AN1003: TG-PM-0523-CICIE R3xx CICIE AT Command Set

This document describes the Home Automation (HA) AT Command Set firmware to be used with Silicon Labs ZigBee® PRO wireless meshing modules on a HA device that can act as a Combined Interface (CI), Control and Indicating Equipment (CIE) device, or On/Off output.

The ZigBee HA Profile is a public application profile which defines device descriptions and standard practices for applications commonly found in a residential or commercial environment. Installation scenarios range from a single room to an entire building. The key applications included in this profile are lighting, HVAC, window shades, and security.

Current Firmware CICIE R311, Hardware ETRX358USB

KEY POINTS

- Provides an overview of the Home Automation (HA) AT command set firmware
- Explains the AT style command conventions
- Provides a list of status codes
- Describes the S-Registers
- Shows how to interpret RSSI energy levels
- Offers step-by-step instructions for upgrading firmware

1. Overview

1.1 Objective

This document describes the Home Automation (HA) AT Command Set firmware to be used with Silicon Labs ZigBee PRO wireless meshing modules on a HA device that can act as a Combined Interface (CI), Control and Indicating Equipment (CIE) device, or On/Off output.

The HA AT Command Set provides a simple AT command interface to implement the CI, CIE, or on/off output of the ZigBee HA Application Profile. The AT command set described in this document can be used to evaluate the Silicon Labs implementation of HA applications. It also enables developers to efficiently design and build ZigBee Home Automation compliant combined interface or CIE products without having to do embedded firmware or RF work. The on/off output can be further developed if required by the customer to satisfy the Home Automation profile specification.

Silicon Labs ZigBee modules are based on the EM35x family of devices, with integrated ARM® Cortex®-M3 processor and IEEE 802.15.4 radio, running the industry-leading EmberZNet ZigBee stack. These modules can be built into any device and provide a low-cost, low-power ZigBee solution. RF experience or expertise is not required to apply this powerful wireless networking capability to your products. Silicon Labs ZigBee Modules offer fast integration opportunities and the shortest possible time to market for your product. This document is meant as an AT-Command and S-Register reference for HA R3xx firmware based on EmberZNet5.x.

1.2 ZigBee Home Automation Profile

The ZigBee HA Profile is a public application profile which defines device descriptions and standard practices for applications commonly found in a residential or commercial environment. Installation scenarios range from a single room to an entire building. The key applications included in this profile are lighting, HVAC, window shades, and security.

1.2.1 Home Automation Devices

Various types of devices can participate in a Home Area Network (HAN) to enable HA applications and market programs. These devices can be categorized into several groups:

Generic Devices	On/Off Switch, Level Control Switch, On/Off Output, Level Controllable Output, Scene Selector, Configuration Tool, Remote Control, Combined Interface, Range Extender, Mains Power Outlet, Door Lock, Door Lock Controller, Simple Sensor
Lighting Devices	On/Off Light, Dimmable Light, Colour Dimmable Light, On/Off Light Switch, Dimmer Switch, Colour Dimmer Switch, Light Sensor, Occupancy Sensor
Closure Devices	Shade, Shade Controller, Window Covering, Window Covering controller
HVAC Devices	Heating/Cooling Unit, Thermostat, Temperature Sensor, Pump, Pump Controller, Pressure Sensor, Flow Sensor
Intruder Alarm System (IAS) Devices	IAS CIE (Control and Indicating Equipment), IAS ACE (Ancillary Control Equipment), IAS Zone, IAS WD (Warning Device)

The Silicon Labs HA device is introduced in this document, and supports three endpoints: 0x01, 0x02, and 0x0A. The endpoint 0x01 has been configured to a combined interface; the endpoint 0x02 has been configured to an IAS CIE device, and the endpoint 0x0A has been configured to an On/Off output device. Table 1.1 Clusters Supported by the Combined Interface on page 2 lists the supported clusters on each endpoint.

1.2.2 Combined Interface

The Combined Interface device is capable of controlling and monitoring other devices. This is typically a mains-powered device, such as a personal computer.

Table 1.1. Clusters Supported by the Combined Interface

Cluster	Cluster ID	Implementation	Description
Basic	0x0000	Server & Client	Provides basic information about a device.
Power configuration	0x0001	Client	Determines detailed information about a device's power source(s).
Device temperature configuration	0x0002	Client	Determines information about a device's internal temperature.
Identify	0x0003	Server & Client	Puts a device into Identification mode.
Group	0x0004	Client	Provides the capability for group addressing.
Scenes	0x0005	Client	Sets/recalls scenes.
On/off	0x0006	Client	Attributes and commands for switching devices between 'on' and 'off' states.
On/off switch configuration	0x0007	Client	Attributes and commands for configuring On/Off switching devices.
Level control	0x0008	Client	Provides an interface for controlling a characteristic of a device that can be set to a level.
Time	0x000A	Server	Provides a basic interface to a real-time clock.
OTA	0x0019	Server	Provides an interoperable mean for devices from different manufacturers to upgrade each other's image over the air.
Poll Control	0x0020	Client	Provides a mechanism to manage an end device's MAC data request rate.
Shade configuration	0x0100	Client	Provides an interface for reading information about a shade and configuring its open/close limits.
Door lock	0x0101	Client	Provides an interface with a generic way to secure a door.
Window covering	0x0102	Client	Provides an interface with the ability to control and adjust automatic window coverings.
Pump configuration and control	0x0200	Client	Provides an interface for setup and control of pump devices.
Thermostat	0x0201	Client	Provides an interface to control the functionality of a thermostat.
Thermostat user interface configuration	0x0204	Client	Provides an interface to allow configuration of the user interface for a thermostat.
Colour control	0x0300	Client	Provides an interface for changing the colour of a light.
Illuminance measurement	0x0400	Client	Provides an interface to the illuminance measurement functionality.
Illuminance level sensing	0x0401	Client	Provides an interface to the illuminance level sensing functionality.
Temperature measurement	0x0402	Client	Provides an interface to the temperature measurement functionality.
Simple Metering	0x0702	Client	Provides a mechanism to retrieve usage information from electric, gas, and water metering devices.

1.2.3 IAS Control and Indicating Equipment (CIE)

The IAS CIE device is the central Control and Indicating Equipment for an Intruder Alarm System. It receives inputs from sensors (Zone Devices) and control equipment (ACE), and sends outputs to a warning device (WD).

Table 1.2. Clusters Supported by IAS CIE

Cluster	Cluster ID	Implementation	Description
Basic	0x0000	Server	Provides basic information about a device.
Identify	0x0003	Server/Client	Puts a device into Identification mode.
IAS Zone	0x0500	Client	Defines an interface to the functionality of an IAS security zone device.
IAS ACE	0x0501	Server	Defines an interface to the functionality of any Ancillary Control Equipment of the IAS system.
IAS WD	0x0502	Client	Provides an interface to the functionality of any Warning Device equipment of the IAS system.

1.2.4 On/Off Output

The On/Off Output device is capable of being switched on and off. This device should only be used when a more specific device specification (for example, a basic light) is not available.

Table 1.3. Clusters Supported by On/Off Output

Cluster	Cluster ID	Implementation	Description
Basic	0x0000	Server	Provides basic information about a device.
Power configuration	0x0001	Server	Determines detailed information about a device’s power source(s).
Device temperature configuration	0x0002	Server	Determines information about a device’s internal temperature.
Identify	0x0003	Server	Puts a device into Identification mode.
Group	0x0004	Server	Provides the capability for group addressing.
Scenes	0x0005	Server	Sets/recalls scenes.
On/off	0x0006	Server	Attributes and commands for switching devices between 'on' and 'off' states.
Generic Tunnel	0x0600	Server/Client	Meets the minimum requirement for use tunnelling.
Tunnelling	0x0704	Server/Client	Provides an interface for tunnelling protocols.

The firmware supports:

- 16 entries of flash-maintained Zone table
- 16 entries of RAM-maintained Zone table
- 22 entries of binding table for ETRX357USB, and 24 entries for ETRX358USB
- 3 entries of reporting table
- 8 entries of address table
- 24 entries of child table for ETRX357USB, and 64 entries for ETRX358USB
- 3 scene entries to be used with an on/off server on endpoint 0x0A

The CICIE device is not designated as a primary cache device, which is capable of buffering messages for sleepy end device with slow polling rate. So the user should consider the sleepy end device's polling settings when using the CICIE to communicate with that sleepy end device.

The CI and CIE endpoints for the ETRX357 module have been HA 1.2.1 certified. However, the CICIE firmware for the 358USB has not yet been certified. The endpoint 0x0A (OnOff output) is not tested, as this endpoint is introduced for demonstration purposes. It contains some customized features, which may affect compliance tests. If the user intends to develop a certified OnOff output (e.g. light) product, it is recommended to contact Silicon Labs for a solution.

1.3 Compatibility with other devices

The CICIE R3xx AT-Command line Interpreter is based on the ZigBee Home Automation specification [6], using the ZigBee PRO feature set [4]. Interoperability is supported with devices that implement the Home Automation Application Profile and use the ZigBee PRO feature set.

Note: The devices should support corresponding clusters, which have been listed in the tables of this chapter.

However, CICIE R3xx is not compatible with devices that do not implement the Home Automation Profile, nor is it compatible with earlier versions of ZigBee.

2. AT Style Command Conventions

To simplify the communication with the Silicon Labs modules, an AT-style command set (similar to the industry standard Hayes modem control language) is used.

Each command must be preceded by an “AT” or “at” prefix. To terminate a command enter <CR>. Any data not following this pattern is either not accepted by the module or will cause an error message in response.

Commands are followed by an optional response that includes <CR><LF><Response><CR><LF> and/or a prompt <CR><LF><Prompt><CR><LF> where the prompt could also be an error message.

Example:

```
ATS00? <CR><CR>
<LF>FFFF<CR><LF>
<CR><LF>OK<CR><LF>
```

It is recommended to wait for an “OK” or “ERROR: XX” prompt before issuing the next command.

Any data prompted to the user is delivered in the format: <CR><LF><prompt><CR><LF>.

Example:

```
<CR><LF><JPAN:11,1789,37BF1CD42CC5E673><CR><LF>
```

A prompt intersecting a command being entered will not affect the command itself.

Throughout this document, only the responses and prompts are presented, and <CR><LF> are omitted intentionally. Sequences of AT commands in a single line are not supported.

Table 2.1. Types of AT Commands

Command	Description
Read Command ATXXX?	Commands ending with a ‘?’ return the current set value of the parameter(s).
Write Command ATXXX=<…>	This command sets user-definable parameters, as indicated by the ‘=’ sign.
Execute Command ATXXX	This command executes routines of the module and returns parameters.

2.1 Parameters

Each parameter must be entered in the correct format for any of the AT commands to execute correctly. Optional parameters are marked with square brackets `[…]`. Some general abbreviations are shown below:

Table 2.2. Different Parameter Formats

Command	Description
`XX`	8-bit hexadecimal number. Valid characters are 0-9, a-f, and A-F
`XXXX`	16-bit hexadecimal number. Valid characters are 0-9, a-f, and A-F
`n`	Numbers from 0-9
`s`	Sign
`b`	Bit (0 or 1)
`c`	Character
`<PANID>`	16-bit hexadecimal PAN ID (0000 to FFFF)
`<EPANID>`	64-bit hexadecimal extended PAN ID
`<channel>`	Decimal channel (802.15.4 channel 11-26)
`<password>`	Eight-character password
`<EUI64>`	64-bit IEEE 802.15.4 address in hexadecimal

2.2 Prompt Overview

The following prompts can show up during the operation of the AT Command set for a CICIE device.

Table 2.3. Prompt Overview

Command	Description
`WRONG HARDWARE`	Check if the module is ETRX3xx.
`OK`	OK terminator.
`ERROR:XX`	Error number XX occurred.
`ACK:XX`	Acknowledgement for message XX was received.
`NACK:XX`	Acknowledgement for message XX was not received.
`LeftPAN`	Local Node has left the PAN.
`LostPAN`	Local Node (not COO) has lost its parent node.
`FFD:<IEEE Address>,<NodeID>`	A router joins CICIE's network.
`SED:<IEEE Address>,<NodeID>`	A sleepy-end device joins CICIE's network.
`ZED:<IEEE Address>,<NodeID>`	A ZigBee-end device joins CICIE's network.
`JPAN:<channel>,<PANID>,<EPANID>`	Local Node has joined PAN with given parameters.
`Bind:<NodeID>,<Status>`	Bind to the target device.
`Unbind: <NodeID>,<Status>`	Unbind from the target device.
`ATTR:<Attribute Value>`	See the +GETATR command description.
`RESPATTR:<NodeID>,<EndPoint>,<ClusterID>,<AttrID>,<Status>,<AttrInfo>`	See the +READATR command description.
`RESPMATTR:<NodeID>,<EndPoint>, <ManufCode>,<ClusterID>,<AttrID>, <Status>,<AttrInfo>`	See the +READMATR command description.
`WRITEATTR:<NodeID>,<EndPoint>,<ClusterID>,<AttrID>,<Status>`	See the +WRITEATR command description.
`WRITEMATTR:<NodeID>,<EndPoint>, <ManufCode>,<ClusterID>,<AttrID>, <Status>`	See the +WRITEMATR command description.
`REPORTATTR:<NodeID>,<EndPoint>,<ClusterID>,<AttrID>,<DataType>,<AttributeValue>`	This prompt is shown when the CICIE receives a *ReportAttribute* command.
`REPORTMATTR:<NodeID>,<EndPoint>,<ManufacturerCode>,<ClusterID>,<AttrID>,<DataType>,<AttributeValue>`	This prompt is shown when the CICIE receives a *ReportAttribute* command for a manufacturer-specified attribute.
`DISCATTR:<NodeID>,<EndPoint>`	See the +ATTRDISC description for more information.
`ENDDISCATTR`	This prompt shows at the end of the attribute discovery response.
`DISCMATTR:<NodeID>,<EndPoint>, <ManufacturerCode>`	See the +MATTRDISC description for more information.
`ENDDISCMATTR`	This prompt shows at the end of a manufacturer-specified attribute discovery response.
`DFTREP:<NodeID>,<EndPoint>,<Cluster>,<CMDID>,<Status>`	Received default response.
`DEV:<Node ID>,<EndPoint>`	ZigBee Service Discovery information. See +DISCOVER for more information.

Command	Description
IDENTIFY:<DestEP>,<SourceNodeID>,<SourceEP>,<TimeOut>	Received an Identify command.
ON:<SourceNodeID>,<SourceEP>	Received an On command.
OFF:<Source NodeID>,<SourceEP>	Received an Off command.
TOGGLE:<Source NodeID>,<SourceEP>	Received a Toggle command.
RX:<NodeID>,<ProfileID>,<DestinationEP>,<SourceEP>,<ClusterID>,<length>: <payload>	Set S0F Bit 9 to enable this prompt (Check 4.2 S-Registers for Module Setup for more information).
CMD:<NodeID>,<CMDID> PAYLOAD: <Raw hexadecimal data>	This prompt is used to show received ZDO commands which are not handled with a specific prompt. For example: MatchDesc for match descriptor response.
EBINDREQ:<NodeID>,<EUI>,<Endpoint>,<ProfileID>INPUTCLUSTER:<ClusterID>…OUTPUTCLUSTER:<ClusterID> …	This prompt is used to show that an End device binding request is received by the CICIE device. The CICIE can handle end device binding request and set up binding for two device endpoint. If the user want to do end device binding for the OnOff output on CICIE firmware with a switch, end device bind request should be sent from the switch to CICIE, then AT+EBIND:0A can be used after EBINDREQ prompt is received.
EBINDRSP:<Status>	This prompt is used to show that an end device binding response is received by the CICIE device.
ZENROLLREQ:<NodeID>,<EndPoint>, <ZoneType>,<ManufactureCode>	This prompt shows when CICIE received an enroll request command. CICIE will automatically handle the request and send a response back if the local zone table is not full.
ENROLLED:<ZID>,<ZoneType>,<EUI>	This prompt shows when a zone device enrolls successfully.
ZONEENTRY:<ZID>,<ZoneType>,<EUI>	Show zone table entry information. See AT+ZENTRY or AT+ZTABLE.
ZMAPSECTION <Index>:<Map>	Show local zone map sections. See AT+PRINTZMAP.
GETZONEINFO:<NodeID>,<EndPoint>, <ZoneID>,<SequenceNumber>	Receive a *GetZoneInfo* command. **Note:** This prompt will only show when the user set S61 bit 6 zone ID is larger than 31.
NOTFOUNDZONE	This prompt will show if the user deletes a non-existent zone entry.
MZONETYPE<index>:<ZoneType>	Show user-specified zone type.
ARM:<NodeID>,<EP>,<ARM Mode>, [<Arm/Disarm code>,<ZoneID>]	Receive an *ARM* command. If the ARM command is sent from a HA1.2 device, <Arm/Disarm code>,<ZoneID> will show.
BYPASS:<NodeID>,<EP>,<Number of Zone>, <ZoneID>…,<Arm/Disarm code>	Receive a *Bypass* command.
EMERGENCY:<NodeID>,<EP>	Receive an *Emergency* command.
FIRE:<NodeID>,<EP>	Receive a *Fire* command.
PANIC:<NodeID>,<EP>	Receive a *Panic* command.

Command	Description
ZONESTATUS:<NodeID>,<EP>, <ZoneStatus>,<ExtendStatus> [,<ZoneID>,<Delay>]	Receive a *Zone Status Change Notification* command. <ZoneStatus> — 16-bit hexadecimal (Check the ZigBee Cluster Library ZoneStatus attribute section for more information.) <ExtendStatus> — 8-bit hexadecimal, this parameter is reserved for additional status If this notification is sent by a HA1.2 device the following information will show: <ZoneID> — 8-bit hexadecimal, zone ID <Delay> — 16-bit hexadecimal, the amount of time (in quarter seconds) from the moment when a change takes place in one or more bits of the *ZoneStatus* and the successful transmission of the *Zone Status Change Notification*. This is designed to help congested networks or offline servers quantify the amount of time from when an event was detected and when it could be reported to the client.
ALARM:<NodeID>,<EP>,<AlarmCode>,<ClusterID>	The CICIE does not support Alarm cluster. This prompt is used to show that the CICIE received an Alarm command from a remote device.
TNLOPEN:<TunnelID>,<Status>	Indicates a tunnel has been opened.
TNLDATA:<TunnelID,<Length>,<Data>	Received tunnel data with length.
TNLERROR:<TunnelID>,<TunnelError>	Received a *TransferDataError* command after sending tunnel data to the target.
TNLCLOCE:<TunnelID>	Indicates a tunnel has been closed.
TNLC00:<EUI>,<ClientEP>,<SourceEP>, <TunnelID>	Local client tunnel information.
TNLS00:<EUI>,<ClientEP>,<SourceEP>, <TunnelID>	Local server tunnel information.
TNLOPEN:<TunnelID>,<Status>	Indicate a tunnel has been opened.
INTERPAN:<ProfileID>,<ClusterID>, <Msgtype>,<Option>,[GroupID],<PanID>, <SrcAddr>,<MsgLength>,<Msg>	This prompt will show when the CICIE receives an interpan message. <ProfileID> — 16-bit hexadecimal <ClusterID> — 16-bit hexadecimal <Msgtype> — 8-bit hexadecimal, 0x00 – Unicast; 0x08 – Broadcast; 0x0C- Multicast <Option> — 16-bit hexadecimal, if it is 0x0002, the <SrcAddr> will be a source long address, and otherwise is a source node ID [GroupID] – 16-bit hexadecimal, if the message is sent to a group, this parameter will show <PanID> — 16-bit hexadecimal, source PANID <SrcAddr> — 16-bit hexadecimal source node ID or EUI <MsgLength> — 8-bit hexadecimal, message length <Msg> — received message in hexadecimal format
GETPANELSTATUS:<NodeID>,<EP>,<Seq>	This prompt will show when the CICIE receives a *GetPanelStatus* command. The user will construct a response and return the same sequence number.
GETBYPASSZLIST:<NodeID>,<EP>,<Seq>	This prompt will show when the CICIE receives a *GetBypassZoneList* command. The user will construct a response and return the same sequence number.
GETZSTATUS:<NodeID>,<EP>, <startingZoneId>,<maxNumberOfZoneIds>,<zoneStatusMaskFlag>, <zoneStatusMask>, <Sequence>	This prompt will show when the CICIE receives a *GetZoneStatus* command. The user will construct a response and return the same sequence number.

Command	Description
`CHECKIN:<NodeId>,<EP>[,Seq]`	This prompt indicates a check-in command is received by CICIE. If S63 Bit 2 is set to TRUE, the sequence number parameter will show. The host will construct a check-in response and return the same sequence number.. By default, S63 will not be set, CICIE will automatically respond. In that case, the host shall use `AT+FPSET` to set the parameter for the check-in response.
`IMGQUERY:<NodeID>,<EP>,<FiledControl>,<ManufCode>,<ImgType>,<CurrentFileVer>[,<HardwareVer>],<SequenceNumber>`[1]	This prompt is used to notify the Host application of a received *QueryNextImageRequest* command. `<NodeID>` — 16-bit hexadecimal. The source network address. `<EP>` — 8-bit hexadecimal. The source endpoint. `<FieldControl>` — 8-bit hexadecimal indicating whether the hardware version parameter is present in this command or not. `<ManufCode>` — 16-bit hexadecimal. The value will be the device's assigned manufacturer code. `<ImgType>` — 16-bit hexadecimal, image type `<CurrentFileVer>` — 32 bit hexadecimal, represents the requesting device's current running image version `<HardwareVer>` — 16-bit hexadecimal, if the <FieldControl> is 1, this parameter is expected. It represents the requesting device's current running hardware. `<SequenceNumber>` — 8-bit hexadecimal, which is the sequence number of the received request command. The user needs to use this sequence number in a response command (`AT+QIMGRSP`). Upon receiving this prompt, the Host application will use a `AT+QIMGRSP` command as a response in 250 ms.

Command	Description
IMGBREQ:<NodeID>,<EP>,<FiledControl>,<ManufCode>,<ImgType>,<FileVer>, <Offset>,<MaxDataSize> [,<RequestNodeAddress>,<BlockRequestDelay>],<SequenceNumber>[1]	This prompt is used to notify the Host application of received *ImageBlockRequest* command. <NodeID> — 16-bit hexadecimal. The source nodeID. <EP> — 8-bit hexadecimal. The source endpoint. <FieldControl> — 8-bit hexadecimal. Used to indicate additional optional fields that may be included in the payload of the Image Block Request command. Field control value 0x00 (bit 0 not set) indicates that the client is requesting a generic OTA upgrade file. Field control value of 0x01 (bit 0 set) means that the client's IEEE address is included in the payload. If Bit 1 is set, the BlockRequestDelay field is present in this prompt. <ManufCode> — 16-bit hexadecimal. The value will be the device's assigned manufacturer code. <ImgType> — 16-bit hexadecimal, image type <FileVer> — 32 bit hexadecimal. Represents the OTA image file version being requested. <Offset> — 32 bit hexadecimal, File offset, its value indicates number of bytes of data offset from the beginning of the file. It essentially points to the location in the OTA image file that the client is requesting the data from. The value reflects the amount of (OTA upgrade image file) data (in bytes) that the client has received so far. <MaxDataSize> — 8-bit hexadecimal. The value indicates the largest possible length of data (in bytes) that the client can receive at once. The server will not send data larger than the maximum data size. If the corresponding bit is set, then the following fields are expected: <RequestNodeAddress> — This is the IEEE address of the client device sending the *ImageBlockRequest* command. <BlockRequestDelay> — 16-bit hexadecimal. This is the current value of the *BlockRequestDelay* attribute of the device making the request. If the device supports the attribute then it will include this field in the request. The value is in milliseconds. <SequenceNumber> — 8-bit hexadecimal. The sequence number of the received request command. The user needs to use this sequence number in the response command (AT+IMGBRSP). Upon receiving this prompt, the Host application shall use a AT+IMGBRSP command as response in 250 ms. If the host stops being able to serve the requested OTA file to a ZigBee module during the OTA file download, it can send a ZCL *DefaultResponse* with status NO_IMAGE_AVAILABLE (0x98) to the client.

Command	Description
`IMGPREQ:<NodeID>,<EP>,<FiledControl>,<ManufCode>,<ImgType>,<FileVer>, <Offset>,<MaxDataSize>,<PageSize>,<ResponseSpacing>[,<RequestNodeAddress>],<SequenceNumber>`[1]	This prompt is used to notify the Host application of a received *ImagePageRequest* command. `<NodeID>` — 16-bit hexadecimal. The source nodeID. `<EP>` — 8-bit hexadecimal. The source endpoint. `<FieldControl>` — 8-bit hexadecimal. Used to indicate additional optional fields that may be included in the payload of the Image Block Request command. Field control value 0x00 (Bit 0 not set) indicates that the client is requesting a generic OTA upgrade file. So *RequestNodeAddress* is not included as an optional field. Field control value of 0x01 (Bit 0 set) means that the client's IEEE address (*RequestNodeAddress*) is included in the payload. `<ManufCode>` — 16-bit hexadecimal. The value will be the device's assigned manufacturer code. `<ImgType>` — 16-bit hexadecimal. Image type. `<FileVer>` — 32-bit hexadecimal. Represents the requested OTA image file version. `<Offset>` — 32-bit hexadecimal. File offset, its value indicates the number of bytes of data offset from the beginning of the file. It essentially points to the location in the OTA image file that the client is requesting the data from. The value reflects the amount of (OTA upgrade image file) data (in bytes) that the client has received so far. `<MaxDataSize>` — 8-bit hexadecimal. The value indicates the largest possible length of data (in bytes) that the client can receive at once. The server will not send data larger than the maximum data size. `<PageSize>` — 8-bit hexadecimal. The value indicates the number of bytes to be sent by the server before the client sends another Image Page Request command. In general, page size value will be larger than the maximum data size value. `<ResponseSpacing>` — 8-bit hexadecimal. The value indicates how fast the server will send the data (via Image Block Response command) to the client. The value is determined by the client. The server shall wait at the minimum the (response) spacing value before sending more data to the client. The value is in milliseconds. If the corresponding bit is set, then the RequestNodeAddress fields is expected: `<RequestNodeAddress>` — This is the IEEE address of the client device sending the Image Block Request command. `<SequenceNumber>` — 8-bit hexadecimal. The sequence number of the received request command. The user needs to use this sequence number in a response command (`AT+IMGBRSP`). Upon receiving this prompt, the Host application shall prepare the OTA file page (usually including multiple blocks), then send blocks in sequence by using `AT+IMGBRSP` commands. The first *ImageBlockResponse* will use the same sequence number contained in this prompt. Then the host will use incremented sequence numbers. Be advised that that the sequence number may not be sequential (for example, if the server is also upgrading another client simultaneously). Additionally, ZCL sequence numbers are only 8-bit and may wrap. If the host no longer able to serve the requested OTA file to a ZigBee module during the OTA file download, it can send a ZCL *DefaultResponse* with status NO_IMAGE_AVAILABLE (0x98) to the client.

Command	Description
`UPGRADEREQ: <NodeID>,<EP>,<Status>,<ManufCode>,<ImgType>,<FileVer>, <SequenceNumber>`[1]	This prompt is used to notify the Host application of a received *UpgradeEndRequest* command. `<NodeID>` — 16-bit hexadecimal. Source network address. `<EP>` — 8-bit hexadecimal. Source endpoint. `<Status>` — 8-bit hexadecimal. Status code. `<ManufCode>` — 16-bit hexadecimal. The value will be the device's assigned manufacturer code. `<ImgType>` — 16-bit hexadecimal. Image type. `<FileVer>` — 32 bit hexadecimal. Newly downloaded OTA upgrade image file version. `<SequenceNumber>` — 8-bit hexadecimal. The sequence number of the received request command. The user needs to use this sequence number in a response command (`AT+UPGRADE`). **Note:** Upon receiving this prompt, the Host application will use a `AT+UPGRADE` command as a response in 250 ms.
`OTAPOLICY:<QueryPolicy>,<UpdatePolicy>`[1]	This prompt is a response of the local command `AT+GETPOLICY`. `<QueryPolicy>` — 8-bit hexadecimal. 00: Upgrade if server has newer 01: Downgrade if server has older 02: Reinstall if server has same 03: No next version `<UpgradeRequestPolicy>` 00: Upgrade now 01: Upgrade in a few minutes 02: Ask me later to upgrade 03: Abort upgrade

Command	Description
OTAIMGID:<ManufCode>,<ImgType>, <FwVer>,<MinHwVer>,<MaxHwVer>, <HostDefinedNodeId>,<ImageSize>[1]	<ManufCode> — 16-bit hexadecimal. The value will be the device's assigned manufacturer code. <ImgType> — 16-bit hexadecimal. OTA file image type id. < FwVer > — 32 bit hexadecimal. Represents the server OTA image's firmware version. < MinHwVer > — 16-bit hexadecimal. The value represents the earliest hardware platform version this image should be used on. The high byte represents the version and the low byte represents the revision. < MaxHwVer>— 16-bit hexadecimal. The value represents the latest hardware platform version this image should be used on. The high byte represents the version and the low byte represents the revision. *On the client side, the hardware version of the device should not be earlier than the minimum (hardware) version or later than the maximum (hardware) version in order to run the OTA upgrade file.* <HostDefinedNodeId> — 16-bit hexadecimal. This firmware provides a feature to allow the host to configure the client's node id in order to provide an image to a specific node (OTA client) for downloading. This parameter can be set by using AT+SETIMGID command. <ImageSize> — 32-bit hexadecimal. Represents the server OTA image's size.

Note:
1. This prompt can appear during the OTA upgrade operation of the AT Command set for a CICIE device.

Prompts related to response will be described in the followings sections, along with their corresponding commands.

2.3 Device Overview

The table below gives an overview of the ZigBee device types mentioned in this document.

Table 2.4. Device Overview

Name	Device Types	ZigBee Naming Convention
COO	Coordinator	ZigBee Coordinator (ZC)
FFD	Router	ZigBee Router (ZR)
ZED	End Device (non-sleepy)	ZigBee End Device (ZED)
SED	Sleepy End Device	

2.4 Addressing modes

Many of the AT commands take a device address as a parameter, which can usually be expressed in several different formats.

EUI64. The Extended Unique Identifier (EUI) of each module consists of 16 hexadecimal characters. This is flashed onto the chip at manufacture and cannot be changed by the user. It can be compared to the permanent MAC address of an IP-based device.

Network address (Node ID). 4 hexadecimal characters. This is allocated to the device when it joins the PAN and cannot be changed or pre-set, except that 0x0000 is always the coordinator. It is analogous to a temporary IP address.

Address table entry. Range 00–07. Used in some ZDO commands.

Binding table entry. Range 10–27 (hexadecimal). Used to send ZCL messages through binding.

FF. In many commands, address FF represents the local device.

2.4.1 AT Command Overview

The following table gives a quick reference of all commands available.

Table 2.5. Command Overview

Command	Description
Module Control and Configuration	
ATI	Display Product Identification Information
ATZ	Software Reset
AT+MTOSTATUS	Concentrator Status
AT+MTOSTART	Concentrator Start
AT+MTOSTOP	Concentrator Stop
AT&F	Restore Local Device's Factory Defaults
AT+RF	Send a Restore Factory Defaults Command to a Remote Device
AT+BLOAD	Enter the Bootloader Menu
ATS	S-Register Access
AT+TOKDUMP	Display All S-Registers
AT+LONOFF	Switch Local Device On/Off
AT+RAWZCL	Construct a Raw ZCL Message and Send to Target
AT+RAWZDO	Construct a Raw ZDO Message and Send to Target
AT+RAWZCLP	Send a Message with Specified Profile ID and Specified Source EP
AT+SENDUCASTB	Send Raw Binary Data to a Remote Node
AT+INTERPAN	Send an Interpan Message
AT+TIMERD	Set Flag to Serve Local Time to the Network
Network Control and Configuration	
AT+ESCAN	Scan the Energy of All Channels
AT+PANSCAN	Scan for Active PAN's
AT+EN	Establish Network
AT+PJOIN	Switch on "Permit Joining" Flag
AT+JN	Join Network
AT+JPAN	Join Specific PAN
AT+SJN	Silent Join
AT+REJOIN	Initiate a Rejoin (used if the device is not a COO)
AT+DASSL	Disassociate Local Node from PAN
AT+DASSR	Disassociate Remote Node from PAN (ZDO)
AT+N	Display Network Information
AT+NTABLE	Display Neighbour Table (ZDO)
AT+RTABLE	Display Routing Table (ZDO)
AT+IDREQ	Request Node's NodeID (ZDO)
AT+EUIREQ	Request Node's EUI (ZDO)

Command	Description
AT+NODEDESC	Request Node's Descriptor (ZDO)
AT+POWERDESC	Request Node's Power Descriptor (ZDO)
AT+ACTEPDESC	Request Node's Active Endpoint List (ZDO)
AT+SIMPLEDESC	Request Endpoint's Simple Descriptor (ZDO)
AT+MATCHREQ	Find Nodes that Match a Specific Descriptor (ZDO)
AT+ANNCE	Announce Local Device in the Network (ZDO)
AT+KEYUPD	Update the Network Key (ZDO)
AT+CCHANGE	Change the Network's Channel
AT+CSWITCH	Switch the Network's Channel without Notifying Other Nodes
Messaging	
AT+ATABLE	Display Address Table
AT+ASET	Set Address Table Entry
AT+LBTABLE	Display Local Binding Table
AT+BSET	Set Local Binding Table Entry
AT+BCLR	Clear Local Binding Table Entry
AT+BTABLE	Display Remote Binding Table (ZDO)
AT+BIND	Create Binding on Remote Device (ZDO)
AT+UNBIND	Delete Binding on Remote Device (ZDO)
AT+EBIND	End Device Binding
ZigBee Discovery and Attributes Commands	
AT+DISCOVER	Discover HA Device on the HAN
AT+DISCOVERP	Discover Devices using Profile ID and Cluster ID
AT+CLUSDISC	Discover All Supported Clusters on a Remote Device
AT+ATTRDISC	Discover Supported Attributes on a Remote Device
AT+READATR	Get Server Attributes from Targets
AT+READCATR	Get Client Attributes from Targets
AT+READMATR	Get Manufacturer Specific Server Attributes from Targets
AT+READMCATR	Get Manufacturer Specific Client Attributes from Targets
AT+WRITEATR	Write Server Attributes to Targets
AT+WRITECATR	Write Client Attributes to Targets
AT+WRITEMATR	Write Manufacturer Specific Server Attributes to Targets
AT+WRITEMCATR	Write Manufacturer Specific Client Attributes to Targets
AT+READMRPTCFG	Read Manufacture Reporting Configuration from Targets
AT+READRPTCFG	Read Reporting Configuration from Targets
AT+CFGMRPT	Configure Manufacture Defined Attribute Reporting
AT+CFGRPT	Configure Attributes Reporting
AT+SETATR	Set Value of a Local ZCL Attribute

Command	Description
AT+GETATR	Get Value of a Local ZCL Attribute
Combined Interface Specified Commands	
AT+IDENTIFY	Starts or Stops the Receiving Device Identifying Itself
AT+IDQUERY	Query if Target Device(s) are in Identifying Mode
AT+GPADD	Add Group on Target Device
AT+GPADIFID	Add Group Membership on Target that is in Identification Mode
AT+GPVIEW	View Group Information of Target Devices
AT+GPGET	Get Group Membership Information from Target Devices
AT+GPRMV	Remove a Group Membership on Target Device
AT+GPRMALL	Remove All Group Associations on a Target Device
AT+SCADD	Add Scenes
AT+SCVIEW	View Scenes
AT+SCRMV	Remove Scenes
AT+SCRMALL	Remove All Scenes
AT+SCSTORE	Store Scenes
AT+SCRECAL	Recall Scenes
AT+SCGETMSH	Get Scene Membership
AT+RONOFF	Switching Target Devices Between ‘On’ and ‘Off’ States
AT+LCMVTOLEV	Level Control Move to Level Command
AT+LCMV	Level Control Move Command
AT+LCSTEP	Level Control Step Command
AT+LCSTOP	Level Control Stop Command
AT+CCMVTOCOL	Colour Control Move to Colour Command
AT+CCMVCOL	Colour Control Move Colour Command
AT+CCSTEPCOL	Colour Control Step Colour Command
AT+CCMVTOHUE	Colour Control Move to Hue Command
AT+CCMVHUE	Colour Control Move Hue Command
AT+CCSTEPHUE	Colour Control Step Hue Command
AT+CCMVTOSAT	Colour Control Move to Saturation Command
AT+CCMVSAT	Colour Control Move Saturation Command
AT+CCSTEPSAT	Colour Control Step Hue Command
AT+CCMVTOHUS	Colour Control Move to Hue and Saturation Command
AT+CCMVTOCT	Colour Control Move to Colour Temperature Command
AT+TSTATSET	Thermostat Set
AT+DRLOCK	Lock/Unlock Door
AT+WNPGSETPT	Window Program Setpoint
AT+WNSETPT	Window go to Lift/Tile Setpoint

Command	Description
AT+WNVALUE	Window go to Lift/Tile Value
AT+WNPCTG	Window go to Lift/Tile Percentage
AT+WNCTRL	Open, Close, or Stop Window
CIE Specified Commands	
AT+WARNING	Send a Start Warning Command to a Warning Device
AT+SQUAWK	Send a Squawk Command to a Warning Device
AT+ADDZENTRY	Add Zone Table Entry
AT+DELZENTRY	Delete Zone Table Entry
AT+ZENTRY	Print a Local Zone Table Entry
AT+ZTABLE	Print Local Zone Table
AT+ZCLEAR	Clear Local Zone Table
AT+PRINTMZTYPE	Print Local Table of Manufacture Defined Zone Type
AT+MZONETYPE	Add Manufacture Defined Zone Type in Local Zone Type Table
AT+ARMPRINT	Show CIE Arm Mode
AT+PRINTZMAP	Print Local Zone Map
AT+EDITZMAP	Edit Local Zone Map
AT+ZINFORSP	Construct and Send a Get Zone Information Response
AT+ZTEST	Place a Remote Zone Server Device in Test Mode
AT+ZNORMAL	Set a Remote Zone Server Device to Normal Working Mode
Tunnel Commands	
AT+TNLOPEN	Send an Open Tunnel Request to the Target
AT+TNLCLOSE	Send a Close Tunnel Command to the Target
AT+TNLSEND	Send Data to the Target Through an Established Tunnel
AT+TNLSTATUS	Check Local Tunnel Information
Time Cluster Commands	
AT+SETTIME	Set the Local Time
AT+GETTIME	Get the Local Time
OTA Communication Commands	
AT+IMGNOTIFY	Notify OTA Client of the Newly Available Image
AT+UPGRADE	Instruct a Device to Upgrade Now
AT+QIMGRSP	Query Next Image Response
AT+IMGBRSP	Image Block Response
AT+DFTRSP	Default Response for OTA Request
OTA Server Configuration Commands	
AT+GETPOLICY	Print the OTA Server Policy
AT+SETPOLICY	Set the OTA Server Policy
AT+SETIMGID	Set the OTA Server Image ID and Other Relevant Information

Command	Description
AT+GETIMGID	Print the OTA Server Image ID and Hardware Version
AT+SETHWVER	Set the Allowable OTA Client's Hardware Version Range
Poll Control Client Commands	
AT+FPSET	Set Local Flag and Timeout for Starting Fast Poll
AT+FPSTOP	Stop Fast Poll
AT+LPINTVL	Set Long Poll Interval
AT+SPINTVL	Set Short Poll Interval

2.4.2 Common Command Set

Module Control and Configuration Commands

Table 2.6. I – Display Product Identification Information

Execute Command	Response
ATI	Telegesis <DeviceName> <Firmware Revision> <EUI64> OK <DeviceName> is the device type <Firmware Revision> is the firmware revision <EUI64> is the device's IEEE 802.15.4 Identifier

Table 2.7. Z – Software Reset

Execute Command	Response
ATZ	OK Module performs software reset. All non-volatile S-Registers keep the user-defined values. If the module was part of a PAN, this will remain unchanged.

Table 2.8. +MTOSTATUS – Check Local Many-To-One Route Discovery

Execute Command	Response
AT+MTOSTATUS	ACTIVE:Y/N OK *Y* means Enabled, and *N* means Disabled.

Table 2.9. +MTOSTART – Start Many-To-One Route Discovery

Execute Command	Response
AT+MTOSTART	MTOSTART OK

Table 2.10. +MTOSTOP – Stop Many-To-One Route Discovery

Execute Command	Response
AT+MTOSTOP If a message is not delivered successfully (usually show ERROR: 66), MTO will be re-activated automatically.	MTOSTOP OK

Table 2.11. &F – Restore Local Device's Factory Defaults

Execute Command	Response
AT&F	OK Module performs a factory reset. All non-volatile S-Registers are updated with their factory defaults and the node leaves the currently joined network.

Table 2.12. +RF – Restore Target Device's Factory Defaults

Execute Command	Response
AT+RF:<Address>,<EP>,<SendMode> <Address> — 16-bit hexadecimal number. It will be the Node ID of a remote device if the command is sent directly to a node or it shall be a group ID if the command is sent to a group. <EP> — 8-bit hexadecimal number, which is the endpoint of a remote device. <SendMode> — A Boolean type to choose transmission mode ('0' sends the command directly, '1' sends the command to a group). If sending the command to a group, the second parameter <EP> will be ignored. The user can leave it blank. For example: AT+RF:0001,0A,0 (sends the command to a node) AT+RF:1234,,1 (sends the command to a group)	OK or ERROR:<errorcode> <errorcode> represents the error code. (See 3. List of Status codes.) A default response may be received to show the device does not support this command, as this is an optional command in Basic cluster.

Table 2.13. +BLOAD – Enter The Boot Loader Menu

Execute Command	Response
AT+BLOAD The bootloader will run at a baud rate of 115 k, no parity, and eight data bits, regardless of the current serial port settings.	<entering bootloader> The device leaves the AT command line and enters the Ember bootloader menu for downloading new firmware. A description of the bootloading process can be found in 6. Firmware Upgrades of this document. A description of the bootloading process can be found in the Evaluation Kits Product Manual.

Table 2.14. S – S-Register Access

Read Command	Response
ATSXX[x]?	<data> OK or ERROR:<errorcode> The module displays the contents of S-register xx or an error message, where <errorcode> represents the error code explained in 3. List of Status codes. All 16-bit registers can also be accessed bit by bit. In order to do this, [x] may specify the bit which is to be read. The result when reading a single bit will always be 0 or 1.
Write Command	**Response**
ATSXX[x]=<data> Some S-Registers require a password for write access. See S-Register description for details. The default password is “password”. Some S-Registers are read-only and will return an error if written to.	OK or ERROR: <errorcode> The data is written to S-register number XX, and if applicable, stored in non-volatile memory. The data format for each individual S-Register is given in the S-Register description. All 16-bit registers with individual bits can also be set or cleared by specifying the bit using [x], and setting it to either 0 or 1.

Table 2.15. TOKDUMP – Display All S-Registers

Execute Command	Response
AT+TOKDUMP Only used on the local node.	<data> OK The module displays the contents of all local S-Registers. The data format for each individual S-Register is given in the S-Register description in 4. S-Registers.

Table 2.16. +LONOFF – Set On or Off Status for the Local Node (For Local LED)

Execute Command	Response
AT+LONOFF:<On/Off> <On/Off> — '1' means on; '0' means off. This parameter can be omitted, and AT+LONOFF can be used to toggle the LED on USB. Use on: Local node	OK or ERROR:<errorcode> (See 3. List of Status codes.)

Table 2.17. +RAWZCL – Construct A Raw ZCL Message And Send To Target

Execute Command	Response

`AT+RAWZCL:<NodeID>,<EP>,<ClusterID>,<data>`
 	
`<NodeID>` — 16-bit hexadecimal number, which is the network address of a remote device.
 	
`<EP>` — 8-bit hexadecimal number, which is the endpoint of a remote device.
 	
`<ClusterID>` — 16-bit hexadecimal number which represents the cluster ID, see 2.4.3 Combined Interface Specified Command Set.
 	
`<data>` — A constructed ZCL command in hexadecimal format, including the frame control and sequence number .
 	
For instance, a raw *ReadAttributeRequest* command to read the Basic cluster attribute (0x0001) can be:
 	
`AT+RAWZCL:1234,01,0000,0002000100`
 	
Payload part: 00: Frame control
 	
02: Sequence number
 	
00: Command ID
 	
0100: Attribute ID (little endian)
 	
Check the ZigBee Cluster Library for reference.
	`OK`
	
	`<Response data>`
	
	or
	
	`ERROR:<errorcode>` (see 3. List of Status codes).
	
	The response may vary depending on the sent command.
	
	When constructing a Raw ZCL message, please check ZigBee ZCL Specification [5] and ZigBee HA Specification [6] for references. Incorrect construction can be transmitted but will not expect any response.

Table 2.18. +RAWZDO – Construct A Raw ZDO Message And Send To Target

Execute Command	Response

`AT+RAWZDO:<NodeID>,<ClusterID>, <data>`
 	
`<NodeID>` — 16-bit hexadecimal number, the network address of a target device or FFFD to broadcast to all routers.
 	
`<ClusterID>` — ZDO Command ID, see ZigBee Specification [4] for more information.
 	
`<data>` — Constructed payload of a ZDO command (Check ZigBee Specification for reference).
	`OK`
	
	`CMD:<ClusterID>,PAYLOAD<Response data>`
	
	or
	
	`ERROR:<errorcode>` (See 3. List of Status codes.)
	
	The response may vary depending on the sent command.
	
	When constructing a Raw ZDO message, check ZigBee Specification [4] for references. Incorrect construction can be transmitted but will not expect any response.

Table 2.19. +RAWZCLP – Send A Raw ZCL Message With Specified ProfileID And Source Endpoint

Execute Command	Response

`AT+RAWZCLP:<NodeID>,<DstEP>,[SrcEP], [ProfileID],<ClusterID>,<data>`
 	
`<NodeID>` — 16-bit hexadecimal number, which is the network address of a remote device.
 	
`<DstEP>` — 8-bit hexadecimal number, which is the destination endpoint of a remote device.
 	
`[SrcEP]` — 8-bit hexadecimal number, which is the source endpoint of a local device. If omitted, the source endpoint is set to 0x02
 	
`[ProfileID]` — 16-bit hexadecimal number which represents the profile ID. (E.g. 0x0104 for a ZigBee home automation profile.) If omitted, the profile ID is set to 0xC216.
 	
`<ClusterID>` — 16-bit hexadecimal number which represents the cluster ID. See 2.4.3 Combined Interface Specified Command Set.
 	
`<data>` — a constructed ZCL command in hexadecimal format (check the ZigBee Cluster Library for references).
	`OK`
	
	`<Response data>`
	
	or
	
	`ERROR:<errorcode>` (See 3. List of Status codes.)
	
	The response may vary depending on the sent command.
	
	When constructing a Raw ZCL message, check the ZigBee ZCL Specification [5] and ZigBee HA Specification [6] for references. Incorrect construction can be transmitted but will not expect any response.

Table 2.20. +SENDUCASTB – Send Raw Binary Data To A Remote Node

Execute Command	Response

`AT+SENDUCASTB:<NodeID>,<DstEP>, <SendMode>,<SrcEP],<ProfileID>, <ClusterID>,<dataLength>`
 	
`<NodeID>` — 16-bit hexadecimal number, which is the network address of a remote device.
 	
`<DstEP>` — 8-bit hexadecimal number, which is the destination endpoint of a remote device.
 	
`<SendMode>` — A Boolean type to choose transmission mode ('0' sends the command directly, and '1' sends the command to a group).
 	
`<SrcEP>` — 8-bit hexadecimal number, which is the source endpoint of a local device.
 	
`<ProfileID>` — 16-bit hexadecimal number which represents the profile ID.
 	
`<ClusterID>` — 16-bit hexadecimal number which represents cluster ID. See 1.2.2 Combined Interface.
 	
`<dataLength>` — 8-bit hex, input data length.
	`><data being entered>`
	
	`OK`
	
	or
	
	`ERROR:<errorcode>` (See 3. List of Status codes.)
	
	The response may vary depending on the sent command.
	
	When constructing a Raw message, check the ZigBee ZCL Specification [5] and ZigBee HA Specification [6] for references. Incorrect construction can be transmitted but will not expect any response.

Table 2.21. +INTERPAN – Send an Interpan Command

Execute Command	Response
`AT+INTERPAN:<AddressMode>,<DstAddress>,<DstPAN>, <ProfileID>,<ClusterID>,<Payload>` `<AddressMode>` — 8-bit hexadecimal number. The user shall use this parameter to specify which type of the destination address is used ('00' is Node ID; '01' is Group ID; '02' is EUI address). `<DstAddress>` — 16-bit hexadecimal number if Node ID or Group ID is used. This may also be the EUI address, if the user uses long destination address. `<DstPAN>` — 16-bit hexadecimal number representing destination PAN ID. `<ProfileID>` — 16-bit hexadecimal number representing profile ID. (E.g. 0x0104 for Home automation or 0xC05E for ZigBee Light Link.) `<ClusterID>` — 16-bit hex number representing Cluster ID. `<Payload>` — Command payload, the user should input the ASCII hex data.	`OK` or `ERROR:<errorcode>` (See 3. List of Status codes.)

Table 2.22. TIMERD – Set Flag to Serve Local Time To The Network

Execute Command	Response
`AT+TIMERD` This command needs to be used after the user sets the Time cluster attributes (time, timeStatus attributes) with valid values. When CICIE initiates, the time will be invalid. So it will respond to a read time attribute command with a default response and failure status. The user (host) needs to set time cluster attributes (time, timeStatus) by using `AT+SETATR` command. After that, the user needs to apply this command in order to serve the time to a HA network. From then on, the CICIE will send read attribute response with its maintained clock time upon receiving a request.	`OK` Check Appendix 3. Time Server Clusters for more information on using the CICIE time server cluster.

Network Control & Configuration Commands

Table 2.23. +ESCAN – Scan The Energy Of All Channels

Execute Command	Response
AT+ESCAN Use on: All nodes Scanning all channels can take up to four seconds. Channels masked out in S00 are not scanned.	+ESCAN: 11:XX … 26:XX OK or ERROR:<errorcode> (See 3. List of Status codes.) *XX* represents the average energy on the respective channel (see description in 5. Interpreting RSSI Energy Levels).

Table 2.24. +PANSCAN – Scan For Active PANs

Execute Command	Response
AT+PANSCAN Use on: All nodes Scanning for active PANs can take up to 4 seconds. The node does not join any of the PANs found. Channels masked out in S00 are not scanned.	+PANSCAN:<channel>,<PANID>,<EPANID>,XX,b OK or +PANSCAN:<channel>,<PID>,<EPID>,XX,b, <rssi>,<LQI> or ERROR:<errorcode> (See 3. List of Status codes.) <channel> represents the channel <PANID> the PAN ID <EPANID> the extended PAN ID The node gives a list of all PANs found. *XX* the ZigBee stack profile (00 = Custom, 01 = ZigBee, 02 = ZigBee PRO) b indicates whether the network is allowing additional nodes to join (1 = joining permitted). If bit 1 of S63 is set the response includes RSSI and LQI

Table 2.25. +EN – Establish Network

Execute Command	Response

`AT+EN:[<channel>],[<POWER>],[<PANID>]`
 	
Use on: Coordinators, which are not a part of a PAN.
 	
Establishing a PAN can take up to four seconds.
 	
This command can only be executed if the local node is not part of a PAN already.
	`OK`
	
	`JPAN:<channel>,<PANID>,<EPANID>`
	
	or
	
	`ERROR:<errorcode>` (See 3. List of Status codes.)
	
	The local node becomes a coordinator and performs an energy scan on all channels selected in 4.1 S-Registers for Network Set-up. It then starts a PAN with a random unused PAN ID and extended PAN ID on the quietest channel. If a PAN ID and/or extended PAN ID is specified, the provided IDs are used instead of random ones, given the selected IDs are not already in use by other networks within range

Table 2.26. +JN – Join Network

Execute Command	Response

`AT+JN`
 	
Use on: All nodes which are not part of a PAN
 	
Joining a PAN can take up to four seconds, depending on the number of channels which need scanning.
 	
This command can only be executed if the local node is not part of a PAN already.
	`OK`
	
	`JPAN:<channel>,<PANID>,<EPANID>`
	
	or
	
	`ERROR:<errorcode>` (See 3. List of Status codes.)
	
	The local node scans all channels selected in register S03 for the existence of a PAN. When finding a PAN which allows joining it will automatically join via the router with the best signal quality.
	
	Where registers S02 and S03 differ from the default value of all zeroes, the node will only join a PAN with the specified PAN ID and/or extended PAN ID.

Table 2.27. +PJOIN – Permit Joining

Read Command	Response
AT+PJOIN[:<sec>,<NodeID>] <sec> — 8-bit hexadecimal number which represents the length of time in seconds during which the ZigBee coordinator or router will allow associations <NodeID> — 16-bit hexadecimal number, network address of a target device. If FFFC is used, the constructed command will be sent as a broadcast. Use on: Coordinator or router The HA network should not leave permit join on permanently, S0A has been set to switch off permit join. If <sec> and <NodeID> are omitted, the CICIE will permit join for 60 seconds. If there is <sec> input, the CICIE will apply the period that is specified by the user. If <NodeID> is filled with FFFC, the CICIE will broadcast a permit join command to the other nodes on the HA network to set permit join on. If the user wants to send a unicast permit join command to a device, <NodeID> should be the network address of that node.	OK or ERROR:<errorcode> (See 3. List of Status codes.) If the command is undeliverable (for unicast case), prompt "NACK" will show. Please note that there is no ACK prompt for this command for both unicast and broadcast.

Table 2.28. +JPAN – Join Specific PAN

Execute Command	Response
AT+JPAN:[<channel>],[<PANID>],[<EPANID>] <channel> — 2 digit decimal number which represents channel number. Range from 11 to 26. <PANID> — 16-bit hexadecimal number which represents short PAN ID of ZigBee network. <EPANID> — 64 bit hexadecimal number which represents extended PAN ID of ZigBee network. Each parameter is optional for input, but "," is required when the parameter is left blank. Examples: AT+JPAN:20 (only specifies channel) AT+JPAN:,1234, (only specifies PANID) The *+JPAN* command can only be executed if the local node is not part of a PAN already.	OK JPAN:<channel>,<PANID>,<EPANID> or ERROR:<errorcode> (See 3. List of Status codes.) The local node joins a particular PAN on <channel> with the specified <PID> or <EPID> via the router with an adequate signal quality and the fewest hops to the COO.

Table 2.29. +SJN – Silent Join

Execute Command	Response
AT+SJN:<channel>,<TCEUI64>,<NMNodeID>,<nwk update ID> "Silent" joining is joining via the commissioning method. All data required to enter the network is provided to the node, so that no joining procedure itself is required. The node will appear in the target network without any joining procedure given that the supplied data is correct. <channel> is a decimal number, while the other parameters are hexadecimal. Use on: All joining devices Example: AT+SJN:11,000D6F00000AAAD0,AFFE,00	JPAN:<channel>,<PANID>,<EPANID> OK or ERROR:<errorcode> (See 3. List of Status codes.) The local node will become part of the network with the channel specified in <channel>, the trust centre EUI64 specified in <TC EUI64>, the NodeID of the network manager specified in <NM NodeID>, the 8-bit network update ID specified in <nwk update ID>, the network key provided in S08, the trust centre link key provided in S09, the PAN ID provided in S02, and the extended PAN ID provided in S03. It is assumed that the key-sequence-number of the network key is 0 when issuing this command.

Table 2.30. +REJOIN – Rejoin the Network

Execute Command	Response
AT+REJOIN:b If *b* is set to 0, join without the known network key (unencrypted). If *b* is set to 1, join encrypted. This command cannot be used on a COO.	LostPAN OK JPAN:<Channel>,<PANID>,<EPANID> or ERROR:<errorcode> (See 5. Interpreting RSSI Energy Levels.) If the contact with the network has been lost because the network has changed channel, or updated its encryption key, the command AT+REJOIN can be used to re-join the network.

Table 2.31. +DASSL – Disassociate Local Device From PAN

Execute Command	Response
AT+DASSL Use on: All Devices Use with care on a Coordinator. It will not be able to re-join the PAN.	OK or ERROR:<errorcode> (See 3. List of Status codes.) Prompt: LeftPAN Instruct local device to leave the PAN.

Table 2.32. +DASSR – Disassociate Remote Node from PAN (ZDO)

Execute Command	Response

`AT+DASSR:<address>`
 	
Where `<address>` can be a node's EUI64, Node ID or address table index
 	
Use on: All Devices
 	
Use with care when targeting a Coordinator. It will not be able to re-join the PAN
 	
Remote Action: Node leaves PAN
	`SEQ:XX`
	
	`OK`
	
	`ERROR:<errorcode>` (See 3. List of Status codes.)
	
	Instruct device to leave the PAN.
	
	Prompt: LeftPAN

Table 2.33. +N – Display Network Information

Read Command	Response

`AT+N`
 	
Use on: All Devices
	`+N=<devicetype>,<channel>,<power>,<PANID>,<EPANID>`
	
	or
	
	`+N=NoPAN`
	
	followed by `OK`
	
	`<devicetype>` represents the node's functionality in the PAN (FFD,COO,ZED,SED).
	
	`<power>` the node's output power measured in dBm.
	
	`<channel>` the IEEE 802.15.4 radio channel (11–26).
	
	`<PANID>` the node's PAN ID.
	
	`<EPANID>` the node's extended PAN ID.

Table 2.34. +NTABLE – Display Neighbour Table (ZDO)

Read Command	Response

`AT+NTABLE:XX,<address>`
 	
Where XX is the start index of the remote LQI table and `<address>` can be the remote node's EUI64, NodeID or address table entry.
 	
Use on: FFD, COO as the target device
 	
The local node can also be the target of this command (e.g. use address table entry FF as the address).
	`SEQ:XX`
	
	`OK`
	
	or
	
	`ERROR<errorcode>`
	
	This command requests the target node to respond by listing its neighbour table, starting from the requested index. Can be used to find the identity of all ZigBee devices in the network, including non-Silicon Labs devices.
	
	Example prompt:
	
	`NTable:<NodeID>,<errorcode>`
	
	`Length:03`
	
	`No.| Type | EUI | ID | LQI`
	
	`0. | FFD | 000D6F000015896B | BC04 | FF`
	
	`1. | FFD | 000D6F00000B3E77 | 739D | FF`
	
	`2. | FFD | 000D6F00000AAD11 | 75E3 | FF`
	
	In this example, the neighbour table of the remote node with the short ID shown in `<NodeID>` contains three entries (hexadecimal), which are displayed. In case the table contains more than three entries, it may be required to repeat this command and increase the index count until the full table is derived.
	
	In case of an error, an errorcode other than 00 will be displayed and the prompt will end after the errorcode.
	
	After successful transmission, the sequence number of the unicast is stated using the "SEQ:XX" prompt. When acknowledged (or not) the accompanying "ACK:XX" (or "NACK:XX") prompt is displayed.

Table 2.35. +RTABLE – Display Routing Table (ZDO)

Read Command	Response

`AT+RTABLE:XX,<address>`
 	
Where XX is the start index of the remote routing table and `<address>` can be the remote node's EUI64, NodeID, or address table entry.
 	
Use on: FFD, COO as the target device
 	
The local node can also be the target of this command (e.g. use address table entry FF as the local address).
	`SEQ:XX`
	
	`OK`
	
	or
	
	`ERROR<errorcode>`
	
	This command requests the target node to respond by listing its routing table starting from the requested index.
	
	Example prompt:
	
	`RTable:<NodeID>,<errorcode>`
	
	`Length:03`
	
	`No.| Dest | Next | Status`
	
	`0. | 1234 | ABCD | 00`
	
	`1. | 4321 | 739D | 00`
	
	`2. | 0000 | 0000 | 03`
	
	In this example, the routing table of the remote node with the short ID shown in `<NodeID>` contains 64 entries (hexadecimal 0x40), of which the first three are displayed. When the table contains more than the displayed entries, it may be required to repeat this command and increase the index count until the full table is derived.
	
	The status will appear as described in Table 2.128 of the ZigBee Pro Specification [4].
	
	In case of an error, an errorcode other than 00 will be displayed and the prompt will end after the errorcode.
	
	After successful transmission, the sequence number of the unicast is stated using the "SEQ:XX" prompt. When acknowledged (or not), the accompanying "ACK:XX" (or "NACK:XX") prompt is displayed.

Table 2.36. +IDREQ – Request Node's NodeID (ZDO)

Execute Command	Response

`AT+IDREQ:<Address>[,XX]`
 	
Where `<Address>` can be a node's EUI64, or address table entry and XX is an optional index number. When an index number is provided, an extended response is requested asking the remote device to list its associated devices (i.e. children).
 	
Sends a broadcast to obtain the specified Device's NodeID and (optionally) also elements of its associated devices list.
 	
Use on: All Devices
 	
Providing FF as an address table entry addresses the local node.
	`OK`
	
	or
	
	`ERROR:<errorcode>`
	
	`<errorcode>` (See 3. List of Status codes.)
	Prompt:
	
	`AddrResp:<errorcode>[,<NodeID>,<EUI64>]`
	
	`[nn. <NodeID>]`
	
	In case of an error, an errorcode other than 00 will be displayed and the prompt will end after the errorcode.
	
	`<EUI64>` is the Remote node's EUI64 and `<NodeID>` is its NodeID. In case an extended response has been requested, the requested NodeIDs from the associated devices list are shown as well.

Table 2.37. +EUIREQ – Request Node's EUI64 (ZDO)

Execute Command	Response

`AT+EUIREQ:< Address>,<NodeID>[,XX]`
 	
Where `<Address>` is the EUI64, NodeID, or address table entry of the node which is to be interrogated about the node with the Node ID specified in `<NodeID>`. XX is an optional index number. In case an index number is provided, an extended response is requested asking the remote device to list its associated devices (i.e. children).
 	
Sends a unicast to obtain the specified device's EUI64 and optionally also elements of its associated devices list (extended response).
 	
Use on: All Devices
 	
Providing FF as an address table entry addresses the local node.
	`SEQ:XX`
	
	`OK`
	
	or
	
	`ERROR:<errorcode>`
	
	`<errorcode>` (See 3. List of Status codes.)
	Prompt:
	
	`AddrResp:<errorcode>[,<NodeID>,<EUI64>]`
	
	In case of an error, an errorcode other than 00 will be displayed and the prompt will end after the errorcode.
	
	`<EUI64>` is the Remote node's EUI64 and `<NodeID>` is its NodeID. Where an extended response has been requested, the requested NodeIDs from the associated devices list are shown.
	
	After successful transmission, the sequence number of the unicast is stated using the "SEQ:XX" prompt. When acknowledged (or not) the accompanying "ACK:XX" (or "NACK:XX") prompt is displayed.

Table 2.38. +NODEDESC – Request Node's Descriptor (ZDO)

Execute Command	Response

`AT+NODEDESC:<Address>,<NodeID>`
 	
Where `<Address>` is the EUI64, NodeID, or Address table entry of the node which is to be interrogated about the node with the NodeID specified in `<NodeID>`.
 	
Sends a unicast to obtain the specified device's node descriptor.
 	
Use on: All Devices
 	
Providing FF as an address table entry addresses the local node.
	`SEQ:XX`
	
	`OK`
	
	or
	
	`ERROR:<errorcode>`
	
	`<errorcode>` (See 3. List of Status codes.)
	Example prompt:
	
	`NodeDesc:<NodeID>,<errorcode>`
	
	`Type:FFD`
	
	`ComplexDesc:No`
	
	`UserDesc:No`
	
	`APSFlags:00`
	
	`FreqBand:40`
	
	`MacCap:8E`
	
	`ManufCode:1010`
	
	`MaxBufSize:52`
	
	`MaxInSize:0080`
	
	`SrvMask:0000`
	
	`MaxOutSize:0080`
	
	`DescCap:00`
	
	In case of an error, an errorcode other than 00 will be displayed and the prompt will end after the errorcode.
	
	`<NodeID>` is the Remote node's NodeID. In addition, the node descriptor is displayed. The individual fields of the Node Descriptor are described in section 2.3.2.3 of the ZigBee Pro specification [4].
	
	After successful transmission, the sequence number of the unicast is stated using the "SEQ:XX" prompt. When acknowledged (or not) the accompanying "ACK:XX" (or "NACK:XX") prompt is displayed.

Table 2.39. +POWERDESC – Request Node's Power Descriptor (ZDO)

Execute Command	Response

`AT+POWERDESC:<Address>,<NodeID>`
 	
Where `<Address>` is the EUI64, NodeID, or Address table entry of the node which is to be interrogated about the node with the NodeID specified in `<NodeID>`.
 	
Sends a unicast to obtain the specified device's power descriptor.
 	
Use on: All Devices
	`SEQ:XX`
	
	`OK`
	
	or
	
	`ERROR:<errorcode>`
	
	`<errorcode>` (See 3. List of Status codes.)
	Prompt:
	
	`PowerDesc:<NodeID>,<errorcode>[,<PowerDescriptor>]`
	
	In case of an error, an errorcode other than 00 will be displayed and the prompt will end after the errorcode.
	
	`<NodeID>` is the Remote node's NodeID. In addition, the `<PowerDescriptor>` is displayed as a 16-bit hexadecimal number described in section 2.3.2.4. of the ZigBee Pro Specification.
	
	After successful transmission, the sequence number of the unicast is stated using the "SEQ:XX" prompt. When acknowledged (or not) the accompanying "ACK:XX" (or "NACK:XX") prompt is displayed.

Table 2.40. +ACTEPDESC – Request Node's Active Endpoint List (ZDO)

Execute Command	Response

`AT+ACTEPDESC:<Address>,<NodeID>`
 	
Where `<Address>` is the EUI64, NodeID, or Address table entry of the node which is to be interrogated about the node with the NodeID specified in `<NodeID>`.
 	
Sends a unicast to obtain the specified device's active endpoint list.
 	
Use on: All Devices
	`SEQ:XX`
	
	`OK`
	
	or
	
	`ERROR:<errorcode>`
	
	`<errorcode>` (See 3. List of Status codes.)
	Prompt:
	
	`ActEpDesc:<NodeID>,<errorcode>[,XX,…]`
	
	`<NodeID>` is the Remote node's NodeID. In addition, all active endpoints are listed as 8-bit hexadecimal numbers separated by commas. In case of an error, an errorcode other than 00 will be displayed and the prompt will end after the errorcode
	
	After successful transmission, the sequence number of the unicast is stated using the "SEQ:XX" prompt. When acknowledged (or not) the accompanying "ACK:XX" (or "NACK:XX") prompt is displayed.

Table 2.41. +SIMPLEDESC – Request Endpoint's Simple Descriptor (ZDO)

Execute Command	Response

`AT+SIMPLEDESC:<Address>,<NodeID>,<XX>`
 	
Where `<Address>` is the EUI64, NodeID, or Address table entry of the node which is to be interrogated about the node with the NodeID specified in `<NodeID>`. XX is the number of the endpoint of which simple descriptor is to be read. XX should be a hexadecimal number.
 	
Sends a unicast to obtain the specified device's active endpoint list.
 	
Use on: All Devices
	`SEQ:XX`
	
	`OK`
	
	or
	
	`ERROR:<errorcode>`
	
	`<errorcode>` (See 3. List of Status codes.)
	Prompt:
	
	`SimpleDesc:<NodeID>,<errorcode>`
	
	`EP:XX`
	
	`ProfileID:XXXX`
	
	`DeviceID:XXXXvXX`
	
	`InCluster:<Cluster List>`
	
	`OutCluster:<Cluster List>`
	
	In case of an error, an errorcode other than 00 will be displayed and the prompt will end after the errorcode.
	
	`<NodeID>` is the Remote node's NodeID. In addition, all active endpoints are listed as 8-bit hexadecimal numbers separated by commas.
	
	After successful transmission, the sequence number of the unicast is stated using the "SEQ:XX" prompt. When acknowledged (or not) the accompanying "ACK:XX" (or "NACK:XX") prompt is displayed.

Table 2.42. +MATCHREQ – Find Nodes which Match a Specific Descriptor (ZDO)

Execute Command	Response

`AT+MATCHREQ:<ProfileID>,<NumInClusters> [,<InClusterList>],<NumOutClusters> [,<OutClusterList>]`
 	
Where `<ProfileID>` is the required profile ID of the device being searched for, followed by a specification of required input and output clusters. If a remote node has a matching ProfileID and matches at least one of the specified clusters, it will respond to the broadcast listing the matching endpoint(s).
 	
`<NumInClusters>` and `<NumOutClusters>` must be two hexadecimal digits.
 	
Example: `AT+MATCHREQ:0109,01,0000,02,0700,0701`
 	
Use on: All Devices
	`OK`
	
	or
	
	`ERROR:<errorcode>`
	
	`<errorcode>` (See 3. List of Status codes.)
	Prompt:
	
	`MatchDesc:<NodeID>,<errorcode>,XX,…`
	
	In case of an error, an errorcode other than 00 will be displayed and the prompt will end after the errorcode.
	
	Where `<NodeID>` is the Remote node's NodeID. In addition, all endpoints of this node matching the search criterion are listed as 8-bit hexadecimal numbers separated by commas.

Table 2.43. +ANNCE – Announce Local Device In The Network (ZDO)

Execute Command	Response

`AT+ANNCE`
 	
Send a ZigBee device announcement. Broadcast announcing the local node on the network.
 	
Use on: All Devices
Remote Action
	`OK`
	
	or
	
	`ERROR:<errorcode>`
	
	`<errorcode>` (See 3. List of Status codes.)
	Prompt:
	
	`FFD:<EUI64>,<NodeID>`
	
	The prompt above will be displayed on all nodes which can hear the announcement.

Table 2.44. +KEYUPD – Update the Network Key

Execute Command	Response

`AT+KEYUPD`
 	
Updates the Network Key with a new random key.
 	
Use on: Trust Centre
 	
This command can only be used on the Trust Centre.
	`OK`
	
	or
	
	`ERROR:<errorcode>`
	
	`<errorcode>` (See 3. List of Status codes.)
	
	Prompt:
	
	`NWKKEYUPDATED` (if the key is updated successfully)

Table 2.45. +CCHANGE – Change the network's channel

Execute Command	Response

`AT+CCHANGE[:channel]`
 	
Ask all nodes in the network to change their channel. Use decimal numbers 11–26 to specify the channel. If no channel is specified, a random channel masked in S00 is selected. The random channel chosen will also not have been previously blacklisted because of excessive packet loss (`NM:ES REPORT WARNING` prompt).
 	
Use on: Network Manager
 	
The New channel needs to be masked in S00 for all nodes on the network. Ideally, S00 should be identical for all nodes on a network.
	`OK`
	
	or
	
	`ERROR:<errorcode>`
	
	`<errorcode>` (See 3. List of Status codes.)
	
	Parameters:
	
	Optional channels range from 11 to 26.

Table 2.46. +CSWITCH – Switch the Network's Channel without Notifying Other Nodes

Execute Command	Response
AT+CSWITCH:<channel> <channel>: decimal number range from 11 to 26. This command can be used to switch channels without notifying all nodes on the network. A usage scenario for this would be when the CICIE wants to move to a channel to make a ZLL light join, but then bring the ZLL light back to its original channel. The user need to use AT+CSWITCH:<The channel the ZLL light is using>. After five seconds, do AT+N to check if the ZLL light has successfully moved to the target channel. Then apply AT+INTERPAN command. After the ZLL light joins, the CICIE should use AT+CCHANGE:<CICIE's original channel>. Use on: Network Manager	OK or ERROR:<errorcode> <errorcode> (See 3. List of Status codes.)

Messaging Commands

Table 2.47. +ATABLE – Display Address Table

Read Command	Response
AT+ATABLE Use on: All Devices The address table is volatile and the contents will be lost if the device is powered down.	No. | Active | ID | EUI 00 | N | 0000 |000D6F0000012345 (…) OK The Address Table contains nodes which can be addressed by referring to the corresponding address table entry. The "Active" column shows nodes to which a message is currently in flight.

Table 2.48. +ASET – Set Address Table Entry

Read Command	Response
AT+ASET:XX,<NodeID>,<EUI64> Where XX is the entry number of the address table entry which is to be written. If the NodeID is unknown, the NodeID must be substituted with "FFFF". Use on: All Devices	OK or ERROR:<errorcode> <errorcode> (See 3. List of Status codes.)

Table 2.49. +LBTABLE – Display Local Binding Table

Read Command	Response
AT+LBTABLE Use on: All Devices The binding table is cleared by a reset. When using this command, the local display binding index starts from 10.	No. | Type | Active | LocalEP | ClusterID | Addr | Rem EP 10. | Ucast | No | 01 | DEAD | 1234567887654321 | 01 11. | MTO | No | 01 | DEAD | E012345678876543 | 88 12. | Mcast | No | 01 | DEAD | CDAB 13. | Unused 14. | Unused 15. | Unused 16. | Unused 17. | Unused 18. | Unused 19. | Unused

Table 2.50. +BSET – Set local Binding Table Entry

Read Command	Response
AT+BSET:<type>,<LocalEP>,<ClusterID>, <DstAddress>[,<DstEP>] Where: <Type> is the type of binding: 1 = Unicast Binding with EUI64, and remote EP specified 2 = Many-to-one Binding with EUI64, and remote EP Specified 3 = Multicast Binding with Multicast ID Specified <LocalEP> is the local endpoint. <ClusterID> is the cluster ID, and the address is either the EUI64 of the target device, or a multicast ID. <DstEP> is the remote endpoint, which is not specified in case of a multicast binding. The new binding is created in the next available free binding table entry. Use on: All Devices	OK or ERROR:<errorcode> <errorcode> (See 3. List of Status codes.)

Table 2.51. +BCLR – Clear local Binding Table Entry

Read Command	Response
AT+BCLR:XX Where XX is the entry number of the binding table to be cleared. Use on: All Devices	OK or ERROR:<errorcode> <errorcode> (See 3. List of Status codes.)

Table 2.52. +BTABLE – Display Binding Table (ZDO)

Read Command	Response

`AT+BTABLE:XX,<address>`
 	
Where XX is the start index of the remote binding table and <address> can be the remote node's EUI64, Node ID or address/binding table entry.
 	
Use on: All devices
 	
The local node can also be the target of this command (e.g. use address table entry FF as the address).
	`SEQ:XX`
	
	`OK`
	
	or
	
	`ERROR:<errorcode>`
	
	`<errorcode>` (See 3. List of Status codes.)
	
	This command requests the target node to respond by listing its binding table starting from the requested index.
	
	The response indicates success or failure in sending this message. The acknowledgement as well as the actual response to this request will follow as asynchronous prompts.
	Example:
	
	`AT+BTABLE:00,0000`
	
	`SEQ:01`
	
	`OK`
	
	`BTable:0000,00`
	
	`Length:03`
	
	`No. | SrcAddr | SrcEP | ClusterID | DstAddr | DstEP`
	
	`00. | 000D6F000059474E | 01 | DEAD |1234567887654321 | 12`
	
	`01. | 000D6F000059474E | 01 | DEAD |E012345678876543 | E0`
	
	`02. | 000D6F000059474E | 01 | DEAD | ABCD`
	
	`ACK:01`
	
	In this example, the neighbour table of the remote node with the short ID shown in `<NodeID>` contains three entries (hexadecimal), which are displayed. In case the table contains more than three entries, it may be required to repeat this command and increase the index count until the full table is derived.
	
	In case of an error, an errorcode other than 00 will be displayed and the prompt will end after the errorcode.

Table 2.53. +BIND – Create Binding on Remote Device (ZDO)

Read Command	Response

`AT+BIND:<address>,<type>,<SrcAddress>,<SrcEP>,<ClusterID>,<DstAddress> [,<DstEP>]`
 	
Create binding on a remote device with:
 	
`<address>` the target Node's EUI64, NodeID, or Address Table entry.
 	
`<type>` the Addressing mode shown as below.
 	
`<SrcAddress>` the EUI64 of the source.
 	
`<SrcEP>` the source endpoint.
 	
`<ClusterID>` The Cluster ID on the source device.
 	
`<DstAddress>` The EUI64 or 16-bit multicast ID, depending on `<type>`.
 	
`<DstEP>` Only in Mode 2: The destination endpoint.
 	
Type:
 	
1 = Multicast Binding with Multicast ID, specified in `<DstAddress>`.
 	
3 = Unicast Binding with destination EUI64 in `<DstAddress>` and destination EP in `<DstEP>`.
 	
The local node can also be the target of this command (e.g. use address table entry FF as the address).
 	
Use on: All devices
	`SEQ:XX`
	
	`OK`
	
	or
	
	`ERROR:<errorcode>`
	
	`<errorcode>` (See 3. List of Status codes.)
	
	The response indicates success or failure in sending this message. The acknowledgement, as well as the actual response to this request, will follow as asynchronous prompts.
	Prompt:
	
	`Bind:<NodeID>,<status>`
	
	In case of an error, a status other than 00 will be displayed
	
	`<NodeID>` is the Remote node's Node ID.
	
	As with all unicasts after successful transmission, the sequence number of the unicast is stated using the "SEQ:XX" prompt. When acknowledged (or not) the accompanying "ACK:XX" (or "NACK:XX") prompt is displayed.
	Example:
	
	`AT+BIND:0000,3,000d6f000059474e,01,abcd,000D6F0000123456,01`
	
	`SEQ:01`
	
	`OK`
	
	`Bind:0000,00`
	
	`ACK:01`

Table 2.54. +UNBIND – Delete Binding on Remote Device (ZDO)

Read Command	Response
AT+UNBIND:<address>,<type>,<SrcAddress>, <SrcEP>,<ClusterID>,<DstAddress>[,<DstEP>] Delete Binding on a remote device with: <address> the target Node's EUI64, Node ID, or Address Table entry. <type> the addressing mode, as shown below. <SrcAddress> the EUI64 of the source. <SrcEP> the source endpoint. <ClusterID> The Cluster ID on the source device. <DstAddress> The EUI64 or 16-bit multicast ID, depending on <type>. <DstEP> Only in Mode 2: The destination endpoint. Types: 1 = Multicast Binding with Multicast ID, specified in <DstAddress>. 3 = Unicast Binding with destination EUI64 in <DstAddress> and destination EP in <DstEP>. Use on: All devices The local node can also be the target of this command (e.g. use address table entry FF as the address).	SEQ:XX OK or ERROR:<errorcode> <errorcode> (See 3. List of Status codes.) The response indicates success or failure in sending this message. The acknowledgement, as well as the actual response to this request, will follow as asynchronous prompts. Prompt: Unbind:<NodeID>,<status> In case of an error, a status other than 00 will be displayed. <NodeID> is the Remote node's NodeID. As with all unicasts after successful transmission, the sequence number of the unicast is stated using the "SEQ:XX" prompt. When acknowledged (or not) the accompanying "ACK:XX" (or "NACK:XX") prompt is displayed. Example: AT+UNBIND:0000,3,000d6f000059474e,01,abcd,000D6F0000123456,01 SEQ:01 OK Unbind:0000,00 ACK:01

Table 2.55. +EBIND – End Device Bind

Execute Command	Response
AT+EBIND:<EP> <EP> — Local endpoint, which will initiate end device binding. This command is only used on the local node.	EBINDACK (meaning the request is acknowledged by coo). EBINDRSP: <Status> OK If status is not 00, it is an error code (see 3. List of Status codes).

ZigBee Discovery Commands

Table 2.56. +DISCOVER – Discover HA Devices On The HAN

Execute Command	Response

`AT+DISCOVER:<Cluster ID>`
 	
`<Cluster ID>` — 16-bit hexadecimal cluster ID (see 1.2.2 Combined Interface). The Combined Interface can search for HA devices based on a specified cluster ID.
 	
Response will show up all HA devices that match. Discovery will only search for server cluster.
	`OK`
	
	or
	
	`ERROR:<errorcode>`
	
	`<errorcode>` (See 3. List of Status codes.)
	
	Prompt:
	
	`DEV:<NodeID>,<EndPoint>`
	
	Carry out the ZigBee Service Discovery to find HA devices that support the given match criteria.
	
	`<NodeID>` — 16-bit hexadecimal number which represents the network address of a found device.
	
	`<EndPoint>` — 8-bit hexadecimal number representing the endpoint on a found remote device supporting the given match criteria.

Table 2.57. +DISCOVERP – Discover Devices Using Profile ID and Cluster ID

Execute Command	Response

`AT+DISCOVERP:[<ProfileID>],<ClusterID> [,<Option>]`
 	
`<ProfileID>` — 16-bit hexadecimal number which represents the profile ID, by default the profile ID is 0x0104 (HA profile).
 	
`<Cluster ID>` — 16-bit hexadecimal cluster ID (see 1.2.2 Combined Interface). The Combined Interface can search for HA devices based on a specified cluster ID.
 	
`<Option>` — A Boolean type ('0' means discover server cluster; '1' means discover client cluster).
	`OK`
	
	or
	
	`ERROR:<errorcode>`
	
	`<errorcode>` (See 3. List of Status codes.)
	
	Prompt:
	
	`DEV:<NodeID>,<EndPoint>`
	
	Carry out the ZigBee Service Discovery to find ZigBee HA devices that support the given match criteria.
	
	`<NodeID>` — 16-bit hexadecimal number which represents the network address of a found device.
	
	`<EndPoint>` — 8-bit hexadecimal number representing endpoint on a found remote device supporting the given match criteria.

Table 2.58. +CLUSDISC – Find All Supported Clusters On A Remote Device endpoint

Execute Command	Response

`AT+CLUSDISC:<Node ID>,<EndPoint>`
 	
`<Node ID>` — 16-bit hexadecimal number. The network address of the target device.
 	
`<EndPoint>` — 8-bit hexadecimal number, which is the endpoint of the target device.
 	
Response show up on all supported clusters on a specified device's endpoint.
	`OK`
	
	`DISCCLUS:<Node ID>,<Status>,<EndPoint>`
	
	or
	
	`ERROR:<errorcode>`
	
	`<errorcode>` (See 3. List of Status codes.)
	
	Prompt (will show if `<Status>` is 00):
	
	`SERVER:<ClusterID>,<ClusterID>,<ClusterID>`
	
	`CLIENT:<ClusterID>,<ClusterID>,< ClusterID>`

Table 2.59. +ATTRDISC — Find Supported Attributes On A Remote Device's Endpoint

Execute Command	Response

`AT+ATTRDISC:<NodeID>,<EP>,<ClusterID>,<AttrID>,<MaxNumofAttr>`
 	
`<NodeID>` — 16-bit hexadecimal number representing the target device's network address.
 	
`<EP>` — 8-bit hexadecimal number representing the target device's endpoint.
 	
`<ClusterID>` — 16-bit hexadecimal number.
 	
`<MaxNumofAttr>` — decimal number representing the number of attributes that require discovery, e.g: 01, 10.
 	
`<MaxNumofAttr>` — decimal number representing the number of attributes required to be discovered, e.g: 01, 10
	`OK`
	
	`DISCATTR:<NodeID>,<EP>,<Complete code>`
	
	`CLUS:<ClusterID>,ATTR:<AttrID>,TYPE:<DataType>`
	
	or
	
	`ERROR:<errorcode>`
	
	`<errorcode>` (See 3. List of Status codes.)
	
	`<Complete code>` — appears if all attributes supported by this cluster have been discovered.
	
	00: Completed discovery
	
	01: Incomplete discovery
	
	If the CICIE receives attribute discovery response, a prompt "ENDDISCATTR" will show at the end of the response.

Table 2.60. +MATTRDISC — Find Supported Manufacturer Defined Attributes On A Remote Device

Execute Command	Response
AT+MATTRDISC:<NodeID>,<EP>,<Mcode>,<ClusterID>,<AttrID>,<MaxNumofAttr> [,<ProfileID>] <NodeID> — 16-bit hexadecimal number representing the target device's network address. <EP> — 8-bit hexadecimal number representing the target device's endpoint. <Mcode> — 16-bit hexadecimal number, which is the manufacturer's code. <ClusterID> — 16-bit hexadecimal number. <AttrID> — 16-bit hexadecimal number, and the attribute discover will start from. <MaxNumofAttr> — decimal number representing the number of attributes that require discovery, e.g: 01, 10. <ProfileID> 16-bit hexadecimal number, representing profile ID. If the manufacturer-specified attributes are defined under a new profile ID, the user needs to specify this parameter. If this parameter is omitted, the Home Automation Profile ID (0x0104) will be used.	OK DISCMATTR:<NodeID>,<EP>,<Mcode>, <Complete code> CLUS:<ClusterID>,ATTR:<AttrID>,TYPE:<DataType> or ERROR:<errorcode> <errorcode> (See 3. List of Status codes.) <Complete code> — appears if all attributes supported by this cluster have been discovered. 00: Completed discovery 01: Incomplete discovery If the CICIE receives an attribute discovery response, a prompt "ENDDISCMATTR" will show at the end of the response.

ZCL Attribute Commands

Table 2.61. +READATR – Read Server Attribute

Execute Command	Response
AT+READATR:<Address>,<EP>,<SendMode>,<ClusterID>,<AttrID>,…<AttrID> AT+READATR:,,,<Cluster>,<AttrID>,… <AttrID> <Address> — 16-bit hexadecimal number, which is the Node ID of a remote device if the command is sent directly to a node, or a group ID if the command is sent to a group. <EP> — 8-bit hexadecimal number, which is the endpoint of a remote device. Valid endpoint addresses are: 0x01 to 0xF0. <SendMode> — A Boolean type to choose transmission mode ('0' sends the command directly; '1' sends the command to a group). <Cluster ID> — 16-bit hexadecimal cluster ID (see 1.2.2 Combined Interface). The Combined Interface can search for HA devices based on a specified cluster ID. <AttrID> — 16-bit hexadecimal number, which represents the attribute ID according to ZigBee Home Automation specification. This command can be used to read multiple attributes (up to 16 in a cluster). If the address is a Node ID, set SendMode to 0, or the address will be recognized as a group ID. If destination address, endpoint, and sendMode are not given (“,” cannot be omitted) the command will search the binding table for a destination. If this is the user's preferred method, then it is necessary to add the required destination to the local binding table in advance. After that, it will not be necessary to fill in the <address> parameter.	OK RESPATTR:<NodeID>,<EP>,<ClusterID>, <AttrID>,<Status>,<AttrInfo> or ERROR:<errorcode> <errorcode> (See 3. List of Status codes.) <NodeID> — 16-bit hexadecimal number, which is the source Node ID of response. <EP> — 8-bit hexadecimal number, which is the source endpoint of the response. <Cluster ID> — 16-bit hexadecimal cluster ID (see 1.2.2 Combined Interface). The Combined Interface can search for HA devices based on a specified cluster ID. <AttrID> — Attribute ID of a 16-bit hexadecimal number <Status> — 8-bit hexadecimal number, which indicates the result of the requested operation. <AttrInfo> — hexadecimal number of a char string (size depends on the attribute requested). <AttrInfo> will only be valid if <Status> = 0x00. If <Status> indicates error, <AttrInfo> is not returned. <errorcode> (See 3. List of Status codes.) Use this command to get the requested attribute from a cluster server on a remote HA device which supports this server cluster. If the attribute is in Temperature measurement cluster or Illuminance measurement cluster, "ILLUMINANCE" and "TEMPERATURE" prompts will show, respectively, and take the place of "RESPATTR". In the response, cluster ID is omitted. For example: ILLUMINANCE:38C8,03,0000,00,07AD

Table 2.62. +READCATR – Read Client Attribute

Execute Command	Response
AT+READCATR:<Address>,<EP>,<SendMode>,<ClusterID>,<AttrID>,…< AttrID > AT+READCATR:,,,<Cluster>,<AttrID>,…<AttrID> <Address> — 16-bit hexadecimal number, which is the Node ID of a remote device if the command is sent directly to a node, or a group ID if the command is sent to a group. <EP> — 8-bit hexadecimal number, endpoint of a remote device. Valid endpoint addresses are: 0x01 to 0xF0. <SendMode> — A Boolean type to choose transmission mode ('0' sends the command directly; '1' sends the command to a group). <Cluster ID> — 16-bit hexadecimal cluster ID (see 1.2.2 Combined Interface). <AttrID> — 16-bit hexadecimal number, which represents attribute ID according to ZigBee Home Automation specification. This command can be used to read multiple attributes (up to 16 in a cluster). If the address is a Node ID, set SendMode to 0, or the address will be recognized as a group ID. If destination address, endpoint, and SendMode are not given (“,” cannot be omitted), the command will search the binding table for a destination. If this is the user's preferred method, then it is necessary to add the required destination to the local binding table in advance. After that, it will not be necessary to fill in the <address> parameter.	OK RESPATTR:<NodeID>,<EP>,<ClusterID>, <AttrID>,<Status>,<AttrInfo> or ERROR:<errorcode> <errorcode> (See 3. List of Status codes.) <NodeID> — 16-bit hexadecimal number, which is the source Node ID of a response. <EP> — 8-bit hexadecimal number, the source endpoint of the response. <Cluster ID> — 16-bit hexadecimal cluster ID (see 1.2.2 Combined Interface). <AttrID> — attribute ID 16-bit hexadecimal number <Status> — 8-bit hexadecimal number, which indicates the result of the requested operation. <AttrInfo> — hexadecimal number of a char string (size depends on the attribute requested). <AttrInfo> will only be valid if <Status> = 0x00. If <Status> indicates error, <AttrInfo> is not returned. <errorcode> (See 3. List of Status codes.) Use this command to get the requested attribute from a cluster server on a remote HA device supporting this server cluster.

Table 2.63. +READMATR – Read Manufacturer Specific Server Attribute

Execute Command	Response

`AT+READMATR:<Address>,<EP>,<SendMode>,<Mcode>,<ClusterID>,<AttrID>,…< AttrID >`
 	
`AT+READMATR:,,,<Mcode>,<ClusterID>, <AttrID>,… <AttrID >`
 	
`<Address>` — 16-bit hexadecimal number, which is the Node ID of a remote device if the command is sent directly to a node, or a group ID if the command is sent to a group.
 	
`<EP>` — 8-bit hexadecimal number, endpoint of a remote device. Valid endpoint addresses are: 0x01 to 0xF0.
 	
`<SendMode>` — A Boolean type to choose transmission mode ('0' sends the command directly; '1' sends the command to a group).
 	
`<Mcode>` — 16-bit hexadecimal manufacturer code.
 	
`<ClusterID>` — 16-bit hexadecimal number, which represents cluster ID.
 	
`<AttrID>` — 16-bit hexadecimal number, which represents attribute ID according to manufacture definition.
 	
This command can be used to read multiple attributes (up to 16 in a cluster).
 	
If the address is a Node ID, set SendMode to 0, or the address will be recognized as a group ID.
 	
If destination address, endpoint, and SendMode are not given (“,” cannot be omitted), the command will search the binding table for a destination. If this is the user's preferred method, then it is necessary to add the required destination to the local binding table in advance. After that, it will not be necessary to fill in the `<address>` parameter.
	`OK`
	
	`RESPMATTR:<NodeID>,<EP>,<Mcode>, <ClusterID>,<AttrID>, <Status>, <AttrInfo>`
	
	or
	
	`ERROR:<errorcode>`
	
	`<errorcode>` (See 3. List of Status codes.)
	
	`<NodeID>` — 16-bit hexadecimal number, which is the source Node ID of a response.
	
	`<EP>` — 8-bit hexadecimal number, which is the source endpoint of the response.
	
	`<Mcode>` — 16-bit hexadecimal manufacture code.
	
	`<ClusterID>` — 16-bit hexadecimal cluster ID.
	
	`<AttrID>` — Attribute ID 16-bit hexadecimal number
	
	`<Status>` — 8-bit hexadecimal number indicating the result of the requested operation.
	
	`<AttrInfo>` — hexadecimal number of a char string (size depends on the attribute requested). `<AttrInfo>` will only be valid if `<Status>` = 0x00. If `<Status>` indicates an error, `<AttrInfo>` is not returned.
	
	`<errorcode>` (See 3. List of Status codes.)
	
	Use this command to get the requested attribute from a cluster server on a remote HA device supporting this server cluster.

Table 2.64. +READMCATR – Read Manufacturer Specific Client Attribute

Execute Command	Response

`AT+READMCATR:<Address>,<EP>, <SendMode>,<Mcode>,<ClusterID>,<AttrID>, …< AttrID >`
 	
`AT+READMCATR:,,,<Mcode>,<ClusterID>, <AttrID>,… <AttrID>`
 	
`<Address>` — 16-bit hexadecimal number, which is the Node ID of a remote device if the command is sent directly to a node, or a group ID if the command is sent to a group.
 	
`<EP>` — 8-bit hexadecimal number, endpoint of a remote device. Valid endpoint addresses are: 0x01 to 0xF0.
 	
`<SendMode>` — A Boolean type to choose transmission mode ('0' sends the command directly; '1' sends the command to a group).
 	
`<Mcode>` — 16-bit hexadecimal manufacturer code.
 	
`<ClusterID>` — 16-bit hexadecimal number representing cluster ID.
 	
`<AttrID>` — 16-bit hexadecimal number representing attribute ID according to manufacture definition.
 	
This command can be used to read multiple attributes (up to 16 in a cluster).
 	
If the address is a Node ID, set SendMode to 0, or the address will be recognized as a group ID.
 	
If destination address, endpoint, and sendMode are not given (“,” cannot be omitted), the command will search the binding table for a destination. If this is the user's preferred method, then it is necessary to add the required destination to the local binding table in advance. After that, it will not be necessary to fill in the `<address>` parameter.
	`OK`
	
	`RESPMATTR:<NodeID>,<EP>,<Mcode>, <ClusterID>,<AttrID>, <Status>,<AttrInfo>`
	
	or
	
	`ERROR:<errorcode>`
	
	`<errorcode>` (See 3. List of Status codes.)
	
	`<NodeID>` — 16-bit hexadecimal number, which is the source Node ID of a response.
	
	`<EP>` — 8-bit hexadecimal number, the source endpoint of a response.
	
	`<Mcode>` — 16-bit hexadecimal manufacture code.
	
	`<ClusterID>` — 16-bit hexadecimal cluster ID.
	
	`<AttrID>` — Attribute ID as a 16-bit hexadecimal number.
	
	`<Status>` — 8-bit hexadecimal number indicating the result of the requested operation.
	
	`<AttrInfo>` — hexadecimal number of a char string (size depends on the attribute requested). `<AttrInfo>` will only be valid if `<Status>` = 0x00. If `<Status>` indicates an error, `<AttrInfo>` is not returned.
	
	`<errorcode>` (See 3. List of Status codes.)
	
	Use this command to get the requested attribute from a cluster server on a remote HA device supporting this server cluster.

Table 2.65. AT+WRITEATR – Write Server Attribute

Execute Command	Response

`AT+WRITEATR:<Address>,<EP>,<SendMode>, <ClusterID>,<AttrID>,<DataType>,<Data>`
 	
`AT+WRITEATR:,,,<ClusterID>,<AttrID>, <DataType>,<AttrValue>`
 	
`<Address>` — 16-bit hexadecimal number, which is the Node ID of a remote device if the command is sent directly to a node, or a group ID if the command is sent to a group.
 	
`<EP>` — 8-bit hexadecimal number, which is the endpoint of a remote device. Valid endpoint addresses are: 0x01 to 0xF0.
 	
`<SendMode>` — A Boolean type to choose transmission mode ('0' sends the command directly; '1' sends the command to a group).
 	
`<Cluster ID>` — 16-bit hexadecimal cluster ID (see 1.2.2 Combined Interface).
 	
`<AttrID>` — 16-bit hexadecimal number representing the attribute ID according to ZigBee Home Automation specification.
 	
`<DataType>` — 8-bit hexadecimal number representing the type of data accepted by this attribute (check the HA specification).
 	
`<AttrValue>` — If the attribute value has an integer type, this field will contain a hexadecimal representation in big-endian format. If the attribute value has a string type, this field will contain a sequence of characters.
 	
If the address is a Node ID, set SendMode to 0, or the address will be recognized as a group ID.
 	
If destination address, endpoint, and SendMode are not given (“,” cannot be omitted), the command will search the binding table for a destination. If this is the user's preferred method, then it is necessary to add the required destination to the local binding tables in advance. After that, it will not be necessary to fill in the `<address>` parameter.
	`OK`
	
	`WRITEATTR:<NodeID>,<EP>,<ClusterID>, <AttrID>,<Status>`
	
	or
	
	`ERROR:<errorcode>`
	
	`<errorcode>` (See 3. List of Status codes.)
	
	`<NodeID>` — 16-bit hexadecimal number, which is the source Node ID of a response.
	
	`<EP>` — 8-bit hexadecimal number, the source endpoint of the response.
	
	`<Cluster ID>` — 16-bit hexadecimal cluster ID (see 1.2.2 Combined Interface).
	
	`<AttrID>` — Attribute ID as a 16-bit hexadecimal number
	
	`<Status>` — 8-bit hexadecimal number indicating the result of the requested operation.
	
	If `<Status >` is not 00, it will be an errorcode listed in 3. List of Status codes.
	
	If the user uses this command to write multiple attributes, and the attributes are written successfully, the response will only contain status code. For example:
	
	`WRITEATTR:1234,01,1010,00`
	
	This is expected behavior according to ZigBee cluster library specification.
	
	Use this command to set the value to a writeable attribute on a remote HA device supporting this cluster.

Table 2.66. AT+WRITECATR – Write Client Attribute

Execute Command	Response

`AT+WRITECATR:<Address>,<EP>,<SendMode>,<ClusterID>,<AttrID>,<DataType>,<Data>`
 	
`AT+WRITECATR:,,,<ClusterID>,<AttrID>, <DataType>,<AttrValue>`
 	
`<Address>` — 16-bit hexadecimal number, which is the Node ID of a remote device if the command is sent directly to a node, or a group ID if the command is sent to a group.
 	
`<EP>` — 8-bit hexadecimal number, which is the endpoint of a remote device. Valid endpoint addresses are: 0x01 to 0xF0.
 	
`<SendMode>` — A Boolean type to choose transmission mode ('0' sends the command directly; '1' sends the command to a group).
 	
`<Cluster ID>` — 16-bit hexadecimal cluster ID (see 1.2.2 Combined Interface).
 	
`<AttrID>` — 16-bit hexadecimal number, which represents the attribute ID according to ZigBee Home Automation specification.
 	
`<DataType>` — 8-bit hexadecimal number representing the type of data accepted by this attribute (check the HA specification).
 	
`<AttrValue>` — If the attribute value has an integer type, this field will contain a hexadecimal representation in big-endian format. If the attribute value has a string type, this field will contain a sequence of characters.
 	
If the address is a Node ID, set SendMode to 0, or the address will be recognized as a group ID.
 	
If destination address, endpoint, and SendMode are not given (“,” cannot be omitted), the command will search the binding table for a destination. If this is the user's preferred method, then it is necessary to add the required destination to the local binding tables in advance. After that, it will not be necessary to fill in the `<address>` parameter.
	`OK`
	
	`WRITEATTR:<NodeID>,<EP>, <ClusterID>, <AttrID>,<Status>`
	
	or
	
	`ERROR:<errorcode>`
	
	`<errorcode>` (See 3. List of Status codes.)
	
	`<NodeID>` — 16-bit hexadecimal number, which is the source Node ID of a response.
	
	`<EP>` — 8-bit hexadecimal number, which is the source endpoint of a response.
	
	`<Cluster ID>` — 16-bit hexadecimal cluster ID (see 1.2.2 Combined Interface).
	
	`<AttrID>` — Attribute ID as a 16-bit hexadecimal number.
	
	`<Status>` — 8-bit hexadecimal number indicating the result of the requested operation.
	
	If `<Status >` is not 00, it is an errorcode listed in 3. List of Status codes.
	
	If the user uses this command to write multiple attributes, and the attributes are written successfully, then the response will only contain status code. For example:
	
	`WRITEATTR:1234,01,1010,00`
	
	This is expected behavior according to the ZigBee cluster library specification.
	
	Use this command to set value to a writeable attribute on a remote HA device supporting this cluster.

Table 2.67. AT+WRITEMATR – Write Manufacturer Specific Server Attribute

Execute Command	Response

`AT+WRITEMATR:<Address>,<EP>, <SendMode>,<Mcode>,<ClusterID>,<AttrID>, <DataType>,<Data>`
 	
`AT+WRITEMATR:,,,<Mcode>,<ClusterID>, <AttrID>, <DataType>,<AttrValue>`
 	
`<Address>` — 16-bit hexadecimal number, which is the Node ID of a remote device if the command is sent directly to a node, or a group ID if the command is sent to a group.
 	
`<EP>` — 8-bit hexadecimal number, endpoint of a remote device. Valid endpoint addresses are: 0x01 to 0xF0.
 	
`<SendMode>` — A Boolean type to choose transmission mode ('0' sends the command directly; '1' sends the command to a group).
 	
`<Mcode>` — 16-bit hexadecimal manufacturer code.
 	
`<ClusterID>` — 16-bit hexadecimal number representing the cluster ID.
 	
`<AttrID>` — 16-bit hexadecimal number representing the attribute ID according to the manufacturer's definition.
 	
`<DataType>` — 8-bit hexadecimal number that represents the type of the data accepted by this Attribute (Depends on the manufacturer's definition.)
 	
`<AttrValue>` — If the attribute value has an integer type, this field will contain a hexadecimal representation in big-endian format. If the attribute value has a string type, this field will contain a sequence of characters.
 	
If the address is a Node ID, set SendMode to 0, or the address will be recognized as a group ID.
 	
If destination address, endpoint, and SendMode are not given (“,” cannot be omitted), the command will search the binding table for a destination. If this is the user's preferred method, then it is necessary to add the required destination to the local binding tables in advance. After that, it will not be necessary to fill in the `<address>` parameter.
	`OK`
	
	`WRITEMATTR:<NodeID>,<EP>, <Mcode>,<ClusterID>,<AttrID>, <Status>`
	
	or
	
	`ERROR:<errorcode>`
	
	`<errorcode>` (See 3. List of Status codes.)
	
	`<NodeID>` — 16-bit hexadecimal number, which is the source Node ID of the response.
	
	`<EP>` — 8-bit hexadecimal number, which is the source endpoint of the response.
	
	`<Mcode>` — 16-bit hexadecimal manufacturer code.
	
	`<ClusterID>` — Cluster ID as a 16-bit hexadecimal number.
	
	`<AttrID>` — Attribute ID as a 16-bit hexadecimal number.
	
	`<Status>` — 8-bit hexadecimal number indicating the result of the requested operation.
	
	If `<Status>` is not 00, it will be an errorcode listed in 3. List of Status codes.
	
	If the user uses this command to write multiple attributes and the attributes are written successfully, then the response will only contain status code. For example:
	
	`WRITEMATTR:1234,01,1010,,00`
	
	This is expected behavior according to ZigBee cluster library specification.
	
	Use this command to write attribute values to a writeable attribute on a remote HA device supporting this cluster.

Table 2.68. AT+WRITEMCATR – Write Manufacturer Specific Client Attribute

Execute Command	Response

`AT+WRITEMCATR:<Address>,<EP>, <SendMode>,<Mcode>,<ClusterID>,<AttrID>, <DataType>,<Data>`
 	
`AT+WRITEMCATR:,,,<Mcode>,<ClusterID>, <AttrID>, <DataType>,<AttrValue>`
 	
`<Address>` — 16-bit hexadecimal number, which is the Node ID of a remote device if the command is sent directly to a node, or a group ID if the command is sent to a group.
 	
`<EP>` — 8-bit hexadecimal number, which is the endpoint of a remote device. Valid endpoint addresses are: 0x01 to 0xF0.
 	
`<SendMode>` — A Boolean type to choose transmission mode ('0' sends the command directly; '1' sends the command to a group).
 	
`<Mcode>` — 16-bit hexadecimal manufacturer code.
 	
`<ClusterID>` — 16-bit hexadecimal number representing the cluster ID.
 	
`<AttrID>` — 16-bit hexadecimal number representing the attribute ID according manufacturer definition.
 	
`<DataType>` — 8-bit hexadecimal number representing the type of data accepted by this attribute. (Depends on the manufacturer's definition.)
 	
`<AttrValue>` — If the attribute value has an integer type, this field will contain a hexadecimal representation in big-endian format. If the attribute value has a string type, this field will contain a sequence of characters.
 	
If the address is a Node ID, set SendMode to 0, or the address will be recognized as a group ID.
 	
If destination address, endpoint, and SendMode are not given (“,” cannot be omitted), the command will search the binding table for a destination. If this is the user's preferred method, then it is necessary to add the required destination to the local binding tables in advance. After that, it will not be necessary to fill in the `<address>` parameter.
	`OK`
	
	`WRITEMATTR:<NodeID>,<EP>,<Mcode>, <ClusterID>,<AttrID>,<Status>`
	
	or
	
	`ERROR:<errorcode>`
	
	`<errorcode>` (See 3. List of Status codes.)
	
	`<NodeID>` — 16-bit hexadecimal number, which is the source Node ID of the response.
	
	`<EP>` — 8-bit hexadecimal number, which is the source endpoint of the response.
	
	`<Mcode>` — 16-bit hexadecimal manufacture code.
	
	`<ClusterID>` — Cluster ID as a 16-bit hexadecimal number.
	
	`<AttrID>` — Attribute ID as a 16-bit hexadecimal number.
	
	`<Status>` — 8-bit hexadecimal number indicating the result of the requested operation.
	
	If `<Status>` is not 00, it will be an errorcode listed in 3. List of Status codes.
	
	If the user uses this command to write multiple attributes and the attributes are written successfully, then the response will only contain status code. For example:
	
	`WRITEMATTR:1234,01,1010,,00`
	
	This is expected behavior according to the ZigBee cluster library specification.
	
	Use this command to write attribute values to a writeable attribute on a remote HA device supporting this cluster.

Table 2.69. +READRPTCFG – Read Reporting Configuration From Remote Node

Execute Command	Response

`AT+READRPTCFG:<Address>,<EP>, <SendMode>,<ClusterID>,<Direction1>, <AttrID1>,…,<Direction16>,<AttrID16>`
 	
`AT+READRPTCFG:,,,<ClusterID>, <Direction1>,<AttrID1>,…,<Direction16>, <AttrID16>`
 	
`<Address>` — 16-bit hexadecimal number, which is the Node ID of a remote device if the command is sent directly to a node, or a group ID if the command is sent to a group.
 	
`<EP>` — 8-bit hexadecimal number, which is the endpoint of a remote device. Valid endpoint addresses are: 0x01 to 0xF0.
 	
`<SendMode>` — A Boolean type to choose transmission mode ('0' sends the command directly; '1' sends the command to a group).
 	
`<Cluster ID>` — 16-bit hexadecimal cluster ID (see 1.2.2 Combined Interface).
 	
`<Direction>` — A Boolean type that specifies whether values of the attribute are reported (0), or whether reports of the attribute are received (1).
 	
`<AttrID>` — 16-bit hexadecimal number representing the attribute ID according to ZigBee Home Automation specification.
 	
This command can be used to read multiple attributes (up to 16 in a cluster).
 	
If the address is a Node ID, set SendMode to 0, or the address will be recognized as a group ID.
 	
If destination address, endpoint, and SendMode are not given (“,” cannot be omitted), the command will search the binding table for a destination. If this is the user's preferred method, then it is necessary to add the required destination to the local binding tables in advance. After that, it will not be necessary to fill in the `<address>` parameter.
	`OK`
	
	`READRPTCFGRSP:<NodeID>,<EP>, <ClusterID>,<Status>,<Direction>,<AttrID>, [<DataType>,<MinimumReportingInterval>, <MaximumReportingInterval>, <ReportableChange>][<Timeout>]`
	
	or
	
	`ERROR:<errorcode>`
	
	`<errorcode>` (See 3. List of Status codes.)
	
	`<NodeID>` — 16-bit hexadecimal number, which is the source Node ID of the response.
	
	`<EP>` — 8-bit hexadecimal number, the source endpoint of the response.
	
	`<Cluster ID>` — 16-bit hexadecimal cluster ID (see 1.2.2 Combined Interface).
	
	`<Status>` — 8-bit hexadecimal number indicating the result of the requested operation.
	
	`<Direction>` — 8-bit hexadecimal number specifies whether values of the attribute are reported (0x00), or whether reports of the attribute are received (0x01).
	
	`<AttrID>` — Attribute ID as a 16-bit hexadecimal number.
	
	If the received `<Direction>` field is 0x00, the following information is expected:
	
	`<DataType>` — 8-bit hexadecimal number representing the attribute data type.
	
	`<MinimumReportingInterval>` — The minimum reporting interval field is 16 bits in length and will contain the minimum interval, in seconds, between issuing reports for the attribute specified in the attribute Identifier field. If the minimum reporting interval has not been configured, this field will contain the value 0xffff.
	
	`<MaximumReportingInterval>` — The maximum reporting interval field is 16 bits in length and will contain the maximum interval, in seconds, between issuing reports for the attribute specified in the attribute Identifier field. If the maximum reporting interval has not been configured, this field will contain the value 0xffff.
	
	`<ReportableChange>` — The reportable change field will contain the minimum change to the attribute that will result in a report being issued. For attributes with 'analog' data type (see ZigBee Cluster Library 075123r04) the field has the same data type as the attribute. If the reportable change has not been configured, this field will contain the invalid value for the relevant data type.
	
	For attributes of 'discrete' data types (see ZigBee Cluster Library) this field is omitted.

Execute Command	Response
	If received, the <Direction> field is 0x01, and the following information is expected: <Timeout> — The timeout period field is 16 bits in length and will contain the maximum expected time, in seconds, between received reports for the attribute specified in the attribute identifier field. If the timeout period has not been configured, this field will contain the value 0xffff. <errorcode> (See 3. List of Status codes.)

Table 2.70. +READMRPTCFG – Read Manufacture Reporting Configuration From Remote Node

Execute Command	Response
AT+READMRPTCFG:<Address>,<EP>, <SendMode>,<Mcode>,<ClusterID>, <Direction1>,<AttrID1>,…,<Direction16>, <AttrID16> AT+READMRPTCFG:,,,<Mcode>,<ClusterID>,<Direction1>,<AttrID1>,…,<DirectionX>, <AttrIDX > <Address> — 16-bit hexadecimal number, which is the Node ID of a remote device if the command is sent directly to a node, or a group ID if the command is sent to a group. <EP> — 8-bit hexadecimal number, which is the endpoint of a remote device. Valid endpoint addresses are: 0x01 to 0xF0. <SendMode> — A Boolean type to choose transmission mode ('0' sends the command directly; '1' sends the command to a group). <Mcode> — 16-bit hexadecimal manufacturer code. <Cluster ID> — 16-bit hexadecimal cluster ID (see 1.2.2 Combined Interface). <Direction> — A Boolean type specifies whether values of the attribute are reported (0), or whether reports of the attribute are received (1). <AttrID> — 16-bit hexadecimal number representing the attribute ID according to manufacturer definition. This command can be used to read multiple attributes (up to 16 in a cluster). If the address is a Node ID, set SendMode to be 0, or the address will be recognized as a group ID. If destination address, endpoint and SendMode are not given (“,” cannot be omitted) the command will search the binding table for a destination. If this is the user's preferred method, then it is necessary to add the required destination to the local binding table in advance. After that, it will not be necessary to fill in the <address> parameter.	OK READMRPTCFGRSP:<NodeID>,<EP>, <Mcode><ClusterID>,<Status>,<Direction>, <AttrID>,[<DataType>, <MinimumReportingInterval>, <MaximumReportingInterval>, <ReportableChange>][<Timeout>] or ERROR:<errorcode> <errorcode> (See 3. List of Status codes.) <NodeID> — 16-bit hexadecimal number, which is the source Node ID of the response. <EP> — 8-bit hexadecimal number, the source endpoint of the response. <Mcode> — 16-bit hexadecimal manufacturer code. <Cluster ID> — 16-bit hexadecimal cluster ID (see 1.2.2 Combined Interface). <Status> — 8-bit hexadecimal number indicating the result of the requested operation. <Direction> — 8-bit hexadecimal number specifies whether values of the attribute are reported (0x00), or whether reports of the attribute are received (0x01). <AttrID> — Attribute ID as a 16-bit hexadecimal number. If the received <Direction> field is 0x00, the following information is expected: <DataType> — 8-bit hexadecimal number which represents the attribute data type. <MinimumReportingInterval> — The minimum reporting interval field is 16 bits in length and will contain the minimum interval, in seconds, between issuing reports for the attribute specified in the attribute Identifier field. If the minimum reporting interval has not been configured, this field will contain the value 0xffff.

Execute Command	Response
	<MaximumReportingInterval> — The maximum reporting interval field is 16 bits in length and will contain the maximum interval, in seconds, between issuing reports for the attribute specified in the attribute Identifier field. If the maximum reporting interval has not been configured, this field will contain the value 0xffff. <ReportableChange> — The reportable change field will contain the minimum change to the attribute that will result in a report being issued. For attributes with 'analog' data type (see ZigBee Cluster Library 075123r04), the field has the same data type as the attribute. If the reportable change has not been configured, this field will contain the invalid value for the relevant data type. For attributes of 'discrete' data types (see ZigBee Cluster Library) this field is omitted. If received, the <Direction> field is 0x01, and the following information is expected: <Timeout> — The timeout period field is 16-bits in length and will contain the maximum expected time, in seconds, between received reports for the attribute specified in the attribute identifier field. If the timeout period has not been configured, this field will contain the value 0xffff. <errorcode> (See 3. List of Status codes.)

Table 2.71. +CFGRPT – Configure Attribute Reporting

Execute Command	Response
AT+CFGRPT:<Address>,<EP>,<SendMode>, <ClusterID>,<Direction>,<AttrID>, [<DataType>,<MinimumReportingInterval>, <MaximumReportingInterval>, <ReportableChange>][<Timeout>] AT+CFGRPT:,,,<ClusterID>,<Direction>, <AttrID>,[<DataType>, <MinimumReportingInterval>, <MaximumReportingInterval>, <ReportableChange>][<Timeout>] <Address> — 16-bit hexadecimal number, which is the Node ID of a remote device if the command is sent directly to a node, or a group ID if the command is sent to a group. <EP> — 8-bit hexadecimal number, which is the endpoint of a remote device. Valid endpoint addresses are: 0x01 to 0xF0. <SendMode> — A Boolean type to choose transmission mode ('0' sends the command directly; '1' sends the command to a group). <Cluster ID> — 16-bit hexadecimal cluster ID (see 1.2.2 Combined Interface). <Direction> — A Boolean type specifies whether values of the attribute are reported (0), or whether reports of the attribute are received (1). <AttrID> — 16-bit hexadecimal number which represents attribute ID according to HA and ZigBee Cluster library. If the <Direction> field is set to 0, the following information needs to be provided: <DataType> — 8-bit hexadecimal number representing the attribute data type	OK CFGRPTRSP:<NodeID>,<EP>,<ClusterID>, <Status>[,<Direction>,<AttrID>] or ERROR:<errorcode> <errorcode> (See 3. List of Status codes.) <NodeID> — 16-bit hexadecimal number, which is the source Node ID of a response. <EP> — 8-bit hexadecimal number, the source endpoint of the response. <Cluster ID> — 16-bit hexadecimal cluster ID (see 1.2.2 Combined Interface). <Status> — 8-bit hexadecimal number indicating the result of the requested operation. If <Status> is not 00, the prompt information will include: <Direction> and <AttrID> <Direction> — A Boolean type that specifies whether values of the attribute are reported (0x00), or whether reports of the attribute are received (0x01). <AttrID> — Attribute ID as a 16-bit hexadecimal number. <errorcode> (See 3. List of Status codes.) Use this command to configure attribute reporting on a remote HA device which supports the corresponding cluster and attribute.

Execute Command	Response
<MinimumReportingInterval> — The minimum reporting interval field is 16 bits in length and will contain the minimum interval, in seconds, between issuing reports for the attribute specified in the attribute Identifier field. <MaximumReportingInterval> — The maximum reporting interval field is 16 bits in length and will contain the maximum interval, in seconds, between issuing reports for the attribute specified in the attribute Identifier field. <ReportableChange> — The reportable change field will contain the minimum change to the attribute that will result in a report being issued. For attributes with 'analog' data type (see ZigBee Cluster Library 075123r04) the field has the same data type as the attribute. If received, and the <Direction> field is 1, the following information is expected: <Timeout> — The timeout period field is 16 bits in length and will contain the maximum expected time, in seconds, between received reports for the attribute specified in the attribute Identifier field. This command can be used to configure multiple attributes reporting (up to five in a cluster). If the address is a Node ID, set SendMode to be 0, or the address will be recognized as a group ID. If destination address, endpoint, and SendMode are not given (“,” cannot be omitted) the command will search the binding table for a destination. If this is the user's preferred method, then it is necessary to add the required destination to the local binding table in advance. After that, it will not be necessary to fill in the <address> parameter.	

Table 2.72. +CFGMRPT – Configure Manufacture Defined Attribute Reporting

Execute Command	Response
AT+CFGMRPT:<Address>,<EP>,<SendMode>,<Mcode>,<ClusterID>,<Direction>,<AttrID>, [<DataType>,<MinimumReportingInterval>, <MaximumReportingInterval>, <ReportableChange>][<Timeout>] <Address> — 16-bit hexadecimal number, which is the Node ID of a remote device if the command is sent directly to a node, or a group ID if the command is sent to a group. <EP> — 8-bit hexadecimal number, which is the endpoint of a remote device. Valid endpoint addresses are: 0x01 to 0xF0. <SendMode> — A Boolean type to choose transmission mode ('0' sends the command directly; '1' sends the command to a group). <Mcode> — 16-bit hexadecimal manufacturer code. <Cluster ID> — 16-bit hexadecimal cluster ID (see 1.2.2 Combined Interface). <Direction> — A Boolean type that specifies whether values of the attribute are reported (0), or whether reports of the attribute are received (1). <AttrID> — 16-bit hexadecimal number representing the attribute ID according to manufacturer definition. If the <Direction> field is set to 0, the following information needs to be provided: <DataType> — 8-bit hexadecimal number representing the attribute data type. <MinimumReportingInterval> — The minimum reporting interval field is 16 bits in length and will contain the minimum interval, in seconds, between issuing reports for the attribute specified in the attribute identifier field. <MaximumReportingInterval> — The maximum reporting interval field is 16 bits in length and will contain the maximum interval, in seconds, between issuing reports for the attribute specified in the attribute identifier field. <ReportableChange> — The reportable change field will contain the minimum change to the attribute that will result in a report being issued. For attributes with 'analog' data type (see ZigBee Cluster Library 075123r04) the field has the same data type as the attribute. If the received <Direction> field is 1, the following information is expected: <Timeout> — The timeout period field is 16 bits in length and will contain the maximum expected time, in seconds, between received reports for the attribute specified in the attribute identifier field.	OK CFGMRPTRSP:<NodeID>,<EP>,<Mcode>, <Cluster>,<Status>[,<Direction>,<AttrID>] or ERROR:<errorcode> <errorcode> (See 3. List of Status codes.) <NodeID> — 16-bit hexadecimal number. It is the source Node ID of response. <EP> — 8-bit hexadecimal number, the source endpoint of the response. <Mcode> — 16-bit hexadecimal manufacturer code. <Cluster ID> — 16-bit hexadecimal cluster ID (see 1.2.2 Combined Interface). <Status> — 8-bit hexadecimal number which indicates the result of the requested operation. If <Status> is not 00, the prompt information will include: <Direction> and <AttrID> <Direction> — a Boolean type specifies whether values of the attribute are reported (0x00), or whether reports of the attribute are received (0x01). <AttrID> — Attribute ID as a 16-bit hexadecimal number. <errorcode> (See 3. List of Status codes.) Use this command to configure reporting for manufacture defined attribute on a remote HA device which supports the corresponding cluster and attribute.

Execute Command	Response
This command can be used to configure multiple attributes reporting (up to five in a cluster). If the address is a Node ID, set SendMode to be 0, or the address will be recognized as a group ID. If destination address, endpoint, and SendMode are not given (“,” cannot be omitted) the command will search the binding table for a destination. If this is the user's preferred method, then it is necessary to add the required destination to the local binding table in advance. After that, it will not be necessary to fill in the `<address>` parameter.	

Table 2.73. +SETATR – Set Value Of A Local ZCL Attribute

Execute Command	Response

`AT+SETATR:<ClusterID>,<AttrID>,<AttrValue>`
 	
`<Cluster ID>` — 16-bit hexadecimal cluster ID (see 1.2.2 Combined Interface).
 	
`<AttrID>` — 16-bit hexadecimal number representing attribute ID. (Refer to ZigBee Home Automation specification.)
 	
`<AttrValue>` — If the attribute value has an integer type, this field will be in big-endian format. If the attribute value has a string type, this field will contain a sequence of characters. Maximum length is 110 characters.
 	
The local and standard time attributes in the time cluster cannot be set using this command.
	`OK`
	
	or
	
	`ERROR:<errorcode>`
	
	`<errorcode>` (See 3. List of Status codes.)
	
	Use this command to set a new value to attribute from a cluster on a local device supporting this cluster.

Table 2.74. +GETATR – Get value of a local ZCL attribute

Execute Command	Response

`AT+GETATR:<Cluster>,<AttrID>`
 	
`<Cluster ID>` — 16-bit hexadecimal cluster ID (see 1.2.2 Combined Interface).
 	
`<AttrID>` — 16-bit hexadecimal number representing the attribute ID, according to ZigBee Home Automation specifications.
	`ATTR:<AttrValue>`
	
	`OK`
	
	or
	
	`ERROR:<errorcode>`
	
	`<errorcode>` (See 3. List of Status codes.)
	
	`<AttrValue>` — If the attribute value has an integer type, this field will contain a hexadecimal representation. If attribute value has a string type, this field contains a sequence of characters. Maximum length is 110 characters.
	
	`<errorcode>` (See 3. List of Status codes.)
	
	Use this command to get a value of an attribute from a cluster server on a local device supporting this cluster.

2.4.3 Combined Interface Specified Command Set

Default Response to ZCL Command Supported By CI

This section introduces the ZCL commands supported by the combined interface. When the user sends a ZCL command to a target device or group, a default response should be expected if that command does not have a ZCL specified response. In such a case, the prompt will show as following:

`DFTREP:<NodeID>,<EP>,<ClusterID>,<CMD>,<Status>`

where `<NodeID>` represents the address of the device which sends back the response.

`<EP>` is the endpoint where is the response from.

`<ClusterID>` shows the cluster which the command belongs to.

`<CMD>` is the command ID which the default response responds to.

`<Status>` indicates whether the command is implemented successfully or not. If this field is not 00, it is an error code. Refer to 3. List of Status codes to find the meaning of the error code.

In the following sections, the default response field will not be repeated.

Identify Cluster – Client

Table 2.75. +IDENTIFY – Start or Stop the Target Identifying itself

Execute Command	Response

`AT+IDENTIFY:<address>,<EP>,<SendMode>,<Time>`
 	
`<address>` — 16-bit hexadecimal number, which is the Node ID of a remote device if the command is sent directly to a node, or a group ID if the command is sent to a group.
 	
`<EP>` — 8-bit hexadecimal number representing the endpoint of the target.
 	
`<SendMode>` — A Boolean type to choose transmission mode ('0' sends the command directly; '1' sends the command to a group).
 	
`<Time>` — 16-bit hexadecimal number representing the Identification time.
 	
Use on: All Devices
 	
If the address is a Node ID, set SendMode to be 0, or the address will be recognized as a group ID.
	`OK`
	
	`DFTREP:<NodeID>,<EP>,<ClusterID>, <CMD>,<Status>`
	
	or
	
	`ERROR<errorcode>`
	
	`<errorcode>` (See 3. List of Status codes.)

Table 2.76. +IDQUERY – Query If Target Device(s) In Identifying Mode

Execute Command	Response
AT+IDQUERY:<address>,<EP>,<SendMode> <address> — 16-bit hexadecimal number, which is the Node ID of a remote device if the command is sent directly to a node, or a group ID if the command is sent to a group. <EP> — 8-bit hexadecimal number representing the endpoint of the target. <SendMode> — A Boolean type to choose transmission mode ('0' sends the command directly; '1' sends the command to a group). Use on: All Devices If the address is a Node ID, set SendMode to be 0, or the address will be recognized as a group ID.	OK IDQUERYRSP:<NodeID>,<EP>,<TimeOut> or ERROR<errorcode> <NodeID> — 16-bit hexadecimal number, which is the source of the response. It should be the same with the target's Node ID. <EP> — 8-bit hexadecimal number represents the endpoint of the target. <TimeOut> — 16-bit hexadecimal number, represents the length of time, in seconds, that the device will continue to identify itself. <errorcode> (See 3. List of Status codes.)

Group Cluster – Client

Table 2.77. +GPADD – Add Group On Target Device

Execute Command	Response
AT+GPADD:<address>,<EP>,<SendMode>, <GPID>,<GPName> AT+GPADD:,,,<GPID>,<GPName> <address> — 16-bit hexadecimal number, which is the Node ID of a remote device if the command is sent directly to a node, or a group ID if the command is sent to a group. <EP> — 8-bit hexadecimal number representing the endpoint of the target. <SendMode> — A Boolean type to choose transmission mode ('0' sends the command directly; '1' sends the command to a group). <GPID> — 16-bit hexadecimal number representing the specified group ID. <GPName> — String up to 16 characters. It can be omitted if the target does not support a group name. If the address is a Node ID, set SendMode to be 0, or the address will be recognized as a group ID. If destination address, endpoint, and SendMode are not given (“,” cannot be omitted) the command will search the binding table for a destination. If this is the user's preferred method, then it is necessary to add the required destination to the local binding table in advance. After that, it will not be necessary to fill in the <address> parameter.	OK ADDGPRSP:<NodeID>,<EP>,<Status>, <GPID> or ERROR:<errorcode> <Node ID> — 16-bit hexadecimal number representing the NodeID of the responding node. <EP> — 8-bit hexadecimal number representing the endpoint of the responding node. <Status> — 8-bit hexadecimal number, which is an error code if it is not 00. <GPID> — 16-bit hexadecimal number representing the group ID that will be added on the target device <errorcode> (See 3. List of Status codes.)

Table 2.78. +GPADIFID – Add Group Membership On Target Devices In Identification Mode

Execute Command	Response
AT+GPADIFID:<address>,<EP>,<SendMode>,<GPID>,<GPName> AT+GPADIFID:,,,<GPID>,<GPName> <address> — 16-bit hexadecimal number, which is the Node ID of a remote device if the command is sent directly to a node, or a group ID if the command is sent to a group. <EP> — 8-bit hexadecimal number representing the endpoint of the target. <SendMode> — A Boolean type to choose transmission mode ('0' sends the command directly; '1' sends the command to a group). <GPID> — 16-bit hexadecimal number representing the specified group ID. <GPName> — String up to 16 characters which can be omitted if the target does not support a group name. If the address is a Node ID, set SendMode to be 0, or the address will be recognized as a group ID. If destination address, endpoint, and SendMode are not given (“,” cannot be omitted) the command will search the binding table for a destination. If this is the user's preferred method, then it is necessary to add the required destination to the local binding table in advance. After that, it will not be necessary to fill in the <address> parameter.	OK DFTREP:<NodeID>,<EP>,<ClusterID>, <CMD>,<Status> or ERROR:<errorcode> <errorcode> (See 3. List of Status codes.) No ZCL response is defined as this command is expected to be multicast or broadcast.

Table 2.79. + GPVIEW – View Group Information Of Target Devices

Execute Command	Response
AT+GPVIEW:<address>,<EP>,<SendMode>,<GPID> AT+GPVIEW:,,, <GPID> <address> — 16-bit hexadecimal number, which is the Node ID of a remote device if the command is sent directly to a node, or a group ID if the command is sent to a group. <EP> — 8-bit hexadecimal number, which is the endpoint of a remote device. Valid endpoint addresses are: 0x01 to 0xF0. <SendMode> — A Boolean type to choose transmission mode ('0' sends the command directly; '1' sends the command to a group). <GPID> — 16-bit hexadecimal number, representing the specified group ID. If the address is a Node ID, set SendMode to be 0, or the address will be recognized as a group ID. If destination address, endpoint, and SendMode are not given (“,” cannot be omitted) the command will search the binding table for a destination. If this is the user's preferred method, then it is necessary to add the required destination to the local binding table in advance. After that, it will not be necessary to fill in the <address> parameter.	OK VIEWGPRSP:<NodeID>,<EP>,<Status>, [<GPID>,<GPName>] or ERROR:<errorcode> <NodeID> — 16-bit hexadecimal number represents the Node ID of the responding node. <EP> — 8-bit hexadecimal number representing the endpoint of the responding node. <Status> — 8-bit hexadecimal number. If status is 00, then <GPID> and <GPName> will show. If status is not 00, it is an error code, and <GPID> and <GPName> will not show. <GPID> — 16-bit hexadecimal number representing the group ID being added to the target device. <errorcode> (See 3. List of Status codes.)

Table 2.80. +GPGET — Get Group Membership Information From Target Devices

Execute Command	Response

`AT+GPGET:<address>,<EP>,<SendMode>,<Count>[,<GPList>]`
 	
`AT+GPGET:,,,<Count>[,<GPList>]`
 	
`<address>` — 16-bit hexadecimal number, which is the Node ID of a remote device if the command is sent directly to a node, or a group ID if the command is sent to a group.
 	
`<EP>` — 8-bit hexadecimal number, endpoint of a remote device. Valid endpoint addresses are: 0x01 to 0xF0.
 	
`<SendMode>` — A Boolean type to choose transmission mode ('0' sends the command directly; '1' sends the command to a group).
 	
`<Count>` — 8-bit hexadecimal number reflecting the number of inquired groups.
 	
`<GPList>` — Series of 16-bit hexadecimal numbers, representing a list of GPIDs (the number of GPIDs should correspond to the count number).
 	
If the address is a Node ID, set SendMode to be 0, or the address will be recognized as a group ID.
 	
If destination address, endpoint, and SendMode are not given (“,” cannot be omitted) the command will search the binding table for a destination. If this is the user's preferred method, then it is necessary to add the required destination to the local binding table in advance. After that, it will not be necessary to fill in the `<address>` parameter.
	`OK`
	
	`GETGPRSP:<NodeID>,<EP>,<Capacity>, <Count>,<GPID>…`
	
	or
	
	`ERROR:<errorcode>`
	
	`<NodeID>` — 16-bit hexadecimal number representing the Node ID of the responding node.
	
	`<EP>` — 8-bit hexadecimal number representing the endpoint of the responding node.
	
	`<Capacity>` — 8-bit hexadecimal number. The Capacity field shall contain the remaining capacity of the group table of the device. The following values apply:
	
	0: No further groups may be added.
	
	0 < Capacity < 0xfe Capacity holds the number of groups that may be added.
	
	0xfe: At least one further group may be added (exact number is unknown).
	
	0xff: It is unknown if any further groups may be added.
	
	`<Count>` — 8-bit hexadecimal number representing the number of groups contained in the group list field.
	
	`<GPID>` — 16-bit hexadecimal number, representing the group ID being added to the target device.
	
	`<errorcode>` (See 3. List of Status codes.)

Table 2.81. +GPRMV — Remove A Group Membership On Target Device

Execute Command	Response

`AT+GPRMV:<address>,<EP>,<SendMode>,<GPID>`
 	
`AT+GPRMV:,,,< GPID>`
 	
`<address>` — 16-bit hexadecimal number, which is the Node ID of a remote device if the command is sent directly to a node, or a group ID if the command is sent to a group.
 	
`<EP>` — 8-bit hexadecimal number, which is the endpoint of a remote device. Valid endpoint addresses are: 0x01 to 0xF0.
 	
`<SendMode>` — A Boolean type to choose transmission mode ('0' sends the command directly; '1' sends the command to a group).
 	
`<GPID>` — 16-bit hexadecimal number, representing the specified group ID.
 	
If the address is a Node ID, set SendMode to be 0, or the address will be recognized as a group ID.
 	
If destination address, endpoint, and SendMode are not given (“,” cannot be omitted) the command will search the binding table for a destination. If this is the user's preferred method, then it is necessary to add the required destination to the local binding table in advance. After that, it will not be necessary to fill in the `<address>` parameter.
	`OK`
	
	`RMVGPRSP:<NodeID>,<EP>,<Status>, <GPID>`
	
	or `ERROR:<errorcode>`
	
	`<NodeID>` — 16-bit hexadecimal number represents the Node ID of the responding node.
	
	`<EP>` — 8-bit hexadecimal number represents the endpoint of the responding node.
	
	`<Status>` — 8-bit hexadecimal number. If it is not 00, it will be an error code (check 3. List of Status codes), then `<GPID>` will not show.
	
	`<GPID>` — 16-bit hexadecimal number, representing the group ID that will be added on the target device
	
	`<errorcode>` (See 3. List of Status codes.)

Table 2.82. +GPRMALL — Remove All Group Associations On Target Device

Execute Command	Response

`AT+GPRMALL:<address>,<EP>,<SendMode>`
 	
`AT+GPRMALL:,,,`
 	
`<address>` — 16-bit hexadecimal number, which is the Node ID of a remote device if the command is sent directly to a node, or a group ID if the command is sent to a group.
 	
`<EP>` — 8-bit hexadecimal number, which is the endpoint of a remote device. Valid endpoint addresses are: 0x01 to 0xF0.
 	
`<SendMode>` — A Boolean type to choose transmission mode ('0' sends the command directly; '1' sends the command to a group).
 	
If the address is a Node ID, set SendMode to be 0, or the address will be recognized as a group ID.
 	
If destination address, endpoint, and SendMode are not given (“,” cannot be omitted) the command will search the binding table for a destination. If this is the user's preferred method, then it is necessary to add the required destination to the local binding table in advance. After that, it will not be necessary to fill in the `<address>` parameter.
 	
Removing all groups requires the removal of all associated scenes as well. Scenes not associated with a group do not have to be removed.
	`OK`
	
	`DFTREP:<NodeID>,<EP>,<ClusterID>, <CMD>,<Status>`
	
	or
	
	`<errorcode>` (See 3. List of Status codes.)

Scenes Cluster – Client

Table 2.83. +SCADD — Add Scenes on Target Devices

Execute Command	Response

`AT+SCADD:<address>,<EP>,<SendMode>, <GPID>,<SCID>,<TransTime>,<SCName>, [Extension field sets]`
 	
`<address>` — 16-bit hexadecimal number, which is the Node ID of a remote device.
 	
`<EP>` — 16-bit hexadecimal number representing the endpoint of the target.
 	
`<SendMode>` — A Boolean type to choose transmission mode ('0' sends the command directly; '1' sends the command to a group). According to ZigBee Cluster Specification, this command is unicast. The SendMode value is 0 for this command.
 	
`<GPID>` — 16-bit hexadecimal number, representing the specified group ID.
 	
`<SCID>` — 8-bit hexadecimal number scene ID.
 	
`<TransTime>` — 16-bit hexadecimal number, which is the amount of time in seconds it will take for the device to change from its current state to the requested scene.
 	
`<SCName>` — String of up to 16 characters representing the scene's name.
 	
`[Extension field sets]` – Holds a set of values of attributes for a cluster implemented on the device. It can be constructed as:
 	
`<ClusterID>`, `<Length>`,`<AttributeValue>`
 	
The sum of all such sets defines a scene. This is an optional parameter which can be omitted. Check the ZigBee Cluster Library for more details.
 	
An example of a constructed command can be:
 	
`AT+SCADD:0234,01,0,0001,01,001C,Light, 0006,01,01`
 	
If there is already a scene in the table with the same Scene ID and Group ID, the new Scene entry will overwrite the previous one.
	`OK`
	
	`ADDSC:<NodeID>,<EP>,<Status>[,<GPID>, <SCID>]`
	
	or
	
	`ERROR:<errorcode>`
	
	`<NodeID>` — 16-bit hexadecimal number represents the Node ID of the responding node.
	
	`<EP>` — 8-bit hexadecimal number represents the endpoint of the responding node.
	
	`<Status>` — 8-bit hexadecimal number. If it is not 00, it is an error code (check 3. List of Status codes). `<GPID>` and `<SCID>` will not show for error codes.
	
	`<GPID>` — 16-bit hexadecimal number, representing the group ID that will be added on the target device.
	
	`<SCID>` — 8-bit hexadecimal number scene ID.
	
	`<errorcode>` (See 3. List of Status codes.)

Table 2.84. +SCVIEW — View Scenes Information of Target Devices

Execute Command	Response
AT+SCVIEW:<address>,<EP>,<SendMode>, <GPID>,<SCID> AT+SCVIEW:,,,<GPID>,<SCID> <address> — 16-bit hexadecimal number, which is the Node ID of a remote device. According to ZigBee Cluster Library, the View Scene command is addressed to a single device (not a group). <EP> — 16-bit hexadecimal number represents the endpoint of the target. <SendMode> — A Boolean type to choose transmission mode ('0' sends the command directly; '1' sends the command to a group). According to the ZigBee Cluster Specification, this command is unicast. SendMode value is set to 0 for this command. <GPID> — 16-bit hexadecimal number, representing the specified group ID. <SCID> — 8-bit hexadecimal number scene ID.	OK VIEWSC:<NodeID>,<EP>,<Status>,[<GPID>,<SCID>,<SCName>] or ERROR:<errorcode> <NodeID> — 16-bit hexadecimal number representing the Node ID of the responding node. <EP> — 8-bit hexadecimal number representing the endpoint of the responding node. <Status> — 8-bit hexadecimal number. If it is not 00, it is an error code (check 3. List of Status codes). <GPID>, <SCID> and <SCName> will not show with an error code. <GPID> — 16-bit hexadecimal number, representing the group ID that will be added to the target device. <SCID> — 8-bit hexadecimal number scene ID. <SCName> — String of up to 16 characters representing the scene's name. <errorcode> (See 3. List of Status codes.)

Table 2.85. +SCRMV — Remove A Scene On Target Device

Execute Command	Response
AT+SCRMV:<address>,<EP>,<SendMode>, <GPID>,<SCID> AT+SCRMV:,,,< GPID>,<SCID> <address> — 16-bit hexadecimal number, which is the Node ID of a remote device if the command is sent directly to a node, or a group ID if the command is sent to a group. <EP> — 8-bit hexadecimal number, which is the endpoint of a remote device. Valid endpoint addresses are: 0x01 to 0xF0. <SendMode> — A Boolean type to choose transmission mode ('0' sends the command directly; '1' sends the command to a group). <GPID> — 16-bit hexadecimal number, representing the specified group ID. <SCID> — 8-bit hexadecimal number, scene ID. If the address is a Node ID, set SendMode to be 0, or the address will be recognized as a group ID. If destination address, endpoint, and SendMode are not given (“,” cannot be omitted) the command will search the binding table for a destination. If this is the user's preferred method, then it is necessary to add the required destination to the local binding table in advance. After that, it will not be necessary to fill in the <address> parameter.	OK RMVSCRSP:<NodeID>,<EP>,<Status>, <GPID>,<SCID> or ERROR:<errorcode> <NodeID> — 16-bit hexadecimal number representing the Node ID of the responding node. <EP> — 8-bit hexadecimal number representing the endpoint of the responding node. <Status> — 8-bit hexadecimal number. If it is not 00, it is an error code (check 3. List of Status codes). <GPID> and <SCID> will not show with an error code. <GPID> — 16-bit hexadecimal number, representing the group ID being added to the target device <SCID> — 8-bit hexadecimal number, scene ID. <errorcode> (See 3. List of Status codes.)

Table 2.86. +SCRMALL — Remove All Scenes On Target Device

Execute Command	Response

`AT+SCRMALL:<address>,<EP>,<SendMode>,<GPID>`
 	
`AT+SCRMALL:,,,<GPID>`
 	
`<address>` — 16-bit hexadecimal number, which is the Node ID of a remote device if the command is sent directly to a node, or a group ID if the command is sent to a group.
 	
`<EP>` — 8-bit hexadecimal number, which is the endpoint of a remote device. Valid endpoint addresses are: 0x01 to 0xF0.
 	
`<SendMode>` — A Boolean type to choose transmission mode ('0' sends the command directly; '1' sends the command to a group).
 	
`<GPID>` — 16-bit hexadecimal number, representing the specified group ID.
 	
If the user does not determine a destination address, the command will search the binding table for a destination. If this is the user's preferred method, then it is necessary to add the required destination to the local binding table in advance. After that, it will not be necessary to fill in the `<address>` parameter.
	`OK`
	
	`RMASCSRSP:<NodeID>,<EP>,<Status>, <GPID>`
	
	or
	
	`ERROR:<errorcode>`
	
	`<NodeID>` — 16-bit hexadecimal number representing the Node ID of the responding node.
	
	`<EP>` — 8-bit hexadecimal number representing the endpoint of the responding node.
	
	`<Status>` — 8-bit hexadecimal number. If it is not 00, it is an error code (check 3. List of Status codes). `<GPID>` and `<SCID>` will not show with an error code.
	
	`<GPID>` — 16-bit hexadecimal number, representing the group ID being added to the target device.
	
	`<errorcode>` (See 3. List of Status codes.)

Table 2.87. +SCSTORE – Store Scenes

Execute Command	Response

`AT+SCSTORE:<address>,<EP>,<SendMode>,<GPID>,<SCID>`
 	
`AT+SCSTORE:,,,<GPID>,<SCID>`
 	
`<address>` — 16-bit hexadecimal number, which is the Node ID of a remote device if the command is sent directly to a node, or a group ID if the command is sent to a group.
 	
`<EP>` — 8-bit hexadecimal number, which is the endpoint of a remote device. Valid endpoint addresses are: 0x01 to 0xF0.
 	
`<SendMode>` — A Boolean type to choose transmission mode ('0' sends the command directly; '1' sends the command to a group).
 	
`<GPID>` — 16-bit hexadecimal number, representing the specified group ID.
 	
`<SCID>` — 8-bit hexadecimal number, scene ID.
 	
If the address is a Node ID, set SendMode to be 0, or the address will be recognized as a group ID.
 	
If destination address, endpoint, and SendMode are not given (“,” cannot be omitted), the command will search the binding table for a destination. If this is the user's preferred method, then it is necessary to add the required destination to the local binding table in advance. After that, it will not be necessary to fill in the `<address>` parameter.
	`OK`
	
	`STORSCRSP:<NodeID>,<EP>,<Status>, <GPID>,<SCID>`
	
	or
	
	`ERROR:<errorcode>`
	
	`<NodeID>` — 16-bit hexadecimal number representing the Node ID of the responding node.
	
	`<EP>` — 8-bit hexadecimal number representing the endpoint of the responding node.
	
	`<Status>` — 8-bit hexadecimal number. If it is not 00, it is an error code (check 3. List of Status codes). `<GPID>` and `<SCID>` will not show with an error code.
	
	`<GPID>` — 16-bit hexadecimal number, representing the group ID being added to the target device.
	
	`<SCID>` — 8-bit hexadecimal number, scene ID.
	
	`<errorcode>` (See 3. List of Status codes.)

Table 2.88. +SCRECAL — Recall Scenes

Execute Command	Response

`AT+SCRECAL:<address>,<EP>,<SendMode>,<GPID>,<SCID>`
 	
`AT+SCRECAL:,,,<GPID>,<SCID>`
 	
`<address>` — 16-bit hexadecimal number, which is the Node ID of a remote device if the command is sent directly to a node, or a group ID if the command is sent to a group.
 	
`<EP>` — 8-bit hexadecimal number, which is the endpoint of a remote device. Valid endpoint addresses are: 0x01 to 0xF0.
 	
`<SendMode>` — A Boolean type to choose transmission mode ('0' sends the command directly; '1' sends the command to a group).
 	
`<GPID>` — 16-bit hexadecimal number, representing the specified group ID.
 	
`<SCID>` — 8-bit hexadecimal number, scene ID.
 	
If the address is a Node ID, set SendMode to be 0, or the address will be recognized as a group ID.
 	
If destination address, endpoint, and SendMode are not given (“,” cannot be omitted), the command will search the binding table for a destination. If this is the user's preferred method, then it is necessary to add the required destination to the local binding table in advance. After that, it will not be necessary to fill in the `<address>` parameter.
	`OK`
	
	or
	
	`ERROR:<errorcode>`
	
	`<errorcode>` (See 3. List of Status codes.)

Table 2.89. +SCGETMSH – Get Scene Membership

Execute Command	Response
AT+SCGETMSH:<address>,<EP>, <SendMode>,<GPID> AT+SCGETMSH:,,,<GPID> <address> — 16-bit hexadecimal number, which is the Node ID of a remote device if the command is sent directly to a node, or a group ID if the command is sent to a group. <EP> — 8-bit hexadecimal number, which is the endpoint of a remote device. Valid endpoint addresses are: 0x01 to 0xF0. <SendMode> — A Boolean type to choose transmission mode ('0' sends the command directly; '1' sends the command to a group). <GPID> — 16-bit hexadecimal number, representing the specified group ID. This command is used to find an unused scene number within the group when no commissioning tool is in the network. It can also be used as a commissioning tool to get used scenes for a group on a single device or on all group members. If the address is a Node ID, set SendMode to be 0, or the address will be recognized as a group ID. If destination address, endpoint, and SendMode are not given (“,” cannot be omitted), the command will search the binding table for a destination. If this is the user's preferred method, then it is necessary to add the required destination to the local binding table in advance. After that, it will not be necessary to fill in the <address> parameter.	OK SCMSHRSP:<NodeID>,<EP>,<Status>, [<Capacity>,<GPID>,<SCCount>,<SCList>] or ERROR:<errorcode> <NodeID> — 16-bit hexadecimal number representing the Node ID of the responding node. <EP> — 8-bit hexadecimal number representing the endpoint of the responding node. <Status> — 8-bit hexadecimal number. If it is not 00, it is an error code, and <Capacity>, <GPID>, <SCCount>, and <SCList> will not show. <Capacity> — 8-bit hexadecimal number. The Capacity field contains the remaining capacity of the group table of the device. The following values apply: 0: No further groups may be added. 0 < Capacity < 0xfe Capacity: Holds the number of groups that may be added. 0xfe: At least one further group may be added (exact number is unknown). 0xff: It is unknown if any further groups may be added. <GPID> — 16-bit hexadecimal number, representing the group ID being added to the target device. <SCCount> — 8-bit hexadecimal number representing the number of scenes contained in the Scene list field. <SCList> — Contains the identifiers of all the scenes in the scene table with the corresponding Group ID. <errorcode> (See 3. List of Status codes.)

On/Off Cluster – Client

Table 2.90. +RONOFF – Switching Target Devices Between 'On' and 'Off' States

Execute Command	Response

`AT+RONOFF:<address>,<EP>,<SendMode>[,<ON/OFF>]`
 	
`AT+RONOFF:,,,[<ON/OFF>]`
 	
`<address>` — 16-bit hexadecimal number, which is the Node ID of a remote device if the command is sent directly to a node, or a group ID if the command is sent to a group.
 	
`<EP>` — 8-bit hexadecimal number, which is the endpoint of a remote device. Valid endpoint addresses are: 0x01 to 0xF0.
 	
`<SendMode>` — A Boolean type to choose transmission mode ('0' sends the command directly; '1' sends the command to a group).
 	
`<ON/OFF>` — A Boolean type to choose transmission mode ('0' means Off; '1' means On). If this field is left blank, this will be a toggle command.
 	
If the address is a Node ID, set SendMode to be 0, or the address will be recognized as a group ID.
 	
If destination address, endpoint, and SendMode are not given (","cannot be omitted), the command will search the binding table for a destination. If this is the user's preferred method, then it is necessary to add the required destination to the local binding table in advance. After that, it will not be necessary to fill in the `<address>` parameter.
	`OK`
	
	or
	
	`ERROR:<errorcode>`
	
	`<errorcode>` (See 3. List of Status codes.)
	
	Default response is expected to show the status of the required operation on the target.

Level Control Cluster – Client

Table 2.91. +LCMVTOLEV – Move To Level

Execute Command	Response

`AT+LCMVTOLEV:<address>,<EP>,<SendMode>,<ON/OFF>,<LevelValue>,<TransTime>`
 	
`AT+LCMVTOLEV:,,,<ON/OFF>,<LevelValue>, <TransTime>`
 	
`<address>` — 16-bit hexadecimal number, which is the Node ID of a remote device if the command is sent directly to a node, or a group ID if the command is sent to a group.
 	
`<EP>` — 8-bit hexadecimal number, which is the endpoint of a remote device. Valid endpoint addresses are: 0x01 to 0xF0.
 	
`<SendMode>` — A Boolean type to choose transmission mode ('0' sends the command directly; '1' sends the command to a group).
 	
`<ON/OFF>` — A Boolean number that represents whether the command is used with On/Off. If set to 0, the command is implemented as a Move to Level command. If set to 1, the command will be implemented as a Move to Level (with On/Off) command.
 	
`<LevelValue>` — 8-bit hexadecimal number. The meaning of 'level' is device dependent, for example, with a light it may mean brightness level.
 	
`<TransTime>` — 16-bit hexadecimal number. The movement should be as continuous as technically practical (i.e. not a step function), and the time taken to move to the new level should be equal to the value of the Transition time field, in tenths of a second, or as close to this as the device is able.
 	
Before commencing any command that has the effect of increasing, CurrentLevel, the On/Off attribute of the On/Off cluster on the same endpoint, if implemented, shall be set to On. If any command that decreases CurrentLevel reduces it to the minimum level allowed by the device, the On/Off attribute of the On/Off cluster on the same endpoint, if implemented, shall be set to Off.
 	
If the address is a Node ID, set SendMode to be 0, or the address will be recognized as a group ID.
 	
If destination address, endpoint, and SendMode are not given (“,” cannot be omitted), the command will search the binding table for a destination. If this is the user's preferred method, then it is necessary to add the required destination to the local binding table in advance. After that, it will not be necessary to fill in the `<address>` parameter.
	`OK`
	
	or
	
	`ERROR:<errorcode>`
	
	`<errorcode>` (See 3. List of Status codes.)
	
	Default response is expected to show the status of the required operation on the target.

Table 2.92. +LCMV – Level Control Move Command

Execute Command	Response

`AT+LCMV:<address>,<EP>,<SendMode>,<ON/OFF>, <Mode>,<Rate>`
 	
`AT+LCMV:,,,<ON/OFF>,<Mode>,<Rate>`
 	
`<address>` — 16-bit hexadecimal number, which is the Node ID of a remote device, or a group ID if the command is sent to a group.
 	
`<EP>` — 8-bit hexadecimal number, which is the endpoint of a remote device. Valid endpoint addresses are: 0x01 to 0xF0.
 	
`<SendMode>` — A Boolean type to choose transmission mode ('0' sends the command directly; '1' sends the command to a group).
 	
`<ON/OFF>` — A Boolean number that represents whether the command is used with On/Off. If set to 0, the command is implemented as a Move command. If set to 1, the command is implemented as a Move (with On/Off) command.
 	
`<Mode>` — 8-bit hexadecimal number representing move mode (0x00 means up; 0x01 means down).
 	
`<Rate>` — 8-bit hexadecimal number, specifies the rate of movement in units per second. The actual rate of movement should be as close to this rate as the device is able. If the Rate field is 0xff, the device should move as fast as it is able.
 	
Before commencing any command that has the effect of increasing, CurrentLevel, the On/Off attribute of the On/Off cluster on the same endpoint, if implemented, shall be set to On. If any command that decreases CurrentLevel reduces it to the minimum level allowed by the device, the On/Off attribute of the On/Off cluster on the same endpoint, if implemented, shall be set to Off.
 	
If the address is a Node ID, set SendMode to be 0, or the address will be recognized as a group ID.
 	
If destination address, endpoint, and SendMode are not given (“,” cannot be omitted), the command will search local binding table for a destination. If this is the user's preferred method, then it is necessary to add the required destination to the local binding table in advance. After that, it will not be necessary to fill in the `<address>` parameter.
	`OK`
	
	or
	
	`ERROR:<errorcode>`
	
	`<errorcode>` (See 3. List of Status codes.)
	
	Default response is expected to show the status of the required operation on the target.

Table 2.93. +LCSTEP – Level Control Step Command

Execute Command	Response

`AT+LCSTEP:<address>,<EP>,<SendMode>,<ON/OFF>,<Mode>,<Size>,<TransTime>`
 	
`AT+LCSTEP:,,,<ON/OFF>,<Mode>,<Size>, <TransTime>`
 	
`<address>` — 16-bit hexadecimal number, which is the Node ID of a remote device if the command is sent directly to a node, or a group ID if the command is sent to a group.
 	
`<EP>` — 8-bit hexadecimal number, which is the endpoint of a remote device. Valid endpoint addresses are: 0x01 to 0xF0.
 	
`<SendMode>` — A Boolean type to choose transmission mode ('0' sends the command directly; '1' sends the command to a group).
 	
`<ON/OFF>` — A Boolean number that represents whether the command is used with On/Off. If set to 0, the command is implemented as a Move command. If set to 1, the command will be implemented as a Move (with On/Off) command.
 	
`<Mode>` — 8-bit hexadecimal number representing move mode (0x00 means up; 0x01 means down).
 	
`<Size>` — 8-bit hexadecimal number.
 	
`<TransTime>` — 16-bit hexadecimal number, The movement should be as continuous as technically practical (i.e. not a step function), and the time taken to move to the new level should be equal to the value of the Transition time field, in tenths of a second, or as close to this as the device is able.
 	
Before commencing any command that has the effect of increasing, CurrentLevel, the On/Off attribute of the On/Off cluster on the same endpoint, if implemented, shall be set to On. If any command that decreases CurrentLevel reduces it to the minimum level allowed by the device, the On/Off attribute of the On/Off cluster on the same endpoint, if implemented, shall be set to Off.
 	
If the address is a Node ID, set SendMode to be 0, or the address will be recognized as a group ID.
 	
If destination address, endpoint, and SendMode are not given (“,” cannot be omitted), the command will search the binding table for a destination. If this is the user's preferred method, then it is necessary to add the required destination to the local binding table in advance. After that, it will not be necessary to fill in the `<address>` parameter.
	`OK`
	
	or
	
	`ERROR:<errorcode>`
	
	`<errorcode>` (See 3. List of Status codes.)
	
	Default response is expected to show the status of the required operation on the target.

Table 2.94. +LCSTOP – Level Control Stop Command

Execute Command	Response

`AT+LCSTOP:<address>,<EP>,<SendMode>,<ON/OFF>`
 	
`AT+LCSTOP:<ON/OFF>`
 	
`<address>` — 16-bit hexadecimal number, which is the Node ID of a remote device if the command is sent directly to a node, or a group ID if the command is sent to a group.
 	
`<EP>` — 8-bit hexadecimal number, which is the endpoint of a remote device. Valid endpoint addresses are: 0x01 to 0xF0.
 	
`<SendMode>` — A Boolean type to choose transmission mode ('0' sends the command directly; '1' sends the command to a group).
 	
`<ON/OFF>` — A Boolean number that represents whether the command is used with On/Off. If set to 0, the command is implemented as a Move command. If set to 1, the command will be implemented as a Move (with On/Off) command.
 	
Before commencing any command that has the effect of increasing, CurrentLevel, the On/Off attribute of the On/Off cluster on the same endpoint, if implemented, should be set to On. If any command that decreases CurrentLevel reduces it to the minimum level allowed by the device, the On/Off attribute of the On/Off cluster on the same endpoint, if implemented, should be set to Off.
 	
If the address is a Node ID, set SendMode to be 0, or the address will be recognized as a group ID.
 	
If destination address, endpoint, and SendMode are not given (“,” cannot be omitted), the command will search the binding table for a destination. If this is the user's preferred method, then it is necessary to add the required destination to the local binding table in advance. After that, it will not be necessary to fill in the `<address>` parameter.
	`OK`
	
	or
	
	`ERROR:<errorcode>`
	
	`<errorcode>` (See 3. List of Status codes.)
	
	Default response is expected to show the status of the required operation on the target.

Color Control Cluster – Client

Table 2.95. +CCMVTOCOL – Color Control Move to Color Command

Execute Command	Response

`AT+CCMVTOCOL:<address>,<EP>,<SendMode>,<ColorX>,<ColorY>,<TransTime>`
 	
`AT+CCMVTOCOL:,,,<ColorX>,<ColorY>,<TransTime>`
 	
`<address>` — 16-bit hexadecimal number, which is the Node ID of a remote device if the command is sent directly to a node, or a group ID if the command is sent to a group.
 	
`<EP>` — 8-bit hexadecimal number, which is the endpoint of a remote device. Valid endpoint addresses are: is 0x01 to 0xF0.
 	
`<SendMode>` — A Boolean type to choose transmission mode ('0' sends the command directly; '1' sends the command to a group).
 	
`<ColorX>` — 16-bit hexadecimal number.
 	
`<ColorY>` — 16-bit hexadecimal number.
 	
`<TransTime>` — 16-bit hexadecimal number.
 	
Check the ZigBee Cluster Library for more field explanations.
 	
If the address is a Node ID, set SendMode to be 0, or the address will be recognized as a group ID.
 	
If destination address, endpoint, and SendMode are not given (“,” cannot be omitted), the command will search the binding table for a destination. If this is the user's preferred method, then it is necessary to add the required destination to the local binding table in advance. After that, it will not be necessary to fill in the `<address>` parameter.
	`OK`
	
	or
	
	`ERROR:<errorcode>`
	
	`<errorcode>` (See 3. List of Status codes.)
	
	Default response is expected to show the status of the required operation on the target.

Table 2.96. +CCMVCOL – Color Control Move Color Command

Execute Command	Response

`AT+CCMVCOL:<address>,<EP>,<SendMode>,<RateX>, <RateY>`
 	
`AT+CCMVCOL:,,,<RateX >,<RateY>`
 	
`<address>` — 16-bit hexadecimal number, which is the Node ID of a remote device if the command is sent directly to a node, or a group ID if the command is sent to a group.
 	
`<EP>` — 8-bit hexadecimal number, which is the endpoint of a remote device. Valid endpoint addresses are: 0x01 to 0xF0.
 	
`<SendMode>` — A Boolean type to choose transmission mode ('0' sends the command directly; '1' sends the command to a group).
 	
`<RateX>` — 16-bit hexadecimal number
 	
`<RateY>` — 16-bit hexadecimal number
 	
Check the ZigBee Cluster Library for more field explanations.
 	
If the address is a Node ID, set SendMode to be 0, or the address will be recognized as a group ID.
 	
If destination address, endpoint, and SendMode are not given (“,” cannot be omitted), the command will search the binding table for a destination. If this is the user's preferred method, then it is necessary to add the required destination to the local binding table in advance. After that, it will not be necessary to fill in the `<address>` parameter.
	`OK`
	
	or
	
	`ERROR:<errorcode>`
	
	`<errorcode>` (See 3. List of Status codes.)
	
	Default response is expected to show the status of the required operation on the target.

Table 2.97. +CCSTEPCOL – Color Control Step Color Command

Execute Command	Response

`AT+CCSTEPCOL:<address>,<EP>,<SendMode>,<StepX>,<StepY>,<TransTime>`
 	
`AT+CCSTEPCOL:,,,<StepX>,<StepY>,<TransTime>`
 	
`<address>` — 16-bit hexadecimal number, which is the Node ID of a remote device if the command is sent directly to a node, or a group ID if the command is sent to a group.
 	
`<EP>` — 8-bit hexadecimal number, which is the endpoint of a remote device. Valid endpoint addresses are: 0x01 to 0xF0.
 	
`<SendMode>` — A Boolean type to choose transmission mode ('0' sends the command directly; '1' sends the command to a group).
 	
The `<StepX>` and `<StepY>` fields specify the change to be added to the device's CurrentX attribute and CurrentY attribute, respectively.
 	
`<TransTime>` — 16-bit hexadecimal number stating the transition time for the step change.
 	
If the address is a Node ID, set SendMode to be 0, or the address will be recognized as a group ID.
 	
If destination address, endpoint, and SendMode are not given (“,” cannot be omitted), the command will search the binding table for a destination. If this is the user's preferred method, then it is necessary to add the required destination to the local binding table in advance. After that, it will not be necessary to fill in the `<address>` parameter.
	`OK`
	
	or
	
	`ERROR:<errorcode>`
	
	`<errorcode>` (See 3. List of Status codes.)
	
	Default response is expected to show the status of the required operation on the target.

Table 2.98. +CCMVTOHUE – Color Control Move to Hue Command

Execute Command	Response

`AT+CCMVTOHUE:<address>,<EP>,<SendMode>, <Hue>,<Direction>,<TransTime>`
 	
`AT+CCMVTOHUE:,,,<Hue>,<Direction>,<TransTime>`
 	
`<address>` — 16-bit hexadecimal number, which is the Node ID of a remote device if the command is sent directly to a node, or a group ID if the command is sent to a group.
 	
`<EP>` — 8-bit hexadecimal number, which is the endpoint of a remote device. Valid endpoint addresses are: 0x01 to 0xF0.
 	
`<SendMode>` — A Boolean type to choose transmission mode ('0' sends the command directly; '1' sends the command to a group).
 	
`<Hue>` — 8-bit hexadecimal number.
 	
`<Direction>` — 8-bit hexadecimal number.
 	
`<TransTime>` — 16-bit hexadecimal number. The Transition time field specifies, in a tenth of a second, the time that should be taken to perform the step. A step is a change in the device's saturation of 'Step size' units.
 	
Check the ZigBee Cluster Library for more field explanations.
 	
If the address is a Node ID, set SendMode to be 0, or the address will be recognized as a group ID.
 	
If destination address, endpoint, and SendMode are not given (",” cannot be omitted), the command will search the binding table for a destination. If this is the user's preferred method, then it is necessary to add the required destination to the local binding table in advance. After that, it will not be necessary to fill in the `<address>` parameter.
	`OK`
	
	or
	
	`ERROR:<errorcode>`
	
	`<errorcode>` (See 3. List of Status codes.)
	
	Default response is expected to show the status of the required operation on the target.

Table 2.99. +CCMVHUE – Color Control Move Hue Command

Execute Command	Response

`AT+CCMVHUE:<address>,<EP>,<SendMode>,<Mode>,<Rate>`
 	
`AT+CCMVHUE:,,,<Mode>,<Rate>`
 	
`<address>` — 16-bit hexadecimal number, which is the Node ID of a remote device if the command is sent directly to a node, or a group ID if the command is sent to a group.
 	
`<EP>` — 8-bit hexadecimal number, which is the endpoint of a remote device. Valid endpoint addresses are: 0x01 to 0xF0.
 	
`<SendMode>` — A Boolean type to choose transmission mode ('0' sends the command directly; '1' sends the command to a group).
 	
`<Mode>` — 8-bit hexadecimal number, move mode.
 	
`<Rate>` — 8-bit hexadecimal number.
 	
Check the ZigBee Cluster Library for more field explanations.
 	
If the address is a Node ID, set SendMode to be 0, or the address will be recognized as a group ID.
 	
If destination address, endpoint, and SendMode are not given (“,” cannot be omitted), the command will search the binding table for a destination. If this is the user's preferred method, then it is necessary to add the required destination to the local binding table in advance. After that, it will not be necessary to fill in the `<address>` parameter.
	`OK`
	
	or
	
	`ERROR:<errorcode>`
	
	`<errorcode>` (See 3. List of Status codes.)
	
	Default response is expected to show the status of the required operation on the target.

Table 2.100. +CCSTEPHUE – Color Control Step Hue Command

Execute Command	Response

`AT+CCSTEPHUE:<address>,<EP>,<SendMode>,<StepMode>, <StepSize>,<TransTime>`
 	
`AT+CCSTEPHUE:,,,<StepMode>,<StepSize>,<TransTime>`
 	
`<address>` — 16-bit hexadecimal number which is the Node ID of a remote device if the command is sent directly to a node, or a group ID if the command is sent to a group.
 	
`<EP>` — 8-bit hexadecimal number, which is the endpoint of a remote device. Valid endpoint addresses are: 0x01 to 0xF0.
 	
`<SendMode>` — A Boolean type to choose transmission mode ('0' sends the command directly; '1' sends the command to a group).
 	
`<StepMode>` — 8-bit hexadecimal number (0x01 means Up; 0x03 means Down).
 	
`<StepSize>` — 8-bit hexadecimal number.
 	
`<TransTime>` — 8-bit hexadecimal number.
 	
Check the ZigBee Cluster Library for more field explanations.
 	
If the address is a Node ID, set SendMode to be 0, or the address will be recognized as a group ID.
 	
If destination address, endpoint, and SendMode are not given (“,” cannot be omitted), the command will search the binding table for a destination. If this is the user's preferred method, then it is necessary to add the required destination to the local binding table in advance. After that, it will not be necessary to fill in the `<address>` parameter.
	`OK`
	
	or
	
	`ERROR:<errorcode>`
	
	`<errorcode>` (See 3. List of Status codes.)
	
	Default response is expected to show the status of the required operation on the target.

Table 2.101. +CCMVTOSAT – Color Control Move to Saturation Command

Execute Command	Response
AT+CCMVTOSAT:<address>,<EP>,<SendMode>,<Sat>,<TransTime> AT+CCMVTOSAT:,,,<Sat>,<TransTime> <address> — 16-bit hexadecimal number, which is the Node ID of a remote device if the command is sent directly to a node, or a group ID if the command is sent to a group. <EP> — 8-bit hexadecimal number, which is the endpoint of a remote device. Valid endpoint addresses are: 0x01 to 0xF0. <SendMode> — A Boolean type to choose transmission mode ('0' sends the command directly; '1' sends the command to a group). <Sat> — 8-bit hexadecimal number representing saturation. <TransTime> — 16-bit hexadecimal number. The Transition time field specifies, in a tenth of a second, the time that should be taken to perform the step. A step is a change in the device's saturation of 'Step size' units. If the address is a Node ID, set SendMode to be 0, or the address will be recognized as a group ID. If destination address, endpoint, and SendMode are not given (",” cannot be omitted), the command will search the binding table for a destination. If this is the user's preferred method, then it is necessary to add the required destination to the local binding table in advance. After that, it will not be necessary to fill in the <address> parameter.	OK or ERROR:<errorcode> <errorcode> (See 3. List of Status codes.) Default response is expected to show the status of the required operation on the target.

Table 2.102. +CCMVSAT – Color Control Move Saturation Command

Execute Command	Response

`AT+CCMVSAT:<address>,<EP>,<SendMode>,<Mode>,<Rate>`
 	
`AT+CCMVSAT:,,,<Mode>,<Rate>`
 	
`<NodeID>` — 16-bit hexadecimal number representing the target's Node ID.
 	
`<EP>` — 8-bit hexadecimal number, which is the endpoint of a remote device. Valid endpoint addresses are: 0x01 to 0xF0.
 	
`<SendMode>` — A Boolean type to choose transmission mode ('0' sends the command directly; '1' sends the command to a group).
 	
`<Mode>` — 8-bit hexadecimal number (0x01 means Up; 0x03 means Down).
 	
`<Rate>` — 8-bit hexadecimal number.
 	
If the address is a Node ID, set SendMode to be 0, or the address will be recognized as a group ID.
 	
If destination address, endpoint, and SendMode are not given (“,” cannot be omitted), the command will search the binding table for a destination. If this is the user's preferred method, then it is necessary to add the required destination to the local binding table in advance. After that, it will not be necessary to fill in the `<address>` parameter.
	`OK`
	
	or
	
	`ERROR:<errorcode>`
	
	`<errorcode>` (See 3. List of Status codes.)
	
	Default response is expected to show the status of the required operation on the target.

Table 2.103. +CCSTEPSAT – Color Control Step Saturation Command

Execute Command	Response

`AT+CCSTEPSAT:<address>,<EP>,<SendMode>,<StepMode>,<StepSize>,<TransTime>`
 	
`AT+CCSTEPSAT:<StepMode>,<StepSize>,<TransTime>`
 	
`<address>` — 16-bit hexadecimal number, which is the Node ID of a remote device if the command is sent directly to a node, or a group ID if the command is sent to a group.
 	
`<EP>` — 8-bit hexadecimal number, which is the endpoint of a remote device. Valid endpoint addresses are: 0x01 to 0xF0.
 	
`<SendMode>` — A Boolean type to choose transmission mode ('0' sends the command directly; '1' sends the command to a group).
 	
`<StepMode>` — 8-bit hexadecimal number (0x01 means Up; 0x03 means Down).
 	
`<StepSize>` — 8-bit hexadecimal number. The change to be added to (or subtracted from) the current value of the device's saturation.
 	
`<TransTime>` — 8-bit hexadecimal number. The Transition time field specifies, in a tenth of a second, the time that should be taken to perform the step. A step is a change in the device's saturation of 'Step size' units.
 	
If the address is a Node ID, set SendMode to be 0, or the address will be recognized as a group ID.
 	
If destination address, endpoint, and SendMode are not given (“,” cannot be omitted), the command will search the binding table for a destination. If this is the user's preferred method, then it is necessary to add the required destination to the local binding table in advance. After that, it will not be necessary to fill in the `<address>` parameter.
	`OK`
	
	or
	
	`ERROR:<errorcode>`
	
	`<errorcode>` (See 3. List of Status codes.)
	
	Default response is expected to show the status of the required operation on the target.

Table 2.104. +CCMVTOCT – Color Control Move to Color Temperature Command

Execute Command	Response

`AT+CCMVTOCT:<address>,<EP>,<SendMode>,<CT>,<TransTime>`
 	
`AT+CCMVTOCT:,,,<CT>,<TransTime>`
 	
`<address>` — 16-bit hexadecimal number, which is the Node ID of a remote device if the command is sent directly to a node, or a group ID if the command is sent to a group.
 	
`<EP>` — 8-bit hexadecimal number, which is the endpoint of a remote device. Valid endpoint addresses are: 0x01 to 0xF0.
 	
`<SendMode>` — A Boolean type to choose transmission mode ('0' sends the command directly; '1' sends the command to a group).
 	
`<CT>` — 16-bit hexadecimal number, Color temperature.
 	
`<TransTime>` — 16-bit hexadecimal number. The Transition time field specifies, in a tenth of a second, the time that should be taken to perform the step. A step is a change in the device's saturation of 'Step size' units.
 	
If the address is a Node ID, set SendMode to be 0, or the address will be recognized as a group ID.
 	
If destination address, endpoint, and SendMode are not given (",," cannot be omitted), the command will search the binding table for a destination. If this is the user's preferred method, then it is necessary to add the required destination to the local binding table in advance. After that, it will not be necessary to fill in the `<address>` parameter.
	`OK`
	
	or
	
	`ERROR:<errorcode>`
	
	`<errorcode>` (See 3. List of Status codes.)
	
	Default response is expected to show the status of the required operation on the target.

Table 2.105. +CCMVTOHUS – Color Control Move to Hue Saturation Command

Execute Command	Response

`AT+CCMVTOHUS:<address>,<EP>,<SendMode>,<Hue><Sat>,<TransTime>`
 	
`AT+CCMVTOHUS:,,,<Hue>,<Sat>,<TransTime>`
 	
`<address>` — 16-bit hexadecimal number, which is the Node ID of a remote device if the command is sent directly to a node, or a group ID if the command is sent to a group.
 	
`<EP>` — 8-bit hexadecimal number, which is the endpoint of a remote device. Valid endpoint addresses are: 0x01 to 0xF0.
 	
`<SendMode>` — A Boolean type to choose transmission mode ('0' sends the command directly; '1' sends the command to a group).
 	
`<Hue>` — 8-bit hexadecimal number.
 	
`<Sat>` — 8-bit hexadecimal number representing saturation.
 	
`<TransTime>` — 16-bit hexadecimal number. The Transition time field specifies, in a tenth of a second, the time that should be taken to perform the step. A step is a change in the device's saturation of 'Step size' units.
 	
Check the ZigBee Cluster Library for more field explanations.
 	
If the address is a Node ID, set SendMode to be 0, or the address will be recognized as a group ID.
 	
If destination address, endpoint, and SendMode are not given (“,” cannot be omitted), the command will search the binding table for a destination. If this is the user's preferred method, then it is necessary to add the required destination to the local binding table in advance. After that, it will not be necessary to fill in the `<address>` parameter.
	`OK`
	
	or
	
	`ERROR:<errorcode>`
	
	`<errorcode>` (See 3. List of Status codes.)
	
	Default response is expected to show the status of the required operation on the target.

Thermostat – Client

Table 2.106. +TSTATSET – Send A Thermostat Set Command

Execute Command	Response

`AT+TSTATSET:<address>,<EP>,<SendMode>,<SetMode>,<Amount>`
 	
`AT+TSTATSET: ,,,<SetMode>,<Amount>`
 	
`<address>` — 16-bit hexadecimal number, which is the Node ID of a remote device if the command is sent directly to a node, or a group ID if the command is sent to a group.
 	
`<EP>` — 8-bit hexadecimal number, which is the endpoint of a remote device. Valid endpoint addresses are: 0x01 to 0xF0.
 	
`<SendMode>` — A Boolean type to choose transmission mode ('0' sends the command directly; '1' sends the command to a group).
 	
`<SetMode>` — 8-bit hexadecimal number, specifies which setpoint is to be configured as following:
 	
0x00: Heat (adjust Heat Setpoint)
 	
0x01: Cool (adjust Cool Setpoint)
 	
0x02: Both (adjust Heat Setpoint and Cool Setpoint)
 	
`<Amount>` — 8-bit hexadecimal number representing the amount the setpoint(s) are to be a increased (or decreased). For example, +01 means increase by 0.1 °C
 	
If the address is a Node ID, set SendMode to be 0, or the address will be recognized as a group ID.
 	
If destination address, endpoint, and SendMode are not given (“,” cannot be omitted), the command will search the binding table for a destination. If this is the user's preferred method, then it is necessary to add the required destination to the local binding table in advance. After that, it will not be necessary to fill in the `<address>` parameter.
	`OK`
	
	or
	
	`ERROR:<errorcode>`
	
	`<errorcode>` (See 3. List of Status codes.)
	
	Default response is expected to show the status of the required operation on the target.

Door Lock – Client

Table 2.107. +DRLOCK – Issue A Lock/Unlock Door Command

Execute Command	Response

`AT+DRLOCK:<address>,<EP>,<SendMode>, <Lock/Unlock>`
 	
`AT+DRLOCK: ,,, <Lock/Unlock>`
 	
`<address>` — 16-bit hexadecimal number. It shall be the Node ID of a remote device if the command is sent directly to a node, or a group ID if the command is sent to a group.
 	
`<EP>` — 8-bit hexadecimal number, which is the endpoint of a remote device. Valid endpoint addresses are: 0x01 to 0xF0.
 	
`<SendMode>` — A Boolean type to choose transmission mode ('0' sends the command directly; '1' sends the command to a group).
 	
`<Lock/Unlock>` — A Boolean type to choose transmission mode ('0' means Unlock; '1' means Lock).
 	
If the address is a Node ID, set SendMode to be 0, or the address will be recognized as a group ID.
 	
If destination address, endpoint, and SendMode are not given (“,” cannot be omitted), the command will search the binding table for a destination. If this is the user's preferred method, then it is necessary to add the required destination to the local binding table in advance. After that, it will not be necessary to fill in the `<address>` parameter.
	`OK`
	
	or
	
	`ERROR:<errorcode>`
	
	`<errorcode>` (See 3. List of Status codes.)
	
	If the destination node supports Door Lock cluster, a door lock/unlock response is expected to report the status and the following prompts may show, respectively:
	
	`DRLOCRSP:<nodeID>,<ep>,<status>`
	
	or
	
	`DRUNLOCKRSP:<nodeID>,<ep>,<status>`
	
	If the destination node does not support the door lock/unlock command, a Default Response will be received and DFTREP prompt shall show.

Window Covering – Client

Table 2.108. +WNPGSETPT – Issue A Window Program Setpoint Command to Target

Execute Command	Response

`AT+WNPGSETPT:<address>,<EP>,<SendMode>,<Lift/Tile>,<SetpointIndex>,<SetpointValue>`
 	
`AT+WNPGSETPT:,,,<Lift/Tile>,<SetpointIndex>, <SetpointValue>`
 	
`<address>` — 16-bit hexadecimal number, which is the Node ID of a remote device if the command is sent directly to a node, or a group ID if the command is sent to a group.
 	
`<EP>` — 8-bit hexadecimal number, which is the endpoint of a remote device. Valid endpoint addresses are: 0x01 to 0xF0.
 	
`<SendMode>` — A Boolean type to choose transmission mode ('0' sends the command directly; '1' sends the command to a group).
 	
`<Lift/Tile>` — A Boolean type specifies the command as following:
 	
0: Go to lift percentage
 	
1: Go to tile percentage
 	
`<SetpointIndex>` — 8-bit hexadecimal number representing the index of Lift/Tile Setpoint.
 	
`<SetpointValue>` — 16-bit hexadecimal number representing the setpoint value.
 	
If the address is a Node ID, set SendMode to be 0, or the address will be recognized as a group ID.
 	
If destination address, endpoint, and SendMode are not given (“,” cannot be omitted), the command will search the binding table for a destination. If this is the user's preferred method, then it is necessary to add the required destination to the local binding table in advance. After that, it will not be necessary to fill in the `<address>` parameter.
	`OK`
	
	or
	
	`ERROR:<errorcode>`
	
	`<errorcode>` (See 3. List of Status codes.)
	
	Default response is expected to show the status of the required operation on the target.

Table 2.109. +WNSETPT – Issue A Window Go To Lift/Tile Setpoint Command to Target

Execute Command	Response

`AT+WNSETPT:<address>,<EP>,<SendMode>, <Lift/Tile>,<Setpoint>`
 	
`AT+WNSETPT:,,, <Lift/Tile>,<Setpoint>`
 	
`<address>` — 16-bit hexadecimal number, which is the Node ID of a remote device if the command is sent directly to a node, or a group ID if the command is sent to a group.
 	
`<EP>` — 8-bit hexadecimal number, which is the endpoint of a remote device. Valid endpoint addresses are: 0x01 to 0xF0.
 	
`<SendMode>` — A Boolean type to choose transmission mode ('0' sends the command directly; '1' sends the command to a group).
 	
`<Lift/Tile>` — A Boolean type specifies the command as following:
 	
0: Go to lift percentage
 	
1: Go to tile percentage
 	
`<Setpoint>` — 8-bit hexadecimal number representing the index of Lift/Tile Setpoint.
 	
If the address is a Node ID, set SendMode to be 0, or the address will be recognized as a group ID.
 	
If destination address, endpoint, and SendMode are not given (“,” cannot be omitted), the command will search the binding table for a destination. If this is the user's preferred method, then it is necessary to add the required destination to the local binding table in advance. After that, it will not be necessary to fill in the `<address>` parameter.
	`OK`
	
	or
	
	`ERROR:<errorcode>`
	
	`<errorcode>` (See 3. List of Status codes.)
	
	Default response is expected to show the status of the required operation on the target.

Table 2.110. +WNVALUE – Issue A Window Go To Lift/Tile Value Command to Target

Execute Command	Response

`AT+WNVALUE:<address>,<EP>,<SendMode>, <Lift/Tile>,<Value>`
 	
`AT+WNVALUE:,,, <Lift/Tile>,<Value>`
 	
`<address>` — 16-bit hexadecimal number, which is the Node ID of a remote device if the command is sent directly to a node, or a group ID if the command is sent to a group.
 	
`<EP>` — 8-bit hexadecimal number, which is the endpoint of a remote device. Valid endpoint addresses are: 0x01 to 0xF0.
 	
`<SendMode>` — A Boolean type to choose transmission mode ('0' sends the command directly; '1' sends the command to a group).
 	
`<Lift/Tile>` — A Boolean type specifies the command as following:
 	
0: Go to lift percentage
 	
1: Go to tile percentage
 	
`<Value>` — 16-bit hexadecimal number representing the value of movement.
 	
If the address is a Node ID, set SendMode to be 0, or the address will be recognized as a group ID.
 	
If destination address, endpoint, and SendMode are not given (“,” cannot be omitted) the command will search the binding table for a destination. If this is the user's preferred method, then it is necessary to add the required destination to the local binding table in advance. After that, it will not be necessary to fill in the `<address>` parameter.
	`OK`
	
	or
	
	`ERROR:<errorcode>`
	
	`<errorcode>` (See 3. List of Status codes.)
	
	Default response is expected to show the status of the required operation on the target.

Table 2.111. +WNPCTG – Issue A Window Go To Lift/Tile Percentage Command to Target

Execute Command	Response

`AT+WNPCTG:<address>,<EP>,<SendMode>, <Lift/Tile>,<Percentage>`
 	
`AT+WNPCTG:,,,<SetMode>,< Percentage >`
 	
`<address>` — 16-bit hexadecimal number, which is the Node ID of a remote device if the command is sent directly to a node, or a group ID if the command is sent to a group.
 	
`<EP>` — 8-bit hexadecimal number, which is the endpoint of a remote device. Valid endpoint addresses are: 0x01 to 0xF0.
 	
`<SendMode>` — A Boolean type to choose transmission mode ('0' sends the command directly; '1' sends the command to a group).
 	
`<Lift/Tile>` — a Boolean type specifies the command as following:
 	
0: Go to lift percentage
 	
1: Go to tile percentage
 	
`<Percentage>` — 8-bit hexadecimal number, which specifies the movement percentage.
 	
If the address is a Node ID, set SendMode to be 0, or the address will be recognized as a group ID.
 	
If destination address, endpoint, and SendMode are not given (“,” cannot be omitted), the command will search the binding table for a destination. If this is the user's preferred method, then it is necessary to add the required destination to the local binding table in advance. After that, it will not be necessary to fill in the `<address>` parameter.
	`OK`
	
	or
	
	`ERROR:<errorcode>`
	
	`<errorcode>` (See 3. List of Status codes.)
	
	Default response is expected to show the status of the required operation on the target.

Table 2.112. +WNCTRL – Issue A Window Control Command to Target (Open, Close or Stop Movement)

Execute Command	Response

`AT+WNCTRL:<address>,<EP>,<SendMode>,<Control>`
 	
`AT+WNCTRL:,,, <Control>`
 	
`<address>` — 16-bit hexadecimal number, which is the Node ID of a remote device if the command is sent directly to a node, or a group ID if the command is sent to a group.
 	
`<EP>` — 8-bit hexadecimal number, which is the endpoint of a remote device. Valid endpoint addresses are: 0x01 to 0xF0.
 	
`<SendMode>` — A Boolean type to choose transmission mode ('0' sends the command directly; '1' sends the command to a group).
 	
`<Control>` — A decimal number specifying the adjusting movement of the target window:
 	
0: Window open or move up
 	
1: Window closing or move down
 	
2: Window stop any adjusting movement
 	
If the address is a Node ID, set SendMode to be 0, or the address will be recognized as a group ID.
 	
If destination address, endpoint, and SendMode are not given (“,” cannot be omitted), the command will search the binding table for a destination. If this is the user's preferred method, then it is necessary to add the required destination to the local binding table in advance. After that, it will not be necessary to fill in the `<address>` parameter.
	`OK`
	
	or
	
	`ERROR:<errorcode>`
	
	`<errorcode>` (See 3. List of Status codes.)
	
	Default response is expected to show the status of the required operation on the target.

2.4.4 CIE Specified Command Set

Table 2.113. +WARNING – Issue A Start Warning Command to Target (Warning Device)

<table>
<tr><th>Execute Command</th><th>Response</th></tr>
<tr><td>
AT+WARNING:<Address>,<EP>,<SendMode>,<Warning>,<Warning Duration>

<Address> — 16-bit hexadecimal number, which is the Node ID of a remote device if the command is sent directly to a node, or a group ID if the command is sent to a group.

<EP> — 8-bit hexadecimal number, which is the endpoint of a remote device. Valid endpoint addresses are: 0x01 to 0xF0.

<SendMode> — A Boolean type to choose transmission mode ('0' sends the command directly; '1' sends the command to a group).

<Warning> — 8-bit hexadecimal number, where:
<table>
<tr><th>4-bit</th><th>2-bit</th><th>2-bit</th></tr>
<tr><td>Warning Mode</td><td>Strobe</td><td>Reserved</td></tr>
</table>
Warning Mode:
<table>
<tr><td>0</td><td>Stop (no warning)</td></tr>
<tr><td>1</td><td>Burglar</td></tr>
<tr><td>2</td><td>Fire</td></tr>
<tr><td>3</td><td>Emergency</td></tr>
<tr><td>4-15</td><td>Reserved</td></tr>
</table>
Strobe:
<table>
<tr><td>0</td><td>No strobe</td></tr>
<tr><td>1</td><td>Use strobe in parallel to warning</td></tr>
<tr><td>2-3</td><td>Reserved</td></tr>
</table>
<Warning Duration> — 16-bit hexadecimal number, which is the requested duration of warning, in seconds. If both Strobe and Warning Mode are '0', this field will be ignored.

If the address is a Node ID, set SendMode to be '0', or the address will be recognized as a group ID.
</td><td>
OK

or

ERROR:<errorcode>

<errorcode> (See 3. List of Status codes.)

A default response is expected to show the status of the required operation on the target.
</td></tr>
</table>

Table 2.114. +SQUAWK– Issue A Squawk Command to Target (Warning Device)

<table>
<tr><th>Execute Command</th><th>Response</th></tr>
<tr><td>

`AT+SQUAWK:<Address>,<EP>,<SendMode>,<Squawk>`

`<Address>` — 16-bit hexadecimal number, which is the Node ID of a remote device if the command is sent directly to a node or it will be a group ID if the command is sent to a group.

`<EP>` — 8-bit hexadecimal number, which is the endpoint of a remote device. Valid endpoint addresses are: 0x01 to 0xF0.

`<SendMode>` — A Boolean type to choose transmission mode ('0' sends the command directly; '1' sends the command to a group).

`<Squawk>` — 8-bit hexadecimal number, where:

<table>
<tr><th>4-bit</th><th>1-bit</th><th>1-bit</th><th>2-bit</th></tr>
<tr><td>Squawk Mode</td><td>Strobe</td><td>Reserved</td><td>Squawk Level</td></tr>
</table>

Squawk Mode:

<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>0</td><td>Notification sound for "System is armed"</td></tr>
<tr><td>1</td><td>Notification sound for "System is disarmed"</td></tr>
<tr><td>2-15</td><td>Reserved</td></tr>
</table>

Strobe:

<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>0</td><td>No strobe</td></tr>
<tr><td>1</td><td>Use strobe blink in parallel to squawk</td></tr>
</table>

Squawk Level:

<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>0</td><td>Low level sound</td></tr>
<tr><td>1</td><td>Medium level sound</td></tr>
<tr><td>2</td><td>High level sound</td></tr>
<tr><td>3</td><td>Very High level sound</td></tr>
</table>

</td><td>

`OK`

or

`ERROR:<errorcode>`

`<errorcode>` (See 3. List of Status codes.)

Default response is expected to show the status of the required operation on the target.

</td></tr>
</table>

Table 2.115. +ADDZENTRY – Add A Zone Table Entry To Local Zone Table

Execute Command	Response

`AT+ADDZENTRY:<Index>,<ZoneType>,<NodeIEEE>`
 	
`<Index>` — 8-bit hexadecimal number, which ranges from 00 to 1F. It is the index of a zone table entry. Entries from 00 to 0F are flash-maintained, while entries 10 to 1F are RAM-maintained.
 	
`<ZoneType>` — 16-bit hexadecimal number, which represents the zone type of the zone device. Check the ZigBee Cluster Library for more information.
 	
`<NodeIEEE>` — A Node EUI address.
 	
This command can be used to add a zone table entry into local zone table. It should be advised to use `AT+ZTABLE` to find and use a free entry. (An entry using only 'F' characters is a free entry).
	`OK`
	
	or
	
	`ERROR:<errorcode>`
	
	`<errorcode>`: the code explained in 3. List of Status codes.

Table 2.116. +DELZENTRY – Delete A Zone Table Entry From Local Zone Table

Execute Command	Response
`AT+DELZENTRY:<Index or EUI>` `<Index or EUI>` — 8-bit hexadecimal Zone ID (00 — 1F) or a device's EUI can be used to remove a zone entry from a local zone table.	`OK` or `ERROR:<errorcode>` the code explained in 3. List of Status codes. If the user attempts to remove a non-existing entry using the `EUI`, `NOTFOUNDZONE` prompt will show.

Table 2.117. +ZENTRY – Print A Zone Table Entry In Local Zone Table

Execute Command	Response

`AT+ZENTRY:<Index>`
 	
`<Index>` — 8-bit hexadecimal number, which ranges from 00 to 1F. This is the index of a zone table entry.
	`ZONEENTRY:<Index>,<ZoneType>,<NodeEUI>`
	
	`OK`
	
	or
	
	`ERROR:<errorcode>` (See 3. List of Status codes.)
	
	`<Index>` — 8-bit hexadecimal number, which ranges from 00 to 0F. This is the index of a zone table entry
	
	`<Zone Type>` — 16-bit hexadecimal number, which represents the zone type of the zone device. Check the ZigBee Cluster Library for more information.
	
	`<Node IEEE>` — EUI address of a node which has been enrolled in CIE's Zone.

Table 2.118. +ZTABLE – Print Local Zone Table

Execute Command	Response
AT+ZTABLE Use this command to show local zone table information.	ZONEENTRY:<Index>,<ZoneType>,<NodeEUI> … OK or ERROR:<errorcode> (See 3. List of Status codes.) <Index> — 8-bit hexadecimal number, which ranges from 00 to 1F. This is the index of a zone table entry <Zone Type> — 16-bit hexadecimal number, which represents the zone type of the zone device. Check the ZigBee Cluster Library for more information. <Node IEEE> — EUI address of a node which has been enrolled in CIE's Zone.

Table 2.119. +ZCLEAR – Delete A Zone Table Entry From Local Zone Table

Execute Command	Response
AT+ZCLEAR Use this command to clear a local zone table. All entries will be deleted. When the CIE leaves the network, the local zone table will be cleared.	OK or ERROR:<errorcode> the code explained in 3. List of Status codes.

Table 2.120. +PRINTMZTYPE – Print Local Table of Manufacture Defined Zone Type

Execute Command	Response
AT+PRINTMZTYPE Using this command to print CIE supported manufacture defined zone type. There are six entries which can be used to add manufacture defined zone type. These entries are not self-maintained during power circle. Check Appendix 2. Zone Enrollment for more information.	MZONETYPE 00:<ZoneType> MZONETYPE 01:<ZoneType> MZONETYPE 02:<ZoneType> MZONETYPE 03:<ZoneType> MZONETYPE 04:<ZoneType> MZONETYPE 05:<ZoneType> OK or ERROR:<errorcode> (See 3. List of Status codes.)

Table 2.121. +MZONETYPE – Add User Defined Zone Type in Local Zone Type Table

Execute Command	Response
`AT+MZONETYPE:<Index>,<ZoneType>` `<Index>` — 8-bit hexadecimal number, which is the index of the local table of supported user-specified zone types. Range from 00 to 05. `<ZoneType>` — 16-bit hexadecimal number, which represents the zone type. Use this command to add a user-defined zone type, which is used to filter out unwanted zone type during enrollment. S61 can be set to check Zone Type before accepting the enroll request. Check 4. S-Registers and Appendix 2. Zone Enrollment for more information.	`OK` or `ERROR:<errorcode>` (See 3. List of Status codes.)

Table 2.122. +PRINTZMAP – Print Local Zone Map

Execute Command	Response

`AT+PRINTZMAP`
 	
Use this command to print the local Zone map. This zone map information is used to respond to the `getZoneIdMap` command sent by ACE client.
	`ZMAPSECTION 00:<Section Value>`
	
	…
	
	`ZMAPSECTION 0F:<Section Value>`
	
	`OK`

Table 2.123. +EDITZMAP – Edit Local Zone Map

Execute Command	Response
`AT+EDITZMAP:<SectionIndex>,<Section>` `<SectionIndex>` — 8-bit hexadecimal, which ranges from 02 to 0F, and represents the index of a section of local zone map. `<Section>` — 16-bit hexadecimal, when set by the user, the module will respond to ACE's request with the edited zone map information. Use this command to print the local Zone map. This zone map information is used to respond to the `getZoneIDMap` command sent by the ACE client.	`OK`

Table 2.124. +ZINFORSP – Construct And Send A Get Zone Information Response

Execute Command	Response
AT+ZINFORSP:<Address>,<EP>,<SendMode>, <ZoneID>,<ZoneType>, <EUI>,<SequenceNumber> <Address> — 16-bit hexadecimal, which will be the Node ID of a remote device. <EP> — 8-bit hexadecimal, which is the endpoint of a remote device. <SendMode> — Send '0' for a unicast command. <ZoneID> 8-bit hexadecimal number representing the zone ID. This zone ID should be used to retrieve corresponding zone entry information. If there is no such a zone in the zone table, use 0xFFFF for Zone Type and 0xFFFFFFFFFFFFFFFF for EUI. <ZoneType> — 16-bit hexadecimal, which represents zone type. <EUI> — the zone device's IEEE address. <SequenceNumber> — 8-bit hexadecimal sequence number, which should be the same as the sequence number contained in <GETZONEINFO>. Use this command to respond to a get zone information command when a GETZONEINFO prompt shows.	The DFTREP prompt is expected after sending this command.

Table 2.125. +ARMPRINT – Show CIE Arm Mode

Execute Command	Response
AT+ARMPRINT Use this command to check the local CIE ARM mode.	ARMMODE:<Arm Mode> OK or ERROR:<errorcode> (See 3. List of Status codes.) According to the ZigBee Cluster Library, the Arm Mode Field Value has following meanings: 0x00 : Disarm 0x01 : Arm Day/Home Zones Only 0x02 : Arm Night/Sleep Zones Only 0x03 : Arm All Zones

Table 2.126. +ZTEST – Place A Remote Zone Server Device into Test Mode

Execute Command	Response
`AT+ZTEST:<Address>,<EP>, <SendMode>,<TestModeDuration>, <CurrentZoneSensitivityLevel>` `<Address>` — 16-bit hexadecimal number, which is the Node ID of a remote device. `<EP>` — 8-bit hexadecimal number, which is the endpoint of a remote device. `<SendMode>` — A Boolean type to choose transmission mode ('0' sends the command directly; '1' sends the command to a group). `<TestModeDuration>` — 8-bit unsigned hexadecimal, which specifies the number of seconds the IAS Zone server will operate in test mode. `<CurrentZoneSensitivityLevel>` — 8-bit unsigned hexadecimal, which specifies the sensitivity level the IAS Zone server will use.	`OK` or `ERROR:<errorcode>` (See 3. List of Status codes.)

Table 2.127. +ZNORMAL – Set A Remote Zone Server Device To Normal Working Mode

Execute Command	Response
`AT+ZNORMAL:<Address>,<EP>, <SendMode>` `<Address>` — 16-bit hexadecimal, which is the Node ID of a remote device. `<EP>` — 8-bit hexadecimal number, which is the endpoint of a remote device. `<SendMode>` — A Boolean type to choose transmission mode ('0' sends the command directly; '1' sends the command to a group).	`OK` or `ERROR:<errorcode>` (See 3. List of Status codes.)

2.4.5 Tunnel Cluster Commands

Table 2.128. +TNLOPEN – Send An Open Tunnel Request To The Target

Execute Command	Response

`AT+TNLOPEN:<NodeID>,<EP>`
 	
`<NodeID>` — 16-bit hexadecimal number representing the target's node ID.
 	
`<EP>` — 8-bit hexadecimal number representing the endpoint number.
	`OK`
	
	`TNLOPEN:<TunnelID>,<Status>`
	
	or
	
	`ERROR:<errorcode>`
	
	`<TunnelID>` — 16-bit hexadecimal number representing the tunnel ID.
	
	`<Status>` — 8-bit hexadecimal number. Rrefer to Table 1.1 Clusters Supported by the Combined Interface on page 2 for additional information.
	
	`<errorcode>` represents the error code. If the errorcode is 01, 02, 03, 04, see the table below for the error message meaning. For other errorcodes, the meaning can be found in 3. List of Status codes.
	
	When the TunnelID is FFFF, check if the Status is 00.

Table 2.129. Tunnel Status Code

Value	Description
0x00	Success
0x01	Busy
0x02	No more tunnel ID
0x03	Protocol not supported
0x04	Flow control not supported
0x05	Reserved

Table 2.130. +TNLCLOSE – Send A Close Tunnel Request To The Target

Execute Command	Response
`AT+TNLCLOSE:<TunnelID>` <TunnelID> — 16-bit hexadecimal number representing the tunnel ID to be closed. Use this command to send a *CloseTunnel* command to a Tunnelling server. This command can only be used to send a Close Tunnel Request. It cannot close a local server tunnel.	`OK` or `ERROR:<errorcode>` (See 3. List of Status codes.) If Tunnelling client does not send a *CloseTunnel* command. The tunnel will be closed automatically by the server when Tunnel life time runs out. If errorcode is 0x05, check if the input tunnelID is correct and the tunnel exists.

Table 2.131. +TNLSEND– Send Tunnel Data To The Target

Execute Command	Response

`AT+TNLSEND:<TunnelID>,<Length>`
 	
`<TunnelID>` — 16-bit hexadecimal number representing the Tunnel ID.
 	
`<Length>` — 8-bit hexadecimal number indicating the length of the transmitted tunnel data.
 	
After issuing the command (using \r), the prompt “>” will appear. Input binary data. For example:
 	
`AT+TNLSEND:5747,06`
 	
`>123456`
 	
The binary data will be counted using `<Length>`. Once the length is reached, data will be sent immediately through the tunnel.
	`<data being entered>`
	
	or
	
	`ERROR:<errorcode>`
	`<errorcode>` (See 3. List of Status codes.)
	
	If the errorcode is 0x05, check if the input tunnelID is correct and that the tunnel exists.
	
	If data is successfully sent to a wrong address or the tunnel has been closed (time out), the following prompt will show:
	
	`TNLERROR:<TunnelID>,<TunnelError>`
	
	`<TunnelID>` — 16-bit hexadecimal number indicating the tunnel which is used for sending data and incur error.
	
	`<TunnelError>` — 8-bit hexadecimal number indicating the reason of error.
	
	0x00: No such tunnel (The TransferData command contains a TunnelID of a non-existent tunnel).
	
	0x01: Wrong device (The TransferData command contains a TunnelID that does not match the device sending the data).
	
	Use this command to send tunnel data to the target.

Table 2.132. +TNLSTATUS – Check Local Tunnel Status

Execute Command	Response

`AT+TNLSTATUS`
 	
Check if there is any opened tunnel on a local server and client tunnel tables.
 	
This firmware only supports one client tunnel entry and one server tunnel entry.
	`TNLC 00: <EUI>,<ClientEP>,<SourceEP>,<TunnelID>`
	
	`TNLS 00:<EUI>,<ClientEP>,<SourceEP>,<TunnelID>`
	
	`OK`
	
	`<EUI>` is the EUI node number the tunnel has been built with.

2.4.6 Time Cluster Commands

Table 2.133. +SETTIME - Set Local Time

Execute Command	Response

`AT+SETTIME: <time>`
 	
`<time>` — 32-bit hexadecimal number representing time in UTC format (number of seconds since 01.01.2000 00:00)
	`OK`
	
	or
	
	`ERROR:<errorcode>`

Table 2.134. +GETTIME - Get The Local Time

Execute Command	Response

`AT+GETTIME`	`TIME:<time>`
	
	`OK`
	
	OR
	
	`ERROR:<errorcode>`
	
	Returns current local time in UTC format.
	
	`<time>` — 32-bit hexadecimal value representing the local time, number of seconds since 0 hours, 0 minutes, 0 seconds, on the 1st of January, 2000 UTC.

The CICIE firmware supports the Time Server cluster on endpoint 0x01. The `+SETTIME` and `+GETTIME` commands can be used to set and get local clock time. `AT+SETATR:000A,<AttrID>,<AttributeValue>` should be used to set other time server cluster attributes. (For example: *timeStatus*, *dstStart*, *dstEnd*, etc.) For more information, refer to Appendix 3. Time Server Clusters.

2.4.7 OTA Communication Commands

Table 2.135. +IMGNOTIFY — Notify OTA Client Of The Newly Available Image

Execute Command	Response

`AT+IMGNOTIFY:<NodeID>,<EP>, <SendMode>,<PayloadType>, <QueryJitter> [,<ManufCode>, <ImageType>,<FileVersion> ]`
 	
`<NodeID>` —16-bit hexadecimal number, network address of a remote device. It can be a node's network address, a group ID, or a broadcast address (e.g FFFC).
 	
`<EP>` —8-bit hexadecimal number, which is the endpoint of a remote device.
 	
`<SendMode>` — A digit set to choose transmission mode ('0' send the command directly, '1' sends the command to a group, '6' sends the command as a broadcast).
 	
`<PayloadType>` —8-bit hexadecimal number representing the Image Notify Command Payload Type. This parameter will determine and indicate which parameter will be included in the command.
 	
Payload Type / Description
0x00 — Query jitter
0x01 — Query jitter and manufacturer code
0x02 — Query jitter, manufacturer code, and image type
0x03 — Query jitter, manufacturer code, image type, and new file version
0x04 – 0xff — Reserved
 	
`<QueryJitter>` — 8-bit hexadecimal number ranging from 0x00 to 0x64. It indicates whether the client receiving the Image Notify Command should send in a *QueryNextImageRequest* command or not.
 	
`<ManufCode>` —16-bit hexadecimal number representing the manufacturer code. Manufacturer code when included in the command should contain the specific value that indicates a certain manufacturer. If the server intends for the command to be applied to all manufacturers, then the value should be omitted.
 	
`<ImageType>` —16-bit hexadecimal number. When imageType is included in the command, it should contain the specific value that indicates a certain file type. If the server intends for the command to be applied to all image type values, then the value "0xffff" should be used.
 	
`<FileVersion>` — 32-bit hexadecimal number. The OTA upgrade file version that the server tries to upgrade client devices in the network. If the server intends for the command to be applied to all file version values, then the wild card value "0xffffffff" should be used.
	`OK`
	
	or
	
	`ERROR:<errorcode>`
	
	This command does not have a response prompt.

Table 2.136. +UPGRADE — Instruct The Remote Device To Upgrade and Apply The Downloaded Image

Execute Command	Response

`AT+UPGRADE:<NodeID>,<EP>,<SendMode>, <ManufacturerCode>,<ImageType>,<FileVersion>, <CurrentTime>,<UpgradeTime>,<Seq>`
 	
`<NodeID>` —16-bit hexadecimal number, which is the network address of a remote device. It can be a node's network address, a group ID, or a broadcast address (e.g. FFFC).
 	
`<EP>` —8-bit hexadecimal number, which is the endpoint of a remote device.
 	
`<SendMode>` — A digit set to choose transmission mode ('0' sends a command directly, '1' sends a command to a group).
 	
`<ManufCode>` —16-bit hexadecimal number representing the manufacturer code. Manufacturer code when included in the command should contain the specific value indicating a certain manufacturer. If the server intends for the command to be applied to all manufacturers, then the value should be omitted.
 	
`<ImageType>` —16-bit hexadecimal number. When imageType is included in the command, it should contain the specific value that indicates a certain file type. If the server intends for the command to be applied to all image type values, then the wild card value "0xffff" should be used.
 	
`<FileVersion>` — 32-bit hexadecimal number. The OTA upgrade file version that the server tries to upgrade client devices to in the network. If the server intends for the command to be applied to all file version values, then the wild card value "0xffffffff" should be used.
 	
`<CurrentTime>` — 32-bit UTC time, which is the server current time.
 	
`<UpgradeTime>` — 32-bit UTC time, which instructs the upgrade time for the OTA client.
 	
`<Seq>` —8-bit hexadecimal number which is the sequence number of the response. The sequence number contained in the `UPGRADE REQ` prompt should be used.
	`OK`
	
	or
	
	`ERROR:<errorcode>`
	
	This command does not have a response prompt.

Table 2.137. +QIMGRSP — Query Next Image Response (Used to respond to Query next image request)

Response Command	Response

`AT+QIMGRSP:<NodeID>,<EP>,<SendMode>, <Status>[,<ManufCode>,<ImgType>, <FileVersion>,<ImgSize>], <Seq>`
 	
`<NodeID>` —16-bit hexadecimal number, which is the network address of a remote device. It can be a node's network address or a group ID.
 	
`<EP>` —8-bit hexadecimal number, which is the endpoint of a remote device.
 	
`<SendMode>` — A digit set to choose transmission mode ('0' sends a command directly, '1' sends a command to a group).
 	
`<Status>` —8-bit hexadecimal number representing the response status. If the status is successful, other fields are included. If the status is not successful, apply the following codes according to the status response.
 	
7E: Server is not authorized to upgrade the client.
 	
80: The command received is badly formatted. It usually means the command is missing certain fields or values included in the fields are invalid.
 	
98: No OTA upgrade image available for a particular client.
 	
More status code can be found in the OTA cluster specifications.
 	
If `<Status>` is `SUCCESS`, the following prompts will appear:
 	
`<ManufCode>` —16-bit hexadecimal number representing the manufacturer code. The value is received by the server in the *QueryNextImageRequest* command.
 	
`<ImgType>` —16-bit hexadecimal number. The value is received by the server in the *QueryNextImageRequest* command.
 	
`<FileVersion>` — 32-bit hexadecimal number indicating the image version that the client is required to install.
 	
`<ImgSize>` — 32-bit hexadecimal number. The value represents the total size of the image (in bytes) including the header and all sub-elements.
 	
`<Seq>` —8-bit hexadecimal number, which is the sequence number of the response. The user should use the sequence number contained in the IMGQUERY prompt.
	`OK`
	
	or
	
	`ERROR:<errorcode>`
	
	This command does not have a response prompt.
	
	This command is only used by the Host application to respond to a received *QueryNextImageRequest* command. The parameters like *ManufCode*, *ImgType*, and so on can be obtained from the *QueryNextImageRequest* command.

Table 2.138. +IMGBRSP — Image Block Response (It is used to respond to Image block request)

Response Command	Response

`AT+IMGBRSP:<NodeID>,<EP>,<SendMode>, <Status> [,Appended parameters], <Seq>`
 	
`<NodeID>` —16-bit hexadecimal number, which is the network address of a remote device. It can be a node's network address or a group ID
 	
`<EP>` —8-bit hexadecimal number, which is the endpoint of a remote device.
 	
`<SendMode>` — A digit set to choose transmission mode ('0' sends a command directly, '1' sends a command to a group).
 	
If this command is used to respond to a ZCL ImagePageRequest command, the user will set this field to 5. This *SendMode* value is specifically used for this command to disable the APS retry option (this is a requirement in OTA Cluster specifications).
 	
`<Status>` —8-bit hexadecimal number representing the response status. This command has three forms depending on `<Status>`. The status can be SUCCESS, ABORT or WAIT_FOR_DATA . If `<Status>` is `SUCCESS` (0x00), the [Appended parameters] will include the following prompts:
 	
`<ManufCode>` —16-bit hexadecimal number representing the manufacture code. The value is the same as the one included in Image Block/Page Request command.
 	
`<ImgType>` —16-bit hexadecimal number. The value is the same as the one included in Image Block/Page Request command.
 	
`<FileVersion>` — 32-bit hexadecimal number indicating the image version that the client is required to install. The version value may be lower than the current image version on the client if the server decides to perform a downgrade. The version value may be the same as the client's current version if the server decides to perform a reinstall. However, in general, the version value should be higher than the current image version on the client to indicate an upgrade.
 	
`<Fileoffset>` — 32-bit hexadecimal number representing the location of the data requested by the client. For most cases, the file offset value included in the (Image Block) response should be the same as the value requested by the client. For (unsolicited) Image Block responses generated as a result of Image Page Request, the file offset value shall be incremented to indicate the next data location.
 	
`<DataSize>` —8-bit hexadecimal number indicating the length of the image data (in bytes) included in the command. The value may be equal or smaller than the maximum data size value requested by the client.
 	
After the parameter `<DataSize>` has been given, use (\r). When the prompt ">" occurs, input `<ImageData>`. For example, if we want to send hex data "0xAB 0xBD 0x02 0x13 0x04 0x05" the raw hex values should be sent over the serial port.
	`><data being entered>`
	
	`OK`
	
	or
	
	`ERROR:<errorcode>`
	
	This command does not have a response prompt.
	
	This command is only used by the Host application for responding to the *ImageBlockRequest* command.
	
	The parameters like *ManufCode*, *ImgType*, and so on can be obtained when receiving the request command.

Response Command	Response
The data will be counted using <DataSize>. Once DataSize is reached, data will be sent immediately to the destination. <ImageData> is the actual OTA upgrade image data with the length equal to the data size value. If <Status> is WAIT_FOR_DATA (0x97), the [Appended parameters] will include the following part. <CurrentTime> — UTC time, which is the OTA server's current time. <RequestTime> — UTC time, which instructs the client to retry the request command and wait at least the request time value before trying again. The client shall wait at least the request time value before trying again. In the case of a sleepy device, it may choose to wait longer than the specified time in order to not disrupt its sleeping cycle. If the current time value is zero, the server does not support UTC time and the client will treat the request time value as offset time. If neither time value is zero, and the client supports UTC time, it will treat the request time value as UTC time. If the client does not support UTC time, it will calculate the offset time from the difference between the two time values. The offset indicates the minimum amount of time to wait in seconds. The UTC time indicates the actual time moment that needs to pass before the client should try again. <BlockRequestDelay> —16-bit hexadecimal number. This value is only included if the server supports rate limiting. If the server does not support rate limiting or does not wish to slow the client's download, the field shall be set to 0. If <Status> is ABORT (0x95), there will be no appended field. No matter which status code is used, the user needs to feed the <Seq>. <Seq> —8-bit hexadecimal number, this is the sequence number of the response. The user should use the sequence number contained in the IMGBREQ prompt.	

Table 2.139. +DFTRSP — Default Response

Response Command	Response

`AT+DFTRSP:<NodeID>,<EP>,<SendMode>, <ClusterID>,<CmdID>,<Status>,<Seq>`
 	
`<NodeID>` —16-bit hexadecimal number, network address of a remote device. It can be a node's network address or a group ID
 	
`<EP>` —8-bit hexadecimal number, endpoint of a remote device.
 	
`<SendMode>` — A digit set to choose transmission mode ('0' sends a command directly, '1' sends a command to a group).
 	
`<ClusterID>` —16-bit hexadecimal number representing the cluster ID. As this AT command is created for responding OTA cluster request, the user should use 0x0019 as cluster ID.
 	
`<CmdID>` —8-bit hexadecimal number. This is the command ID the default response is responding to. The codes are be 0x03 for imageBlockRequest or 0x04 for imagepageRequest.
 	
`<Status>` —8-bit hexadecimal number representing the response status.
 	
No matter which status code is used, the user need to feed the <Seq>.
 	
`<Seq>` —8-bit hexadecimal number, this is the sequence number of the response. The user should use the sequence number contained in the IMGBREQ or IMGPREQ prompt.
	`OK`
	
	or
	
	`ERROR:<errorcode>`
	
	This command does not have a response prompt.
	
	This command is specifically created for OTA cluster implementation.
	
	The only usage scenario is that the host loses the OTA file after an OTA upgrading progress has started. In this case, the user may apply this command to respond to an *ImageBlockRequest* or *ImagePageRequest* to stop an OTA upgrade process.
	
	An example command is shown below:
	
	`AT+DFTRSP:53F3,01,0,0019,04,98,06`

2.4.8 OTA Server Configuration Commands

Table 2.140. +GETPOLICY — Print The Policy Of Local OTA Server

Execute Command	Response
AT+GETPOLICY	OK OTAPOLICY:<QueryPolicy>,<UpgradeRequestPolicy> <QueryPolicy> — 8-bit hexadecimal 00: Upgrade if server has newer 01: Downgrade if server has older 02: Reinstall if server has same 03: No next version <UpgradeRequestPolicy> 00: Upgrade now 01: Upgrade in a few minutes 02: Ask me to upgrade later 03: Abort upgrade

Table 2.141. +SETPOLICY — Set The Policy Of Local OTA Server

Execute Command	Response
AT+SETPOLICY:<QueryPolicy>,<UpgradeRequestPolicy> <QueryPolicy> 00: Upgrade if server has newer 01: Downgrade if server has older 02: Reinstall if server has same 03: No next version <UpgradeRequestPolicy> 00: Upgrade now 01: Upgrade in two minutes 02: Ask me to upgrade later 03: Abort upgrade	OK This command does not have a response prompt.

Table 2.142. +SETIMGID — Set The OTA Server Image ID And Some Other Relevant Image Information

Execute Command	Response
`AT+SETIMGID:<ManufCode>,<ImgType>,<FileVersion>, <SpecificNodeId>,<ImageSize>` `<ManufCode>` — 16-bit hexadecimal number representing the manufacturer code. `<ImgType>` — 16-bit hexadecimal number representing the image type ID. `<FileVersion>` — 32-bit hexadecimal number indicating the image version of the server equipped firmware. `<SpecificNodeId>` — 16-bit hexadecimal used to specify this parameter to indicate the equipped OTA file is for a specific device. `<ImageSize>` — 32-bit hexadecimal number indicating the image size of the server-equipped firmware. This parameter is used in the *QueryNextImageResponse* command. This command should be used to set the image ID, which includes manufacture ID, image type ID, and firmware version. These parameters will be used to filter out the *queryNextImageRequest* from devices that are not supposed to be upgraded. These settings will take effect when the user sets S62.	`OK` This command does not have a response prompt.

Table 2.143. +GETIMGID — Print The OTA Server Image ID And Hardware Version

Execute Command	Response
AT+GETIMGID	OTAIMGID:<ManufCode>,<ImgType>,<FwVer>,<MinHwVer>, <MaxHwVer>,<HostDefinedNodeId>,<ImageSize> OK <ManufCode> — 16-bit hexadecimal. The value will be the device's assigned manufacturer code. <ImgType> — 16-bit hexadecimal. This is the image type ID of the OTA file. <FwVer> — 32-bit hexadecimal representing the server OTA image's firmware version. <MinHwVer> — 16-bit hexadecimal, representing the earliest hardware platform version this image should be used on. The high byte represents the version and the low byte represents the revision. <MaxHwVer>- 16-bit hexadecimal, the value representing the latest hardware platform version this image should be used on. The high byte represents the version and the low byte represents the revision. The <MinHwVer> and <MaxHwVer> can be set by using the AT+SETHWVER command. On the client side, the hardware version of the device should not be earlier than the minimum (hardware) version and should not be later than the maximum (hardware) version in order to run the OTA upgrade file. <HostDefinedNodeId> — 16-bit hexadecimal. This firmware provides a feature to allow the host to configure the client's Node ID in order to provide an image to a specific node (OTA client) for downloading. This parameter can be set by using the AT+SETIMGID command. <ImageSize> — 32-bit hexadecimal representing the server OTA image size.

Table 2.144. +SETHWVER — Set The Allowable OTA Client's Hardware Version Range

Execute Command	Response
AT+SETHWVER:<MinHwVer>,<MaxHwVer> <MinHwVer> — 16-bit hexadecimal representing the earliest hardware platform version this image should be used on. The high byte represents the version and the low byte represents the revision. <MaxHwVer>- 16-bit hexadecimal representing the latest hardware platform version this image should be used on. The high byte represents the version and the low byte represents the revision.	OK This command does not have a response prompt.

2.4.9 Poll Control Cluster Commands

Table 2.145. +FPSET — Set Local Flag and Timeout for Starting Fast Poll

Execute Command	Response
`AT+FPSET:<FastPollFlag>,<Timeout>` `<FastPollFlag>` — Boolean type, set to 01 for instructing the poll control server to do fast poll (00 is not for doing a fast poll). `<Timeout>`— 16-bit hexadecimal, the value represents the fast polling period. If the *FastPollFlag* is set to 00, the poll control server device will omit the set timeout and send a *checkInResponse* command. This command is used to set the local flag and timeout that will be used in the *checkInReponse* command. If S63 is used to disable the automatic check-in response, the CICIE will not send a response when it receives a check-in command.	`OK` This command does not have a response prompt.

Table 2.146. +LPINTVL — Set Long Poll Interval

Execute Command	Response

`AT+LPINTVL:<NodeID>,<EP>,<SendMode>,<LongPollInterval>`
 	
`<NodeID>` — 16-bit hexadecimal number, the network address of a remote device. It can be a node's network address or a group ID.
 	
`<EP>` — 8-bit hexadecimal number, which is the endpoint of a remote device.
 	
`<SendMode>` — A digit set to choose transmission mode ('0' sends a command directly, '1' sends a command to a group).
 	
`<LongPollInterval>` — 32-bit hexadecimal number representing the *LongPollInterval* to be set on the poll control server cluster.
 	
This command can be sent to a poll control server device to configure the long poll interval.
	`OK`
	
	or
	
	`ERROR:<error_code>`
	
	If the set value is not acceptable, the Poll Control Server will send back a default response of INVALID_VALUE (0x87).

Table 2.147. +FPSTOP — Stop Fast Poll

Execute Command	Response

`AT+FPSTOP:<NodeID>,<EP>,<SendMode>`
 	
This command can be sent to a poll control server device to quit fast poll mode.
 	
`<NodeID>` — 16-bit hexadecimal number, the network address of a remote device. It can be a node's network address or a group ID.
 	
`<EP>` — 8-bit hexadecimal number, which is the endpoint of a remote device.
 	
`<SendMode>` — A digit set to choose transmission mode ('0' sends a command directly, '1' sends a command to a group).
	`OK`
	
	or
	
	`ERROR:<error_code>`

Table 2.148. +SPINTVL — Set Short Poll Interval On Remote Device

Execute Command	Response

`AT+SPINTVL:<NodeID>,<EP>,<SendMode>,<ShortPollInterval>`
 	
`<NodeID>` — 16-bit hexadecimal number, the network address of a remote device. It can be a node's network address or a group ID.
 	
`<EP>` — 8-bit hexadecimal number, which is the endpoint of a remote device.
 	
`<SendMode>` — A digit set to choose transmission mode ('0' sends a command directly, '1' sends a command to a group).
 	
`<ShortPollInterval>` — 16-bit hexadecimal number representing the longPollInterval to be set on the poll control server cluster.
 	
This command can be sent to a poll control server device to set the short poll interval.
	`OK`
	
	or
	
	`ERROR:<error_code>`
	
	If the set value is not acceptable, the Poll Control Server will send back a default response of INVALID_VALUE (0x87).

3. List of Status codes

00	Everything OK — Success
01	Fatal Error
02	Unknown command
04	Invalid S-Register
05	Invalid parameter
06	Recipient could not be reached
07	Message was not acknowledged
0A	Message could not be sent
0C	Too many characters
0D	License Problem
0E	PAN could not be established because duplicate PAN ID was detected
0F	Fatal error initializing the network
10	Error bootloading
12	Fatal error initializing the stack
18	Node has run out of Buffers
19	Trying to write read-only register
20	Invalid password
25	Cannot form network
27	No network found
28	Operation cannot be completed if node is part of a PAN
2C	Error leaving the PAN
2D	Error scanning for PANs
33	No response from the remote bootloader
35	Binary data input timeout
39	MAC transmit queue is full
6C	Invalid binding index
66	Message is not sent to the target successfully
70	Invalid operation
72	The maximum number of in-flight messages has been reached
74	Message too long
80	ZDP invalid request type (if ZDO command has been sent)
81	ZDP device not found (if ZDO command has been sent)
82	ZDP invalid endpoint (if ZDO command has been sent)
83	ZDP not active (if ZDO command has been sent)
84	ZDP not supported (if ZDO command has been sent)
91	Operation only possible if joined to a PAN
93	Node is not part of a Network
94	Cannot join network

96	Mobile End Device Move to new Parent Failed
98	Cannot join ZigBee 2006 Network as Router
A1	More than eight broadcasts were sent within eight seconds
A6	Error in trying to encrypt at APS level. No link key entry in the table for the destination
AB	Trying to join, but no beacons could be heard
AC	Network key was sent in the clear when trying to join a secured network
AD	Did not receive Network Key
AE	No Link Key received
AF	Preconfigured Key required
B1	Not attached to a meter
B2	ESI end point not known
C5	NWK already present
C7	NWK table full
C8	NWK unknown device

Additional codes for ZigBee Cluster (ZCL) Command:

Note: ZCL commands are referred to in 2.4.3 Combined Interface Specified Command Set. The following code will show ZCL command responses (e.g. read attribute response, add group response, etc.), or default responses.

80	Malformed Command
81	Unsupported Cluster Command
82	Unsupported General Command
83	Unsupported Manufacturer Cluster Command
84	Unsupported Manufacturer General Command
85	Invalid Field
86	Unsupported Attribute
87	Invalid Value
88	Read Only
89	Insufficient Space
8A	Duplicate Exists
8B	Not Found
8C	Unreportable Attribute
8D	Invalid Data Type
C0	Hardware Failure
C1	Software Failure
C2	Calibration Error
C3	Unsupported Cluster

4. S-Registers

Most S-Registers of the modules can be read and written. All S-registers are stored in non-volatile memory and will keep their user-defined settings, unless reset to the factory defaults using the “AT&F” command. The S-Registers used in this firmware are summarized in the table below.

Table 4.1. S-Register Overview[1]

S-Register Overview		Local R/W
S00	Channel Mask	(•/•)
S01	Transmit Power Level	(•/•)
S02	Preferred PAN ID	(•/•)
S03	Preferred Extended PAN ID	(•/•)
S04	Local EUI	(•/-)
S05	Local NodeID	(•/-)
S06	Parent’s EUI	(•/-)
S07	Parent’s NodeID	(•/-)
S08	Network Key 1	(-/•)
S09	Link Key 1	(-/•)
S0A	Main Function 1	(•/•)
S0B	User Readable Name 1	(•/•)
S0C	Password 1	(-/•)
S0D	Device Information	(•/-)
S0F	Prompt Enable	(•/•)
S12	UART Setup	(•/•)
S48	Endpoint 1 Profile ID	(•/•)
S49	Endpoint 1 Device ID	(•/•)
S4A	Endpoint 1 Device Version	(•/•)
S4B	Endpoint 1 Input Cluster List	(•/•)
S4C	Endpoint 1 Output Cluster List	(•/•)
S4E	End Device Poll Timeout	(•/•)
S4F	MAC Timeout	(•/•)
S60	Manufacturer Code	(•/•)
S61	IAS Enrol Control	(•/•)
S62	OTA Upgrade Control	(•/•)
S63	Enable RSSI, LQI printing and Check-in response control	(•/•)
S64	Licence Code	(•/•)
S65	Protocol ID	(•/•)
S66	Server Tunnel ID	(•/•)

Note:

1. “1” signifies a password-protected register.

4.1 S-Registers for Network Setup

Table 4.2. S00 – Channel Mask

Description	Parameters
The 802.15.4 channel mask. Operations: R/W LOCAL Becomes effective: When joining, scanning, or establishing a PAN The channel mask does not affect the AT+JPAN command. Storage: Non-Volatile	XXXX Where XXXX represents a 16-bit decimal number enabling IEEE 802.15.4 channel numbers 11 to 26. Writing a bit to 1 enables a channel, and subsequently writing a bit to 0 disables a channel for scanning, joining, and establishing networks. For example, when setting S00 to 0001, only channel 11 will be used for all following operations. Range: 0001 - FFFF Factory Default: FFFF

Table 4.3. S01 – Transmit Power Level

Description	Parameters
The device's transmit power level in dBm. Operations: R/W LOCAL The output power of the"-PA" and "-LR" variants is higher than the value in S01. Refer to the respective hardware manuals for more information. Becomes effective: When joining or establishing a PAN Storage: Non-Volatile	snn Where snn represents a signed 8-bit decimal number. Range: ETRX3: 8 to -43 ETRX3 LRS Variants: -7 to -43 Actual values are: {8, 7, 6, 5, 4, 3, 2, 1, -1, -2, -3, -4, -5, -6, -7, -8, -9, -11, -12, -14, -17, -20, -26, -43}. Entering a value not on this list (such as –19) will result in the next lowest output power. Entering a value higher than three will automatically enable boost mode. Factory Default: 8 or (-7 for LRS)

Table 4.4. S02 – Preferred PAN ID

Description	Parameters
The 802.15.4 PAN ID. Operations: R/W LOCAL Becomes effective: When joining or establishing a PAN Two networks operating on the same channel with the same PAN ID, but a different EPANID are detected to be in conflict with each other. PAN ID conflicts are detected by the stack and resolved by one of the networks dynamically changing its PAN ID. The preferred PAN ID in S02 does not affect the AT+JPAN command. Storage: Non-Volatile	<PANID> Where <PANID> represents a 16-bit hexadecimal number. Range: 0000 – FFFF When establishing a PAN the coordinator will pick a random PAN ID if S02 is set to 0000. If set to any value between 0001 and FFFF, this number will be used as PAN ID instead, unless it is trying to use a PAN ID which already exists on the same channel. In this case, a random PAN ID will be used instead. When joining, only a PAN with the ID stored in S02 will be successful, unless S02 is set to 0000. In this case, the next best PAN which allows joining is used. Factory Default: 0000

Table 4.5. S03 – Preferred Extended PAN ID

Description	Parameters
The extended PAN ID. Operations: R/W LOCAL Becomes effective: When joining or establishing a PAN The EPANID is used for PAN ID conflict detection. It is recommended to use a random EPANID at all times. The preferred EPANID does not affect the AT+JPAN command. Storage: Non-Volatile	<EPANID> Where <EPANID> represents a 64-bit hexadecimal number. Range: 0000000000000000 – FFFFFFFFFFFFFFFF When establishing a PAN, the coordinator will pick a random EPANID if S03 is set to all 0's. If set to any other value, this number will be used as the EPANID instead. When the command AT+JN is used to join a PAN, the module will check if the PAN it finds has the EPID which matches the value in S03. If the value in S03 is set to 0, then there will be no EPID check. By default, S03 is set at all zeros Factory Default: 0000000000000000

Table 4.6. S04 – Local EUI64

Description	Parameters
The local node's unique EUI64 identifier. Operations: R LOCAL Storage: Non-Volatile	<EUI64> Range: 0000000000000000 – FFFFFFFFFFFFFFFF Factory Default: <unique number>

Table 4.7. S05 – Local 16-Bit NodeID

Description	Parameters
The local node's 16-bit NodeID. Reading this register while not associated with a network will result in an undefined return value. Operations: R LOCAL Storage: Non-Volatile	<NodeID> Range: 0000-FFFF Factory Default: n/a

Table 4.8. S06 – Parent's EUI64

Description	Parameters
The parent node's unique EUI64 identifier. The return value is undefined for nodes without parents (coordinator and nodes that are not joined to a network). Operations: R LOCAL Storage: Non-Volatile	<EUI64> Range: 0000000000000000 – FFFFFFFFFFFFFFFF Factory Default: n/a

Table 4.9. S07 – Parent's 16-Bit NodeID

Description	Parameters
The parent node's 16-bit NodeID. Operations: R LOCAL The return value is undefined for nodes without parents (coordinator and nodes that are not joined to a network). Storage: Non-Volatile	<NodeID> Range: 0000-FFFF Factory Default: n/a

Table 4.10. S08 – Network Key

Description	Parameters
The network key can be written using the password. The default password is "password". Operations: W LOCAL Write operation: ATS08=<key>:<password> Storage: Non-Volatile	Range: 0 to 2128-1 The 128-bit AES network key in hexadecimal representation (32 characters). This register has been set to all 0's (default) and a random network key is generated when establishing a PAN. Set when using silent join (AT+SJN). Factory Default: 00000000000000000000000000000000

Table 4.11. S09 – Trust Centre Link Key

Description	Parameters
The link key can be written using the password. The default password is "password". Operations: W LOCAL Write operation: ATS09=<key>:<password> Becomes effective: When joining or establishing a PAN Storage: Non-Volatile	Range: 0 to 2128-1 The 128-bit trust centre link key in hexadecimal representation (32 characters). When set to all 0s (default) a random trust centre link key is generated when establishing a PAN. Factor Default: 5A6967426565416C6C69616E63653039

4.2 S-Registers for Module Setup

Table 4.12. S0A – Main Function

Description	Parameters
Defines the behavior of the device. Operations: R/W LOCAL Write operation: `ATS0A=XXXX:<Password>` For security reasons, this register is password protected. The default password is “password”. Storage: Non-Volatile	`XXXX` Where XXXX represents a 16-bit hexadecimal number. Range: 0000 to FFFF Bit 0: Set to not allow other nodes to join the network. Use `AT+PJOIN` to allow other nodes to join a HA network. Bit 3: If this bit is set to true, CICIE will not accept a remote node attempting to rejoin its network using insecure rejoin. Bit 4: Set to Send Network key unencrypted to joining nodes Bit 7: Device uses S09 stored, pre-configured link key when joining. By default, S09 stores HA link key, which can be used in joining or silent join. The user can specify a link key and use it in forming/joining a network. Other Bits: Reserved Factory Default: 0001

Table 4.13. S0B – User Readable Name

Description	Parameters
Password protected user-defined name, which can be used to identify the node. Operations: R/W LOCAL Write operation: `ATS0B=<name>:<password>` Becomes effective: Instantly Storage: Non-Volatile	0123456789ABCDEF Name with up to 16 characters.

Table 4.14. S0C – Password

Description	Parameters
The local node’s password. Operations: W LOCAL Write operation: `ATS0C=<NEW>:<OLD>` Becomes effective: Instantly Storage: Non-Volatile	12345678 Eight case-sensitive characters (8 bytes). The password must have exactly eight characters. Factory Default: password

Table 4.15. S0D – Firmware Revision

Description	Parameters
String containing the module's order code and firmware revision. Operations: R LOCAL Storage: Non-Volatile	Text string: ccc…ccc Example: CI R300 Factory Default: N/A

Table 4.16. S0F – Prompt Enable

Description	Parameters
Enable/Disable default response prompts and prompts for message from other profiles. Operations: R/W LOCAL Becomes effective: Instantly Use of Bit 8 requires Bit 1 to be reset. Storage: Non-Volatile	XXXX Where XXXX represents a 16-bit hexadecimal number. Range: 0000 to FFFF Bit A: Set: Show received default response. Bit 9: Set: Show received raw messages. Used to display unhandled messages with non-HA profile IDs. All the other bits are reserved. Factory Default: 0400

Table 4.17. S48 – Endpoint 1 Profile ID

Description	Parameters
Register defining the Profile ID of the Node's endpoint 1. Operations: R/W LOCAL Becomes effective: Instantly Storage: Non-Volatile	XXXX The 16-bit Profile ID. Factory Default: 0000 If the default value is used, CICIE will use 0x0104 as default Profile ID.

Table 4.18. S49 – Endpoint 1 Device ID

Description	Parameters
Register defining the Device ID of the Node's endpoint 1. Operations: R/W LOCAL Becomes effective: Instantly Storage: Non-Volatile	XXXX The 16-bit device ID. Factory Default: 0000 If the default value is used, CICIE will use 0x0007 as default Device ID for endpoint 1.

Table 4.19. S4A – Endpoint 1 Device Version

Description	Parameters
Register defining the version of the device attached to the Node's endpoint 1. Operations: R/W LOCAL Becomes effective: Instantly Storage: Non-Volatile	00XX The 8-bit version preceded by two leading 0's. Factory Default: 0000 If default value is used, the CICIE will use 0x0002 as default Device version number for endpoint 1.

Table 4.20. S4B – Endpoint 1 Input Cluster List

Description	Parameters
Register defining the input cluster list of the device attached to the Node's endpoint 1. Operations: R/W LOCAL Becomes effective: Instantly Storage: Non-Volatile	[XXXX,XXXX,XXXX,XXXX,XXXX,XXXX, XXXX,XXXX,XXXX,XXXX,XXXX,XXXX] A list of a maximum of 4x16-bit cluster IDs separated by commas. The list may have any length from 0 to 4 clusters. Example: ATS4B=0000,0003 Factory Default: NULL If the default value is used, the CICIE ep1 will use its original configuration and support clusters listed in 2.4.3 Combined Interface Specified Command Set.

Table 4.21. S4C – Endpoint 1 Output Cluster List

Description	Parameters
Register defining the output cluster list of the device attached to the Node's endpoint 2. Operations: R/W LOCAL Becomes effective: Instantly Storage: Non-Volatile	[XXXX,XXXX,XXXX,XXXX,XXXX,XXXX, XXXX,XXXX,XXXX,XXXX,XXXX,XXXX] A list of a maximum of 21x16-bit cluster IDs separated by commas. The list may have any length from 0 to 21 clusters. Factory Default: NULL If the default value is used, the CICIE ep1 will use its original configuration and support clusters listed in 2.4.3 Combined Interface Specified Command Set.

Table 4.22. S4E – End Device Poll Timeout

Description	Parameters
Register in a parent node defining the amount of time a SED or ZED times out of its parent's child table if it has not polled. Operations: R/W LOCAL Becomes effective: After soft or hard reset Storage: Non-Volatile	XXYY Where YY is the timeout in seconds left-shifted by XX (YY * 2^XX). The default number results in a timeout of five minutes, whereas the maximum number results in a timeout of approximately 48 days. Maximum: 0EFF Factory Default: 0605

Table 4.23. S4F – MAC Timeout

Description	Parameters
Register defining the MAC timeout. Operations: R/W LOCAL Becomes effective: After Soft or Hard Reset Storage: Non-Volatile	XXXX The Ember_Indirect_Transmission_Timeout is the amount of time in milliseconds that the MAC in a parent node will hold a message for indirect transmission to a child. This is also the basis for the timeout after which an acknowledged unicast to an end device is nacked when no ACK is received. The default value is 7680ms. The maximum is 30s Maximum: 7530 Factory Default: 1E00

Table 4.24. S60 – Manufacturer Code

Description	Parameters
String containing the manufacture code used to define manufacturer defined clusters and attributes. Operations: R/W LOCAL Storage: Non-Volatile	Text string: cccc Example: 1010 Factory Default: N/A

Table 4.25. S61 – IAS Enrol Control

Description	Parameters
Set to enable IAS Zone Device enroll. Operations: R/W LOCAL Storage: Non-Volatile Check Appendix 2. Zone Enrollment for more information.	XXXX XXXX is a 16-bit hexadecimal number. Bit C: When this bit is set to 1, the CIE will clear flash-maintained zone table entries during software reset. Bit 8: When this bit is set, the user needs to use the AT+RAWZCLP command to respond to ACE client commands. Other relevant prompts are shown below: GETPANELSTATUS:<NodeID>,<EP>,<Sequence> GETBYPASSZLIST:<NodeID>,<EP>,<Sequence> GETZSTATUS:<NodeID>,<EP>,<StartingZoneId>, <maxNumberOfZoneIds>,<zoneStatusMaskFlag>,<zoneStatusMask>,<Sequence> Bit 6: When set this bit to 1, the user needs to handle the ACE Get Zone Information command if the Zone ID is larger than 31. (See Appendix 2. Zone Enrollment.) Bit 5: When this bit is set to 1, the CIE will allow zone devices to enroll even when the Zone table is full. Bit 4: When this bit is set to 1, the CIE will allow zone devices with user-specified zone types to enroll. Bit 3: When this bit is set to 1, the CIE will only allow zone devices with ZigBee Cluster Library defined zone type to enroll. Bit 2: When this bit is set to 1, the CIE will check the Zone Type of the received Enroll Request, then decide if the Zone device is allowed to enroll. If the Zone Type is not ZigBee Specification or user-specified, the enroll request is rejected. AT+MZONETYPE can be used to add a user-specified zone type in order to allow a device with user-defined zone type to be enrolled. Bit 1: When this bit is set to 1, the CIE will check the local Zone Table and only allow the device which has been resisted in the Zone Table to enroll. The AT+ADDZENTRY command can be used to configure the local zone table before commissioning one or a group of IAS devices, then set this bit for commissioning. Bit 0: Set to 1, the CIE will permit enroll when a Zone device joins the network and sends an enroll request to the CIE. If this bit is set to 0, other bits of this S-register setting will not take effect. All the other bits are reserved. Factory Default: 0001

Table 4.26. S62 – OTA Upgrade Control

Description	Parameters
This S-register will be used as a switch to determine whether the CICIE firmware will handle the OTA client commands or pass the commands to host application. Operations: R/W LOCAL Becomes effective: Instantly	XXXX Where XXXX represents a 16-bit hexadecimal number. Bit 0: Set this bit to 1 for instructing the CICIE to handle the *queryNextImageRequest* command sent by the OTA clients. The CICIE firmware will automatically respond to the request and filter out the request from the clients which are not supposed to be upgraded. The CICIE will send a response with IMAGE_NOT_AVAILIABLE to those devices. If this bit is not set, the CICIE firmware will print out an IMGQUERY prompt to notify the host application and wait for the host to handle the request. In that case, the host is required to contract an `AT+QIMGRSP` command and send it back to the OTA client. Bit 1: Set this bit to 1 for instructing the CICIE to handle the *UpdateEndRequest* command sent by the OTA clients. The CICIE firmware will automatically respond to the request with the user preset response parameters. If this bit is set, the user need to use `AT+SETPOLICY` to specify the upgrade policy. By default, the upgrade policy is set to UPDATE_NOW indicating the client should update itself immediately. If this bit is not set, the CICIE firmware will print out an UPGRADEREQ prompt to notify the host application and wait for the host to handle the request. In this case, the host is required to contract an `AT+UPGRADE` command and send it back to the OTA client. By default, CICIE will not wait for the Host application to construct a response and send back to the requesting device automatically. The user should check and set the policy and image ID using the corresponding AT commands If Bit 0 and 1 are not set to 1, the CICIE firmware will respond to the receiving request command. However, the IMGQUERY and UPGRADEREQ prompts will still show when the requests are handled and the responses are sent to client. Factory Default: 0003

Table 4.27. S63 – Enable RSSI and LQI Printing

Description	Parameters
This S-register is used to enable/disable the RSSI and LQI value shown in the incoming message prompt. Operations: R/W LOCAL Becomes effective: Instantly	XXXX Where XXXX represents a 16-bit hexadecimal number. Bit 0: Set this bit to 1, then the incoming message prompts will include the RSSI and LQI value at the end. For instance, the read attribute response prompt will be: `RESPATTR:<NodeID>,<EndPoint>,<ClusterID>,<AttrID>, <Status>,<AttributeValue>,<RSSI>,<LQI>` Bit 1: Set this bit to 1, then the `+PANSCAN` prompt will have an RSSI and LQI reading of the beacon at the end. Bit 2: Set this bit to 1 to disable the automatic check-in response sent by CICIE. If this bit is set, the `CHECKIN` prompt will show the sequence number of the check-in command. In this case, the host will use this sequence number in its constructed check-in command and send to the originator node. CICIE will automatically send check-in responses by default. The host will apply `AT+FPSET` to set the payload for the check-in response. Factory Default: 0000

Table 4.28. S64 – License Key

Description	Parameters
This S-register is used to store the license key. This key is used to activate the firmware. Check the document "TG-APP-0022 Copy Protection Release" for more information Operations: R/W LOCAL Becomes effective: Instantly	XXXXXXXXXXXXXXXX Where XXXXXXXXXXXXXXXX represents a 128-bit hexadecimal number. Factory Default: 0000000000000000

Table 4.29. S65 – Protocol ID

Description	Parameters
The 8-bit hexadecimal number represents the protocol ID for using a tunnel. Operations: R/W LOCAL Storage: Non-Volatile	01 Factory Default: C8

Table 4.30. S66 – Server Tunnel ID

Description	Parameters
The 16-bit hexadecimal number represents the server Tunnel ID, which will be used when a client requests a tunnel from CICIE. Operations: R/W LOCAL Storage: Non-Volatile	0000 Factory Default: 5747

4.3 I/O Related S-Registers

Table 4.31. S12 – UART Setup[1]

Description	Parameters
The device's RS232 Baud rate and mode. The default setting of 0500 results in: 19200 bps, no parity, 1 stop bit, and 8 data bits. Operations: R/W LOCAL Becomes effective: Instantly When the 128-byte output buffer of the module is full, data will be dropped. Storage: Non-Volatile	XXXX Where XXXX represents a 16-bit hexadecimal number. Range of the most significant byte: 00 to 0C 00: 1200 baud 01: 2400 baud 02: 4800 baud 03: 9600 baud 04: 14400 baud 05: 19200 baud 06: 28800 baud 07: 38400 baud 08: 50000 baud 09: 57600 baud 0A: 76800 baud 0B: 100000 baud 0C: 115200 baud 00 to FF Bit 6-7: Reserved Bit 5: H/W flow control enabled Bit 4: No command echo Bit 3: Seven data bits instead of eight Bit 2: Two stop bits instead of one Bit 1: Odd parity enabled Bit 0: Even parity enabled Factory Default: 0500

Note:

1. Baudrate settings do not apply to ETRX358USB.

5. Interpreting RSSI Energy Levels

The readings from AT+ESCAN represent the hexadecimal numbers with offset by +127 to make them positive numbers.

The RSSI is calculated over an 8-symbol period, as well as at the end of a received packet. The algorithm utilizes the RX gain settings and the output level of the ADC. The linear range of RSSI is specified as 40 dB over all temperatures. At room temperature, the linear range is approximately 60 dB (–90 dBm to –30 dBm).

6. Firmware Upgrades

If required, the firmware of the ETRX3 modules can be upgraded serially.

6.1 Firmware Upgrades via Serial Port

In order to upgrade the firmware of the ETRX3 module using the serial bootloader, issue the AT+BL0AD command either by typing it in, or by pressing the respective button in the [**Module Control**] group of the Telegesis Terminal Application.

After entering the bootloader, the connection parameters need to be changed to 115200bps, 8 data bits, 1 stop bit, no parity, no flow control (providing that it is not already set to these values). This is achieved by pressing the [**Disconnect**] button, changing the settings, and then pressing the [**Connect**] button. (If you only need to change the connection speed, disconnecting and reconnecting is not required).

After pressing [**Enter**], the bootloader menu will be shown in the terminal window, as shown in the figure below.

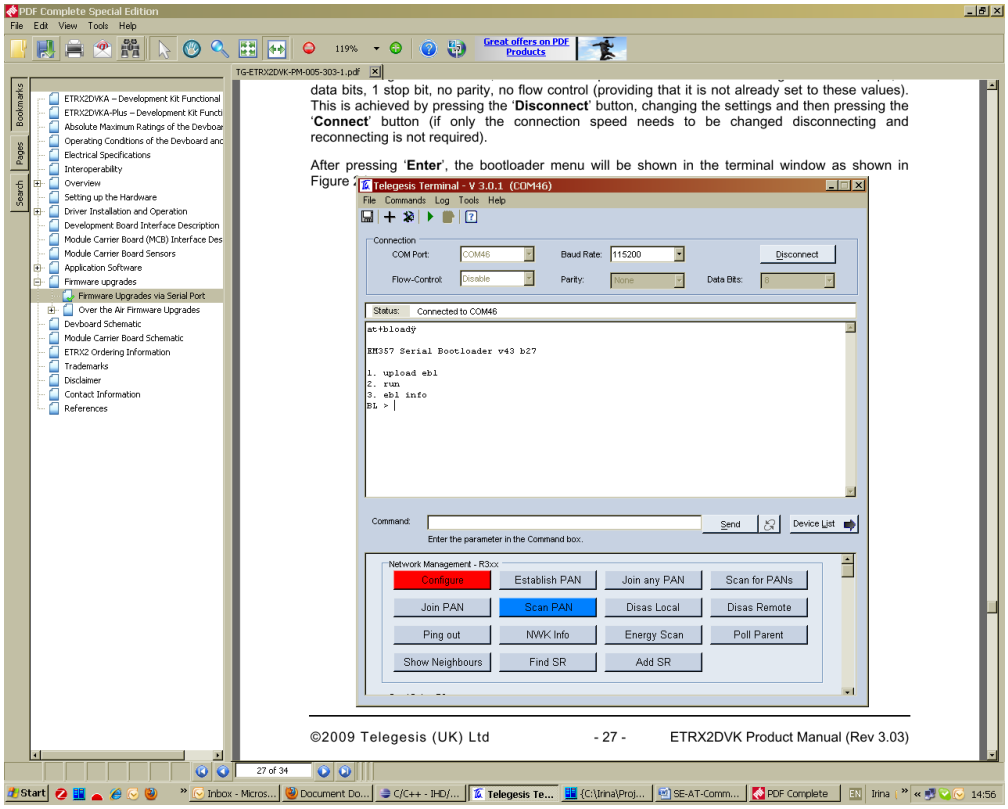


Figure 6.1. Options in Bootload Mode

Pressing [**1**] initiates the upload of the new firmware and a number of [**C**] characters will indicate that the ETRX3 is ready to receive data. Within 60 seconds, select [**Tools**] →[**Transfer File**], and browse for the new firmware file.

Firmware files for the ETRX3 will be in the .ebl format. After checking that the protocol is set to XMODEM (128 Bytes), press the [**Send**] button and the new firmware will be downloaded, as shown in the figure below.

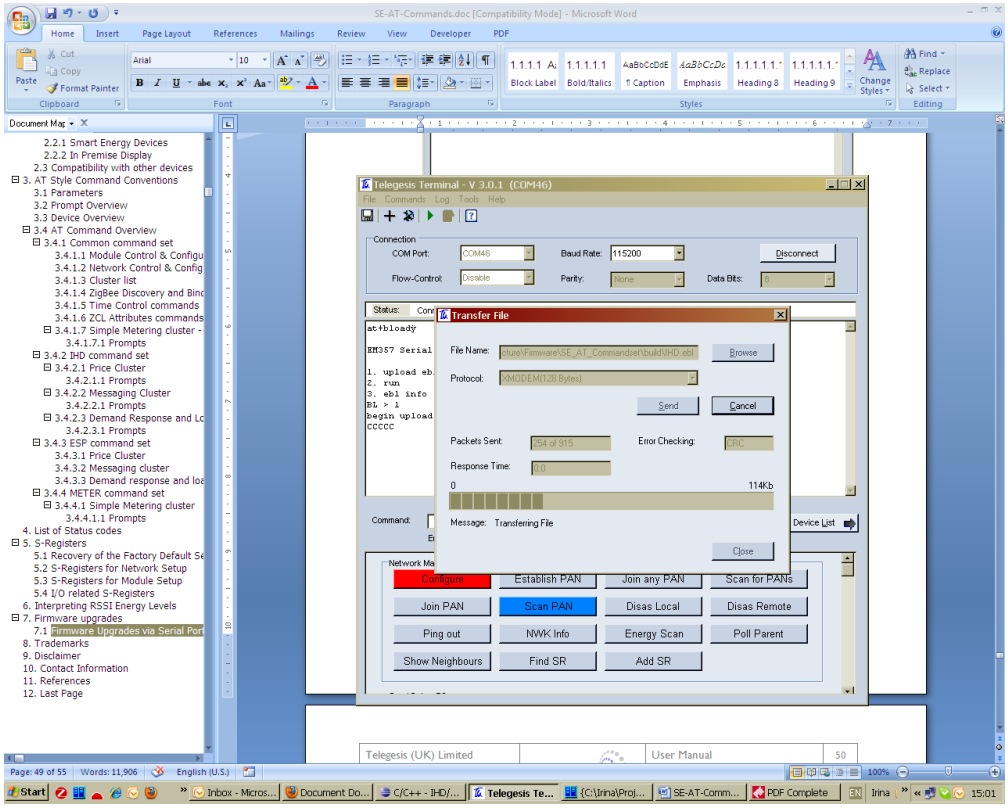


Figure 6.2. EBL Fill Transferred in Bootload Mode

When the transfer has been completed successfully, press [**Enter**] again to return to the bootloader menu (shown in Figure 6.1 Options in Bootload Mode on page 130) and option [**2**] to run the downloaded application software. If the application software has a baud rate other than 115200bps, this will need to be changed to the application baud rate described above.

7. References

This document was developed based on the following resources:

[1] ETRX3 Series Product Manuals — The Product Manuals concentrate on the hardware specifications of the modules.

[2] Evaluation Kit User's Guides — The Evaluation Kit User's Guides contains all information required to set up Evaluation Kits and run firmware upgrades.

[3] TG-APP-0022 Copy Protection Implementation Release Ver r3

[4] ZigBee PRO specification, document 053474r20

[5] ZigBee ZCL specification, document 075123r04ZB

[6] ZigBee Home Automation Public Application Profile Version 1.1 specification Revision 26, document 053520r26

[7] ZigBee Home Automation Public Application Profile Version 1.2 specification Revision 29, document 053520r29

It is highly recommended to refer to the ZigBee Cluster Library and ZigBee Home Automation Profile specifications to further understand the various AT Commands and their usage.

Appendix 1. Specified Cluster Command/Response Prompts

This appendix includes information about prompts defined for user-specified cluster commands/responses. The cluster commands are communicated with user-specified profiles (ProfileID: 0xC216).

Table 1.1. Specified Cluster Command/Response Prompt Overview

Command	Description
`LIFESIGN:<NodeID>,<EP>,<Seq>,<StatusFlag>,<MsTimer>,[PsuVoltage],[Temperature],[Rssi], [Lqi],[SwitchMask], [SwitchStates]` This prompt will show when CI-CIE R31x receives a lifesign command.	`<NodeID>` — 16-bit hexadecimal number representing the source network address. `<EP>` — 8-bit hexadecimal number representing the source endpoint. `<Seq>` — 8-bit hexadecimal number representing the ZCL sequence number. `<StatusFlag>` — 8-bit hexadecimal bitmap. Bit 0: If set, Battery Voltage is included in the lifesign command. Bit 1: If set, Temperature is included. Bit 2: If set, Switch Status is included. Bit 3: If set, LQI is included. Bit 4: If set, RSSI is included. Bit 5–7: Reserved. `<MsTimer>` — 32-bit hexadecimal. Current reading from the millisecond timer. `[PsuVoltage]` — 16-bit hexadecimal. Current power supply voltage reading in mV. `[Temperature]` — 16-bit hexadecimal. Current temperature reading. `[Rssi]` — Signed decimal number. Received Signal Strength Indication. dBm `[Lqi]` — 8-bit hexadecimal, Link Quality Indicator. `[SwitchMask]` — 8-bit hexadecimal. Availability mask of first eight logical switches. `[SwitchStates]` — 8-bit hexadecimal. Logical activation state of first eight logical switches
`GETRTC:<NodeID>,<EP>,<Seq>` This prompt will show when CI-CIE R31x receives a get rtc command.	`<NodeID>` — 16-bit hexadecimal number representing the source network address. `<EP>` — 8-bit hexadecimal number representing the source endpoint. `<Seq>` — 8-bit hexadecimal number representing the ZCL sequence number.
`FAULTREPORT:<NodeID>,<EP>,<Seq>, <ManufID>,<ModeID>,<FaultID>` This prompt will show when CI-CIE R31x receives a fault report command.	`<NodeID>` — 16-bit hexadecimal number representing source network address. `<EP>` — 8-bit hexadecimal number representing the source endpoint. `<Seq>` — 8-bit hexadecimal number representing the ZCL sequence number. `<ManufID>` — 16-bit hexadecimal number representing manufacturer ID. `<ModeID>` — 16-bit hexadecimal number representing mode ID. `<FaultID>` — 16-bit hexadecimal number representing fault ID.
`STDOUT:<NodeID>,<EP>,<Seq>,<Message>` This prompt will show when CI-CIE R31x receives a std out command.	`<NodeID>` — 16-bit hexadecimal number representing the source network address. `<EP>` — 8-bit hexadecimal number representing the source endpoint. `<Seq>` — 8-bit hexadecimal number representing the ZCL sequence number. `<Message>` — A null terminated ASCII hex string.

Command	Description
`HELLORSP:<NodeID>,<EP>,<Seq>,<Address>,<EUI>,<ManufID>,<DeviceType>, <ReleaseCode>,<AppVer>,<HWMinorVer>, <HWMajorVer>,<MfgString>,<ModelString>, <DataCodeString>` This prompt will show when CI-CIE R31x receives a hello response.	`<NodeID>` — 16-bit hexadecimal number representing the source network address. `<EP>` — 8-bit hexadecimal number representing the source endpoint. `<Seq>` — 8-bit hexadecimal number representing the ZCL sequence number. `<Address>` — 16-bit hexadecimal. Network node identification. `<EUI>` — IEEE address of the source node. `<ManufID>` — 16-bit hexadecimal. Manufacturer ID code. `<DeviceType>` — 16-bit hexadecimal. Device type code. `<ReleaseCode>` — 8-bit hexadecimal. Release code information for application code `<AppVer>` — 8-bit hexadecimal. Version information for application code. `<HWMinorVer>` — 8-bit hexadecimal. Version information for hardware platform (minor). `<HWMajorVer>` — 8-bit hexadecimal. Version information for hardware platform (major). `<MfgString>` — Character string. Manufacturer information string. `<ModelString>` — Character string. Model information string. `<DataCodeString>` — Character string. Date code string.
`RANGETEST:<NodeID>,<EP>,<Seq>,<Rssi>, <Lqi>` This prompt will show when CI-CIE R31x receives a range test command.	`<NodeID>` — 16-bit hexadecimal number representing the source network address. `<EP>` — 8-bit hexadecimal number representing the source endpoint. `<Seq>` — 8-bit hexadecimal number representing the ZCL sequence number. `[Rssi]` — Signed decimal number. Received Signal Strength Indication in dBm. `[Lqi]` — 8-bit hexadecimal, Link Quality Indicator.

Appendix 2. Zone Enrollment

This appendix provides some information about zone enrollment and example usage of supported AT commands and S registers.

Note: This appendix does not cover all possible usage scenarios. The user should check the ZigBee Cluster Library and ZigBee Home Automation Profile specifications for more information.

From Build 030714 onwards, CICIE R31x supports 16 flash-maintained zone entries and 16 RAM-maintained zone entries. Therefore, it can allow up to 32 zone devices to enroll. In order to allow more devices to enroll, the user can set S61 bit 5. But in this situation, CICIE will not be able to maintain the zone entry, and the user will need to use external memory and maintain the zone entries with a Zone ID larger than 31. The following figure illustrates how CICIE processes and responds to a zone enroll request.

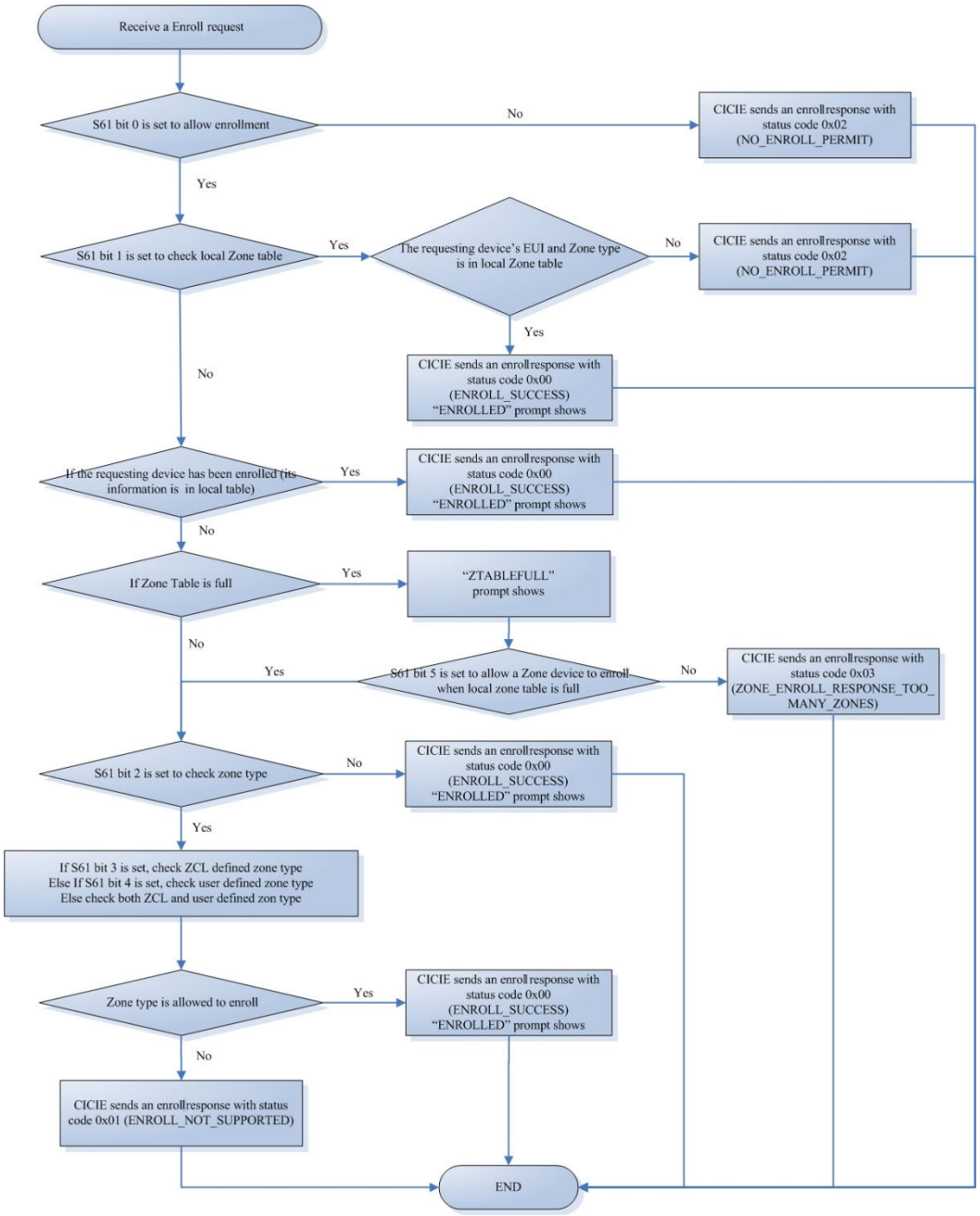


Figure 2.1. Zone Enroll Request

An IAS zone device needs to complete an enrollment process before it can work on a HA network. After joining a HA network, a zone device will send an enroll request to the IAS CIE device. The CIE will then respond with a status code. If the status code is successful, the zone device will start working on a network. Usually, it will periodically send a Zone Status Change Notification command to CIE. If the status code is not successful, the zone device may choose to leave the network.

Before sending an enroll request, a zone device needs to have knowledge of CIE's IEEE address (IAS_CIE_Address attribute 0x0010). The zone's specific implementation will determine whether it auto-detects for CIE, requires the intervention of a Configuration Tool (CT), or waits for CIE to notify the zone of its IEEE address.

If a zone device requires the intervention of a CT, the CT usually sets CIE's IEEE address in the zone device's local IAS_CIE_Address attribute 0x0010, before commissioning a zone device to join a network.

The following provides two examples of enrolling a zone device.

Zone Device (ZD) discovers CIE and auto-sets IAS_CIE_Address:

1. CIE forms a network and permits join.
2. ZD joins the network and broadcasts an announcement.
3. ZD sends a broadcast match descriptor to search for Zone Client cluster (0x0500).
4. CIE responds to the match descriptor with its node ID and endpoint.
5. ZD sends IEEE address requests to CIE using CIE's node ID.
6. CIE sends IEEE address response to ZD.
7. ZD sets CIE's IEEE address to local IAS_CIE_Address attribute.
8. ZD sends an enroll request to CIE.
9. CIE responds to ZD with an enroll response.

CIE writes its IEEE Address to ZD's IAS_CIE_Address:

1. CIE forms a network and permits join.
2. ZD joins the network and broadcasts an announcement.
3. CIE sends a broadcast match descriptor request to discover Zone Server cluster (0x0500).
4. ZD sends a match descriptor response with its node ID and endpoint.
5. CIE sends a write attribute request to ZD to set its IEEE address to ZD's IAS_CIE_Address.
6. If ZD permits a remote write attribute, it sends a write attribute response to CIE with a success status code.
7. ZD sends a enroll request to CIE.
8. CIE responds to ZD with an enroll response.

Appendix 2.1 Enroll Additional Zone Devices

From Build B030714, the CICIE firmware supports 32 zones entries. Zone entries 00 - 0F are flash-maintained and entries 10 – 1F are RAM-maintained. The user can also set S61 Bit 5 to allow more zone devices to enroll even when CIE's zone table is full. The following describes an example procedure for using an extended zone table (up to 255 entries).

To use an extended zone table, several assumptions need to be made:

1. CIE's zone table is full.
2. The user has set S61 to allow a zone device with any zone types to enroll.
3. The joining zone device has never enrollled with the CIE.
4. The total number of enrollled devices has not reached 255.

Commissioning procedure:

1. Set S61 Bit 5 (using command ATS615 = 1).
2. When the joining zone device joins CIE's network, it sends an enroll request. When CIE receives this enroll request, it shows the prompt `ZenrollLREQ`.
3. As S61 Bit 5 has been set, the CIE will send an enroll response with the success status code and a Zone ID larger than 0x1F. Meanwhile, the CIE will show prompt `enrollLED`. The user should consider copying the information provided with the `enrollLED` prompt and saving it as a zone entry.
4. The user should use the `AT+EDITZMAP` command to edit zone map information. The zone map information will be used to respond to ACE device's get zone map request.

   A zone ID map has 16 sections. Each section is a 16-bit bitmap. If CIE's local zone table is full, Section 0 and Section 1 will be both 0xFFFF.

   Note: Section 0 and Section 1 are automatically generated by CIE, according to the usage of local zone entries. The user cannot edit Sections 0 and 1.

   An example is shown below:

   If CIE shows prompt `enrollLED:20,1234, 0021ED1000000003`, it means that the CIE has sent an enroll response to a joining zone device with a success status code and zone ID 0x20. The user should change to the corresponding bit for this zone, indicating that the zone device has been enrollled. To do this, the user can use `AT+EDITZMAP:02,0001`, as Bit 0 of Section 2 is associated with zone ID 0x20.
5. By default, CIE will handle the `GETZONEINFO` command sent by ACE device. If CIE receives a `GETZONEINFO` with zone ID less than 32, it will check local zone table to retrieve the information and respond.

   If the extended zone table is used (zone ID larger than 31), the user should set S61 Bit 6 to enable user handling of the zone information request by using `AT+ZINFORSP` command. In such a case, when CIE receives a get zone information command with zone ID that is larger than 31, the `GETZONEINFO` prompt will be presented as a notification. The user should respond with the same sequence number contained in the `GETZONEINFO` request command.

Appendix 3. Time Server Clusters

This appendix provides information about using the time server cluster of CICIE as a time master on a HA network.

The CICIE supports Time Server cluster (0x000A). The Time attribute (0x0000) of time server cluster holds the time value of a real time clock. When the module is reset or in a power cycle, this attribute will reset and start counting from 0x00000000. Therefore, the user should set the time (using `AT+SETTIME`) after reset or power cycle of the module. After setting Time and Time status attributes to valid values, use the `AT+TIMERD` command to notify the module that the clock time is now ready to be used.

Note: If the `AT+TIMERD` command is not used, CICIE will respond to a read time cluster attribute command with a default response (status code failuer).

In addition, some HA devices need to synchronize time with the network COO's local time, so the CICE supports *StandardTime* (0x0006) and *LocalTime* (0x0007).

The local Standard Time (time adjusted for the time zone, but not adjusted for Daylight Savings Time (DST)), is given by:

$$Standard\ Time = Time + TimeZone$$

The Local Time (time adjusted for the time zone and DST), is given by:

$$Local\ Time = Standard\ Time + DstShift \text{ (if } DstStart \leq Time \leq DstEnd\text{)}$$

$$Local\ Time = Standard\ Time \text{ (if } Time < DstStart \text{ or } Time > DstEnd\text{)}$$

The CICIE firmware has integrated the mechanism to calculate standard time and local time. The calculated values are stored in attributes *StandardTime* (0x0006) and *LocalTime* (0x0007), respectively.

In order to serve Local Time to other devices on a network, the CICIE needs to have valid *TimeZone* (0x0002), *DstStart* (0x0003), *DstEnd* (0x0004), and *DstShift* (0x0005). Therefore, it is recommended to utilize the `AT+SETATR` command to set these four attributes after initiating the CICIE. By default, the four attributes will be set to 0, so both standard time and local time will be the same as clock time.

It is strongly recommended that the user maintain the *TimeStatus* (0x0002) attribute, which holds a number of bit fields to indicate the status of local clock time.

Table 3.1. Time Attributes

Bit	Meaning	Values	Notes
0	Master	1: Master clock 0: No master clock	This bit is not writeable. By default, this bit is set to 1.
1	Synchronized	1: Synchronized 0: Not synchronized	By default, this bit is set to 0.
2	MasterZoneDst	1: Master for Time Zone and DST 0: No master for Time Zone and DST	This bit is not writeable. By default, this bit is set to 0.
3	Superseding	1: Time synchronization should be superseded 0: Time synchronization should not be superseded	By default, this bit is set to 0.
Others	Reserved		

Together, the Master and Synchronized bits provide information on how closely the Time attribute conforms to the time standard.

The Master bit specifies whether the real time clock corresponding to the Time attribute is internally set to the time standard.

The Synchronized bit specifies whether Time has been set over the ZigBee network to synchronize it (as close as may be practical) to the time standard.

The MasterZoneDst bit specifies whether the *TimeZone*, *DstStart*, *DstEnd*, and *DstShift* attributes are set internally to correct values for the location of the clock.

Follow the steps below to set up a time server:

1. After the module power up, set local clock time. e.g. `AT+SETTIME:1BDBE518` (`1BDBE518` is UTC time representing 16:11:00 23/10/2014).
2. Set Bit 1 of *TimeStatus*, since local clock time has been synchronized with current time. Use `AT+SETATR:000A,0001,03` (Set Bit 0 to 1, as the time is used as master time from the local clock).
3. Set *TimeZone*, *DstStart*, *DstEnd* and *DstShift* attributes accordingly.
4. Set Bit 2 of TimeStatus attribute. Use `AT+SETATR:000A,0001,07` (Set bit 0 to 1).

Then CICIE can produce the local time for remote devices to use during time synchronization.

Appendix 4. OTA Upgrade Process

The OTA cluster provides a standard way to upgrade devices in the network via OTA messages. Thus, the upgrade process may be performed between two devices from different manufacturers. Devices are required to have application bootloader and additional memory space in order to successfully implement the cluster. This appendix will briefly describe the OTA upgrade progress.

According to OTA cluster specification, it is the responsibility of the server to indicate to the clients when the update images are available. The client may be upgraded, downgraded, or reinstalled. The upgrade server must know which client devices to upgrade and to what file version. The upgrade server may be notified of such information via the backend system. In this project, the Host application will be responsible for storing and managing the information of client devices for upgrading.

The OTA cluster is implemented in a way that the client service works on both ZED and ZR devices. Being able to handle polling is mandatory for all server devices. Therefore, the CICIE will provide APIs to the Host application to handle the poll. Meanwhile, a notify command is supported for sending broadcast or unicast new image notifications, while being able to send a notification is optional. The following picture illustrates the way the OTA upgrade cluster works. First, the host needs to prepare an OTA image and set the upgrade policy for the OTA server. Then, the CICIE OTA server can either send a notification to the OTA client or wait for the client to query the image. Once the OTA client is informed of the availability of a new image, it will start downloading the image.

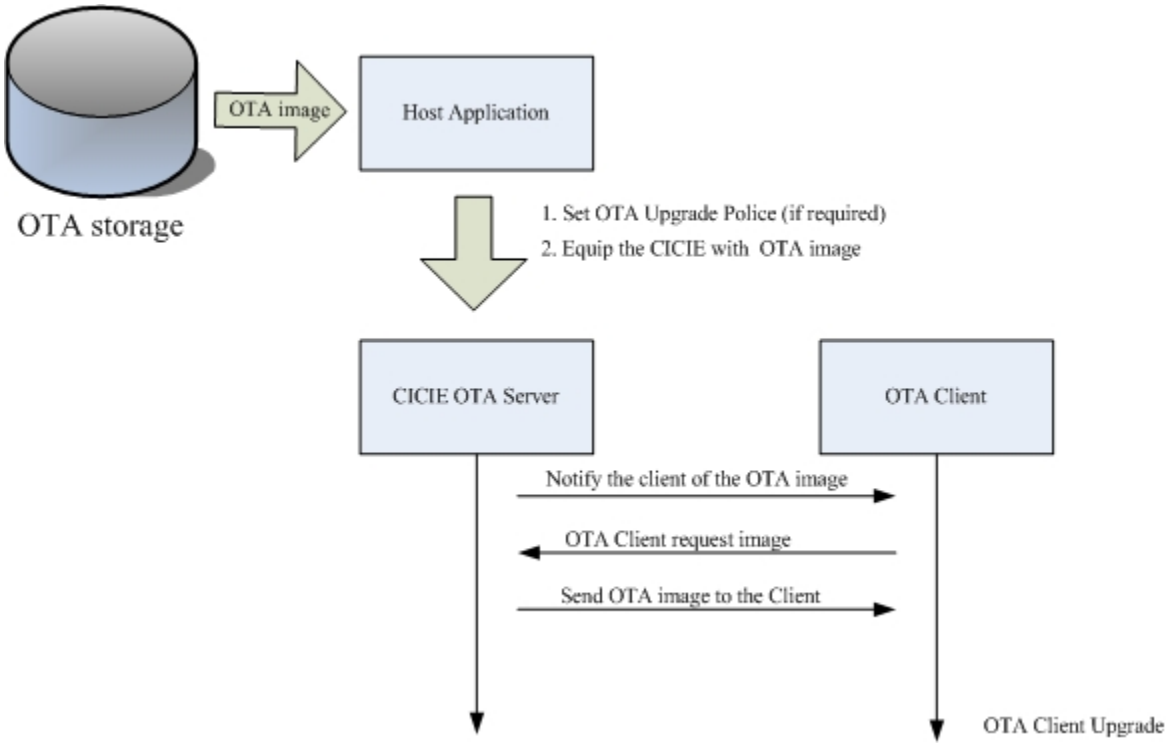


Figure 4.1. OTA Upgrade

The OTA upgrade progress may include multiple command or message exchange between the OTA server and client. The following table lists these commands:

Table 4.1. OTA Upgrade Cluster Commands

Command ID	Description	Direction	Mandatory/Optional
0x00	Image Notify	Server	O
0x01	Query Next Image Request	Client	M
0x02	Query Next Image Response	Server	M
0x03	Image Block Request	Client	M
0x04	Image Page Request	Client	O
0x05	Image Block Response	Server	M
0x06	Upgrade End Request	Client	M
0x07	Upgrade End Response	Server	M
0x08	Query Specific File Request	Client	O (not supported by CICIE)
0x09	Query Specific File Response	Server	O (not supported by CICIE)

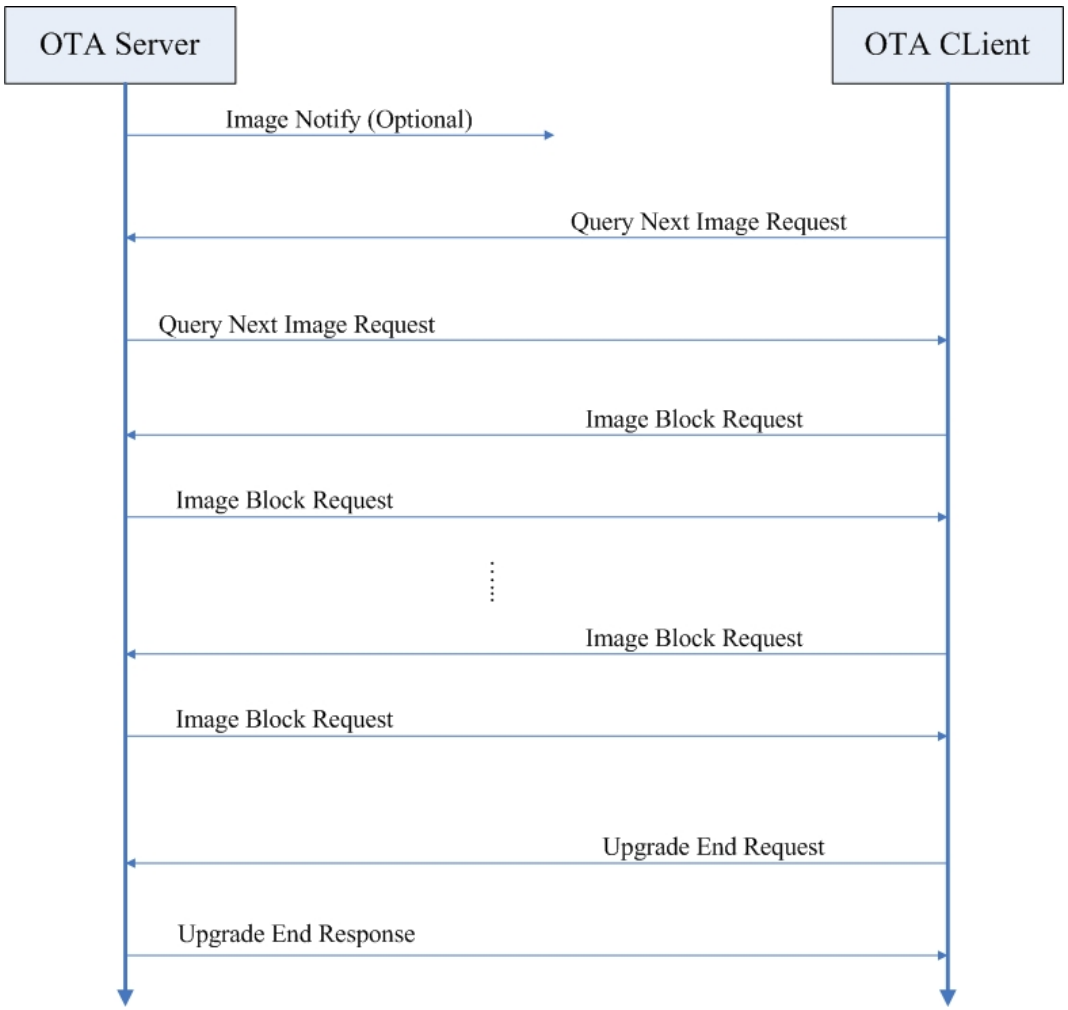


Figure 4.2. Typical OTA Upgrade Process

The above figure illustrates a common OTA upgrade process using the commands listed in Table 4.1 OTA Upgrade Cluster Commands on page 141. An OTA upgrade progress may be divided into three main parts:

1. *The OTA client gets notification of availability of a new image.*

   According to the OTA cluster specification, the server may notify devices in the network when it receives a new OTA upgrade image by sending an "Image Notify" Command.

   The client device will send a "Query Next Image Request" Command if the information in the "Image Notify" Command is of interest after applying the jitter value.

   Sending the "Image Notify" Command to an OTA client is optional. All OTA client devices send a "Query Next Image Request" Command periodically, regardless of whether an "Image Notify" was sent by the OTA server.

   Once the OTA server receives a "Query Next Image Request", it checks the information contained in the request and makes a decision for responding to the requesting device.
2. *The OTA client downloads image blocks from server.*

   When the device has received a response to its query indicating a new OTA upgrade image is available, the client device requests blocks of the OTA upgrade image. The process continues until the client receives all image data.
3. *The OTA client queries the server for upgrade time.*

   When the client has finished downloading the image, it verifies the integrity of the image received and sends an "Upgrade End Request" Command along with the upgrade status. The server will notify the client when it is time to to upgrade to the new image in the "Upgrade End Response".

Appendix 4.1 CICIE OTA Server Implementation

From the description above, it can be found that an OTA server will normally receive three commands and issue three response commands. In addition, the server should be able to send an image notify command. Therefore, three prompts and four AT commands are defined in OTA server implementation.

The host can use the `AT+IMGNOTIFY` command to send unicast/broadcast to an OTA client to notify them of the availability of a new image. Then it will receive query commands.

1. `IMGQUERY` — This prompt is shown when the "Query Next Image Request" command is received. The host can use the `AT+QIMGRSP` command to send back a response.
2. `IMGBREQ` — This prompt is shown when the "Image Block Request" command is received. The host can use the `AT+IMGBRSP` command to send back a response.
3. `UPGRADEREQ` — This prompt is shown when the "Image Block Request" command is received. The host can use the `AT+UPGRADE` command to send back a response.

The prompt `IMGPREQ` is introduced to support image page requests. When this prompt shows, the host should prepare the request OTA file page and then use a series of `AT+IMGBRSP` commands to send the page block by block. *When the host issues `AT+IMGBRSP` commands in response to a page request, the host will set to 5 in the `<sendMode>` parameter to disable APS retries as per OTA cluster specifications.*

Appendix 4.2 Use Cases

This section describes some use cases to help the user understand how to test and use the OTA server features.

In CICIE firmware design, an S-register S62 is introduced for configuring OTA commands handling. By changing the setting in S62, the user can select to handle all incoming OTA client commands, or partially handle them. Block request commands will always be passed to a host application since the host takes in charge of storing and serving OTA images.

Appendix 4.2.1 Host Application Handling Incoming OTA Client Cluster Commands

In this scenario, S62 should be set to 0000. The CICIE will not check and handle the OTA client request commands, all incoming requests will be passed to the host application. In such a case, the host application needs to wait for the `IMGQUERY` prompt, and based on the parameters present, will follow the prompt to find the relevant image in its file system. Then the Host will use `AT+QIMGRSP` to respond to the requestor and indicate the availability of the requested file.

If the host can serve the requested OTA file, the next step will be that the remote device start requesting OTA block data. Upon receiving a block request, CICIE will use an `IMGBREQ` prompt to notify the host. Then the host will use `AT+IMGBRSP` to transmit the block to the requestor. The requestor will send another block request until it receives the entire OTA file.

If the host gets an `IMGPREQ` prompt, it will prepare the page of the OTA file (contains multiple blocks) and apply a series of `AT+IMGBRSP` to transmit all blocks of the requested page to the requestor.

If the host loses the OTA file and cannot serve it anymore after an OTA progress has started, the `AT+DFTRSP` command should be used with status 0x98 (IMAGE_NOT_AVAILIABLE) to terminate the OTA upgrade progress. This is a specified behaviour in an OTA cluster specification.

Once an OTA file is received by the requestor, it will send an upgrade end request command to query the update time. The CICIE firmware will show an `UPGRADEREQ` prompt to the host, and the host will respond with `AT+UPGRADE` to finish off the OTA upgrage process.

The above requires the host application to parse the OTA image file to retrieve the manufacturer id, hw/fw firmware versions, etc., to supply the correct image data to the CICIE firmware.

Appendix 4.2.2 CICIE OTA Plugin Handle Incoming OTA Client Cluster Commands

All OTA client devices will periodically send *QueryNextImageRequest* commands to the server, and the host may require a filtering mechanism to handle some incoming requests. Also, the host may want to set the CICIE to respond to its clients about the upgrade time. In such a case, S62 should be set to 0003 (both Bit 0 and Bit 1 are set to true).

If the host has a new image for downloading, `AT+SETPOLICY`, `AT+SETIMGID`, and the `AT+SETHWVER` command can be used to set the OTA upgrade policy and image ID, hardware (hw) version, software (sw) version, etc. If the new image is for a specific node, the host can also use `AT+SETIMGID` to set that node's network address. This information will be used to filter out all requests which contained unmatched information. Only the request with matching information will be shown to the host application with the `IMGQUERY` prompt.

In the case that a request contained unmatched information (e.g. manufacture Id, image type, or fw version), the CICIE will automatically send a *QueryNextImageResponse* with status code NO_IMAGE_AVAILABLE to the requesting client.

If a request is received with matching information, the CICIE will automatically send a *QueryNextImageResponse* with status code "Success" to the requesting client. In the meantime, the host will receive the `IMGQUERY` prompt, which is the notification for preparing the image download.

The remote device may start requesting the block for data from the OTA server. At that point the host application will receive an `IMGBREQ` prompt with a block address for the host application to supply the data block from the OTA image to the ZigBee for sending to the remote device using `AT+IMGBRSP`.

Once all blocks have been transmitted to the remote device, the remote device sends an upgrade end request, which is provided to the host application as `UPGRADEREQ` prompt. In this scenario, the `UPGRADEREQ` prompt is used to notify the host that the client has finished image downloading. Since the S62 Bit 1 is set to true, the CICIE will automatically send an *UpgradeEndResponse* to the client. If the host did not change the upgrade policy, the CICIE will send the response to inform the client to upgrade the downloaded firmware immediately. The host can change the policy using the `AT+SETPOLICY` command. This concept is introduced in 4. S-Registers.

Appendix 4.2.3 Both Host and CICIE OTA Plugin Partially Handle Incoming OTA Cluster Commands

If the S62 is set to 0001 or 0002, CICIE will only handle *QueryNextImageRequest* or *UpgradeEndRequest*, respectively. In such a case, the host needs to use corresponding AT command to construct and send a response when `IMGQUERY` or `UPGRADEREQ` is shown.

For instance, the host only wants the CICIE to handle incoming *QueryNextImageRequest* commands. It should set S62 to 0001. Then the CICIE will only pass the requests containing matching information (manufacture ID, device type, etc.) and print the `IMGQUERY` prompt. It will also send back a response to the requesting clients.

The remote device may start requesting the block for data from the OTA server, and at that point the host application will receive an `IMGBREQ` prompt with the block address for the host application to supply that data block from the OTA image to the ZigBee for sending to the remote device using `AT+IMGBRSP`.

Once all blocks have been transmitted to the remote device, the remote device sends an upgrade end request, which is provided to the host application as `UPGRADEREQ` prompt. The host then responds to the message with `AT+UPGRADE` to finish off the process.

Appendix 4.3 Explanation of Upgrade Policy

The upgrade policy will only be used when S62 is set to none (0) value. The upgrade policy includes *QueryPolicy* and *UpgradeRequestPolicy*. The `AT+SETPOLICY` command can be used to set these two policies. `AT+GETPOLICY` can be used to check the pre-set policy.

Appendix 4.3.1 QueryPolicy

Note: *QueryPolicy* is used when S62 Bit 0 is set to true.

00: Upgrade if server has newer

01: Downgrade if server has older

02: Reinstall if server has same

03: No next version

Assume:

1. Parameters (such as Manufacturer code, ImageType, hardware version, etc.) contained in the "Query Next Image request" match the OTA server's corresponding information.
2. The OTA server gets an image with Firmware Version: 0x00000002.
3. The "Query Next Image" request contained in the Current Firmware Version is: 0x00000001.

If set to 00, the device has current firmware version 0x00000001, and CICIE has a newer version (0x00000002). CICIE will print a prompt notifying the host to prepare the block for downloading. The prompt is `IMGQUERY`. Meanwhile, it sends a response notifying the client that there is new image for downloading.

If set to 01, the device has current firmware version 0x00000001, and the CICIE has an older one (maybe 0x00000000). CICIE will not print a prompt. Meanwhile, it sends a response notifying the client that there is no image available for downloading.

If set to 02, the device has current firmware version 0x00000001, and the CICIE has the same version (should be 0x00000001). CICIE will not print a prompt. Meanwhile, it sends a response notifying the client that there is no image available for downloading.

If set to 03, CICIE will send a response notifying the client that there is no image available for downloading.

Appendix 4.3.2 UpgradeRequestPolicy

Note: *UpgradeRequestPolicy* is used when S62 Bit 1 is set to true.

00: Upgrade now

01: Upgrade in two minutes

02: Ask me later to upgrade

03: Abort upgrade

Note: This policy will be used only if the ATS63 Bit 1 is set to true. In that case, the host does not need to issue an AT+UPGRADE command, as CICIE will automatically respond to the *UpgradeEndRequest* command.

After the OTA client finished downloading the OTA image, it will send an *UpgradeEndRequest* command to the CICIE.

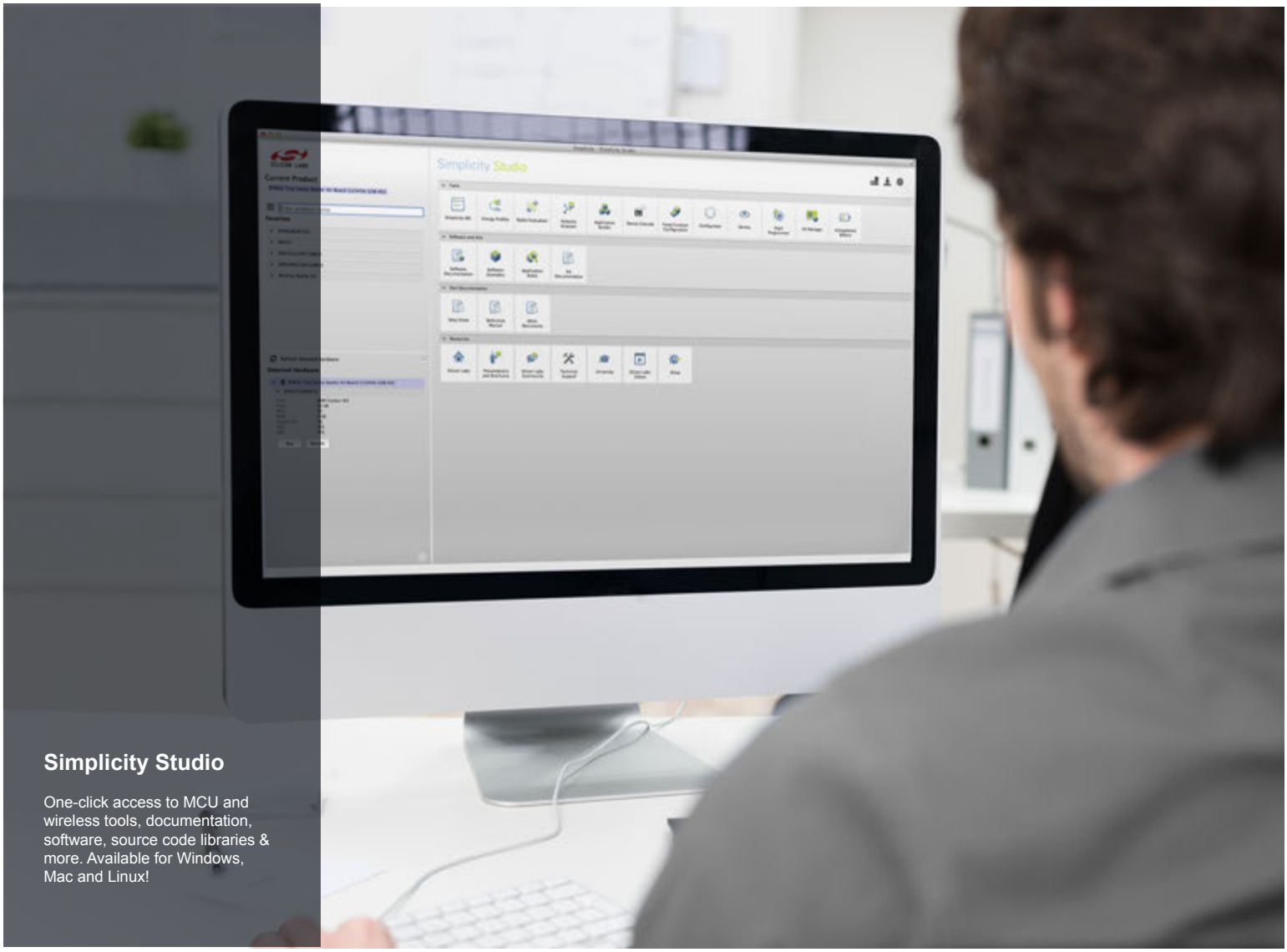
If *UpgradeRequestPolicy* is set to 00, the CICIE will send back an *UpgradeEndResponse* with the current time (0x00000000) and upgrade time (0x00000000). In this case, the OTA client is expected to upgrade itself immediately.

If *UpgradeRequestPolicy* is set to 01, the CICIE will send back an *UpgradeEndResponse* with the current time (0x00000000) and upgrade time (0x00000078). In this case, the OTA client is expected to upgrade itself after two minutes.

If *UpgradeRequestPolicy* is set to 02, the CICIE will send back an *UpgradeEndResponse* with the current time (0x00000000) and upgrade time (0xFFFFFFFF). In this case, the OTA client is expected not to upgrade itself until receiving an unsolicited *UpgradeEndResponse* command with an update time which is not all 'F'.

If *UpgradeRequestPolicy* is set to 03, the CICIE will send back a default response with error code 0x95 indicating Abort status.

Note: This appendix only provides information to assist the user in evaluating the HA OTA product. More information about OTA upgrade clusters can be found in "docs-09-5264-23-00zi-zigbee-ota-upgrade-cluster-specification.pdf".



Simplicity Studio

One-click access to MCU and wireless tools, documentation, software, source code libraries & more. Available for Windows, Mac and Linux!

IoT Portfolio
www.silabs.com/IoT

SW/HW
www.silabs.com/simplicity

Quality
www.silabs.com/quality

Support and Community
community.silabs.com

SILICON LABS®

Silicon Laboratories Inc.
400 West Cesar Chavez
Austin, TX 78701
USA

http://www.silabs.com