ZIGBEE CLUSTER LIBRARY SPECIFICATION

ZigBee Document 075123r04ZB	
May 29, 2012 10:50 am	
Sponsored by: ZigBee Alliance	
Accepted by	This document has been accepted for release by the ZigBee Alliance Board of Directors.
Abstract	This document defines the ZigBee Cluster Library.
Keywords	ZigBee, Application Framework, Cluster Library, ZCL

May 31, 2012

PARTICIPANTS

The following is a list of those who were members of the ZigBee Cluster Library Development Board when this document was released:

Cam Williams Co-Chair, Technical Editor

Drew Gislason Co-Chair

Contributions were made to this document from the following ZigBee members:

Rob Alexander
Marcel Beij
Claudio Borean
Jason Choong
David Clark
Ettore Colicchio
Kent Crouse
Tim Gillman
Drew Gislason
Ted Humpal
Phil Jamieson
William Keith
Tom Klein
Jared Lemke
Jens Klostergaard Lyngsø
Marco Naeve
Philip Orlik
Isaac Pinhas
Phil Rudland
Zachary Smith
Don Sturek
Mads Westergreen
Urban Wicklander
Cam Williams
Ian Winterburn
Walter Young

NOTICE OF USE AND DISCLOSURE

The ZigBee Specification is available to individuals, companies and institutions free of charge for all non-commercial purposes (including university research, technical evaluation, and development of non-commercial software, tools, or documentation). No part of this specification may be used in development of a product for sale without becoming a member of ZigBee Alliance.

Elements of ZigBee Alliance specifications may be subject to third party intellectual property rights, including without limitation, patent, copyright or trademark rights (such a third party may or may not be a member of ZigBee). ZigBee is not responsible and shall not be held responsible in any manner for identifying or failing to identify any or all such third party intellectual property rights.

This document and the information contained herein are provided on an “AS IS” basis and ZigBee DISCLAIMS ALL WARRANTIES EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO (A) ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OF THIRD PARTIES (INCLUDING WITHOUT LIMITATION ANY INTELLECTUAL PROPERTY RIGHTS INCLUDING PATENT, COPYRIGHT OR TRADEMARK RIGHTS) OR (B) ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE OR NON-INFRINGEMENT. IN NO EVENT WILL ZIGBEE BE LIABLE FOR ANY LOSS OF PROFITS, LOSS OF BUSINESS, LOSS OF USE OF DATA, INTERRUPTION OF BUSINESS, OR FOR ANY OTHER DIRECT, INDIRECT, SPECIAL OR EXEMPLARY, INCIDENTIAL, PUNITIVE OR CONSEQUENTIAL DAMAGES OF ANY KIND, IN CONTRACT OR IN TORT, IN CONNECTION WITH THIS DOCUMENT OR THE INFORMATION CONTAINED HEREIN, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH LOSS OR DAMAGE. All Company, brand and product names may be trademarks that are the sole property of their respective owners.

The above notice and this paragraph must be included on all copies of this document that are made.

ZigBee Alliance, Inc.
2400 Camino Ramon, Suite 375
San Ramon, CA 94583

This page left intentionally blank

DOCUMENT HISTORY

ZigBee Cluster Library History

Revision Number	Date	Comments
00	11th July 2007	Document created
01	19th Oct 2007	First release
02	29th May 2008	Added Commissioning Cluster from 064699r12. • Added material from annex of CBA Profile 053516r10 • Structured types (arrays etc) and structured R/W commands • Input / Output / Value clusters (Basic) • Input / Output / Value clusters (BACnet Regular & Extended) • Generic Tunnel cluster • BACnet Protocol Tunnel cluster Made changes to the Color Control cluster re. CCB 870 • Added x,y control according to CIE 1931 Color Space Added long data types (as required by SE profile 075356r12 etc) • 40-64bit integers etc, long strings Made changes to time cluster (as required by CCBs 890, 914) • Added time zone & DST + UTCtime type Made minor changes as requested by the following CCBs • 627, 714, 781, 853, 854, 867, 878, 879, 880, 881, 883, 893, 897, 898, 919, 958

ZigBee Cluster Library History

03	18th Sept 2009	The following changes were made to the Editor's Copy of the ZCL, 095254r00. Made change to the Basic cluster, re CCB comment #606 • Added optional attribute *DisableLocalConfig*. Updated Pressure Measurement cluster re CCB comment #961 • Added extra attributes to allow wider range of pressure. Updated Color Control cluster re CCB comment #1006 • Clarification of stop commands, color mode switching etc. Made changes to RSSI Location cluster, re CCB comment #1053 • Added mechanism for centralized location. Made change to Generic Tunnel cluster, re CCB comment #1068 • Added extra fields to Match Protocol Address Response command
	24th Dec 2009	Made minor changes and clarifications re the following CCBs • 960, 1001, 1004, 1061, 1097. • Added Door Lock cluster. Updated Occupancy Sensor re CCB comments 1092, 1093, 1094
04	2010	CCB 1174: Fixed references CCB 1176: Added new status codes CCB 1202: Corrected default value in thermostat cluster
	April 2012	CCB 1381: Default Response clarification CCB 1260: Generic Tune l cluster clarification CCB 1377: Commissioning Cluster minor change CCB 1146: Report Attributes without Configuration CCB 1169: Dependencies on Optional Attributes CCB 1379: Generic Tunnel *ProtocolAddress* attribute ReadOnly option CCB 1420: Time cluster ESI bit CCB 1390: Reporting destination clarification

TABLE OF CONTENTS

This page left intentionally blank

LIST OF TABLES

This page left intentionally blank

LIST OF FIGURES

This page left intentionally blank

CHAPTER

1

INTRODUCTION

1.1 Scope and Purpose

This document specifies the ZigBee Cluster Library (ZCL). The ZCL is a repository for cluster functionality that is developed by the ZigBee Alliance, and, as a consequence, it will be a working library with regular updates as new functionality is added.

A developer constructing a new application profile should use the ZCL to find relevant cluster functionality that can be incorporated into the new profile. Correspondingly, new clusters that are defined for application profiles should be considered for inclusion in the ZCL.

The ZCL consists of the ZCL Foundation, a set of elements that apply across the entire library (such as frame structures, attribute access commands and data types), and a number of sets of clusters. Clusters that are generally useful across many application domains are included in the General set. Clusters that are intended for use mainly in specific application domains are grouped together in domain oriented sets.

1.2 Acronyms and Abbreviations

Acronym / Abbreviation	Meaning
ACE	Ancillary Control Equipment
AIB	Application support sub-layer Information Base
APS	Application support Sub-layer

Acronym / Abbreviation	Meaning
CIE	Control and Indicating Equipment
EUI-64	IEEE-defined 64-bit Extended Unique Identifiers
HVAC	Heating, Ventilation, Air Conditioning
IAS	Intruder Alarm System
MAC PIB	Medium Access Control sub-layer PAN Information Base
NIB	Network layer Information Base
NWK	Network layer
PAN	Personal Area Network
PIR	Pyroelectric Infra-Red (a type of motion detection sensor)
RSSI	Received Signal Strength Indication
TC	Trust Center
WD	Warning Device
ZCL	ZigBee Cluster Library

1.3 Definitions

1.3.1 ZigBee Definitions

Cluster: A related collection of attributes and commands, which together define a communications interface between two devices. The devices implement server and client sides of the interface respectively.

Client: A cluster interface which is listed in the output cluster list of the simple descriptor on an endpoint. Typically this interface sends commands that manipulate the attributes on the corresponding server cluster.

Server: A cluster interface which is listed in the input cluster list of the simple descriptor on an endpoint. Typically this interface supports all or most of the attributes of the cluster.

1.3.2 Application Domain Definitions

4-pipes: In a 4-pipe HVAC fan coil system, heated and chilled water each have their own supply and return pipes, while in a 2 pipe system they share the same supply and return. With a 4-pipes system, heating and cooling can take place at

the same time in different locations of a building. With a 2-pipes system, only heating or cooling can take place in the whole building.

Ballast Factor: A measure of the light output (lumens) of a ballast and lamp combination in comparison to an ANSI standard ballast operated with the same lamp. Multiply the ballast factor by the rated lumens of the lamp to get the light output of the lamp/ballast combination.

HSV: Hue, Saturation, Value. A color space, also known as HSB (Hue, Saturation, Brightness). This is a well-known transformation of the RGB (Red, Green, Blue) color space. For more information see e.g. http://en.wikipedia.org/wiki/HSV_color_space.

Illuminance: The density of incident luminous flux on a surface. Illuminance is the standard metric for lighting levels, and is measured in lux (lx).

Precooling: Cooling a building in the early (cooler) part of the day, so that the thermal mass of the building decreases cooling needs in the later (hotter) part of the day.

1.4 Conformance Levels

- **Expected:** A key word used to describe the behavior of the hardware or software in the design models *assumed* by this Draft. Other hardware and software design models may also be implemented.
- **May:** A key word that indicates flexibility of choice with *no implied preference*.
- **Shall:** A key word indicating a mandatory requirement. Designers are *required* to implement all such mandatory requirements.
- **Should:** A key word indicating flexibility of choice with a strongly preferred alternative. Equivalent to the phrase *is recommended*.

1.5 References

The following standards and specifications contain provisions, which through reference in this document constitute provisions of this specification. All the standards and specifications listed are normative references. At the time of publication, the editions indicated were valid. All standards and specifications are subject to revision, and parties to agreements based on this specification are encouraged to investigate the possibility of applying the most recent editions of the standards and specifications indicated below.

1.5.1 ZigBee Alliance Documents

[B1] ZigBee document 053474, ZigBee Specification, ZigBee Alliance.

[B2] ZigBee Document 064321, The ZigBee Stack Profile, ZigBee Alliance

[B3] ZigBee Document 074855, The ZigBee PRO Stack Profile, ZigBee Alliance

1.5.2 International Standards Documents

[B4] CIE 1931 Color Space. Commission Internationale de l'Eclairage Proceedings. Cambridge University Press, Cambridge

1.5.3 National Standards Documents

[B5] EN 50131 European Standards Series for Intruder Alarm Systems

1.5.4 IEEE Documents

[B6] IEEE Standards 802, Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) specifications for Low Rate Wireless Personal Area Networks (LR-WPANs), IEEE, October 2003.

[B7] IEEE 754-1985, IEEE Standard for Binary Floating-Point Arithmetic, IEEE, 1985.

1.5.5 ASHRAE Documents

[B8] ASHRAE 135-2004 standard, Data Communication Protocol for Building Automation and Control Networks

CHAPTER

2

FOUNDATION SPECIFICATION

2.1 Scope and Purpose

This chapter provides an entry point into the documentation for the ZigBee cluster library (ZCL), and specifies the elements that are general across the entire library.

The ZCL frame structure is specified along with ZCL wide commands used to manipulate attributes from all the clusters defined throughout the ZCL. In addition, a set of data types is defined that can be used to represent attributes and a common set of status values returned by commands throughout the ZCL.

An overview is included which lists all the domains specified in the ZCL and the clusters contained therein.

2.2 Cluster Library Overview

The ZigBee Cluster Library (ZCL) is intended to act as a repository for cluster functionality that is developed by ZigBee and, as a consequence, it will be a working library with regular updates as new functionality is added. A developer constructing a new application profile should use the ZCL to find relevant cluster functionality that can be incorporated into the new profile so as not to “re-invent the wheel”. This also allows ZigBee profiles to be developed with more of an object oriented style approach.

2.2.1 Client/Server Model

Throughout the ZCL, a client/server model is employed. This model is illustrated in Figure 2.1

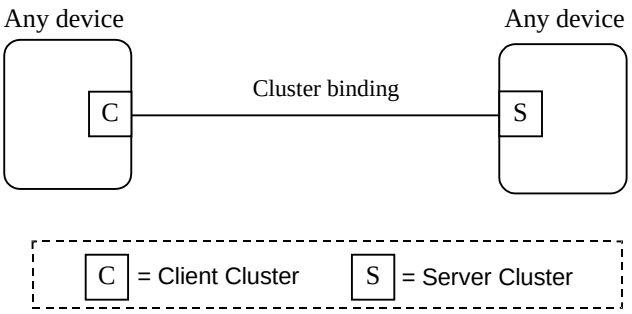

Figure 2.1 The ZCL Client Server Model

A cluster is a related collection of commands and attributes, which together define an interface to specific functionality. Typically, the entity that stores the attributes of a cluster is referred to as the server of that cluster and an entity that affects or manipulates those attributes is referred to as the client of that cluster. However, if required, attributes may also be present on the client of a cluster.

Commands that allow devices to manipulate attributes, e.g. in this document the read attribute (see 2.4.1) or write attribute (see 2.4.3) commands, are (typically) sent from a client device and received by the server device. Any response to those commands, e.g. in this document the read attribute response (see 2.4.2) or the write attribute response (see 2.4.5) commands, are sent from the server device and received by the client device.

Conversely, the command that facilitates dynamic attribute reporting, i.e. the report attribute command (see 2.4.11) is (typically) sent from the server device (as typically this is where the attribute data itself is stored) and sent to the client device that has been bound to the server device.

The clusters supported by an application object within an application profile are identified through the simple descriptor (see [B1]), specified on each active endpoint of a device. In the simple descriptor, the application input cluster list shall contain the list of server clusters supported on the device and the application output cluster list shall contain the list of client clusters supported on the device.

2.2.2 Functional Domains

The ZCL is divided into a number of functional domains, each domain addressing clusters relating to specific functionality. The functional domains defined in the ZCL are listed in Table 2.1.

Table 2.1 Functional Domains Defined in the ZCL

Functional Domain	Cluster ID Range
General	0x0000 – 0x00ff
Closures	0x0100 – 0x01ff
HVAC	0x0200 – 0x02ff
Lighting	0x0300 – 0x03ff
Measurement and sensing	0x0400 – 0x04ff
Security and safety	0x0500 – 0x05ff
Protocol interfaces	0x0600 – 0x06ff

The structure of each of these functional domains is described in the following sub-clauses.

2.2.2.1 General

The general functional domain contains clusters and information that provides generally applicable functions and attributes that are not specific to other functional domains.

This functional domain specifies the clusters listed in Table 2.2.

Table 2.2 Clusters of the General Functional Domain

Cluster ID	Cluster Name	Description
0x0000	Basic	Attributes for determining basic information about a device, setting user device information such as location, and enabling a device.
0x0001	Power configuration	Attributes for determining more detailed information about a device's power source(s), and for configuring under/over voltage alarms.
0x0002	Device Temperature Configuration	Attributes for determining information about a device's internal temperature, and for configuring under/over temperature alarms.
0x0003	Identify	Attributes and commands for putting a device into Identification mode (e.g. flashing a light)

Table 2.2 Clusters of the General Functional Domain (Continued)

Cluster ID	Cluster Name	Description
0x0004	Groups	Attributes and commands for group configuration and manipulation.
0x0005	Scenes	Attributes and commands for scene configuration and manipulation.
0x0006	On/off	Attributes and commands for switching devices between ‘On’ and ‘Off’ states.
0x0007	On/off Switch Configuration	Attributes and commands for configuring On/Off switching devices
0x0008	Level Control	Attributes and commands for controlling devices that can be set to a level between fully ‘On’ and fully ‘Off’.
0x0009	Alarms	Attributes and commands for sending notifications and configuring alarm functionality.
0x000a	Time	Attributes and commands that provide a basic interface to a real-time clock.
0x000b	RSSI Location	Attributes and commands that provide a means for exchanging location information and channel parameters among devices.
0x000c	Analog Input (Basic)	An interface for reading the value of an analog measurement and accessing various characteristics of that measurement.
0x000d	Analog Output (Basic)	An interface for setting the value of an analog output (typically to the environment) and accessing various characteristics of that value.
0x000e	Analog Value (Basic)	An interface for setting an analog value, typically used as a control system parameter, and accessing various characteristics of that value.
0x000f	Binary Input (Basic)	An interface for reading the value of a binary measurement and accessing various characteristics of that measurement.
0x0010	Binary Output (Basic)	An interface for setting the value of a binary output (typically to the environment) and accessing various characteristics of that value.
0x0011	Binary Value (Basic)	An interface for setting a binary value, typically used as a control system parameter, and accessing various characteristics of that value.
0x0012	Multistate Input (Basic)	An interface for reading the value of a multistate measurement and accessing various characteristics of that measurement.

Table 2.2 Clusters of the General Functional Domain (Continued)

Cluster ID	Cluster Name	Description
0x0013	Multistate Output (Basic)	An interface for setting the value of a multistate output (typically to the environment) and accessing various characteristics of that value.
0x0014	Multistate Value (Basic)	An interface for setting a multistate value, typically used as a control system parameter, and accessing various characteristics of that value.
0x0015	Commissioning	Attributes and commands for commissioning and managing a ZigBee device.
0x0016 – 0x00ff	-	Reserved.

2.2.2.2 Closures

The closures functional domain contains clusters and information to build devices in the closure domain, e.g. shade controllers.

This functional domain specifies the clusters listed in Table 2.3.

Table 2.3 Clusters of the Closures Functional Domain

Cluster ID	Cluster Name	Description
0x0100	Shade Configuration	Attributes and commands for configuring a shade.
0x0101	Door Lock	An interface for controlling a door lock.
0x0102 – 0x01ff	-	Reserved.

2.2.2.3 HVAC

The HVAC functional domain contains clusters and information to build devices in the HVAC domain, e.g. pumps.

This functional domain specifies the clusters listed in Table 2.4

Table 2.4 Clusters of the HVAC Functional Domain

Cluster ID	Cluster Name	Description
0x0200	Pump Configuration and Control	An interface for configuring and controlling pumps.
0x0201	Thermostat	An interface for configuring and controlling the functionality of a thermostat.
0x0202	Fan Control	An interface for controlling a fan in a heating / cooling system.
0x0203	Dehumidification Control	An interface for controlling dehumidification.
0x0204	Thermostat User Interface Configuration	An interface for configuring the user interface of a thermostat (which may be remote from the thermostat).
0x0205 – 0x02ff	-	Reserved

2.2.2.4 Lighting

The lighting functional domain contains clusters and information to build devices in the lighting domain, e.g. ballast units.

This functional domain specifies the clusters listed in Table 2.5

Table 2.5 Clusters of the Lighting Functional Domain

Cluster ID	Cluster Name	Description
0x0300	Color control	Attributes and commands for controlling the color properties of a color-capable light
0x0301	Ballast Configuration	Attributes and commands for configuring a lighting ballast
0x0302 – 0x03ff	-	Reserved.

2.2.2.5 Measurement and Sensing

The measurement and sensing functional domain contains clusters and information to build devices in the measurement and sensing domain, e.g. a temperature sensor or an occupancy sensor.

This functional domain specifies the clusters listed in Table 2.6.

Table 2.6 Clusters - Measurement and Sensing Functional Domain

Cluster ID	Cluster Name	Description
0x0400	Illuminance measurement	Attributes and commands for configuring the measurement of illuminance, and reporting illuminance measurements.
0x0401	Illuminance level sensing	Attributes and commands for configuring the sensing of illuminance levels, and reporting whether illuminance is above, below, or on target.
0x0402	Temperature measurement	Attributes and commands for configuring the measurement of temperature, and reporting temperature measurements.
0x0403	Pressure measurement	Attributes and commands for configuring the measurement of pressure, and reporting pressure measurements.
0x0404	Flow measurement	Attributes and commands for configuring the measurement of flow, and reporting flow rates.
0x0405	Relative humidity measurement	Attributes and commands for configuring the measurement of relative humidity, and reporting relative humidity measurements.
0x0406	Occupancy sensing	Attributes and commands for configuring occupancy sensing, and reporting occupancy status.
0x0407 – 0x04ff	-	Reserved.

2.2.2.6 Security and Safety

The security and safety functional domain contains clusters and information to build devices in the security and safety domain, e.g. alarm units.

This functional domain specifies the clusters listed in Table 2.7.

Table 2.7 Clusters of the Security and Safety Functional Domain

Cluster ID	Cluster Name	Description
0x0500	IAS Zone	Attributes and commands for IAS security zone devices.
0x0501	IAS ACE	Attributes and commands for IAS Ancillary Control Equipment.
0x0502	IAS WD	Attributes and commands for IAS Warning Devices.
0x0503 – 0x05ff	-	Reserved.

2.2.2.7 Protocol Interfaces

The protocol interfaces functional domain contains clusters and information to build devices to interface to other protocols, e.g. BACnet.

This functional domain specifies the clusters listed in Table 2.8.

Table 2.8 Clusters of the Protocol Interfaces Functional Domain

Cluster ID	Cluster Name	Description
0x0600	Generic Tunnel	The minimum common commands and attributes required to tunnel any protocol.
0x0601	BACnet Protocol Tunnel	Commands and attributes required to tunnel the BACnet protocol.
0x0602	Analog Input (BACnet Regular)	An interface for accessing a number of commonly used BACnet based attributes of an analog measurement.
0x0603	Analog Input (BACnet Extended)	An interface for accessing a number of BACnet based attributes of an analog measurement.
0x0604	Analog Output (BACnet Regular)	An interface for accessing a number of commonly used BACnet based attributes of an analog output.
0x0605	Analog Output (BACnet Extended)	An interface for accessing a number of BACnet based attributes of an analog output.
0x0606	Analog Value (BACnet Regular)	An interface for accessing a number of commonly used BACnet based attributes of an analog value, typically used as a control system parameter.
0x0607	Analog Value (BACnet Extended)	An interface for accessing a number of BACnet based attributes of an analog value, typically used as a control system parameter.
0x0608	Binary Input (BACnet Regular)	An interface for accessing a number of commonly used BACnet based attributes of a binary measurement.
0x0609	Binary Input (BACnet Extended)	An interface for accessing a number of BACnet based attributes of a binary measurement.
0x060a	Binary Output (BACnet Regular)	An interface for accessing a number of commonly used BACnet based attributes of a binary output.
0x060b	Binary Output (BACnet Extended)	An interface for accessing a number of BACnet based attributes of a binary output.
0x060c	Binary Value (BACnet Regular)	An interface for accessing a number of commonly used BACnet based attributes of a binary value, typically used as a control system parameter.

Table 2.8 Clusters of the Protocol Interfaces Functional Domain

Cluster ID	Cluster Name	Description
0x060d	Binary Value (BACnet Extended)	An interface for accessing a number of BACnet based attributes of a binary value, typically used as a control system parameter.
0x060e	Multistate Input (BACnet Regular)	An interface for accessing a number of commonly used BACnet based attributes of a multistate measurement.
0x060f	Multistate Input (BACnet Extended)	An interface for accessing a number of BACnet based attributes of a multistate measurement.
0x0610	Multistate Output (BACnet Regular)	An interface for accessing a number of commonly used BACnet based attributes of a multistate output.
0x0611	Multistate Output (BACnet Extended)	An interface for accessing a number of BACnet based attributes of a multistate output.
0x0612	Multistate Value (BACnet Regular)	An interface for accessing a number of commonly used BACnet based attributes of a multistate value, typically used as a control system parameter.
0x0613	Multistate Value (BACnet Extended)	An interface for accessing a number of BACnet based attributes of a multistate value, typically used as a control system parameter.
0x0614 – 0x06ff	-	Reserved.

2.3 Command Frame Formats

All commands, defined in this specification, shall be transmitted to the ZigBee stack using the message service.

The transmission order for octets and bits of all ZCL elements is as specified in section 1.2.1.3 of the ZigBee Specification [B1], i.e. least significant octet and bit first.

2.3.1 General ZCL Frame Format

The ZCL frame format is composed of a ZCL header and a ZCL payload. The general ZCL frame shall be formatted as illustrated in Figure 2.2.

Bits: 8	0/16	8	8	Variable
Frame control	Manufacturer code	Transaction sequence number	Command identifier	Frame payload
ZCL header				ZCL payload

Figure 2.2 Format of the General ZCL Frame

2.3.1.1 Frame Control Field

The frame control field is 8 bits in length and contains information defining the command type and other control flags. The frame control field shall be formatted as shown in Figure 2.3. Bits 5-7 are reserved for future use and shall be set to 0.

Bits: 0-1	2	3	4	5-7
Frame type	Manufacturer specific	Direction	Disable default response	Reserved

Figure 2.3 Format of the Frame Control Field

2.3.1.1.1 Frame Type Sub-field

The frame type sub-field is 2 bits in length and shall be set to one of the non-reserved values listed in Figure 2.4.

Frame Type Value b_1b_0	Description
00	Command acts across the entire profile
01	Command is specific to a cluster
10-11	Reserved

Figure 2.4 Values of the Frame Type Sub-field

2.3.1.1.2 Manufacturer Specific Sub-field

The manufacturer specific sub-field is 1 bit in length and specifies whether this command refers to a manufacturer specific extension to a profile. If this value is set to 1, the manufacturer code field shall be present in the ZCL frame. If this value is set to 0, the manufacturer code field shall not be included in the ZCL frame.

2.3.1.1.3 Direction Sub-field

The direction sub-field specifies the client/server direction for this command. If this value is set to 1, the command is being sent from the server side of a cluster to the client side of a cluster. If this value is set to 0, the command is being sent from the client side of a cluster to the server side of a cluster.

2.3.1.1.4 Disable Default Response Sub-field

The disable default response sub-field is 1 bit in length. If it is set to 0, the Default response command will be returned, under the conditions specified in 2.4.12.2. If it is set to 1, the Default response command will only be returned if there is an error, also under the conditions specified in 2.4.12.2.

2.3.1.2 Manufacturer Code Field

The manufacturer code field is 16 bits in length and specifies the ZigBee assigned manufacturer code for proprietary extensions to a profile. This field shall only be included in the ZCL frame if the manufacturer specific sub-field of the frame control field is set to 1.

2.3.1.3 Transaction Sequence Number

The transaction sequence number field is 8 bits in length and specifies an identification number for the transaction so that a response-style command frame can be related to a request-style command frame. The application object itself shall maintain an 8-bit counter that is copied into this field and incremented by one for each command sent. When a value of 0xff is reached, the next command shall re-start the counter with a value of 0x00.

The transaction sequence number field can be used by a controlling device, which may have issued multiple commands, so that it can match the incoming responses to the relevant command.

2.3.1.4 Command Identifier Field

The command identifier field is 8 bits in length and specifies the cluster command being used. If the frame type sub-field of the frame control field is set to 0b00, the command identifier corresponds to one of the non-reserved values of Table 2.9. If the frame type sub-field of the frame control field is set to 0b01, the command identifier corresponds to a cluster specific command. The cluster specific command identifiers can be found in each individual document describing the clusters (see also 2.2.1).

2.3.1.5 Frame Payload Field

The frame payload field has a variable length and contains information specific to individual command types. The maximum payload length for a given command is

limited by the stack profile in use, in conjunction with the applicable cluster specification and application profile. Fragmentation will be used where available.

2.4 General Command Frames

General command frames are used for manipulating attributes and other general tasks that are not specific to an individual cluster.

The command frames defined in this document are listed in Table 2.9. Each command frame shall be constructed with the frame type sub-field of the frame control field set to 0b00.

All clusters (server and client) shall support generation, reception and execution of the Default response command.

Each cluster (server or client) that implements attributes shall support reception of, execution of, and response to all commands to discover, read, and write these attributes. However, if no attributes with structured types are supported, it is not required to support the structured read and write commands.

Implementation of commands to report, configure reporting of, and read reporting configuration of attributes is only mandatory if the cluster has attributes whose reportability is mandatory.

Generation of these commands is application dependent.

Table 2.9 ZCL Command Frames

Command Identifier Field Value	Description
0x00	Read attributes
0x01	Read attributes response
0x02	Write attributes
0x03	Write attributes undivided
0x04	Write attributes response
0x05	Write attributes no response
0x06	Configure reporting
0x07	Configure reporting response
0x08	Read reporting configuration
0x09	Read reporting configuration response
0x0a	Report attributes

Table 2.9 ZCL Command Frames (Continued)

0x0b	Default response
0x0c	Discover attributes
0x0d	Discover attributes response
0x0e	Read attributes structured
0x0f	Write attributes structured
0x10	Write attributes structured response
0x11 – 0xff	Reserved

2.4.1 Read Attributes Command

2.4.1.1 Read Attributes Command Frame Format

The read attributes command frame shall be formatted as illustrated in Figure 2.5.

Octets: Variable	**2**	**2**	**…**	**2**
ZCL header	Attribute identifier 1	Attribute identifier 2	…	Attribute identifier *n*

Figure 2.5 Format of the Read Attributes Command Frame

2.4.1.1.1 ZCL Header Fields

The frame control field shall be specified as follows. The frame type sub-field shall be set to indicate a profile wide command (0b00). The manufacturer specific sub-field shall be set to 0 if this command is being used to read attributes defined for any cluster in the ZCL or 1 if this command is being used to read manufacturer specific attributes.

The command identifier field shall be set to indicate the read attributes command (see Table 2.9).

2.4.1.1.2 Attribute Identifier Field

The attribute identifier field is 16 bits in length and shall contain the identifier of the attribute that is to be read.

2.4.1.2 When Generated

The read attributes command is generated when a device wishes to determine the values of one or more attributes located on another device. Each attribute identifier field shall contain the identifier of the attribute to be read.

2.4.1.3 Effect on Receipt

On receipt of this command, the device shall process each specified attribute identifier and generate a read attributes response command. The read attributes response command shall contain as many read attribute status records as attribute identifiers included in this command frame. Each read attribute status record shall contain the corresponding attribute identifier from this command frame, a status value evaluated as described below, and, depending on the status value, the value of the attribute itself.

For each attribute identifier included in the command frame, the device shall first check that it corresponds to an attribute that exists on this device. If it does not, the device shall set the status field of the corresponding read attribute status record to UNSUPPORTED_ATTRIBUTE and shall not include an attribute value field. The device shall then move on to the next attribute identifier.

If the attribute identified by the attribute identifier is supported, the device shall set the status field of the corresponding read attribute status record to SUCCESS and shall set the attribute value field to its current value. The device shall then move on to the next attribute identifier.

2.4.2 Read Attributes Response Command

2.4.2.1 Read Attributes Response Command Frame Format

The read attributes response command frame shall be formatted as illustrated in Figure 2.6.

Octets: Variable	**Variable**	**Variable**	**…**	**Variable**
ZCL header	Read attribute status record 1	Read attribute status record 2	…	Read attribute status record *n*

Figure 2.6 Format of Read Attributes Response Command Frame

Each read attribute status record shall be formatted as illustrated in Figure 2.7

Octets: 2	1	0 / 1	0 / Variable
Attribute identifier	Status	Attribute data type	Attribute value

Figure 2.7 Format of the Read Attributes Status Record Field

2.4.2.1.1 ZCL Header Fields

The frame control field shall be specified as follows. The frame type sub-field shall be set to indicate a profile wide command (0b00). The manufacturer specific sub-field shall be set to 0 if this command is being used as a response to reading attributes defined for any cluster in the ZCL or 1 if this command is being used as a response to reading manufacturer specific attributes.

The command identifier field shall be set to indicate the read attributes response command (see Table 2.9).

2.4.2.1.2 Attribute Identifier Field

The attribute identifier field is 16 bits in length and shall contain the identifier of the attribute that has been read (or of which an element has been read). This field shall contain the same value that was included in the corresponding attribute identifier field of the original read attributes or read attributes structured command.

2.4.2.1.3 Status Field

The status field is 8 bits in length and specifies the status of the read operation on this attribute. This field shall be set to SUCCESS, if the operation was successful, or an error code, as specified in 2.4.1.3, if the operation was not successful.

2.4.2.1.4 Attribute Data Type Field

The attribute data type field shall contain the data type of the attribute in the same read attributes status record (see Table 2.16). This field shall only be included if the associated status field contains a value of SUCCESS.

2.4.2.1.5 Attribute Value Field

The attribute value field is variable in length and shall contain the current value of this attribute. This field shall only be included if the associated status field contains a value of SUCCESS.

For an attribute or element of simple type (not array, structure, set or bag), this field has the format shown in the Table of Data Types (Table 2.16).

For an attribute or element of type array, set or bag, this field has the format shown in Figure 2.8.

Octets: 1	2	Variable	…	Variable
Element type	Number of elements (*m*)	Element value 1	…	Element value *m*

Figure 2.8 Format of the Attribute Value Field for an Array, Set or Bag

(NB The reason that the Element type field is before the Number of elements field is so that the latter field is in the logical position for the zeroth element.)

If the Number of elements field has the value 0xffff, this indicates that the attribute or element being read is invalid / undefined. In this case, or if the Number of elements field has the value 0, no Element value fields are included.

For an attribute or element of type structure, this field has the format shown in Figure 2.9.

Octets: 2	1	Variable	…	1	Variable
Number of elements (*m*)	Element type 1	Element value 1	…	Element type *m*	Element value *m*

Figure 2.9 Format of the Attribute Value Field for a Structure

In both figures, the Element value subfield follows the same format as that of the attribute value field. This format is thus recursive to any required depth (see sub-clause 2.4.15.1.3 for limitations).

If the Number of elements field has the value 0xffff, this indicates that the attribute or element being read is invalid / undefined. In this case, or if the Number of elements field has the value 0, no Element type or Element value fields are included.

2.4.2.2 When Generated

The read attributes response command is generated in response to a read attributes or read attributes structured command. The command frame shall contain a read attribute status record for each attribute identifier specified in the original read attributes or read attributes structured command. For each read attribute status record, the attribute identifier field shall contain the identifier specified in the original read attributes or read attributes structured command. The status field shall contain a suitable status code, as detailed in Clause 2.4.1.3.

The attribute data type and attribute value field shall only be included in the read attribute status record if the associated status field contains a value of SUCCESS

and, where present, shall contain the data type and current value, respectively, of the attribute, or element thereof, that was read.

The length of this command may exceed a single frame, and thus fragmentation support may be needed to return the entire response If fragmentation is not supported, only as many read attribute status records as will fit in the frame shall be returned.

2.4.2.3 Effect on Receipt

On receipt of this command, the originator is notified of the results of its original read attributes attempt and, for each successful request, the value of the requested attribute.

If fragmentation is not supported, and some trailing attribute status records have not been returned, due to space limitations in the frame, the originator may issue an additional read attributes or read attributes structured command to obtain their values.

2.4.3 Write Attributes Command

2.4.3.1 Write Attributes Command Frame Format

The write attributes command frame shall be formatted as illustrated in Figure 2.10.

Octets: Variable	**Variable**	**Variable**	**…**	**Variable**
ZCL header	Write attribute record 1	Write attribute record 2	…	Write attribute record *n*

Figure 2.10 Format of the Write Attributes Command Frame

Each write attribute record shall be formatted as illustrated in Figure 2.11.

Octets: 2	**1**	**Variable**
Attribute identifier	Attribute data type	Attribute data

Figure 2.11 Format of the Write Attribute Record Field

2.4.3.1.1 ZCL Header Fields

The frame control field shall be specified as follows. The frame type sub-field shall be set to indicate a profile wide command (0b00). The manufacturer specific sub-field shall be set to 0 if this command is being used to write attributes defined for any cluster in the ZCL or 1 if this command is being used to write manufacturer specific attributes.

The command identifier field shall be set to indicate the write attributes command (see Table 2.9).

2.4.3.1.2 Attribute Identifier Field

The attribute identifier field is 16 bits in length and shall contain the identifier of the attribute that is to be written.

2.4.3.1.3 Attribute Data Type Field

The attribute data type field shall contain the data type of the attribute that is to be written.

2.4.3.1.4 Attribute Data Field

The attribute data field is variable in length and shall contain the actual value of the attribute that is to be written.

2.4.3.2 When Generated

The write attributes command is generated when a device wishes to change the values of one or more attributes located on another device. Each write attribute record shall contain the identifier and the actual value of the attribute to be written.

2.4.3.3 Effect on Receipt

On receipt of this command, the device shall attempt to process each specified write attribute record and shall construct a write attribute response command (2.4.5). Each write attribute status record of the constructed command shall contain the identifier from the corresponding write attribute record and a status value evaluated as described below.

For each write attribute record included in the command frame, the device shall make the error checks listed below, in the order shown. If an error is detected, a corresponding write attribute status record shall be generated, the status shall be set according to the check below, and the device shall move on to the next write attribute record.

1) If the attribute is not supported on this device, the status field of the corresponding write attribute status record shall be set to UNSUPPORTED_ATTRIBUTE.

2) If the attribute data type field is incorrect, the device shall set the status field of the corresponding write attribute status record to INVALID_DATA_TYPE.

3) If the attribute is designated as read only, the device shall set the status field of the corresponding write attribute status record to READ_ONLY.

4) If the device is not currently accepting write attribute commands for the attribute, the status field of the corresponding write attribute status record shall be set to NOT_AUTHORIZED or READ_ONLY.[1]

5) If the supplied value is not within the specified range of the attribute, the status field of the corresponding write attribute status record shall be set to INVALID_VALUE.

If the above error checks pass without generating a write attribute status record, the device shall write the supplied value to the identified attribute, and shall move on to the next write attribute record.

When all write attribute records have been processed, the device shall generate the constructed write attributes response command. If there are no write attribute status records in the constructed command, indicating that all attributes were written successfully, a single write attribute status record shall be included in the command, with the status field set to SUCCESS and the attribute identifier field omitted.

2.4.4 Write Attributes Undivided Command

The write attributes undivided command is generated when a device wishes to change the values of one or more attributes located on another device, in such a way that if any attribute cannot be written (e.g. if an attribute is not implemented on the device, or a value to be written is outside its valid range), no attribute values are changed.

In all other respects, including generation of a write attributes response command, the format and operation of the command is the same as that of the write attributes command, except that the command identifier field shall be set to indicate the write attributes undivided command (see Table 2.9).

2.4.5 Write Attributes Response Command

2.4.5.1 Write Attributes Response Command Frame Format

The write attributes response command frame shall be formatted as illustrated in Figure 2.12.

1. CCB 1379

Octets: Variable	3	3	…	3
ZCL header	Write attribute status record 1	Write attribute status record 2	…	Write attribute status record *n*

Figure 2.12 Format of Write Attributes Response Command Frame

Each write attribute status record shall be formatted as illustrated in Figure 2.13.

Octets: 1	2
Status	Attribute identifier

Figure 2.13 Format of the Write Attribute Status Record Field

2.4.5.1.1 ZCL Header Fields

The frame control field shall be specified as follows. The frame type sub-field shall be set to indicate a profile wide command (0b00). The manufacturer specific sub-field shall be set to 0 if this command is being used as a response to writing attributes defined for any cluster in the ZCL or 1 if this command is being used as a response to writing manufacturer specific attributes.

The command identifier field shall be set to indicate the write attributes response command (see Table 2.9).

2.4.5.1.2 Status Field

The status field is 8 bits in length and specifies the status of the write operation attempted on this attribute, as detailed in 2.4.3.3.

Note that write attribute status records are not included for successfully written attributes, in order to save bandwidth. In the case of successful writing of all attributes, only a single write attribute status record shall be included in the command, with the status field set to SUCCESS and the attribute identifier field omitted.

2.4.5.1.3 Attribute Identifier Field

The attribute identifier field is 16 bits in length and shall contain the identifier of the attribute on which the write operation was attempted.

2.4.5.2 When Generated

The write attributes response command is generated in response to a write attributes command.

2.4.5.3 Effect on Receipt

On receipt of this command, the device is notified of the results of its original write attributes command.

2.4.6 Write Attributes No Response Command

2.4.6.1 Write Attributes No Response Command Frame Format

The write attributes no response command frame shall be formatted as illustrated in Figure 2.14.

Octets: Variable	Variable	Variable	…	Variable
ZCL header	Write attribute record 1	Write attribute record 2	…	Write attribute record *n*

Figure 2.14 Write Attributes No Response Command Frame

Each write attribute record shall be formatted as illustrated in Figure 2.11.

2.4.6.1.1 ZCL Header Fields

The frame control field shall be specified as follows. The frame type sub-field shall be set to indicate a profile wide command (0b00). The manufacturer specific sub-field shall be set to 0 if this command is being used to write attributes defined for any cluster in the ZCL or 1 if this command is being used to write manufacturer specific attributes.

The command identifier field shall be set to indicate the write attributes no response command (see Table 2.9).

2.4.6.1.2 Write Attribute Records

Each write attribute record shall be formatted as illustrated in Figure 2.11 Its fields have the same meaning and contents as the corresponding fields of the Write attributes command.

2.4.6.2 When Generated

The write attributes no response command is generated when a device wishes to change the value of one or more attributes located on another device but does not require a response. Each write attribute record shall contain the identifier and the actual value of the attribute to be written.

2.4.6.3 Effect on Receipt

On receipt of this command, the device shall attempt to process each specified write attribute record.

For each write attribute record included in the command frame, the device shall first check that it corresponds to an attribute that is implemented on this device. If it does not, the device shall ignore the attribute and move on to the next write attribute record.

If the attribute identified by the attribute identifier is supported, the device shall check whether the attribute is writable. If the attribute is designated as read only, the device shall ignore the attribute and move on to the next write attribute record.

If the attribute is writable, the device shall check that the supplied value in the attribute data field is within the specified range of the attribute. If the supplied value does not fall within the specified range of the attribute, the device shall ignore the attribute and move on to the next write attribute record.

If the value supplied in the attribute data field is within the specified range of the attribute, the device shall write the supplied value to the identified attribute and move on to the next write attribute record.

2.4.7 Configure Reporting Command

The Configure Reporting command is used to configure the reporting mechanism for one or more of the attributes of a cluster.

The individual cluster definitions specify which attributes shall be available to this reporting mechanism, however specific implementations of a cluster may make additional attributes available.

Note that attributes with data types of array, structure, set or bag cannot be reported.

2.4.7.1 Configure Reporting Command Frame Format

The Configure Reporting command frame shall be formatted as illustrated in Figure 2.15.

Octets: Variable	Variable	Variable	…	Variable
ZCL header	Attribute reporting configuration record 1	Attribute reporting configuration record 2	…	Attribute reporting configuration record n

Figure 2.15 Format of the Configure Reporting Command Frame

There shall be one attribute reporting configuration record for each attribute to be configured. Each such record shall be formatted as illustrated in Figure 2.16.

Octets: 1	2	0/1	0/2	0/2	0/Variable	0/2
Direction	Attribute identifier	Attribute data type	Minimum reporting interval	Maximum reporting interval	Reportable change	Timeout period

Figure 2.16 Format of the Attribute Reporting Configuration Record

2.4.7.1.1 ZCL Header Fields

The frame control field shall be specified as follows. The frame type sub-field shall be set to indicate a profile wide command (0b00). The manufacturer specific sub-field shall be set to 0 if this command is being used to configure attribute reports defined for any cluster in the ZCL or 1 if this command is being used to configure attribute reports for manufacturer specific attributes.

The command identifier field shall be set to indicate the report configuration command (see Table 2.9).

2.4.7.1.2 Direction Field

The direction field specifies whether values of the attribute are be reported, or whether reports of the attribute are to be received.

If this value is set to 0x00, then the attribute data type field, the minimum reporting interval field, the maximum reporting interval field and the reportable change field are included in the payload, and the timeout period field is omitted. The record is sent to a cluster server (or client) to configure how it sends reports to a client (or server) of the same cluster.

If this value is set to 0x01, then the timeout period field is included in the payload, and the attribute data type field, the minimum reporting interval field, the maximum reporting interval field and the reportable change field are omitted. The record is sent to a cluster client (or server) to configure how it should expect reports from a server (or client) of the same cluster.

All other values of this field are reserved.

Table 2.10 Destination of Reporting Based on Direction Field[a]

Direction Field	Destinations
0x00	The receiver of the Configure Reporting command shall configure reporting to send to each destination as resolved by the bindings for the cluster hosting the attributes to be reported.
0x01	This indicates to the receiver of the Configure Reporting command that the sender has configured its reporting mechanism to transmit reports and that, based on the current state of the sender's bindings, the sender will send reports to the receiver.
Other values	Reserved

a. CCB 1390

2.4.7.1.3 Attribute Identifier Field

If the direction field is 0x00, this field contains the identifier of the attribute that is to be reported. If instead the direction field is 0x01, the device shall expect reports of values of this attribute.

2.4.7.1.4 Attribute Data Type Field

The Attribute data type field contains the data type of the attribute that is to be reported.

2.4.7.1.5 Minimum Reporting Interval Field

The minimum reporting interval field is 16 bits in length and shall contain the minimum interval, in seconds, between issuing reports of the specified attribute.

If this value is set to 0x0000, then there is no minimum limit, unless one is imposed by the specification of the cluster using this reporting mechanism or by the applicable profile.

2.4.7.1.6 Maximum Reporting Interval Field

The maximum reporting interval field is 16 bits in length and shall contain the maximum interval, in seconds, between issuing reports of the specified attribute.

If this value is set to 0xffff, then the device shall not issue reports for the specified attribute, and the configuration information for that attribute need not be maintained. (Note: in an implementation using dynamic memory allocation, the memory space for that information may then be reclaimed).

2.4.7.1.7 Reportable Change Field

The reportable change field shall contain the minimum change to the attribute that will result in a report being issued. This field is of variable length. For attributes with 'analog' data type (see Table 2.16) the field has the same data type as the attribute. The sign (if any) of the reportable change field is ignored.

For attributes of 'discrete' data type (see Table 2.16) this field is omitted.

2.4.7.1.8 Timeout Period Field

The timeout period field is 16 bits in length and shall contain the maximum expected time, in seconds, between received reports for the attribute specified in the attribute identifier field. If more time than this elapses between reports, this may be an indication that there is a problem with reporting.

If this value is set to 0x0000, reports of the attribute are not subject to timeout.

Note that, for a server/client connection to work properly using automatic reporting, the timeout value set for attribute reports to be received by the client (or server) cluster must be set somewhat higher than the maximum reporting interval set for the attribute on the server (or client) cluster.

2.4.7.2 When Generated

The report configuration command is generated when a device wishes to configure a device to automatically report the values of one or more of its attributes, or to receive such reports.

2.4.7.3 Effect on Receipt

On receipt of this command, the device shall attempt to process each attribute reporting configuration record and shall construct a configure reporting response command. Each attribute status record of the constructed command shall contain an identifier from an attribute reporting configuration record and a status value evaluated as described below.

If the direction field is 0x00, indicating that the reporting intervals and reportable change are being configured, then

- If the attribute specified in the attribute identifier field is not implemented on this device or if the attribute type is set to array, structure, set or bag, the device shall construct an attribute status record with the status field set to UNSUPPORTED_ATTRIBUTE.
- Else, if the attribute identifier in this field cannot be reported (because it is not in the list of mandatory reportable attributes in the relevant cluster specification, and support has also not been implemented as a manufacturer option), the device shall construct an attribute status record with the status field set to UNREPORTABLE_ATTRIBUTE.

- Else, if the attribute data type field is incorrect, the device shall construct an attribute status record with the status field set to INVALID_DATA_TYPE.

Else, if the minimum reporting interval field is less than any minimum set by the relevant cluster specification or application profile, or the value of the maximum reporting interval field is non-zero and is less than that of the minimum reporting interval field, the device shall construct an attribute status record with the status field set to INVALID_VALUE.

Else the device shall set the minimum and maximum reporting intervals and the reportable change for the attribute to the values contained in the corresponding fields.

Else the direction field is 0x01, indicating that the timeout period is being configured, then

If reports of values of the attribute identifier specified in the attribute identifier field cannot be received (because it is not in the list of mandatory reportable attributes in the relevant cluster specification, and support has also not been implemented as a manufacturer option), or the timeout feature is not supported, the device shall construct an attribute status record with the status field set to UNSUPPORTED_ATTRIBUTE.

Else the device shall set the timeout value for the attribute identifier specified in the attribute identifier field to the value of the timeout period field. Note that the action to be taken by the device if the timeout period is exceeded is cluster and device dependent, including optionally taking no action.

When all attribute reporting configuration records have been processed, the device shall generate the constructed configure reporting response command. If there are no attribute status records in the constructed command, indicating that all attributes were configured successfully, a single attribute status record shall be included in the command, with the status field set to SUCCESS and the direction and attribute identifier fields omitted.

The device shall then proceed to generate or receive attribute reports according the configuration just set up, by means of the Report attributes command (see 2.4.11.2.1 through 2.4.11.2.4). See Table 2.10, “Destination of Reporting Based on Direction Field” to determine the destination of the Report Attributes command.[2]

2.4.8 Configure Reporting Response Command

The Configure Reporting Response command is used to respond to a Configure Reporting command.

2. CCB 1390

2.4.8.1 Configure Reporting Response Command Frame Format

The Configure Reporting Response command frame shall be formatted as illustrated in Figure 2.17

Octets: Variable	4	4	…	4
ZCL header	Attribute status record 1	Attribute status record 2	…	Attribute status record *n*

Figure 2.17 Format of the Configure Reporting Response Command Frame

Each attribute status record shall be formatted as illustrated in Figure 2.18.

Octets: 1	1	2
Status	Direction	Attribute identifier

Figure 2.18 Format of the Attribute Status Record Field

2.4.8.1.1 ZCL Header Fields

The frame control field is specified as follows. The frame type sub-field shall be set to indicate a profile wide command (0b00). The manufacturer specific sub-field shall be set to 0 if this command is being used as a response to configuring attribute reports defined for any cluster in the ZCL or 1 if this command is being used as a response to configuring attribute reports for manufacturer specific attributes.

The command identifier field shall be set to indicate the report configuration response command (see Table 2.9).

2.4.8.1.2 Direction Field

The direction field specifies whether values of the attribute are reported (0x00), or whether reports of the attribute are received (0x01).

All other values of this field are reserved.

2.4.8.1.3 Status Field

The status field specifies the status of the configure reporting operation attempted on this attribute, as detailed in 2.4.7.3.

Note that attribute status records are not included for successfully configured attributes, in order to save bandwidth. In the case of successful configuration of all attributes, only a single attribute status record shall be included in the command,

with the status field set to SUCCESS and the direction and attribute identifier fields omitted.

2.4.8.2 When Generated

The Configure Reporting Response command is generated in response to a Configure Reporting command.

2.4.8.3 Effect on Receipt

On receipt of this command, the device is notified of the success (or otherwise) of its original configure reporting command, for each attribute.

2.4.9 Read Reporting Configuration Command

The Read Reporting Configuration command is used to read the configuration details of the reporting mechanism for one or more of the attributes of a cluster.

2.4.9.1 Read Reporting Configuration Command Frame Format

The Read Reporting Configuration command frame shall be formatted as illustrated in Figure 2.19

Octets: Variable	**3**	**3**	**…**	**3**
ZCL header	Attribute record 1	Attribute record 2	…	Attribute record *n*

Figure 2.19 Read Reporting Configuration Command Frame

Each attribute record shall be formatted as illustrated in Figure 2.20.

Octets: 1	**2**
Direction	Attribute identifier

Figure 2.20 Format of the Attribute Status Record Field

2.4.9.1.1 ZCL Header Fields

The frame control field shall be specified as follows. The frame type sub-field shall be set to indicate a profile wide command (0b00). The manufacturer specific sub-field shall be set to 0 if this command is being used to read the reporting configuration of attributes defined for any cluster in the ZCL or 1 if this command

is being used to read the reporting configuration of manufacturer specific attributes.

The command identifier field shall be set to indicate the read reporting configuration command (see Table 2.9).

2.4.9.1.2 Direction Field

The direction field specifies whether values of the attribute are reported (0x00), or whether reports of the attribute are received (0x01).

All other values of this field are reserved.

2.4.9.1.3 Attribute Identifier Field

The attribute identifier field shall contain the identifier of the attribute whose reporting configuration details are to be read.

2.4.9.2 Effect on Receipt

On receipt of this command, a device shall generate a read reporting configuration response command containing the details of the reporting configuration for each of the attributes specified in the command (see 2.4.10).

2.4.10 Read Reporting Configuration Response Command

The Read Reporting Configuration Response command is used to respond to a Read Reporting Configuration command.

2.4.10.1 Read Reporting Configuration Response Command Frame Format

The read reporting configuration response command frame shall be formatted as illustrated in Figure 2.21

Octets: Variable	Variable	Variable	…	Variable
ZCL header	Attribute reporting configuration record 1	Attribute reporting configuration record 2	…	Attribute reporting configuration record n

Figure 2.21 Format of the Read Reporting Configuration Response Command Frame

There shall be one attribute reporting configuration record for each attribute record of the received read reporting configuration command. Each such record shall be formatted as illustrated in Figure 2.22.

Octets: 1	1	2	0/1	0/2	0/2	0/Variable	0/2
Status	Direction	Attribute identifier	Attribute data type	Minimum reporting interval	Maximum reporting interval	Reportable change	Timeout period

Figure 2.22 Attribute Reporting Configuration Record Field

2.4.10.1.1 ZCL Header Fields

The frame control field shall be specified as follows. The frame type sub-field shall be set to indicate a profile wide command (0b00). The manufacturer specific sub-field shall be set to 0 if this command is being used to for attributes specified in the ZCL or 1 if this command is being used for manufacturer specific attributes.

The command identifier field shall be set to indicate the Read reporting configuration response command (see Table 2.9).

2.4.10.1.2 Status Field

If the attribute is not implemented on the sender or receiver of the command, whichever is relevant (depending on direction), this field shall be set to UNSUPPORTED_ATTRIBUTE. If the attribute is supported, but is not capable of being reported, this field shall be set to UNREPORTABLE_ATTRIBUTE. Otherwise, this field shall be set to SUCCESS.

If the status field is not set to SUCCESS, all fields except the direction and attribute identifier fields shall be omitted.

2.4.10.1.3 Direction Field

The direction field specifies whether values of the attribute are reported (0x00), or whether reports of the attribute are received (0x01).

If this value is set to 0x00, then the attribute data type field, the minimum reporting interval field, the maximum reporting interval field and the reportable change field are included in the payload, and the timeout period field is omitted. If this value is set to 0x01, then the timeout period field is included in the payload, and the attribute data type field, the minimum reporting interval field, the maximum reporting interval field and the reportable change field are omitted.

All other values of this field are reserved.

2.4.10.1.4 Attribute Identifier Field

The attribute identifier field is 16 bits in length and shall contain the identifier of the attribute that the reporting configuration details apply to.

2.4.10.1.5 Minimum Reporting Interval Field

The minimum reporting interval field is 16 bits in length and shall 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 shall contain the value 0xffff.

2.4.10.1.6 Maximum Reporting Interval Field

The maximum reporting interval field is 16 bits in length and shall 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 shall contain the value 0xffff.

2.4.10.1.7 Reportable Change Field

The reportable change field shall contain the minimum change to the attribute that will result in a report being issued. For attributes with 'analog' data type (see Table 2.16) the field has the same data type as the attribute. If the reportable change has not been configured, this field shall contain the invalid value for the relevant data type.

For attributes of 'discrete' data type (see Table 2.16) this field is omitted.

2.4.10.1.8 Timeout Period Field

The timeout period field is 16 bits in length and shall 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 shall contain the value 0xffff.

2.4.10.2 When Generated

The read reporting configuration response command is generated in response to a read reporting configuration command. Only as many attribute reporting configuration records as will fit in the frame shall be returned.

2.4.10.3 Effect on Receipt

On receipt of this command, the originator is notified of the results of its original read reporting configuration command.

If some trailing attribute reporting configuration records have not been returned, due to space limitations in the frame, the originator may issue a further read reporting configuration command to obtain their values.

2.4.11 Report Attributes Command

The report attributes command is used by a device to report the values of one or more of its attributes to another device. Individual clusters, defined elsewhere in the ZCL, define which attributes are to be reported and at what interval. See sub-clause 2.4.7 “Configure Reporting Command” to determine the destination of the Report Attributes command.[3]

2.4.11.1 Report Attributes Command Frame Format

The report attributes command frame shall be formatted as illustrated in Figure 2.23.

Octets: Variable	Variable	Variable	…	Variable
ZCL header	Attribute report 1	Attribute report 2	…	Attribute report *n*

Figure 2.23 Format of the Report Attributes Command Frame

Each attribute report field shall be formatted as illustrated in Figure 2.24.

Octets: 2	1	Variable
Attribute identifier	Attribute data type	Attribute data

Figure 2.24 Format of the Attribute Report Fields

2.4.11.1.1 ZCL Header Fields

The frame control field shall be specified as follows. The frame type sub-field shall be set to indicate a profile wide command (0b00). The manufacturer specific sub-field shall be set to 0 if this command is being used to report attributes defined for any cluster in the ZCL or 1 if this command is being used to report manufacturer specific attributes.

The command identifier field shall be set to indicate the report attributes command (see Table 2.9).

3. CCB 1390

2.4.11.1.2 Attribute Identifier Field

The attribute identifier field is 16 bits in length and shall contain the identifier of the attribute that is being reported.

2.4.11.1.3 Attribute Data Type Field

The attribute data type field contains the data type of the attribute that is being reported.

2.4.11.1.4 Attribute Data Field

The attribute data field is variable in length and shall contain the actual value of the attribute being reported.

2.4.11.2 When Generated

The report attributes command is generated when a device has been configured to report the values of one or more of its attributes to another device., and when the conditions that have been configured are satisfied. These conditions are detailed in the following sections.

A report attributes command may also be configured locally on a device at any time. Except for the source, a locally created report configuration shall be no different than a configuration received externally. A locally created report configuration shall support the same services as a configuration received externally.[4]

If the destination of the Report Attributes Command can not be determined, then the command shall not be generated. See sub-clause 2.4.7 “Configure Reporting Command” to determine the destination of the Report Attributes command.[5]

2.4.11.2.1 Periodic Reporting

A report shall be generated when the time that has elapsed since the previous report of the same attribute is equal to the Maximum Reporting Interval for that attribute (see 2.4.7.1.6). The time of the first report after configuration is not specified.

If the Maximum Reporting Interval is set to 0x0000, there is no periodic reporting, but change based reporting is still operational.

If the Maximum Reporting Interval is set to 0xffff, no reports shall be generated, whatever other conditions are satisfied.

4. CCB 1146
5. CCB 1390

2.4.11.2.2 Changes to 'Discrete' Attributes

If the attribute has a 'discrete' data type, a report shall be generated when the attribute undergoes any change of value. Discrete types are general data types (which are often used as sets of bit fields), logical types, bitmap types, enumerations, strings, identifiers, IEEE address and security key (see Table 2.16).

Reporting is subject to the Minimum Reporting Interval for that attribute (see 2.4.7.1.5). After a report, no further reports are sent during this interval.

2.4.11.2.3 Changes to 'Analog' Attributes

If the attribute has an 'analog' data type, a report shall be generated when the attribute undergoes a change of value, in a positive or negative direction, equal to or greater than the Reportable Change for that attribute (see 2.4.7.1.7). The change is measured from the value of the attribute when the Reportable Change is configured, and thereafter from the previously reported value of the attribute.

Analog types are signed and unsigned integer types, floating point types and time types (see Table 2.16).

Reporting is subject to the Minimum Reporting Interval for that attribute (see 2.4.7.1.5). After a report, no further reports are sent during this interval.

2.4.11.2.4 Cluster Specific Conditions

The specification for a cluster may add additional conditions for specific attributes of that cluster.

2.4.11.2.5 Consolidation of Attribute Reporting

In order to reduce the resources (such as the number of timers) required for attribute reporting, a device may adapt the timing of reports by relaxing the configured minimum and maximum periods as described below. By employing these techniques a device may limit the number of timers required to any manufacturer specific value, including use of only a single timer, though at the cost of some side effects, such as increased network traffic in some cases.

In consolidating timers, a number of principles apply:

1/ The maximum reporting interval of an attribute may be reduced, as it should not normally cause a problem to devices to receive reports more frequently than expected – typical reporting intervals are seconds to minutes. It may not be increased, as this may be incompatible with any timeout period set.

2/ The minimum reporting interval of an attribute may also be reduced. However, it may not be increased, as an application may be relying on receiving reports of changes to an attribute within a given delay time. Minimum values are generally used to reduce network traffic, but this is less important than ensuring that the application timing needs are satisfied.

3/ From (1), when consolidating the maximum reporting periods of two or more attributes together, the consolidated reporting period shall be equal to the lowest of the configured maximum intervals of the attributes to be reported.

4/ Similarly, from (2), when consolidating the minimum reporting periods of two or more attributes together, the consolidated reporting period shall be equal to the lowest of the configured minimum intervals of the attributes to be reported.

As a first step, timers for attributes on the same cluster may be consolidated. Such adaptations should aim to send attribute reports for different attributes of the same cluster at the same time, so that they can be consolidated into fewer attribute reports, thus reducing network traffic.

To reduce the number of timers further, timers may be consolidated across clusters and endpoints if needed.

(Note that it is not generally possible to consolidate timeout values (see 2.4.7.1.8) of received attribute reports.)

2.4.11.3 Effect on Receipt

On receipt of this command, a device is notified of the latest values of one or more of the attributes of another device.

2.4.12 Default Response Command

2.4.12.1 Default Response Command Frame Format

The default response command frame shall be formatted as illustrated in Figure 2.25.

Octets: Variable	1	1
ZCL header	Command identifier	Status code

Figure 2.25 Format of the Default Response Command Frame

2.4.12.1.1 ZCL Header Fields

The frame control field shall be specified as follows. The frame type sub-field shall be set to indicate a profile wide command (0b00). The manufacturer specific sub-field shall be set to 0 if this command is being sent in response to a command defined for any cluster in the ZCL or 1 if this command is being sent in response to a manufacturer specific command. (Note, this requires all manufacturers to

include this command, as defined here, in any manufacturer specific command set(s) they specify).

The command identifier sub-field shall be set to indicate the default response command (see Table 2.9).

2.4.12.1.2 Command Identifier Field

The command identifier field is 8 bits in length and specifies the identifier of the received command to which this command is a response.

2.4.12.1.3 Status Code Field

The status code field is 8 bits in length and specifies either SUCCESS or the nature of the error that was detected in the received command. It shall be one of the status enumerations listed in Table 2.17.

2.4.12.2 When Generated

The default response command is generated when all 3 of these criteria are met:

1. A device receives a unicast command that is not a default response command
2. No other command is sent in response to the received command, using the same Transaction sequence number as the received command
3. The Disable default response bit of its Frame control field is set to 0 (see 2.3.1.1.4) or when an error results.[6]

If a device receives a command in error through a broadcast or multicast transmission, the command shall be discarded and the default response command shall not be generated.

If the identifier of the received command is not supported on the device, it shall set the command identifier field to the value of the identifier of the command received in error. The error code field shall be set to the either:
UNSUP_CLUSTER_COMMAND,
UNSUP_GENERAL_COMMAND,
UNSUP_MANUF_CLUSTER_COMMAND or
UNSUP_MANUF_GENERAL_COMMAND, as appropriate.

The default response command shall be generated in response to reception of all commands, including response commands (such as the write attributes response command), under the conditions specified above. However, the default response command shall not be generated in response to reception of another default response command.

6. CCB 1381

2.4.12.3 Effect on Receipt

On receipt of this command, the device is notified of the success or otherwise of the generated command with the same transaction sequence number (see 2.3.1.3).

2.4.13 Discover Attributes Command

2.4.13.1 Discover Attributes Command Frame Format

The discover attributes command frame shall be formatted as illustrated in Figure 2.26.

Octets: Variable	2	1
ZCL header	Start attribute identifier	Maximum attribute identifiers

Figure 2.26 Format of the Discover Attributes Command Frame

2.4.13.1.1 ZCL Header Fields

The frame control field shall be specified as follows. The frame type sub-field shall be set to indicate a profile wide command (0b00). The manufacturer specific sub-field shall be set to 0 to discover standard attributes in a ZigBee cluster or 1 to discover manufacturer specific attributes in either a standard or a manufacturer specific cluster.

The command identifier field shall be set to indicate the discover attributes command (see Table 2.9).

2.4.13.1.2 Start Attribute Identifier Field

The start attribute identifier field is 16 bits in length and specifies the value of the identifier at which to begin the attribute discovery.

2.4.13.1.3 Maximum Attribute Identifiers Field

The maximum attribute identifiers field is 8 bits in length and specifies the maximum number of attribute identifiers that are to be returned in the resulting discover attributes response command.

2.4.13.2 When Generated

The discover attributes command is generated when a remote device wishes to discover the identifiers and types of the attributes on a device which are supported within the cluster to which this command is directed.

2.4.13.3 Effect on Receipt

On receipt of this command, the device shall construct an ordered list of attribute information records, each containing a discovered attribute identifier and its data type, in ascending order of attribute identifiers. This list shall start with the first attribute that has an identifier that is equal to or greater than the identifier specified in the start attribute identifier field. The number of attribute identifiers included in the list shall not exceed that specified in the maximum attribute identifiers field.

The device shall then generate a discover attributes response command containing the discovered attributes and their types, and shall return it to the originator of the discover attributes command.

2.4.14 Discover Attributes Response Command

2.4.14.1 Discover Attributes Response Command Frame Format

The discover attributes response command frame shall be formatted as illustrated in Figure 2.27.

Octets: Variable	**1**	**3**	**3**	**…**	**3**
ZCL header	Discovery complete	Attribute information 1	Attribute information 2	…	Attribute information n

Figure 2.27 Discover Attributes Response Command Frame

Each attribute report field shall be formatted as illustrated in Figure 2.28.

Octets: 2	**1**
Attribute identifier	Attribute data type

Figure 2.28 Format of the Attribute Report Fields

2.4.14.1.1 ZCL Header Fields

The frame control field shall be specified as follows. The frame type sub-field shall be set to indicate a profile wide command (0b00). The manufacturer specific sub-field shall be set to the same value included in the original discover attributes command.

The command identifier field shall be set to indicate the discover attributes response command (see Table 2.9).

2.4.14.1.2 Discovery Complete Field

The discovery complete field is a boolean field. A value of 0 indicates that there are more attributes to be discovered. A value of 1 indicates that there are no more attributes to be discovered.

2.4.14.1.3 Attribute Identifier Field

The attribute identifier field shall contain the identifier of a discovered attribute. Attributes shall be included in ascending order, starting with the lowest attribute identifier that is greater than or equal to the start attribute identifier field of the received discover attributes command.

2.4.14.1.4 Attribute Data Type Field

The attribute data type field shall contain the data type of the attribute in the same attribute report field (see Table 2.16).

2.4.14.2 When Generated

The discover attributes response command is generated in response to a discover attributes command.

2.4.14.3 Effect on Receipt

On receipt of this command, the device is notified of the results of its attribute discovery request.

Following the receipt of this command, if the discovery complete field indicates that there are more attributes to be discovered, the device may choose to send subsequent discover attribute request commands to obtain the rest of the attribute identifiers. In this case, the start attribute identifier specified in the next attribute discovery request command should be set equal to one plus the last attribute identifier received in the discover attributes response command.

2.4.15 Read Attributes Structured Command

2.4.15.1 Read Attributes Structured Command Frame Format

The read attributes structured command frame shall be formatted as illustrated in Figure 2.29.

Octets: Variable	2	Variable	...	2	Variable
ZCL header	Attribute identifier 1	Selector 1	...	Attribute identifier *n*	Selector *n*

Figure 2.29 Format of Read Attributes Structured Command Frame

2.4.15.1.1 ZCL Header Fields

The frame control field shall be specified as follows. The frame type sub-field shall be set to indicate a profile wide command (0b00). The manufacturer specific sub-field shall be set to 0 if this command is being used to read attributes defined for any cluster in the ZCL or 1 if this command is being used to read manufacturer specific attributes.

The command identifier field shall be set to indicate the Read Attributes Structured command (see Table 2.9).

2.4.15.1.2 Attribute Identifier Field

The attribute identifier field is 16 bits in length and shall contain the identifier of the attribute that is to be read.

2.4.15.1.3 Selector Field

Each attribute identifier field is followed by a selector field, which specifies whether the whole of the attribute value is to be read, or only an individual element of it. An individual element may only be read from attributes with types of Array or Structure.

The Selector field shall be formatted as illustrated in Figure 2.30.

Octets: 1	2	...	2
Indicator (*m*)	Index 1	...	Index *m*

Figure 2.30 Format of the Selector Field

The Indicator subfield indicates the number of index fields that follow it. This number is limited to the range 0 - 15. It may be further limited by any relevant profile or application. All other values of this field are reserved.

If this subfield is 0, there are no index fields, and the whole of the attribute value is to be read. For attributes of type other than array or structure, this subfield shall have the value 0.

If this subfield is 1 or greater, the index fields indicate which element is to be read, nested to a depth of m. For example, if the attribute is an array of arrays (or structures), then if m = 2, index 1 = 5 and index 2 = 3, the third element of the fifth element of the attribute will be read.

Note that elements are numbered from 1 upwards for both arrays and structures. The zeroth element of an array or structure is readable, always has type 16 bit unsigned integer, and returns the number of elements contained in the array or structure.

2.4.15.2 When Generated

The read attributes command is generated when a device wishes to determine the values of one or more attributes, or elements of attributes, located on another device. Each attribute identifier field shall contain the identifier of the attribute to be read.

2.4.15.3 Effect on Receipt

On receipt of this command, the device shall process each specified attribute identifier and associated selector, and shall generate a read attributes response command. The read attributes response command shall contain as many read attribute status records as there are attribute identifiers included in this command frame. Each read attribute status record shall contain the corresponding attribute identifier from this command frame, a status value evaluated as described below, and, depending on the status value, the value of the attribute (or attribute element) itself.

For each attribute identifier included in the command frame, the device shall first check that it corresponds to an attribute that exists on this device, and that its associated selector field correctly indicates either the whole of the attribute or an element of the attribute. If it does not, the device shall set the status field of the corresponding read attribute status record to either UNSUPPORTED_ATTRIBUTE or INVALID_SELECTOR as appropriate, and shall not include an attribute value field. The device shall then move on to the next attribute identifier.

If the attribute identified by the attribute identifier is supported, and its associated selector field is valid, the device shall set the status field of the corresponding read attribute status record to SUCCESS and shall set the attribute value field to the value of the attribute (or its selected element). The device shall then move on to the next attribute identifier.

2.4.16 Write Attributes Structured Command

2.4.16.1 Write Attributes Structured Command Frame Format

The write attributes structured command frame shall be formatted as illustrated in Figure 2.31.

Octets: Variable	Variable	Variable	...	Variable
ZCL header	Write attribute record 1	Write attribute record 2	...	Write attribute record *n*

Figure 2.31 Write Attributes Structured Command Frame

Each write attribute record shall be formatted as illustrated in Figure 2.32.

Octets: 2	Variable	1	Variable
Attribute identifier	Selector	Attribute data type	Attribute value

Figure 2.32 Format of the Write Attribute Record Field

2.4.16.1.1 ZCL Header Fields

The frame control field shall be specified as follows. The frame type sub-field shall be set to indicate a profile wide command (0b00). The manufacturer specific sub-field shall be set to 0 if this command is being used to write attributes defined for any cluster in the ZCL or 1 if this command is being used to write manufacturer specific attributes.

The command identifier field shall be set to indicate the write attributes structured command (see Table 2.9).

2.4.16.1.2 Attribute Identifier Field

The attribute identifier field is 16 bits in length and shall contain the identifier of the attribute that is to be written (or an element of which is to be written).

2.4.16.1.3 Selector Field

The selector field specifies whether the whole of the attribute value is to be written, or only an individual element of it. An individual element may only be written to attributes with types of Array, Structure, Set or Bag.

The Selector field shall be formatted as illustrated in Figure 2.33.

Octets: 1	2	...	2
Indicator (*m*)	Index 1	...	Index *m*

Figure 2.33 Format of the Selector Field

2.4.16.1.3.1 Writing an Element to an Array or Structure

When writing an element to an array or structure, the Indicator subfield indicates the number of index fields that follow it. This number is limited to the range 0 - 15 (i.e. the upper 4 bits of the Indicator field are set to zero). It may be further limited by any relevant profile or application.

If the Indicator subfield is 0, there are no index fields, and the whole of the attribute value is to be written.

If this subfield is 1 or greater, the index fields indicate which element is to be written, nested to a depth of m. For example, if the attribute is an array of arrays (or structures), then if m = 2, index 1 = 5 and index 2 = 3, the third element of the fifth element of the attribute will be written.

Note that elements are numbered from 1 upwards for both arrays and structures.

The zeroth element of an array or structure has type 16 bit unsigned integer, and holds the number of elements in the array or structure. The zeroth element of an array may optionally be written (this is application dependent) and has the effect of changing the number of elements of the array. If the number is reduced, the array is truncated. If the number is increased, the content of new elements is application dependent.

The zeroth element of a structure may not be written to. Writing to an element with an index greater than the number of elements in an array or structure is always an error.

2.4.16.1.3.2 Adding/Removing an Element to/from a Set or Bag

This command may also be used to add an element to a set or bag, or to remove an element from a set or bag.

In this case, the lower 4 bits of the Indicator subfield still indicate the number of index fields that follow it, as the set may be an element of an array or structure, which may itself be nested inside other arrays or structures.

The upper 4 bits of the Indicator subfield have the following values:

0b0000 Write whole set / bag

0b0001 Add element to the set / bag

0b0010 Remove element from the set / bag

All other values are reserved.

2.4.16.1.4 Attribute Data Type Field

The attribute data type field shall contain the data type of the attribute or element thereof that is to be written.

2.4.16.1.5 Attribute Value Field

The attribute value field is variable in length and shall contain the actual value of the attribute, or element thereof, that is to be written. For an attribute or element of type array, structure, set or bag, this field has the same format as for the read attributes structured command (see sub-clause 2.4.15).

2.4.16.2 When Generated

The write attributes structured command is generated when a device wishes to change the values of one or more attributes located on another device. Each write attribute record shall contain the identifier and the actual value of the attribute, or element thereof, to be written.

2.4.16.3 Effect on Receipt

On receipt of this command, the device shall attempt to process each specified write attribute record and shall construct a write attribute structured response command. Each write attribute status record of the constructed command shall contain the identifier from the corresponding write attribute record and a status value evaluated as described below.

For each write attribute record included in the command frame, the device shall first check that it corresponds to an attribute that is implemented on this device and that its associated selector field correctly indicates either the whole of the attribute or an element of the attribute. If it does not (e.g. an index is greater than the number of elements of an array), the device shall set the status field of the corresponding write attribute status record to either UNSUPPORTED_ATTRIBUTE or INVALID_SELECTOR as appropriate and move on to the next write attribute record.

If the attribute identified by the attribute identifier is supported, the device shall check whether the attribute data type field is correct. (Note - if the element being written is the zeroth element of an array (in order to change the length of the array) the data type must be 16 bit unsigned integer). If not, the device shall set the status field of the corresponding write attribute status record to INVALID_DATA_TYPE and move on to the next write attribute record.

If the attribute data type is correct, the device shall check whether the attribute is writable. If the attribute is designated as read only, the device shall set the status

field of the corresponding write attribute status record to READ_ONLY and move on to the next write attribute record. (Note - if an array may not have its length changed, its zeroth element is read only).

If the attribute is writable, the device shall check that all the supplied basic (e.g. integer, floating point) values in the attribute value field are within the specified ranges of the elements they are to be written to. If a supplied value does not fall within the specified range of its target element, the device shall set the status field of the corresponding write attribute status record to INVALID_VALUE, shall set the selector field of that record to indicate that target element, and shall move on to the next write attribute record.

The returned selector shall have the number of indices necessary to specify the specific low-level element that failed, which will be the same as or greater than the number of indices in the selector of the write attribute record. Note that if the element being written is the zeroth element of an array (in order to change the length of the array) and the requested new length is not acceptable to the application, the value being written is considered outside the specified range of the element.

If the value supplied in the attribute value field is within the specified range of the attribute, the device shall proceed as follows.

- If an element is being added to a set, and there is an element of the set that has the same value as the value to be added, the device shall set the status field of the corresponding write attribute status record to DUPLICATE_ENTRY and move on to the next write attribute record.
- Else, if an element is being removed from a set or a bag, and there is no element of the set or bag that has the same value as the value to be removed, the device shall set the status field of the corresponding write attribute status record to NOT_FOUND and move on to the next write attribute record.
- Otherwise, the device shall write, add or remove the supplied value to/from the identified attribute or element, as appropriate, and shall move on to the next write attribute record. In this (successful) case, a write attribute status record shall not be generated. (Note - if the element being written is the zeroth element of an array, the length of the array shall be changed. If the length is reduced, the array is truncated. If the length is increased, the content of new elements is application dependent.)

When all write attribute records have been processed, the device shall generate the constructed write attributes response command. If there are no write attribute status records in the constructed command, because all attributes were written successfully, a single write attribute status record shall be included in the command, with the status field set to SUCCESS and the attribute identifier field omitted.

2.4.17 Write Attributes Structured Response Command

2.4.17.1 Write Attributes Structured Response Command Frame Format

The write attributes response command frame shall be formatted as illustrated in Figure 2.34.

Octets: Variable	Variable	Variable	...	Variable
ZCL header	Write attribute status record 1	Write attribute status record 2	...	Write attribute status record *n*

Figure 2.34 Write Attributes Structured Response Command Frame

Each write attribute status record shall be formatted as illustrated in Figure 2.35.

Octets: 1	2	Variable
Status	Attribute identifier	Selector

Figure 2.35 Format of the Write Attribute Status Record Field

2.4.17.1.1 ZCL Header Fields

The frame control field shall be specified as follows. The frame type sub-field shall be set to indicate a profile wide command (0b00). The manufacturer specific sub-field shall be set to 0 if this command is being used to write attributes defined for any cluster in the ZCL or 1 if this command is being used to write manufacturer specific attributes.

The command identifier field shall be set to indicate the write attributes structured response command (see Table 2.9).

2.4.17.1.2 Status Field

The status field is 8 bits in length and specifies the status of the write operation attempted on this attribute, as detailed in sub-clause 2.4.16.3.

Note that write attribute status records are not included for successfully written attributes, in order to save bandwidth. In the case of successful writing of all attributes, only a single write attribute status record shall be included in the

command, with the status field set to SUCCESS and the attribute identifier and selector fields omitted.

2.4.17.1.3 Attribute Identifier Field

The attribute identifier field is 16 bits in length and shall contain the identifier of the attribute on which the write operation was attempted.

2.4.17.1.4 Selector Field

The selector field shall specify the element of the attribute on which the write operation that failed was attempted. See Figure 2.33 for the structure of this field.

From the structure shown in Figure 2.33, note that for all attribute data types other than array or structure this field consists of a single octet with value zero. For array or structure types, a single octet with value zero indicates that no information is available about which element of the attribute caused the failure.

2.4.17.2 When Generated

The write attributes structured response command is generated in response to a write attributes structured command.

2.4.17.3 Effect on Receipt

On receipt of this command, the device is notified of the results of its original write attributes structured command.

2.5 Addressing, Types and Enumerations

2.5.1 Addressing

ZigBee uses a number of concepts to address application profiles, clusters, device descriptions, attributes and commands, each with their own constraints. This sub-clause details these constraints.

2.5.1.1 Profile Identifier

A profile identifier is 16 bits in length and specifies the application profile being used. A profile identifier shall be set to one of the non-reserved values listed in

Table 2.11. Within a manufacturer specific application profile, the full ranges of clusters, device, attribute and command identifiers can be used.

Table 2.11 Valid Profile Identifier Values

Profile Identifier	Description
0x0000 – 0x7fff	Standard ZigBee application profile.
0x8000 – 0xbfff	Reserved.
0xc000 – 0xffff	Manufacturer Specific application profile.

2.5.1.2 Device Identifier

A device identifier is 16 bits in length and specifies a specific device within a standard application profile. A device identifier shall be set to one of the non-reserved values listed in Table 2.12.

Table 2.12 Valid Device Identifier Values

Device Identifier	Description
0x0000 – 0xbfff	Standard ZigBee device description.
0xc000 – 0xffff	Reserved.

2.5.1.3 Cluster Identifier

A cluster identifier is 16 bits in length and specifies the set of related commands and attributes within a standard application profile. It shall be set to one of the non-reserved values listed in Table 2.13.

Table 2.13 Valid Cluster Identifier Values

Cluster Identifier	Description
0x0000 – 0x7fff	Standard ZigBee cluster.
0x8000 – 0xfbff	Reserved.
0xfc00 – 0xffff	Manufacturer specific cluster within a standard ZigBee profile.

2.5.1.4 Attribute Identifier

An attribute identifier is 16 bits in length and specifies a single attribute within a standard application profile. An attribute identifier, defined within the ZCL, shall be set to one of the non-reserved values listed in Table 2.14.

Table 2.14 Valid ZCL Defined Attribute Identifier Values

Attribute Identifier	Description
0x0000 – 0x3fff	Standard ZigBee attribute.
0x4000 – 0xffff	Reserved.

Manufacturer specific attributes within a standard ZigBee cluster can be defined over the full 16-bit range. These may manipulated using the commands listed in Table 2.9, but the frame control field must be set to indicate a manufacturer specific command (see 2.3.1.4). (Note that, alternatively, the manufacturer may define his own commands, re-using these command IDs if desired).

2.5.1.5 Command Identifier

A command identifier is 8 bits in length and specifies a specific command within the ZCL as a whole or within a specific cluster. A command identifier shall be set to one of the non-reserved values listed in Table 2.15. Manufacturer specific commands within a standard ZigBee cluster can be defined over the full 8-bit range but each shall use the appropriate manufacturer code.

Table 2.15 Valid ZCL Defined Command Identifier Values

Command Identifier	Description
0x00 – 0x7f	Standard ZigBee command.
0x80 – 0xff	Reserved.

2.5.2 Data Types

ZigBee devices, such as thermostats, lamps, etc., are defined in terms of the attributes they contain, which can be written, read or reported using the commands defined in clause 2.4. Table 2.16 details the data types and formats that can be used for these attributes. Note that individual clusters, which may use different or new types, show valid values, ranges, and units for the attributes they represent.

Each data type is allocated an 8-bit data type ID. The most significant 5 bits of this ID is used to divide the types into 32 type classes, and the least significant 3 bits specify a specific data type within this class.

Table 2.16 also indicates for each data type whether it is considered to be 'analog' or 'discrete'. Values of analog types may be added to or subtracted from other values of the same type, and are typically used to measure the value of properties in the real world that vary continuously over a range. Values of discrete data types only have meaning as individual values, and may not be added or subtracted.

Table 2.16 Data Types

Type Class	Data Type ID	Data Type	Length Of Data (Octets)	Invalid Number	Analog / Discrete
Null	0x00	No data	0	-	-
	0x01 – 0x7	Reserved	-	-	
General data	0x08	8-bit data	1	-	D
	0x09	16-bit data	2	-	
	0x0a	24-bit data	3	-	
	0x0b	32-bit data	4	-	
	0x0c	40-bit data	5	-	
	0x0d	48-bit data	6	-	
	0x0e	56-bit data	7	-	
	0x0f	64-bit data	8	-	
Logical	0x10	Boolean	1	0xff	D
	0x11 – 0x17	Reserved	-	-	
Bitmap	0x18	8-bit bitmap	1	-	D
	0x19	16-bit bitmap	2	-	
	0x1a	24-bit bitmap	3	-	
	0x1b	32-bit bitmap	4	-	
	0x1c	40-bit bitmap	5	-	
	0x1d	48-bit bitmap	6	-	
	0x1e	56-bit bitmap	7	-	
	0x1f	64-bit bitmap	8	-	

Table 2.16 Data Types (Continued)

Type Class	Data Type ID	Data Type	Length Of Data (Octets)	Invalid Number	Analog / Discrete
Unsigned integer	0x20	Unsigned 8-bit integer	1	0xff	A
	0x21	Unsigned 16-bit integer	2	0xffff	
	0x22	Unsigned 24-bit integer	3	0xffffff	
	0x23	Unsigned 32-bit integer	4	0xffffffff	
	0x24	Unsigned 40-bit integer	5	0xffffffffff	
	0x25	Unsigned 48-bit integer	6	0xffffffffffff	
	0x26	Unsigned 56-bit integer	7	0xffffffffffffff	
	0x27	Unsigned 64-bit integer	8	0xffffffffffffffff	
Signed integer	0x28	Signed 8-bit integer	1	0x80	A
	0x29	Signed 16-bit integer	2	0x8000	
	0x2a	Signed 24-bit integer	3	0x800000	
	0x2b	Signed 32-bit integer	4	0x80000000	
	0x2c	Signed 40-bit integer	5	0x8000000000	
	0x2d	Signed 48-bit integer	6	0x800000000000	
	0x2e	Signed 56-bit integer	7	0x80000000000000	
	0x2f	Signed 64-bit integer	8	0x8000000000000000	

Table 2.16 Data Types (Continued)

Type Class	Data Type ID	Data Type	Length Of Data (Octets)	Invalid Number	Analog / Discrete
Enumeration	0x30	8-bit enumeration	1	0xff	D
	0x31	16-bit enumeration	2	0xffff	
	0x32 – 0x37	Reserved	-	-	
Floating point	0x38	Semi-precision	2	Not a Number	A
	0x39	Single precision	4	Not a Number	
	0x3a	Double precision	8	Not a Number	
	0x3b – 0x3f	Reserved	-	-	
String	0x40	Reserved	-	-	D
	0x41	Octet string	Defined in first octet	0xff in first octet	
	0x42	Character string	Defined in first octet	0xff in first octet	
	0x43	Long octet string	Defined in first two octets	0xffff in first two octets	
	0x44	Long character string	Defined in first two octets	0xffff in first two octets	
	0x45 – 0x47	Reserved	-	-	
Ordered sequence	0x48	Array	2 + sum of lengths of contents	0xffff in first 2 octets	D
	0x49 - 0x4b	Reserved	-	-	
	0x4c	Structure	2 + sum of lengths of contents	0xffff in first 2 octets	
	0x4d - 0x4f	Reserved	-	-	

Table 2.16 Data Types (Continued)

Type Class	Data Type ID	Data Type	Length Of Data (Octets)	Invalid Number	Analog / Discrete
Collection	0x50	Set	Sum of lengths of contents	Number of elements returned as 0xffff	D
	0x51	Bag	Sum of lengths of contents	Number of elements returned as 0xffff	
	0x52 - 0x57	Reserved	-	-	
Reserved	0x58 – 0xdf	-	-	-	-
Time	0xe0	Time of day	4	0xffffffff	A
	0xe1	Date	4	0xffffffff	
	0xe2	UTCTime	4	0xffffffff	
	0xe3 – 0xe7	Reserved	-	-	
Identifier	0xe8	Cluster ID	2	0xffff	D
	0xe9	Attribute ID	2	0xffff	
	0xea	BACnet OID	4	0xffffffff	
	0xeb – 0xef	Reserved	-	-	
Miscellaneous	0xf0	IEEE address	8	0xffffffffffffffff	D
	0xf1	128-bit security key	16	-	-
	0xf2 – 0xfe	Reserved	-	-	
Unknown	0xff	Unknown	0	-	-

2.5.2.1 No Data Type

The no data type is a special type to represent an attribute with no associated data.

2.5.2.2 General Data (8, 16, 24, 32, 40, 48, 56 and 64-bit)

This type has no rules about its use, and may be used when a data element is needed but its use does not conform to any of the standard types.

2.5.2.3 Boolean

The Boolean type represents a logical value, either FALSE (0x00) or TRUE (0x01). The value 0xff represents an invalid value of this type. All other values of this type are forbidden.

2.5.2.4 Bitmap (8, 16, 24, 32, 40, 48, 56 and 64-bit)

The Bitmap type holds 8, 16, 24, 32, 40, 48, 56 or 64 logical values, one per bit, depending on its length. There is no value that represents an invalid value of this type.

2.5.2.5 Unsigned Integer (8, 16, 24, 32, 40, 48, 56 and 64-bit)

This type represents an unsigned integer with a decimal range of 0 to 2^8-1, 0 to 2^{16}-1, 0 to 2^{24}-1, 0 to 2^{32}-1, 0 to 2^{40}-1, 0 to 2^{48}-1, 0 to 2^{56}-1, or 0 to 2^{64}-1, depending on its length. The values that represents an invalid value of this type are 0xff, 0xffff, 0xffffff, 0xffffffff, 0xffffffffff, 0xffffffffffff, 0xffffffffffffff and 0xffffffffffffffff respectively.

2.5.2.6 Signed Integer (8, 16, 24, 32, 40, 48, 56 and 64-bit)

This type represents a signed integer with a decimal range of -(2^7-1) to 2^7-1, -(2^{15}-1) to 2^{15}-1, -(2^{23}-1) to 2^{23}-1, -(2^{31}-1) to 2^{31}-1, -(2^{39}-1) to 2^{39}-1, -(2^{47}-1) to 2^{47}-1, -(2^{55}-1) to 2^{55}-1, or -(2^{63}-1) to 2^{63}-1, depending on its length. The values that represents an invalid value of this type are 0x80, 0x8000, 0x800000, 0x80000000, 0x8000000000, 0x800000000000, 0x80000000000000 and 0x8000000000000000 respectively.

2.5.2.7 Enumeration (8-bit, 16-bit)

The Enumeration type represents an index into a lookup table to determine the final value. The values 0xff and 0xffff represent invalid values of the 8-bit and 16-bit types respectively.

2.5.2.8 Semi-precision

The ZigBee semi-precision number format is based on the IEEE 754 standard for binary floating-point arithmetic [B7]. This number format should be used very sparingly, when absolutely necessary, keeping in mind the code and processing required supporting it.

The value is calculated as:

$$\text{Value} = -1^{\text{Sign}} * (\text{Hidden} + \text{Mantissa}/1024) * 2^{(\text{Exponent}-15)}$$

	Sign	Exponent					Hidden		Mantissa									
	S	E_4	E_3	E_2	E_1	E_0	H	.	M_9	M_8	M_7	M_6	M_5	M_4	M_3	M_2	M_1	M_0
bit	15					10			9									0

Figure 2.36 Format of the ZigBee Semi-precision Number

Note: The transmission order for the format in Figure 2.36 is bit 0 first.

For normalized numbers ($>2^{-14}$), the hidden bit = 1 and the resolution is constant at 11 bits (1 in 2048).

For un-normalized numbers, the hidden bit = 0. Note that this does not maintain 11-bit resolution and that the resolution becomes coarser as the number gets smaller.

The hidden bit is not sent over the link. It shall have the value ‘1’ (i.e. normalized) in order to be classified as a ZigBee semi-precision number.

The sign bit is set to 0 for positive values, 1 for negative.

The exponent is 5 bits. The actual exponent of 2 is calculated as (exponent – 15).

Certain values are reserved for specific purposes:

- **Not a Number**: this is used for undefined values (e.g. at switch-on and before initialization) and is indicated by an exponent of 31 with a non-zero mantissa.
- **Infinity:** this is indicated by an exponent of 31 and a zero mantissa. The sign bit indicates whether this represents + infinity or – infinity, the figure of 0x7c00 representing +∞ and 0xfc00 representing -∞.
- **Zero:** this is indicated by both a zero exponent and zero mantissa. The sign bit indicates whether this is + or – zero, the value 0x0000 representing +zero and 0x8000 representing –zero.
- **Un-normalised numbers:** numbers $< 2^{-14}$ are indicated by a value of 0 for the exponent. The hidden bit is set to zero.

The maximum value represented by the mantissa is 0x3ff / 1024. The largest number that can be represented is therefore:

$$-1^{Sign} * (1 + 1023/1024) * 2^{(30 - 15)} = \pm 1.9990234 * 32768 = \pm 65504$$

Certain applications may choose to scale this value to allow representation of larger values (with a correspondingly more coarse resolution). For details, see the relevant device descriptions.

For example, a value of +2 is represented by $+2^{(16-15)} * 1.0 = 0x4000$, while a value of –2 is represented by 0xc000.

Similarly, a value of +0.625 is represented by $+2^{(17-15)} * 1.625 = 0x4680$, while –0.625 is represented by 0xc680.

2.5.2.9 Single Precision

The format of the single precision data type is based on the IEEE 754 standard for binary floating-point arithmetic [B7]. This number format should be used very sparingly, when absolutely necessary, keeping in mind the code and processing required supporting it.

The format and interpretation of values of this data type follow the same rules as given for the semi-precision data type, but with longer sub-fields, as follows.

Length of mantissa = 23 bits, length of exponent = 8 bits

For further details, see [B7].

2.5.2.10 Double Precision

The format of the double precision data type is based on the IEEE 754 standard for binary floating-point arithmetic [B7]. This number format should be used very sparingly, when absolutely necessary, keeping in mind the code and processing required supporting it.

The format and interpretation of values of this data type follow the same rules as given for the semi-precision data type, but with longer sub-fields, as follows.

Length of mantissa = 52 bits, length of exponent = 11 bits

For further details, see [B7].

2.5.2.11 Octet String

The octet string data type contains data in an application-defined format, not defined in this specification. The octet string data type is formatted as illustrated in Figure 2.37.

Octets: 1	Variable
Octet count	Octet data

Figure 2.37 Format of the Octet String Type

The octet count sub-field is one octet in length and specifies the number of octets contained in the octet data sub-field.

Setting this sub-field to 0x00 represents an octet string with no octet data (an “empty string”). Setting this sub-field to 0xff represents an invalid octet string value. In both cases the octet data sub-field has zero length.

The octet data sub-field is *n* octets in length, where *n* is the value of the octet count sub-field. This sub-field contains the application-defined data.

2.5.2.12 Character String

The character string data type contains data octets encoding characters according to the language and character set field of the complex descriptor (see [B1]). The character string data type shall be formatted as illustrated in Figure 2.38.

Octets: 1	Variable
Character data length	Character data

Figure 2.38 Format of the Character String Type

The character data length sub-field is one octet in length and specifies the length of the character data sub-field. (Note - for the ISO 646 ASCII character set, this is the same as the number of characters in the string. For other codings, this may not be the case.)

Setting this sub-field to 0x00 represents a character string with no character data (an “empty string”). Setting this sub-field to 0xff represents an invalid character string value. In both cases the character data sub-field has zero length.

The character data sub-field contains the encoded characters that comprise the desired character string. Its length is the sum of the lengths of the characters as specified by the language and character set fields of the complex descriptor.

A character string with no contents, i.e. with the character count sub-field equal to 0x00 and a zero length character data sub-field, shall be referred to as an 'empty string'.

2.5.2.13 Long Octet String

The long octet string data type contains data in an application-defined format, not defined in this specification. The long octet string data type is formatted as illustrated in Figure 2.39.

Octets: 2	Variable
Octet count	Octet data

Figure 2.39 Format of the Long Octet String Type

The octet count sub-field is two octets in length and specifies the number of octets contained in the octet data sub-field. It has the same format as a 16-bit unsigned integer (see 2.5.2.5).

Setting this sub-field to 0x0000 represents a long octet string with no octet data (an “empty string”). Setting this sub-field to 0xffff represents an invalid long octet string value. In both cases the octet data sub-field has zero length.

The octet data sub-field is *n* octets in length, where *n* is the value of the octet count sub-field. This sub-field contains the application-defined data.

2.5.2.14 Long Character String

The long character string data type contains data octets encoding characters according to the language and character set field of the complex descriptor (see [B1]). The long character string data type is formatted as illustrated in Figure 2.40.

Octets: 2	Variable
Character count	Character data

Figure 2.40 Format of the Long Character String Type

The character count sub-field is two octets in length and specifies the length of the character data sub-field. (Note - for the ISO 646 ASCII character set, this is the same as the number of characters in the string. For other codings, this may not be the case.) It has the same format as a 16-bit unsigned integer (see 2.5.2.5).

Setting this sub-field to 0x0000 represents a long character string with no character data (an “empty string”). Setting this sub-field to 0xffff represents an invalid long character string value. In both cases the character data sub-field has zero length.

The character data sub-field contains the encoded characters that comprise the desired character string. Its length is the sum of the lengths of the characters as specified by the language and character set fields of the complex descriptor.

A character string with no contents, i.e. with the character count sub-field equal to 0x0000 and a zero length character data sub-field, shall be referred to as an 'empty string'.

2.5.2.15 Array

An array is an ordered sequence of zero or more elements, all of the same data type. This data type may be any ZCL defined data type, including array, structure, bag or set. The total nesting depth is limited to 15, and may be further limited by any relevant profile or application.

Individual elements may be accessed by an index of type 16-bit unsigned integer. Elements are numbered from 1 upwards. The zeroth element is readable, always has type 16 bit unsigned integer, and holds the number of elements contained in the array, which may be zero. If the zeroth element contains 0xffff, the array is considered invalid / undefined.

The zeroth element may also, as an implementation option, be writeable, in order to change the size of the array (see 2.4.16 for details).

Arrays are 'packed', i.e. there is no concept of a 'null' element. However, if an element has a simple (unstructured) type, and that type has an 'invalid number' value defined (see Table 2.16, that value indicates that the element is invalid / undefined.

2.5.2.16 Structure

A structure is an ordered sequence of elements, which may be of different data types. Each data type may be any ZCL defined data type, including array, structure, bag or set. The total nesting depth is limited to 15, and may be further limited by any relevant profile or application.

Individual elements may be accessed by an index of type 16-bit unsigned integer. Elements are numbered from 1 upwards. The zeroth element is readable, always has type 16 bit unsigned integer, and holds the number of elements contained in the structure, which may be zero. If the zeroth element contains 0xffff, the array is considered invalid / undefined. The zeroth element may not be written to.

Structures are 'packed', i.e. there is no concept of a 'null' element. However, if an element has a simple (unstructured) type, and that type has an 'invalid number' value defined (see Table 2.16), that value indicates that the element is undefined.

2.5.2.17 Set

A set is a collection of elements with no associated order. Each element has the same data type, which may be any ZCL defined data type, including array, structure, bag or set. The nesting depth is limited to 15, and may be further limited by any relevant profile or application.

Elements of a set are not individually addressable, so may not be individually read or modified. Sets may only be read in their entirety. Individual elements may be added to a set or removed from a set; removal is done by value.

The maximum number of elements in a set is 0xfffe. If the number of elements is returned by a read command as 0xffff, this indicates that it is invalid / undefined.

No two elements of a set may have the same value.

2.5.2.18 Bag

A bag behaves exactly the same as a set, except that the restriction that no two elements may have the same value is removed.

2.5.2.19 Time of Day

The Time of Day data type shall be formatted as illustrated in Figure 2.41.

Octets: 1	**1**	**1**	**1**
Hours	Minutes	Seconds	Hundredths

Figure 2.41 Format of the Time of Day Type

The hours subfield represents hours according to a 24 hour clock. The range is from 0 to 23.

The minutes subfield represents minutes of the current hour. The range is from 0 to 59.

The seconds subfield represents seconds of the current minute. The range is from 0 to 59.

The hundredths subfield represents 100ths of the current second. The range is from 0 to 99.

A value of 0xff in any subfield indicates an unused subfield. If all subfields have the value 0xff, this indicates an invalid or 'don't care' value of the data type.

2.5.2.20 Date

The Time of day data type shall be formatted as illustrated in Figure 2.42.

Octets: 1	1	1	1
Year - 1900	Month	Day of month	Day of week

Figure 2.42 Format of the Date Type

The year - 1900 subfield has a range of 0 to 255, representing years from 1900 to 2155.

The month subfield has a range of 1 to 12, representing January to December.

The day of month subfield has a range of 1 to 31. Note that values in the range 29 to 31 may be invalid, depending on the month and year.

The day of week subfield has a range of 1 to 7, representing Monday to Sunday.

A value of 0xff in any subfield indicates an unused subfield. If all subfields have the value 0xff, this indicates an invalid or 'don't care' value of the data type.

2.5.2.21 UTCTime

UTCTime is an unsigned 32-bit value representing the number of seconds since 0 hours, 0 minutes, 0 seconds, on the 1st of January, 2000 UTC (Universal Coordinated Time). The value that represents an invalid value of this type is 0xffffffffff.

Note that UTCTime does not hold a standard textual representation of Universal Coordinated Time (UTC). However, UTC (to a precision of one second) may be derived from it.

2.5.2.22 Cluster ID

This type represents a cluster identifier as defined in 2.5.1.3.

2.5.2.23 Attribute ID

This type represents an attribute identifier as defined in 2.5.1.4.

2.5.2.24 BACnet OID (Object Identifier)

The BACnet OID data type is included to allow interworking with BACnet (see [B8]). The format is described in the referenced standard.

2.5.2.25 IEEE Address

The IEEE Address data type is a 64-bit IEEE address that is unique to every ZigBee device. A value of 0xffffffffffffffff indicates that the address is unknown.

2.5.2.26 128-bit Security Key

The 128-bit Security Key data type is for use in ZigBee security, and may take any 128-bit value.

2.5.3 Status Enumerations

Where a ZCL command contains a status field, the actual value of the enumerated status values are listed in Table 2.17.

Table 2.17 Enumerated Status Values Used in the ZCL

Enumerated Status	Value	Description
SUCCESS	0x00	Operation was successful.
FAILURE	0x01	Operation was not successful.
-	0x02 – 0x7d	Reserved.
NOT_AUTHORIZED	0x7e	The sender of the command does not have authorization to carry out this command.
RESERVED_FIELD_NOT_ZERO	0x7f	A reserved field/subfield/bit contains a non-zero value.
MALFORMED_COMMAND	0x80	The command appears to contain the wrong fields, as detected either by the presence of one or more invalid field entries or by there being missing fields. Command not carried out. Implementer has discretion as to whether to return this error or INVALID_FIELD.
UNSUP_CLUSTER_COMMAND	0x81	The specified cluster command is not supported on the device. Command not carried out.
UNSUP_GENERAL_COMMAND	0x82	The specified general ZCL command is not supported on the device.
UNSUP_MANUF_CLUSTER_ COMMAND	0x83	A manufacturer specific unicast, cluster specific command was received with an unknown manufacturer code, or the manufacturer code was recognized but the command is not supported.
UNSUP_MANUF_GENERAL_ COMMAND	0x84	A manufacturer specific unicast, ZCL specific command was received with an unknown manufacturer code, or the manufacturer code was recognized but the command is not supported.
INVALID_FIELD	0x85	At least one field of the command contains an incorrect value, according to the specification the device is implemented to.
UNSUPPORTED_ATTRIBUTE	0x86	The specified attribute does not exist on the device.
INVALID_VALUE	0x87	Out of range error, or set to a reserved value. Attribute keeps its old value. Note that an attribute value may be out of range if an attribute is related to another, e.g. with minimum and maximum attributes. See the individual attribute descriptions for specific details.

Table 2.17 Enumerated Status Values Used in the ZCL (Continued)

Enumerated Status	Value	Description
READ_ONLY	0x88	Attempt to write a read only attribute.
INSUFFICIENT_SPACE	0x89	An operation (e.g. an attempt to create an entry in a table) failed due to an insufficient amount of free space available.
DUPLICATE_EXISTS	0x8a	An attempt to create an entry in a table failed due to a duplicate entry already being present in the table.
NOT_FOUND	0x8b	The requested information (e.g. table entry) could not be found.
UNREPORTABLE_ATTRIBUTE	0x8c	Periodic reports cannot be issued for this attribute.
INVALID_DATA_TYPE	0x8d	The data type given for an attribute is incorrect. Command not carried out.
INVALID_SELECTOR	0x8e	The selector for an attribute is incorrect.
WRITE_ONLY	0x8f	A request has been made to read an attribute that the requestor is not authorized to read. No action taken.
INCONSISTENT_STARTUP_STATE	0x90	Setting the requested values would put the device in an inconsistent state on startup. No action taken.
DEFINED_OUT_OF_BAND	0x91	An attempt has been made to write an attribute that is present but is defined using an out-of-band method and not over the air.
INCONSISTENT	0x92	The supplied values (e.g. contents of table cells) are inconsistent.
ACTION_DENIED	0x93	The credentials presented by the device sending the command are not sufficient to perform this action.
TIMEOUT	0x94	The exchange was aborted due to excessive response time.
ABORT	0x95	Failed case when a client or a server decides to abort the upgrade process.
INVALID_IMAGE	0x96	Invalid OTA upgrade image (ex. failed signature validation or signer information check or CRC check).
WAIT_FOR_DATA	0x97	Server does not have data block available yet.
NO_IMAGE_AVAILABLE	0x98	No OTA upgrade image available for a particular client.
REQUIRE_MORE_IMAGE	0x99	The client still requires more OTA upgrade image files in order to successfully upgrade.
-	0x9a–0xbf	Reserved

Table 2.17 Enumerated Status Values Used in the ZCL (Continued)

Enumerated Status	Value	Description
HARDWARE_FAILURE	0xc0	An operation was unsuccessful due to a hardware failure.
SOFTWARE_FAILURE	0xc1	An operation was unsuccessful due to a software failure.
CALIBRATION_ERROR	0xc2	An error occurred during calibration.
-	0xc3 – 0xff	Reserved.

2.6 Functional Description

2.6.1 Transmission

ZCL frames are transmitted via the APS sub-layer by issuing the APSDE-DATA.request primitive.

All sub-fields of ZCL frames, including individual bits, that are specified as reserved, shall be set to zero for transmission. This applies to all ZCL frames, including cluster-specific frames. Similarly all reserved bits of attributes of type bitmap shall be set to zero for transmission.

2.6.2 Reception

ZCL frames are received via the APS sub-layer by the reception of the APSDE-DATA.indication primitive.

On receipt of a command (including both general and cluster-specific commands) the device shall attempt to parse and execute the command. During the parsing process for a non-manufacturer-specific command, it shall ignore all reserved sub-fields of the ZCL frame, including individual reserved bits.

Note that, if any of these sub-fields are not set to zero, this may indicate that the format or interpretation of the frame has been updated. However, it is the responsibility of the specifier of such an updated format that it be backward compatible, i.e. any new format will result in the same functionality as before when parsed by a device that supports the previous version of the cluster. Any additional octets found appended to the frame shall also be ignored, as these may be added as part of such an updated frame format.

If the command is manufacturer-specific, handling of reserved sub-fields is determined by the manufacturer.

If required, the device shall then generate a response to the command. Responses are detailed in the specification of each command. If there is no response specified for a particular set of circumstances, (e.g. if the command has been rejected or is not recognized, or the command has succeeded but there is no response specified to indicate success), the default response command shall be generated, taking into account the conditions in 2.4.12.2. The status code returned by the default response command shall be one of the status enumerations listed in Table 2.17.

On receipt of a frame containing a broadcast endpoint (0xff), the APS sub-layer shall direct the frame payload to each active endpoint, through its APSDE-DATA.indication primitive.

2.6.3 Manufacturer Specific Extensions

Manufacturers are free to extend a standard profile in the following ways:

- Add manufacturer specific clusters to a standard profile.
- Add manufacturer specific commands to a standard cluster.
- Add manufacturer specific attributes to a standard cluster.

All communications regarding manufacturer specific extensions shall be transmitted with the manufacturer specific sub-field of the frame control field set to 1 and the manufacturer code included in the frame.

If the manufacturer code in a command frame is not recognized, the command is not carried out.

2.6.4 Dependencies on Optional Attribute

If the specification of an attribute of a cluster depends on the value of another optional attribute of the same cluster, then the optional attribute shall have a well defined default value. When the optional attribute is not supported by a device, then the default value shall be used for those attributes that depend on the unsupported optional attribute value. This rule shall be recursive if there is a chain of dependencies.[7]

7. CCB 1169

This page left intentionally blank

CHAPTER

3

GENERAL SPECIFICATION

3.1 General Description

3.1.1 Introduction

The clusters specified in this document are included here because they are sufficiently general to be of use across a wide range of application domains.

3.1.2 Cluster List

The clusters defined in this document are listed in Table 3.1 to Table 3.5.

Table 3.1 Device Configuration and Installation Clusters

Cluster Name	Description
Basic	Attributes for determining basic information about a device, setting user device information such as description of location, and enabling a device.
Power configuration	Attributes for determining more detailed information about a device's power source(s), and for configuring under/over voltage alarms.
Device temperature configuration	Attributes for determining information about a device's internal temperature, and for configuring under/over temperature alarms.
Identify	Attributes and commands for putting a device into Identification mode (e.g. flashing a light)

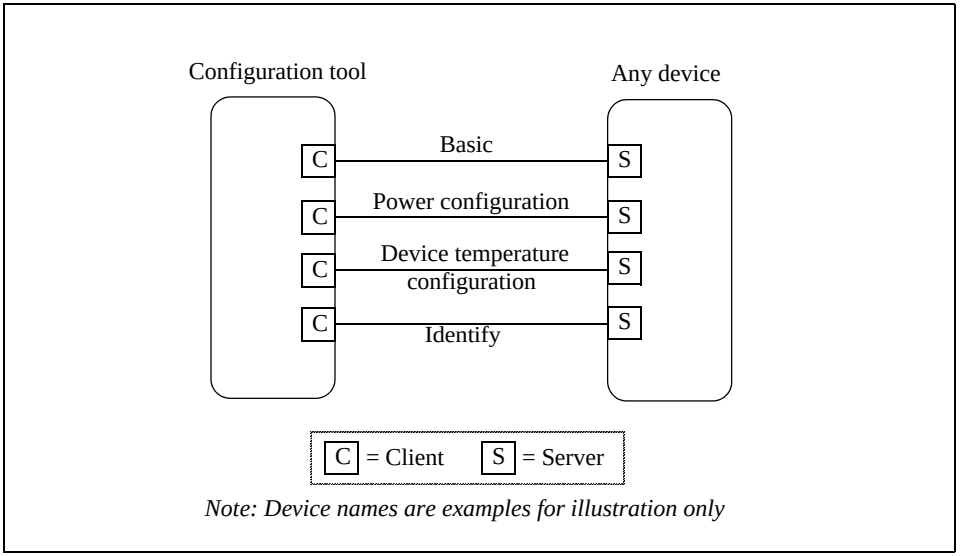

Figure 3.1 Typical Usage of Device Configuration and Installation Clusters

Table 3.2 Groups and Scenes Clusters

Cluster Name	Description
Groups	Attributes and commands for allocating a device to one or more of a number of groups of devices, where each group is addressable by a group address.
Scenes	Attributes and commands for setting up and recalling a number of scenes for a device. Each scene corresponds to a set of stored values of specified device attributes.

Table 3.3 On/Off and Level Control Clusters

Cluster Name	Description
On/off	Attributes and commands for switching devices between ‘On’ and ‘Off’ states.
On/off switch configuration	Attributes and commands for configuring on/off switching devices
Level control	Attributes and commands for controlling a characteristic of devices that can be set to a level between fully ‘On’ and fully ‘Off’.

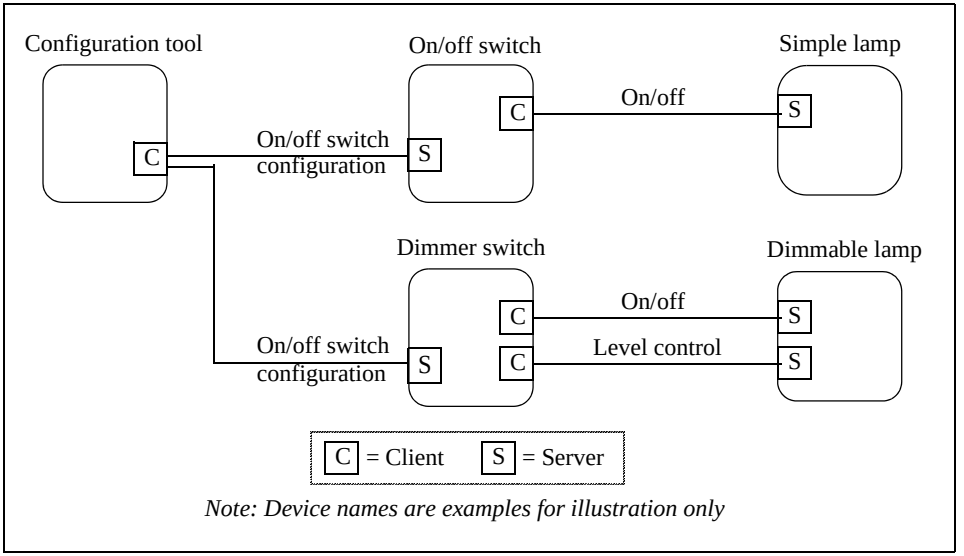

Figure 3.2 Typical Usage of On / Off and Level Control Clusters

Table 3.4 Alarms Cluster

Cluster Name	Description
Alarms	Attributes and commands for sending alarm notifications and configuring alarm functionality.

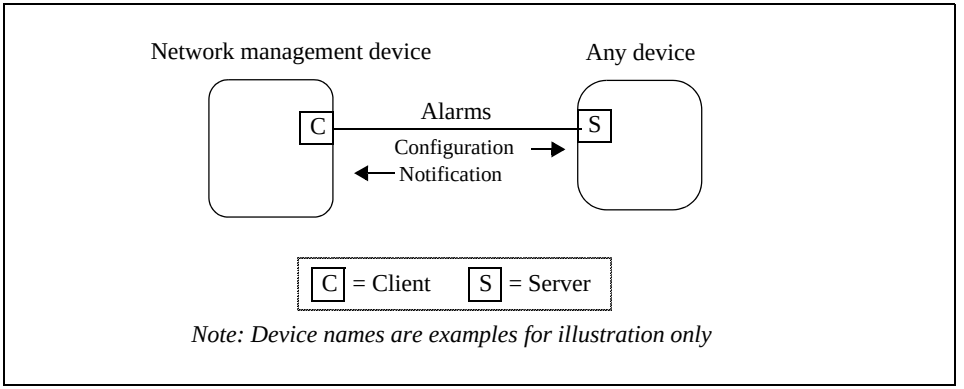

Figure 3.3 Typical Usage of the Alarms Cluster

Table 3.5 Other Clusters

Cluster Name	Description
Time	Attributes and commands that provide an interface to a real-time clock.
Location	Attributes and commands for exchanging location information and channel parameters among devices, and (optionally) reporting data to a centralized device that collects data from devices in the network and calculates their positions from the set of collected data.
Commissioning	Attributes and commands for commissioning and managing a ZigBee device.

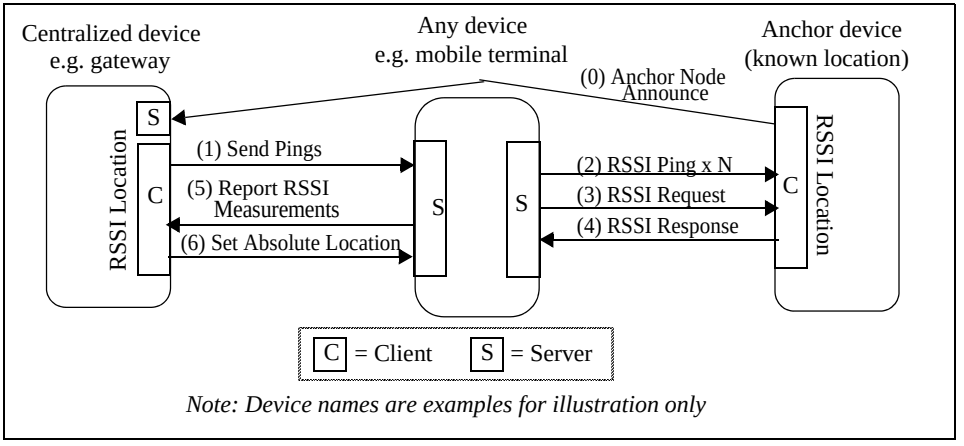

Figure 3.4 Typical Usage of the Location Cluster, with Centralized Device

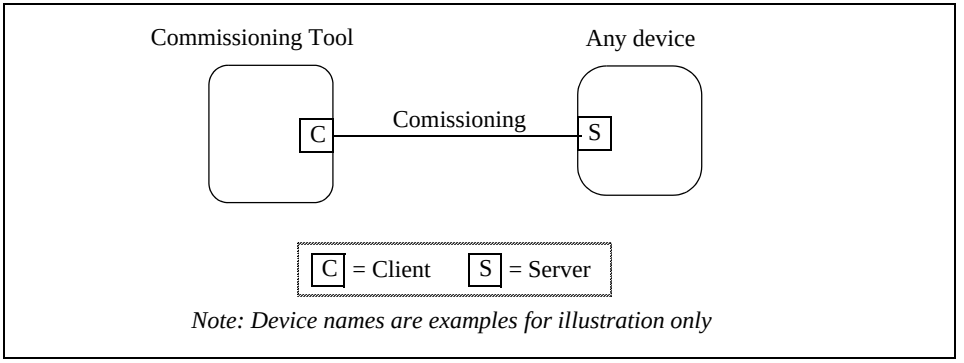

Figure 3.5 Typical Usage of the Commissioning Cluster

Table 3.6 Generic Clusters

Cluster Name	Description
Analog input (basic)	An interface for reading the value of an analog measurement and accessing various characteristics of that measurement.
Analog output (basic)	An interface for setting the value of an analog output (typically to the environment) and accessing various characteristics of that value.
Analog value (basic)	An interface for setting an analog value, typically used as a control system parameter, and accessing various characteristics of that value.
Binary input (basic)	An interface for reading the value of a binary measurement and accessing various characteristics of that measurement.
Binary output (basic)	An interface for setting the value of a binary output (typically to the environment) and accessing various characteristics of that value.
Binary value (basic)	An interface for setting a binary value, typically used as a control system parameter, and accessing various characteristics of that value.
Multistate input (basic)	An interface for reading the value of a multistate measurement and accessing various characteristics of that measurement.
Multistate output (basic)	An interface for setting the value of a multistate output (typically to the environment) and accessing various characteristics of that value.
Multistate value (basic)	An interface for setting a multistate value, typically used as a control system parameter, and accessing various characteristics of that value.

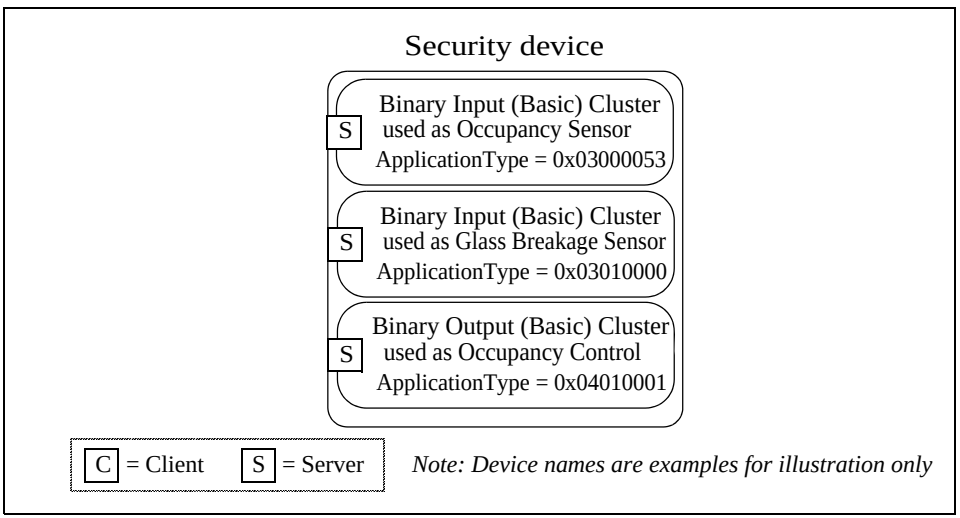

Figure 3.6 Example Usage of the Input, Output and Value Clusters

3.2 Basic Cluster

3.2.1 Overview

Attributes and commands for determining basic information about a device, setting user device information such as location, enabling a device and resetting it to factory defaults.

Note: Where a physical ZigBee node supports multiple endpoints it will often be the case that many of these settings will apply to the whole node, that is they are the same for every endpoint on the device. In such cases they can be implemented once for the node, and mapped to each endpoint.

3.2.2 Server

3.2.2.1 Dependencies

For the alarms functionality of this cluster to be operational, the Alarms cluster server shall be implemented on the same endpoint.

3.2.2.2 Attributes

For convenience, the attributes defined in this specification are arranged into sets of related attributes; each set can contain up to 16 attributes. Attribute identifiers are encoded such that the most significant three nibbles specify the attribute set and the least significant nibble specifies the attribute within the set. The currently defined attribute sets are listed in Table 3.7.

Table 3.7 General Attribute Sets

Attribute Set IDentifier	Description
0x000	Basic Device Information
0x001	Basic Device Settings
0x002 – 0xfff	Reserved

3.2.2.2.1 Basic Device Information Attribute Set

The Basic Device Information attribute set contains the attributes summarized in Table 3.8.

Table 3.8 Attributes of the Basic Device Information Attribute Set

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0000	*ZCLVersion*	Unsigned 8-bit integer	0x00 – 0xff	Read only	0x01	M
0x0001	*ApplicationVersion*	Unsigned 8-bit integer	0x00 – 0xff	Read only	0x00	O
0x0002	*StackVersion*	Unsigned 8-bit integer	0x00 – 0xff	Read only	0x00	O
0x0003	*HWVersion*	Unsigned 8-bit integer	0x00 – 0xff	Read only	0x00	O
0x0004	*ManufacturerName*	Character string	0 – 32 bytes	Read only	Empty string	O
0x0005	*ModelIdentifier*	Character string	0 – 32 bytes	Read only	Empty string	O
0x0006	*DateCode*	Character string	0 – 16 bytes	Read only	Empty string	O
0x0007	*PowerSource*	8-bit enumeration	0x00 – 0xff	Read only	0x00	M

3.2.2.2.2 *ZCLVersion* Attribute

The *ZCLVersion* attribute is 8 bits in length and specifies the version number of the ZigBee Cluster Library that all clusters on this endpoint conform to. For the this version of the ZCL, this attribute shall be set to 0x01.

Note: It is strongly recommended that new functionality is added to the ZCL either in the form of new clusters or by addition of optional attributes and optional commands to existing clusters. New functionality should be added in an 'orthogonal' way, making use of existing clusters to perform the functions they offer rather than re-implementing these functions as part of the new clusters, so that devices whose functionality has been extended via the new clusters can still interwork with devices using existing clusters. When increasing the version of the ZCL, no changes should be made to the functionality of individual clusters that prevent interworking with previous versions of the same cluster.

3.2.2.2.3 *ApplicationVersion* Attribute

The *ApplicationVersion* attribute is 8 bits in length and specifies the version number of the application software contained in the device. The usage of this attribute is manufacturer dependent.

3.2.2.2.4 *StackVersion* Attribute

The *StackVersion* attribute is 8 bits in length and specifies the version number of the implementation of the ZigBee stack contained in the device. The usage of this attribute is manufacturer dependent.

3.2.2.2.5 *HWVersion* Attribute

The *HWVersion* attribute is 8 bits in length and specifies the version number of the hardware of the device. The usage of this attribute is manufacturer dependent.

3.2.2.2.6 *ManufacturerName* Attribute

The *ManufacturerName* attribute is a maximum of 32 bytes in length and specifies the name of the manufacturer as a ZigBee character string.

3.2.2.2.7 *ModelIdentifier* Attribute

The *ModelIdentifier* attribute is a maximum of 32 bytes in length and specifies the model number (or other identifier) assigned by the manufacturer as a ZigBee character string.

3.2.2.2.8 *DateCode* Attribute

The *DateCode* attribute is a ZigBee character string with a maximum length of 16 bytes . The first 8 characters specify the date of manufacturer of the device in international date notation according to ISO 8601, i.e. YYYYMMDD, e.g. 20060814.

The final 8 characters may include country, factory, line, shift or other related information at the option of the manufacturer. The format of this information is manufacturer dependent.

3.2.2.2.9 *PowerSource* Attribute

The *PowerSource* attribute is 8 bits in length and specifies the source(s) of power available to the device. Bits b_0–b_6 of this attribute represent the primary power source of the device and bit b_7 indicates whether the device has a secondary power source in the form of a battery backup.

Bits b_0–b_6 of this attribute shall be set to one of the non-reserved values listed in Table 3.9.

Table 3.9 Values of the *PowerSource* Attribute

Attribute Value b_6-b_0	Description
0x00	Unknown
0x01	Mains (single phase)
0x02	Mains (3 phase)
0x03	Battery
0x04	DC source
0x05	Emergency mains constantly powered
0x06	Emergency mains and transfer switch
0x07 – 0x7f	Reserved

Bit b_7 of this attribute shall be set to 1 if the device has a secondary power source in the form of a battery backup. Otherwise, bit b_7 shall be set to 0.

3.2.2.2.10 Basic Device Settings Attribute Set

The device configuration attribute set contains the attributes summarized in Table 3.10.

Table 3.10 Attributes of the Device Configuration Attribute Set

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0010	*LocationDescription*	Character string	0 – 16 bytes	Read/ write	Empty string	O
0x0011	*PhysicalEnvironment*	8-bit enumeration	0x00 – 0xff	Read/ write	0x00	O
0x0012	*DeviceEnabled*	Boolean	0x00 – 0x01	Read/ write	0x01	O
0x0013	*AlarmMask*	8-bit bitmap	000000xx	Read/ write	0x00	O
0x0014	*DisableLocalConfig*	8-bit bitmap	000000xx	Read/ write	0x00	O

3.2.2.2.11 *LocationDescription* Attribute

The *LocationDescription* attribute is a maximum of 16 bytes in length and describes the physical location of the device as a ZigBee character string. This location description may be added into the device during commissioning.

3.2.2.2.12 *PhysicalEnvironment* Attribute

The *PhysicalEnvironment* attribute is 8 bits in length and specifies the type of physical environment in which the device will operate. This attribute shall be set to one of the non-reserved values listed in Table 3.11.

Table 3.11 Values of the *PhysicalEnvironment* Attribute

***PhysicalEnvironment* Attribute Value**	**Description**
0x00	Unspecified environment
0x01 – 0x7f	Specified per Profile
0x80 – 0xfe	Reserved
0xff	Unknown environment

3.2.2.2.13 *DeviceEnabled* Attribute

The *DeviceEnabled* attribute is a boolean and specifies whether the device is enabled or disabled. This attribute shall be set to one of the non-reserved values listed in Table 3.12.

Table 3.12 Values of the *DeviceEnable* Attribute

***DeviceEnable* Attribute Value**	**Description**
0x00	Disabled
0x01	Enabled

'Disabled' means that the device does not send or respond to application level commands, other than commands to read or write attributes. Values of attributes which depend on the operation of the application may be invalid, and any functionality triggered by writing to such attributes may be disabled. ZigBee networking functionality remains operational.

If implemented, the identify cluster cannot be disabled, i.e. it remains functional regardless of this setting.

3.2.2.2.14 *AlarmMask* Attribute

The *AlarmMask* attribute is 8 bits in length and specifies which of a number of general alarms may be generated, as listed in Table 3.13. A '1' in each bit position enables the associated alarm.

Table 3.13 Values of the *AlarmMask* Attribute

***AlarmMask* Attribute Bit Number**	**Alarm Code**	**Alarm**
0	0	General hardware fault
1	1	General software fault
2 – 7	-	Reserved

These alarms are provided as basic alarms that a device may use even if no other clusters with alarms are present on the device.

3.2.2.2.15 *DisableLocalConfig* Attribute

The *DisableLocalConfig* attribute allows a number of local device configuration functions to be disabled.

Table 3.14 Values of the *DisableLocalConfig* Attribute

***DisableLocalConfig* Attribute Bit Number**	**Description**
0	0 = Reset (to factory defaults) enabled 1 = Reset (to factory defaults) disabled
1	0 = Device configuration enabled 1 = Device configuration disabled
2 – 7	Reserved

The intention of this attribute is to allow disabling of any local configuration user interface, for example to prevent reset or binding buttons being activated by non-authorised persons in a public building.

Bit 0 of the *DisableLocalConfig* attribute disables any factory reset button (or equivalent) on the device. Bit 1 disables any device configuration button(s) (or equivalent) - for example a bind button.

The effect of these bits is manufacturer dependent - for example, if a reset button is not physically accessible, bit 0 may be ignored.

3.2.2.3 Commands Received

The command IDs for the Basic cluster are listed in Table 3.15.

Table 3.15 Received Command IDs for the Basic Cluster

Command Identifier Field Value	Description	Mandatory / Optional
0x00	Reset to Factory Defaults	0
0x01 – 0xff	Reserved	-

3.2.2.3.1 Reset to Factory Defaults Command

This command does not have a payload.

3.2.2.3.1.1 Effect on Receipt

On receipt of this command, the device resets all the attributes of all its clusters to their factory defaults.

Note that ZigBee networking functionality and any bindings are not affected by this command.

3.2.2.4 Commands Generated

No commands are generated by the server cluster.

3.2.3 Client

3.2.3.1 Dependencies

None

3.2.3.2 Attributes

The Client cluster has no attributes.

3.2.3.3 Commands Received

No cluster specific commands are received by the client cluster.

3.2.3.4 Commands Generated

The cluster specific commands generated by the client cluster are those received by the server, as required by the application.

3.3 Power Configuration Cluster

3.3.1 Overview

Attributes for determining detailed information about a device's power source(s), and for configuring under/over voltage alarms.

3.3.2 Server

3.3.2.1 Dependencies

Any endpoint that implements this server cluster shall also implement the Basic server cluster.

For the alarm functionality described in this cluster to be operational, any endpoint that implements the Power Configuration server cluster must also implement the Alarms server cluster (see sub-clause 3.11).

3.3.2.2 Attributes

For convenience, the attributes defined in this specification are arranged into sets of related attributes; each set can contain up to 16 attributes. Attribute identifiers are encoded such that the most significant three nibbles specify the attribute set and the least significant nibble specifies the attribute within the set. The currently defined attribute sets are listed in Table 3.16.

Table 3.16 Power Configuration Attribute Sets

Attribute Set Identifier	Description
0x000	Mains Information
0x001	Mains Settings
0x002	Battery Information
0x003	Battery Settings
0x004 – 0xfff	Reserved

3.3.2.2.1 Mains Information Attribute Set

The Mains Information attribute set contains the attributes summarized in Table 3.17.

Table 3.17 Attributes of the Mains Information Attribute Set

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0000	*MainsVoltage*	Unsigned 16-bit integer	0x0000 – 0xffff	Read only	-	O
0x0001	*MainsFrequency*	Unsigned 8-bit integer	0x00 – 0xff	Read only	-	O

3.3.2.2.1.1 *MainsVoltage* Attribute

The *MainsVoltage* attribute is 16 bits in length and specifies the actual (measured) RMS voltage (or DC voltage in the case of a DC supply) currently applied to the device, measured in units of 100mV.

3.3.2.2.1.2 *MainsFrequency* Attribute

The *MainsFrequency* attribute is 8 bits in length and represents the frequency, in Hertz, of the mains as determined by the device as follows:

MainsFrequency = 0.5 x measured frequency

Where 2 Hz <= measured frequency <= 506 Hz, corresponding to a *MainsFrequency* in the range 1 to 0xfd.

The maximum resolution this format allows is 2 Hz.

The following special values of *MainsFrequency* apply.

0x00 indicates a frequency that is too low to be measured.

0xfe indicates a frequency that is too high to be measured.

0xff indicates that the frequency could not be measured.

In the case of a DC supply, this attribute shall also have the value zero.

3.3.2.2.2 Mains Settings Attribute Set

The Mains Settings attribute set contains the attributes summarized in Table 3.18.

Table 3.18 Attributes of the Mains Settings Attribute Set

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0010	*MainsAlarmMask*	8-bit bitmap	0000 00xx	Read/ write	0000 0000	O
0x0011	*MainsVoltageMinThreshold*	Unsigned 16-bit integer	0x0000 – 0xffff	Read/ write	0x0000	O
0x0012	*MainsVoltageMaxThreshold*	Unsigned 16-bit integer	0x0000 – 0xffff	Read/ write	0xffff	O
0x0013	*MainsVoltageDwellTripPoint*	Unsigned 16-bit integer	0x0000 – 0xffff	Read/ write	0x0000	O

The alarm settings in this table require the Alarms cluster to be implemented on the same device - see Dependencies. If the Alarms cluster is not present on the same device they may be omitted.

3.3.2.2.2.1 *MainsAlarmMask* Attribute

The *MainsAlarmMask* attribute is 8 bits in length and specifies which mains alarms may be generated, as listed in Table 3.19. A ‘1’ in each bit position enables the alarm.

Table 3.19 Values of the *MainsAlarmMask* Attribute

***MainsAlarmMask* Attribute Bit Number**	**Alarm**
0	Mains Voltage too low (7.2.2.2.2)
1	Mains Voltage too high (7.2.2.2.3)
2 – 7	Reserved

3.3.2.2.2.2 *MainsVoltageMinThreshold* Attribute

The *MainsVoltageMinThreshold* attribute is 16 bits in length and specifies the lower alarm threshold, measured in units of 100mV, for the *MainsVoltage* attribute. The value of this attribute shall be less than *MainsVoltageMaxThreshold*.

If the value of *MainsVoltage* drops below the threshold specified by *MainsVoltageMinThreshold*, the device shall start a timer to expire after *MainsVoltageDwellTripPoint* seconds. If the value of this attribute increases to greater than or equal to *MainsVoltageMinThreshold* before the timer expires, the device shall stop and reset the timer. If the timer expires, an alarm shall be generated.

The Alarm Code field (see 3.11.2.3.1) included in the generated alarm shall be 0x00.

If this attribute takes the value 0xffff then this alarm shall not be generated.

3.3.2.2.2.3 *MainsVoltageMaxThreshold* Attribute

The *MainsVoltageMaxThreshold* attribute is 16 bits in length and specifies the upper alarm threshold, measured in units of 100mV, for the *MainsVoltage* attribute. The value of this attribute shall be greater than *MainsVoltageMinThreshold.*

If the value of *MainsVoltage* rises above the threshold specified by *MainsVoltageMaxThreshold*, the device shall start a timer to expire after *MainsVoltageDwellTripPoint* seconds. If the value of this attribute drops to lower than or equal to *MainsVoltageMaxThreshold* before the timer expires, the device shall stop and reset the timer. If the timer expires, an alarm shall be generated.

The Alarm Code field (see 3.11.2.3.1) included in the generated alarm shall be 0x01.

If this attribute takes the value 0xffff then this alarm shall not be generated.

3.3.2.2.2.4 *MainsVoltageDwellTripPoint* Attribute

The *MainsVoltageDwellTripPoint* attribute is 16 bits in length and specifies the length of time, in seconds that the value of *MainsVoltage* may exist beyond either of its thresholds before an alarm is generated.

If this attribute takes the value 0xffff then the associated alarms shall not be generated.

3.3.2.2.3 Battery Information Attribute Set

The Battery Information attribute set contains the attributes summarized in Table 3.20.

Table 3.20 Attributes of the Battery Information Attribute Set

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0020	*BatteryVoltage*	Unsigned 8-bit integer	0x00 – 0xff	Read only	-	O

3.3.2.2.3.1 *BatteryVoltage* Attribute

The *BatteryVoltage* attribute is 8 bits in length and specifies the current actual (measured) battery voltage, in units of 100mV.

The value 0xff indicates an invalid or unknown reading.

3.3.2.2.4 Battery Settings Attribute Set

Table 3.21 Attributes of the Battery Information Attribute Set

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0030	*BatteryManufacturer*	Character string	0 – 16 bytes	Read/write	Empty string	O
0x0031	*BatterySize*	8-bit enumeration	0x00 – 0xff	Read/write	0xff	O
0x0032	*BatteryAHrRating*	Unsigned 16-bit integer	0x0000 – 0xffff	Read/write	-	O
0x0033	*BatteryQuantity*	Unsigned 8-bit integer	0x00 – 0xff	Read/write	-	O
0x0034	*BatteryRatedVoltage*	Unsigned 8-bit integer	0x00 – 0xff	Read/write	-	O
0x0035	*BatteryAlarmMask*	8-bit bitmap	0000 000x	Read/write	0000 0000	O
0x0036	*BatteryVoltageMin Threshold*	Unsigned 8-bit integer	0x00 – 0xff	Read/write	0x0000	O

3.3.2.2.4.1 *BatteryManufacturer* Attribute

The *BatteryManufacturer* attribute is a maximum of 16 bytes in length and specifies the name of the battery manufacturer as a ZigBee character string.

3.3.2.2.4.2 *BatterySize* Attribute

The *BatterySize* attribute is an enumeration which specifies the type of battery being used by the device. This attribute shall be set to one of the non-reserved values listed in Table 3.22.

Table 3.22 Values of the *BatterySize* Attribute

Attribute Value	Description
0x00	No battery
0x01	Built in
0x02	Other
0x03	AA
0x04	AAA

Table 3.22 Values of the *BatterySize* Attribute (Continued)

Attribute Value	Description
0x05	C
0x06	D
0x07 – 0xfe	Reserved
0xff	Unknown

3.3.2.2.4.3 *BatteryAHrRating* Attribute

The *BatteryAHrRating* attribute is 16 bits in length and specifies the Ampere-hour rating of the battery, measured in units of 10mAHr.

3.3.2.2.4.4 *BatteryQuantity* Attribute

The *BatteryQuantity* attribute is 8 bits in length and specifies the number of battery cells used to power the device.

3.3.2.2.4.5 *BatteryRatedVoltage* Attribute

The *BatteryRatedVoltage* attribute is 8 bits in length and specifies the rated voltage of the battery being used in the device, measured in units of 100mV.

3.3.2.2.4.6 *BatteryAlarmMask* Attribute

The *BatteryAlarmMask* attribute is 8 bits in length and specifies which mains alarms may be generated, as listed in Table 3.23. A ‘1’ in each bit position enables the alarm.

Table 3.23 Values of the *MainsAlarmMask* Attribute

Attribute Bit Number	Alarm
0	Battery voltage too low (7.2.2.4.7)
1 – 7	Reserved

3.3.2.2.4.7 *BatteryVoltageMinThreshold* Attribute

The *BatteryVoltageMinThreshold* attribute is 8 bits in length and specifies the low voltage alarm threshold, measured in units of 100mV, for the *BatteryVoltage* attribute.

If the value of *BatteryVoltage* drops below the threshold specified by *BatteryVoltageMinThreshold* an alarm shall be generated.

The Alarm Code field (see 3.11.2.3.1) included in the generated alarm shall be 0x10.

If this attribute takes the value 0xff then this alarm shall not be generated.

3.3.2.3 Commands Received

No commands are received by the server.

3.3.2.4 Commands Generated

The server generates no commands.

3.3.3 Client

3.3.3.1 Dependencies

None

3.3.3.2 Attributes

The client has no attributes.

3.3.3.3 Commands Received

No cluster specific commands are received by the client.

3.3.3.4 Commands Generated

No cluster specific commands are generated by the client.

3.4 Device Temperature Configuration Cluster

3.4.1 Overview

Attributes for determining information about a device’s internal temperature, and for configuring under/over temperature alarms for temperatures that are outside the device's operating range.

3.4.2 Server

3.4.2.1 Dependencies

For the alarm functionality described in this cluster to be operational, any endpoint that implements the Device Temperature Configuration server cluster shall also implement the Alarms server cluster (see 3.11).

3.4.2.2 Attributes

For convenience, the attributes defined in this specification are arranged into sets of related attributes; each set can contain up to 16 attributes. Attribute identifiers are encoded such that the most significant three nibbles specify the attribute set and the least significant nibble specifies the attribute within the set. The currently defined attribute sets are listed in Table 3.24.

Table 3.24 Device Temperature Configuration Attribute Sets

Attribute Set Identifier	Description
0x000	Device Temperature Information
0x001	Device Temperature Settings
0x002 – 0xfff	Reserved

3.4.2.2.1 Device Temperature Information Attribute Set

The Device Temperature Information attribute set contains the attributes summarized in Table 3.25.

Table 3.25 Device Temperature Information Attribute Set

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0000	*CurrentTemperature*	Signed 16-bit integer	-200 to +200	Read only	-	M
0x0001	*MinTempExperienced*	Signed 16-bit integer	-200 to +200	Read only	-	O
0x0002	*MaxTempExperienced*	Signed 16-bit integer	-200 to +200	Read only	-	O
0x0003	*OverTempTotalDwell*	Unsigned 16-bit integer	0x0000 – 0xffff	Read only	0	O

3.4.2.2.1.1 *CurrentTemperature* Attribute

The *CurrentTemperature* attribute is 16 bits in length and specifies the current internal temperature, in degrees Celsius, of the device. This attribute shall be specified in the range –200 to +200.

The value 0xffff indicates an invalid reading.

3.4.2.2.1.2 *MinTempExperienced* Attribute

The *MinTempExperienced* attribute is 16 bits in length and specifies the minimum internal temperature, in degrees Celsius, the device has experienced while powered. This attribute shall be specified in the range –200 to +200.

The value 0xffff indicates an invalid reading.

3.4.2.2.1.3 *MaxTempExperienced* Attribute

The *MaxTempExperienced* attribute is 16 bits in length and specifies the maximum internal temperature, in degrees Celsius, the device has experienced while powered. This attribute shall be specified in the range –200 to +200.

The value 0xffff indicates an invalid reading.

3.4.2.2.1.4 *OverTempTotalDwell* Attribute

The *OverTempTotalDwell* attribute is 16 bits in length and specifies the length of time, in hours, the device has spent above the temperature specified by the *HighTempThreshold* attribute 3.4.2.2.2.3, cumulative over the lifetime of the device.

The value 0xffff indicates an invalid time.

3.4.2.2.2 Device Temperature Settings Attribute Set

The Device Temperature Settings attribute set contains the attributes summarized in Table 3.26.

Table 3.26 Device Temperature Settings Attribute Set

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0010	*DeviceTempAlarmMask*	8-bit bitmap	0000 00xx	Read/ write	0000 0000	O
0x0011	*LowTempThreshold*	Signed 16-bit integer	-200 to +200	Read/ write	-	O
0x0012	*HighTempThreshold*	Signed 16-bit integer	-200 to +200	Read/ write	-	O
0x0013	*LowTempDwellTripPoint*	Unsigned 24-bit integer	0x000000 – 0xffffff	Read/ write	-	O
0x0014	*HighTempDwellTripPoint*	Unsigned 24-bit integer	0x000000 – 0xffffff	Read/ write	-	O

All attributes in this table require the Alarms cluster to be implemented on the same device - see Dependencies. If the Alarms cluster is not present on the same device they may be omitted.

3.4.2.2.2.1 *DeviceTempAlarmMask* Attribute

The *DeviceTempAlarmMask* attribute is 8 bits in length and specifies which alarms may be generated, as listed in Table 3.27. A ‘1’ in each bit position enables the corresponding alarm.

Table 3.27 Values of the *DeviceTempAlarmMask* Attribute

Attribute Bit Number	Alarm
0	Device Temperature too low (8.2.2.2.2)
1	Device Temperature too high (8.2.2.2.3)
2 – 7	Reserved

3.4.2.2.2.2 *LowTempThreshold* Attribute

The *LowTempThreshold* attribute is 16 bits in length and specifies the lower alarm threshold, measured in degrees Celsius (range -200°C to 200°C), for the

CurrentTemperature attribute. The value of this attribute shall be less than *HighTempThreshold.*

If the value of *CurrentTemperature* drops below the threshold specified by *LowTempThreshold*, the device shall start a timer to expire after *LowTempDwellTripPoint* seconds. If the value of this attribute increases to greater than or equal to *LowTempThreshold* before the timer expires, the device shall stop and reset the timer. If the timer expires, an alarm shall be generated.

The Alarm Code field (see 3.11.2.3.1) included in the generated alarm shall be 0x00.

If this attribute takes the value 0x8000 then this alarm shall not be generated.

3.4.2.2.2.3 *HighTempThreshold* Attribute

The *HighTempThreshold* attribute is 16 bits in length and specifies the upper alarm threshold, measured in degrees Celsius (range -200°C to 200°C), for the *CurrentTemperature* attribute. The value of this attribute shall be greater than *LowTempThreshold.*

If the value of *CurrentTemperature* rises above the threshold specified by *HighTempThreshold*, the device shall start a timer to expire after *HighTempDwellTripPoint* seconds. If the value of this attribute drops to lower than or equal to *HighTempThreshold* before the timer expires, the device shall stop and reset the timer. If the timer expires, an alarm shall be generated.

The Alarm Code field (see 3.11.2.3.1) included in the generated alarm shall be 0x01.

If this attribute takes the value 0x8000 then this alarm shall not be generated.

3.4.2.2.2.4 *LowTempDwellTripPoint* Attribute

The *LowTempDwellTripPoint* attribute is 24 bits in length and specifies the length of time, in seconds, that the value of *CurrentTemperature* may exist below *LowTempThreshold* before an alarm is generated.

If this attribute takes the value 0xffffff then this alarm shall not be generated.

3.4.2.2.2.5 *HighTempDwellTripPoint* Attribute

The *HighTempDwellTripPoint* attribute is 24 bits in length and specifies the length of time, in seconds, that the value of *CurrentTemperature* may exist above *HighTempThreshold* before an alarm is generated.

If this attribute takes the value 0xffffff then this alarm shall not be generated.

3.4.2.3 Commands Received

No commands are received by the server.

3.4.2.4 Commands Generated

The server generates no commands.

3.4.3 Client

3.4.3.1 Dependencies

None

3.4.3.2 Attributes

The client has no attributes.

3.4.3.3 Commands Received

No cluster specific commands are received by the client.

3.4.3.4 Commands Generated

No cluster specific commands are generated by the client.

3.5 Identify Cluster

3.5.1 Overview

Attributes and commands to put a device into an Identification mode (e.g. flashing a light), that indicates to an observer – e.g. an installer - which of several devices it is, also to request any device that is identifying itself to respond to the initiator.

Note that this cluster cannot be disabled, and remains functional regardless of the setting of the *DeviceEnable* attribute in the Basic cluster.

3.5.2 Server

3.5.2.1 Dependencies

None

3.5.2.2 Attributes

The server supports the attribute shown in Table 3.28.

Table 3.28 Attributes of the Identify Server Cluster

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0000	*IdentifyTime*	Unsigned 16-bit integer	0x0000 – 0xffff	Read / write	0x0000	M

3.5.2.2.1 *IdentifyTime* Attribute

The *IdentifyTime* attribute specifies the remaining length of time, in seconds, that the device will continue to identify itself.

If this attribute is set to a value other than 0x0000 then the device shall enter its identification procedure, in order to indicate to an observer which of several devices it is. It is recommended that this procedure consists of flashing a light with a period of 0.5 seconds. The *IdentifyTime* attribute shall be decremented every second.

If this attribute reaches or is set to the value 0x0000 then the device shall terminate its identification procedure.

3.5.2.3 Commands Received

The server side of the identify cluster is capable of receiving the commands listed in Table 3.29.

Table 3.29 Received Command IDs for the Identify Cluster

Command Identifier Field Value	Description	Mandatory / Optional
0x00	Identify	M
0x01	Identify Query	M
0x02 – 0xff	Reserved	

3.5.2.3.1 Identify Command

The identify command starts or stops the receiving device identifying itself.

3.5.2.3.1.1 Payload Format

The identify query response command payload shall be formatted as illustrated in Figure 3.7.

Octets	2
Data Type	Unsigned 16-bit integer
Field Name	Identify Time

Figure 3.7 Format of Identify Query Response Command Payload

3.5.2.3.1.2 Effect on Receipt

On receipt of this command, the device shall set the *IdentifyTime* attribute to the value of the Identify Time field. This then starts, continues, or stops the device's identification procedure as detailed in 3.5.2.2.1.

3.5.2.3.2 Identify Query Command

The identify query command allows the sending device to request the target or targets to respond if they are currently identifying themselves.

This command has no payload.

3.5.2.3.2.1 Effect on Receipt

On receipt of this command, if the device is currently identifying itself then it shall generate an appropriate Identify Query Response command, see 3.5.2.4.1, and unicast it to the requester. If the device is not currently identifying itself it shall take no further action.

3.5.2.4 Commands Generated

The server side of the identify cluster is capable of generating the commands listed in Table 3.30.

Table 3.30 Generated Command IDs for the Identify Cluster

Command Identifier Field Value	Description	Mandatory / Optional
0x00	Identify Query Response	M
0x01 – 0xff	Reserved	

3.5.2.4.1 Identify Query Response Command

The identify query response command is generated in response to receiving an Identify Query command, see 3.5.2.3.1, in the case that the device is currently identifying itself.

3.5.2.4.1.1 Payload Format

The identify query response command payload shall be formatted as illustrated in Figure 3.8.

Octets	2
Data Type	Unsigned 16-bit integer
Field Name	Timeout

Figure 3.8 Format of Identify Query Response Command Payload

3.5.2.4.1.2 Timeout Field

The Timeout field contains the current value of the *IdentifyTime* attribute, and specifies the length of time, in seconds, that the device will continue to identify itself.

3.5.2.4.1.3 Effect on Receipt

On receipt of this command, the device is informed of a device in the network which is currently identifying itself. This information may be particularly beneficial in situations where there is no commissioning tool. Note that there may be multiple responses.

3.5.3 Client

3.5.3.1 Dependencies

None.

3.5.3.2 Attributes

The client has no attributes.

3.5.3.3 Commands Received

The client receives the cluster specific response commands detailed in 3.5.2.4.

3.5.3.4 Commands Generated

The client generates the cluster specific commands detailed in 3.5.2.3, as required by the application.

3.6 Groups Cluster

3.6.1 Overview

The ZigBee specification provides the capability for group addressing. That is, any endpoint on any device may be assigned to one or more groups, each labeled with a 16-bit identifier (0x0001 – 0xfff7), which acts for all intents and purposes like a network address. Once a group is established, frames, sent using the APSDE-DATA.request primitive and having a DstAddrMode of 0x01, denoting group addressing, will be delivered to every endpoint assigned to the group address named in the DstAddr parameter of the outgoing APSDE-DATA.request primitive on every device in the network for which there are such endpoints.

Management of group membership on each device and endpoint is implemented by the APS, but the over-the-air messages that allow for remote management and commissioning of groups are defined here in the cluster library on the theory that, while the basic group addressing facilities are integral to the operation of the stack, not every device will need or want to implement this management cluster. Furthermore, the placement of the management commands here allows developers of proprietary profiles to avoid implementing the library cluster but still exploit group addressing.

Commands are defined here for discovering the group membership of a device, adding a group, removing a group and removing all groups.

Finally, the group cluster allows application entities to store a name string for each group to which they are assigned and to report that name string in response to a client request.

Note that configuration of group addresses for outgoing commands is achieved using the APS binding mechanisms, and is not part of this cluster.

3.6.1.1 Security

In order to ensure that only authorized devices are able to set up groups (particularly if application link keys are to be used) the following approach should be employed. The security Permissions Configuration Table (see [B1]) provides a mechanism by which certain commands can be restricted to specified authorized devices. Configuration of groups via the Groups cluster should use the ApplicationSettings permissions entry of this table to specify from which devices group configuration commands may be received, and whether a link key is required.

3.6.2 Server

Each ZigBee device that implements this cluster may be thought of as a group management server in the sense that it responds to information requests and configuration commands regarding the contents of its group table.

Note that, since these commands are simply data frames sent using the APSDE_SAP, they must be addressed with respect to device and endpoint. In particular the destination device and endpoint of a group management command must be unambiguous at the time of the issuance of the primitive either because:

- They are explicitly spelled out in the DstAddr and DstEndpoint parameters of the primitive.
- They are not explicitly spelled out but may be derived from the binding table in the APS of the sending device.
- Broadcast addressing is being employed, either with respect to the device address or the endpoint identifier.
- Group addressing is being employed.

On receipt of a group cluster command, the APS will, at least conceptually, deliver the frame to each destination endpoint spelled out in the addressing portion of the APS header and, again conceptually speaking, the application entity resident at that endpoint will process the command and respond as necessary. From an implementation standpoint, of course, this may be done in a more economical way that does not involve duplication and separate processing, e.g by providing a hook in the APS whereby group cluster commands could be delivered to a special application entity without duplication.

3.6.2.1 Dependencies

For correct operation of the 'Add group if identifying' command, any endpoint that implements the Groups server cluster shall also implement the Identify server cluster.

3.6.2.2 Attributes

The server supports the attribute shown in Table 3.31.

Table 3.31 Attributes of the Groups Server Cluster

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0000	*NameSupport*	8-bit bitmap	x0000000	Read only	-	M

3.6.2.2.1 *NameSupport* Attribute

The most significant bit of the *NameSupport* attribute indicates whether or not group names are supported. A value of 1 indicates that they are supported, and a value of 0 indicates that they are not supported.

3.6.2.2.2 Group Names

Group names are between 0 and 16 characters long. Support of group names is optional, and is indicated by the *NameSupport* attribute. Group names, if supported, must be stored in a separate data structure managed by the application in which the entries correspond to group table entries.

3.6.2.3 Commands Received

The groups cluster is concerned with management of the group table on a device. In practice, the group table is managed by the APS and the table itself is available to the next higher layer as an AIB attribute. A command set is defined here and the implementation details of that command set in terms of the facilities provided by the APS is left up to the implementer of the cluster library itself.

The server side of the groups cluster is capable of receiving the commands listed in Table 3.32.

Table 3.32 Received Command IDs for the Groups Cluster

Command Identifier Field Value	Description	Mandatory / Optional
0x00	Add group	M
0x01	View group	M
0x02	Get group membership	M
0x03	Remove group	M
0x04	Remove all groups	M
0x05	Add group if identifying	M
0x06 – 0xff	Reserved	-

3.6.2.3.1 Add Group Command

The add group command allows the sending device to add group membership in a particular group for one or more endpoints on the receiving device.

3.6.2.3.1.1 Payload Format

The Add Group command payload shall be formatted as illustrated in Figure 3.9

Octets		Variable
Data Type	Unsigned 16-bit integer	Character string
Field Name	Group ID	Group Name

Figure 3.9 Format of the Add Group Command Payload

3.6.2.3.1.2 Effect on Receipt

On receipt of this command, the device shall (if possible) add the Group ID and Group Name to its Group Table. It shall then generate an appropriate Add Group Response command indicating success or failure. See 3.6.2.4.1.

3.6.2.3.2 View Group Command

The view group command allows the sending device to request that the receiving entity or entities respond with a view group response command containing the application name string for a particular group.

3.6.2.3.2.1 Payload Format

The View Group command payload shall be formatted as illustrated in Figure 3.10.

Octets	2
Data Type	Unsigned 16-bit integer
Field Name	Group ID

Figure 3.10 Format of the View Group Command Payload

3.6.2.3.2.2 Effect on Receipt

On receipt of this command, the device shall generate an appropriate View Group Response command. 3.6.2.4.2.

3.6.2.3.3 Get Group Membership Command

The get group membership command allows the sending device to inquire about the group membership of the receiving device and endpoint in a number of ways.

3.6.2.3.3.1 Payload Format

The get group membership command payload shall be formatted as illustrated in Figure 3.11.

Octets	1	Variable
Data Type	Unsigned 8-bit integer	List of 16-bit integers
Field Name	Group count	Group list

Figure 3.11 Format of Get Group Membership Command Payload

3.6.2.3.3.2 Effect on Receipt

On receipt of the get group membership command, each receiving entity shall respond, as necessary, with group membership information using the get group membership response frame shown below. An entity shall respond if and only if:

- The group count field of the command frame has a value of 0 indicating that the group list field is empty, or
- The group list field of the command frame contains at least one group of which the entity is a member. In this case the response frame will contain the identifiers of all such groups.

3.6.2.3.4 Remove Group Command

The remove group command allows the sender to request that the receiving entity or entities remove their membership, if any, in a particular group.

Note that if a group is removed the scenes associated with that group should be removed.

3.6.2.3.4.1 Payload Format

The Remove Group command payload shall be formatted as illustrated in Figure 3.12.

Octets	2
Data Type	Unsigned 16-bit integer
Field Name	Group ID

Figure 3.12 Format of the Remove Group Command Payload

3.6.2.3.4.2 Effect on Receipt

On receipt of this command, the device shall (if possible) remove the Group ID and Group Name from its Group Table. It shall then generate an appropriate Remove Group Response command indicating success or failure. See 3.6.2.4.4.

3.6.2.3.5 Remove All Groups Command

The remove all groups command allows the sending device to direct the receiving entity or entities to remove all group associations.

Note that removing all groups necessitates the removal of all associated scenes as well. (Note: scenes not associated with a group need not be removed).

3.6.2.3.5.1 Payload Format

The Remove All Groups command has no payload.

3.6.2.3.5.2 Effect on Receipt

On receipt of this command, the device shall remove all groups on this endpoint from its Group Table.

3.6.2.3.6 Add Group If Identifying Command

The add group if identifying command allows the sending device to add group membership in a particular group for one or more endpoints on the receiving device, on condition that it is identifying itself. Identifying functionality is controlled using the identify cluster, (see 3.5).

This command might be used to assist configuring group membership in the absence of a commissioning tool.

3.6.2.3.6.1 Payload Format

The Add Group If Identifying command payload shall be formatted as illustrated in Figure 3.13.

Octets	2	Variable
Data Type	Unsigned 16-bit integer	Character string
Field Name	Group ID	Group Name

Figure 3.13 Add Group If Identifying Command Payload

3.6.2.3.6.2 Effect on Receipt

On receipt of this command, the device shall first check whether it is currently identifying itself. If so then the device shall (if possible) add the Group ID and Group Name to its Group Table. If the device it not currently identifying itself then no action shall be taken.

No response is defined as this command is expected to be multicast or broadcast.

3.6.2.4 Commands Generated

The commands generated by the server side of the groups cluster, as listed in Table 3.33, are responses to the received commands listed above in sub-clause 3.6.2.3.

Table 3.33 Generated Command IDs for the Groups Cluster

Command Identifier Field Value	Description	Mandatory / Optional
0x00	Add group response	M
0x01	View group response	M
0x02	Get group membership response	M
0x03	Remove group response	M
0x04 – 0xff	Reserved	

(**Note:** There is no need for a response to the Remove all Groups command, as, at an application level, this command always succeeds)

3.6.2.4.1 Add Group Response Command

The add group response is sent by the groups cluster server in response to an add group command.

3.6.2.4.1.1 Payload Format

The Add Group Response command payload shall be formatted as illustrated in Figure 3.14.

Octets	1	2
Data Type	8-bit enumeration	Unsigned 16-bit integer
Field Name	Status	Group ID

Figure 3.14 Format of the Add Group Response Command Payload

3.6.2.4.1.2 When Generated

This command is generated in response to a received Add Group command 10.2.2.3. The Status field is set to SUCCESS, DUPLICATE_EXISTS, or INSUFFICIENT_SPACE as appropriate. The Group ID field is set to the Group ID field of the received Add Group command.

3.6.2.4.2 View Group Response Command

The view group response command is sent by the groups cluster server in response to a view group command.

3.6.2.4.2.1 Payload Format

The View Group Response command payload shall be formatted as illustrated in Figure 3.15.

Octets	1	2	Variable
Data Type	8-bit enumeration	Unsigned 16-bit integer	Character string
Field Name	Status	Group ID	Group Name

Figure 3.15 Format of the View Group Response Command Payload

3.6.2.4.2.2 When Generated

This command is generated in response to a received View Group command 10.2.2.4. The Status field is set to SUCCESS or NOT_FOUND as appropriate. The Group ID field is set to the Group ID field of the received View Group command. If the status is SUCCESS, and group names are supported, the Group Name field is set to the Group Name associated with that Group ID in the Group Table; otherwise it is set to the null (empty) string, i.e. a single octet of value 0.

3.6.2.4.3 Get Group Membership Response Command

The get group membership response command is sent by the groups cluster server in response to a get group membership command.

3.6.2.4.3.1 Payload Format

The payload of the get group membership response command is formatted as shown in Figure 3.16.

Octets	1	1	Variable
Data Type	Unsigned 8-bit integer	Unsigned 8-bit integer	List of 16-bit group ID
Field Name	Capacity	Group count	Group list

Figure 3.16 Format of the Get Group Membership Response Command Payload

The fields of the get group membership response command have the following semantics:

- The Capacity field shall contain the remaining capacity of the group table of the device. The following values apply:
  0No further groups may be added.
  0 < Capacity < 0xfeCapacity holds the number of groups that may be added
  0xfeAt least 1 further group may be added (exact number is unknown)
  0xffIt is unknown if any further groups may be added
- The Group count field shall contain the number of groups contained in the group list field.
- The Group list field shall contain the identifiers either of all the groups in the group table (in the case where the group list field of the received get group membership command was empty) or all the groups from the group list field of the received get group membership command which are in the group table.

3.6.2.4.3.2 When Generated

When an application entity receives the get group membership command and either the group list of the command payload is empty or the group list contains at least one group to which the entity belongs, the entity shall respond with a get group membership response command.

3.6.2.4.4 Remove Group Response Command

The remove group response command is generated by an application entity in response to the receipt of a remove group command.

3.6.2.4.4.1 Payload Format

The Remove Group Response command payload shall be formatted as illustrated in Figure 3.17.

Octets	1	2
Data Type	8-bit enumeration	Unsigned 16-bit integer
Field Name	Status	Group ID

Figure 3.17 Format of Remove Group Response Command Payload

3.6.2.4.4.2 When Generated

This command is generated in response to a received Remove Group command 3.6.2.3.4. The Status field is set to SUCCESS or NOT_FOUND as appropriate. The Group ID field is set to the Group ID field of the received Remove Group command.

3.6.3 Client

3.6.3.1 Dependencies

None.

3.6.3.2 Attributes

The Client cluster has no attributes.

3.6.3.3 Commands Received

The client receives the cluster specific response commands detailed in 3.6.2.4.

3.6.3.4 Commands Generated

The client generates the cluster specific commands detailed in 3.6.2.3.

3.7 Scenes Cluster

3.7.1 Overview

The scenes cluster provides attributes and commands for setting up and recalling scenes. Each scene corresponds to a set of stored values of specified attributes for one or more clusters on the same end point as the scenes cluster.

In most cases scenes are associated with a particular group ID. Scenes may also exist without a group, in which case the value 0x0000 replaces the group ID. Note that extra care is required in these cases to avoid a scene ID collision, and that commands related to scenes without a group may only be unicast, i.e.: they may not be multicast or broadcast.

3.7.2 Server

3.7.2.1 Dependencies

Any endpoint that implements the Scenes server cluster shall also implement the Groups server cluster.

3.7.2.2 Attributes

For convenience, the attributes defined in this specification are arranged into sets of related attributes; each set can contain up to 16 attributes. Attribute identifiers are encoded such that the most significant three nibbles specify the attribute set and the least significant nibble specifies the attribute within the set. The currently defined attribute sets are listed in Table 3.34.

Table 3.34 Scenes Attribute Sets

Attribute Set Identifier	Description
0x000	Scene Management Information
0x001 – 0xfff	Reserved

3.7.2.2.1 Scene Management Information Attribute Set

The Scene Management Information attribute set contains the attributes summarized in Table 3.35.

Table 3.35 Scene Management Information Attribute Set

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0000	*SceneCount*	Unsigned 8-bit integer	0x00 – 0xff (see 3.7.2.3.1)	Read only	0x00	M
0x0001	*CurrentScene*	Unsigned 8-bit integer	0x00 – 0xff (see 3.7.2.3.1)	Read only	0x00	M
0x0002	*CurrentGroup*	Unsigned 16-bit integer	0x0000 – 0xfff7	Read only	0x00	M
0x0003	*SceneValid*	Boolean	0x00 – 0x01	Read only	0x00	M
0x0004	*NameSupport*	8-bit bitmap	x0000000	Read only	-	M
0x0005	*LastConfiguredBy*	IEEE Address	-	Read only	-	O

3.7.2.2.1.1 *SceneCount* Attribute

The *SceneCount* attribute specifies the number of scenes currently in the device's scene table.

3.7.2.2.1.2 *CurrentScene* Attribute

The *CurrentScene* attribute holds the Scene ID of the scene last invoked.

3.7.2.2.1.3 *CurrentGroup* Attribute

The *CurrentGroup* attribute holds the Group ID of the scene last invoked, or 0x0000 if the scene last invoked is not associated with a group.

3.7.2.2.1.4 *SceneValid* Attribute

The *SceneValid* attribute indicates whether the state of the device corresponds to that associated with the *CurrentScene* and *CurrentGroup* attributes. TRUE indicates that these attributes are valid, FALSE indicates that they are not valid.

Before a scene has been stored or recalled, this attribute is set to FALSE. After a successful Store Scene or Recall Scene command it is set to TRUE. If, after a

scene is stored or recalled, the state of the device is modified, this attribute is set to FALSE.

3.7.2.2.1.5 *NameSupport* Attribute

The most significant bit of the *NameSupport* attribute indicates whether or not scene names are supported. A value of 1 indicates that they are supported, and a value of 0 indicates that they are not supported.

3.7.2.2.1.6 *LastConfiguredBy* Attribute

The *LastConfiguredBy* attribute is 64 bits in length and specifies the IEEE address of the device that last configured the scene table.

The value 0xffffffffffffffff indicates that the device has not been configured, or that the address of the device that last configured the scenes cluster is not known.

3.7.2.3 Scene Table

The scene table is used to store information for each scene capable of being invoked on a device. Each scene is defined for a particular group.

The fields of each scene table entry consist of a number of sets. The base set consists of the first four fields of Table 3.36. A set of extension fields can be added by each additional cluster implemented on a device.

Table 3.36 Fields of a Scene Table Entry

Field	Type	Valid Range	Description
Scene group ID	Unsigned 16-bit integer	0x0000 – 0xfff7	The group ID for which this scene applies, or 0x0000 if the scene is not associated with a group.
Scene ID	Unsigned 8-bit integer	0x00 – 0xff (see 3.7.2.3.1)	The identifier, unique within this group, which is used to identify this scene.

Table 3.36 Fields of a Scene Table Entry (Continued)

Field	Type	Valid Range	Description
Scene name	Character string	0 – 16 characters	The name of the scene (optional)
Scene transition time	Unsigned 16-bit integer	0x0000 – 0xffff	The amount of time, in seconds, it will take for the device to change from its current state to the requested scene.
Extension field sets	Variable	Variable	See the Scene Table Extensions subsections of individual clusters. Each extension field set holds a set of values of attributes for a cluster implemented on the device. The sum of all such sets defines a scene.

3.7.2.3.1 Scene Names

Scene names are between 0 and 16 characters long. Support of scene names is optional, and is indicated by the *NameSupport* attribute. If scene names are not supported, any commands that writes a scene name shall simply discard the name, and any command that returns a scene names shall return the null string.

3.7.2.3.2 Maximum Number of Scenes

The number of scenes capable of being stored in the table is defined by the profile in which this cluster is used. The default maximum, in the absence of specification by the profile, is 16.

3.7.2.4 Commands Received

The received command IDs for the Scenes cluster are listed in Table 3.37.

Table 3.37 Received Command IDs for the Scenes Cluster

Command Identifier Field Value	Description	Mandatory / Optional
0x00	Add scene	M
0x01	View scene	M
0x02	Remove scene	M
0x03	Remove all scenes	M
0x04	Store scene	M

Table 3.37 Received Command IDs for the Scenes Cluster (Continued)

Command Identifier Field Value	Description	Mandatory / Optional
0x05	Recall scene	M
0x06	Get scene membership	M
0x07 – 0xff	Reserved	

3.7.2.4.1 Add Scene Command

The Add Scene command shall be addressed to a single device (not a group).

3.7.2.4.1.1 Payload Format

The payload shall be formatted as illustrated in Figure 3.18.

Octets	2	1	2	Variable	Variable
Data Type	Unsigned 16-bit integer	Unsigned 8-bit integer	Unsigned 16-bit integer	Character string	Variable (multiple types)
Field Name	Group ID	Scene ID	Transition time	Scene Name	Extension field sets, one per cluster

Figure 3.18 Format of the Add Scene Command Payload

The format of each extension field set is a 16 bit field carrying the cluster ID, followed by an 8 bit length field and the set of scene extension fields specified in the relevant cluster. The length field holds the length in octets of that extension field set.

Extension field sets =
{{ClusterID 1, length 1, {extension field set 1}}, {ClusterID 2, length 2, {extension field set 2}}, ... }.

The attributes included in the extension field set for each cluster are defined in the specification for that cluster in this document (the ZigBee Cluster Library). The field set consists of values for these attributes concatenated together, in the order given in the cluster specification, with no attribute identifiers or data type indicators.

For forward compatibility, reception of this command shall allow for the possible future addition of other attributes to the trailing ends of the lists given in the cluster specifications (by ignoring them). Similarly, it shall allow for one or more attributes to be omitted from the trailing ends of these lists (see 3.7.2.4.6.2).

It is not mandatory for a field set to be included in the command for every cluster on that endpoint that has a defined field set. Extension field sets may be omitted, including the case of no field sets at all.

3.7.2.4.1.2 Effect on Receipt

On receipt of this command, the device shall (if possible) create an entry in the Scene Table with fields copied from the command payload. If there is already a scene in the table with the same Scene ID and Group ID, it shall overwrite it, (i.e. it shall first remove all information included in the original scene entry).

It shall then generate an appropriate Add Scene Response command indicating success or failure. See 3.7.2.5.1.

3.7.2.4.2 View Scene Command

The View Scene command shall be addressed to a single device (not a group).

3.7.2.4.2.1 Payload Format

The payload shall be formatted as illustrated in Figure 3.19.

Octets	2	1
Data Type	Unsigned 16-bit integer	Unsigned 8-bit integer
Field Name	Group ID	Scene ID

Figure 3.19 Format of the View Scene Command Payload

3.7.2.4.2.2 Effect on Receipt

On receipt of this command, the device shall generate an appropriate View Scene Response command. See 3.7.2.5.2.

3.7.2.4.3 Remove Scene Command

The Remove Scene command may be addressed to a single device or to a group.

3.7.2.4.3.1 Payload Format

The Remove Scene command payload shall be formatted as illustrated in Figure 3.20.

Octets	2	1
Data Type	Unsigned 16-bit integer	Unsigned 8-bit integer
Field Name	Group ID	Scene ID

Figure 3.20 Format of the Remove Scene Command Payload

3.7.2.4.3.2 Effect on Receipt

On receipt of this command, the device shall (if possible) remove from its Scene Table the entry with this Scene ID and group ID. If the command was addressed to a single device (not a group) then it shall generate an appropriate Remove Scene Response command indicating success or failure. See 3.7.2.5.3.

3.7.2.4.4 Remove All Scenes Command

The Remove All Scenes may be addressed to a single device or to a group.

3.7.2.4.4.1 Payload Format

The Remove All Scenes command payload shall be formatted as illustrated in Figure 3.21.

Octets	2
Data Type	Unsigned 16-bit integer
Field Name	Group ID

Figure 3.21 Format of the Remove All Scenes Command Payload

3.7.2.4.4.2 Effect on Receipt

On receipt of this command, the device shall, if possible, remove from its Scene Table all entries with this Group ID. If the command was addressed to a single device (not to a group) it shall then generate an appropriate Remove All Scenes Response command indicating success or failure. See 3.7.2.5.4.

3.7.2.4.5 Store Scene Command

The Store Scene command may be addressed to a single device or to a group.

3.7.2.4.5.1 Payload Format

The Store Scene command payload shall be formatted as illustrated in Figure 3.22.

Octets	2	1
Data Type	Unsigned 16-bit integer	Unsigned 8-bit integer
Field Name	Group ID	Scene ID

Figure 3.22 Format of the Store Scene Command Payload

3.7.2.4.5.2 Effect on Receipt

On receipt of this command, the device shall (if possible) add an entry to the Scene Table with the Scene ID and Group ID given in the command, and all extension field sets corresponding to the current state of other clusters on the device.

If an entry already exists with the same Scene ID and Group ID, as a result of a previous Add Scene command, the extension field sets are overwritten (i.e. completely replaced), but the transition time field and the scene name field are left unaltered. If no such entry exists, the transition time field shall be set to 0, and the scene name field shall be set to the null string.

Note that, accordingly, if a scene to be stored requires a transition time field and/ or a scene name field, these must be set up by a prior Add Scene command, e.g. with no scene extension field sets.

If the Group ID field is not zero, and the device is not a member of this group, the scene will not be added.

If the command was addressed to a single device (not to a group) then it shall generate an appropriate Store Scene Response command indicating success or failure. See 3.7.2.5.5.

3.7.2.4.6 Recall Scene Command

The Recall Scene command may be addressed to a single device or to a group.

3.7.2.4.6.1 Payload Format

The Recall Scene command payload shall be formatted as illustrated in Figure 3.23.

Octets	2	1
Data Type	Unsigned 16-bit integer	Unsigned 8-bit integer
Field Name	Group ID	Scene ID

Figure 3.23 Format of the Recall Scene Command Payload

3.7.2.4.6.2 Effect on Receipt

On receipt of this command, the device shall (if possible) locate the entry in its Scene Table with the Group ID and Scene ID given in the command. For each other cluster on the device, it shall then retrieve any corresponding extension fields from the Scene Table and set the attributes and corresponding state of the cluster accordingly.

If there is no extension field set for a cluster, the state of that cluster shall remain unchanged. If an extension field set omits the values of any trailing attributes, the values of these attributes shall remain unchanged.

The scene transition time field of the entry indicates how long the transition takes from the old cluster state to the new cluster state. It is recommended that, where possible (e.g. it is not possible for attributes with boolean data type), a gradual transition should take place from the old to the new state over this time. However, the interpretation of this field is manufacturer dependent.

This command does not result in a response command.

3.7.2.4.7 Get Scene Membership Command

The Get Scene Membership command can be used to find an unused scene number within the group when no commissioning tool is in the network, or for a commissioning tool to get used scenes for a group on a single device or on all devices in the group.

3.7.2.4.7.1 Payload Format

The Get Scene Membership command may be addressed to a single device or to a group.

The Get Scene Membership command payload shall be formatted as illustrated in Figure 3.24.

Ocets	2
Data Type	Unsigned 16-bit integer
Field Name	Group ID

Figure 3.24 Format of Get Scene Membership Command Payload

3.7.2.4.7.2 Effect on Receipt

On receipt of this command, the device shall if addressed to a single device generate an appropriate Get Scene Membership Response command, otherwise it shall only generate an appropriate Get Scene Membership Response command if an entry within the Scene Table corresponds to the Group ID. See 3.7.2.5.6.

3.7.2.5 Commands Generated

The generated command IDs for the Scenes cluster are listed in Table 3.38.

Table 3.38 Generated Command IDs for the Scenes Cluster

Command Identifier Field Value	Description	Mandatory / Optional
0x00	Add scene response	M
0x01	View scene response	M
0x02	Remove scene response	M
0x03	Remove all scenes response	M
0x04	Store scene response	M
0x05	Reserved	-
0x06	Get scene membership response	M
0x07 – 0xff	Reserved	-

3.7.2.5.1 Add Scene Response Command

3.7.2.5.1.1 Payload Format

The Add Scene Response command payload shall be formatted as illustrated in Figure 3.25.

Octets	1	2	1
Data Type	8-bit enumeration	Unsigned 16-bit integer	Unsigned 8-bit integer
Field Name	Status	Group ID	Scene ID

Figure 3.25 Format of the Add Scene Response Command Payload

3.7.2.5.1.2 When Generated

This command is generated in response to a received Add Scene command 11.2.4.1. The Status field is set to SUCCESS, INSUFFICIENT_SPACE or INVALID_FIELD (the group is not present in the Group Table) as appropriate. The Group ID and Scene ID fields are set to the corresponding fields of the received Add Scene command.

3.7.2.5.2 View Scene Response Command

3.7.2.5.2.1 Payload Format

The View Scene Response command payload shall be formatted as illustrated in Figure 3.26.

Octets	1	2	1	0 / 2	0 / Variable	0 / Variable
Data Type	8-bit enumeration	Unsigned 16-bit integer	Unsigned 8-bit integer	Unsigned 16-bit integer	Character string	Variable (multiple types)
Field Name	Status	Group ID	Scene ID	Transition time	Scene Name	Extension field sets, one per cluster

Figure 3.26 Format of the View Scene Response Command Payload

The format of each extension field set is a 16 bit field carrying the cluster ID, followed by an 8 bit data length field and the set of scene extension fields specified in the relevant cluster. These fields are concatenated together in the order given in the cluster.

Extension field sets =
{{ClusterID 1, length 1, {extension field set 1}}, {ClusterID 2, length 2, {extension field set 2 }}, ... }.

3.7.2.5.2.2 When Generated

This command is generated in response to a received View Scene command 11.2.4.2.

The entry in the Scene Table with Scene ID and Group ID given in the received View Scene command is located (if possible). The Status field is set to SUCCESS, NOT_FOUND (the scene is not present in the Scene Table) or INVALID_FIELD (the group is not present in the Group Table) as appropriate. The Group ID and Scene ID fields are set to the corresponding fields in the received View Scene command.

If the status is SUCCESS, the Transition time, Scene Name and Extension field fields are copied from the corresponding fields in the table entry, otherwise they are omitted.

3.7.2.5.3 Remove Scene Response Command

3.7.2.5.3.1 Payload Format

The Remove Scene Response command payload shall be formatted as illustrated in Figure 3.27.

Octets	1	2	1
Data Type	8-bit enumeration	Unsigned 16-bit integer	Unsigned 8-bit integer
Field Name	Status	Group ID	Scene ID

Figure 3.27 Format of Remove Scene Response Command Payload

3.7.2.5.3.2 When Generated

This command is generated in response to a received Remove Scene command 10.2.2.4. The Status field is set to SUCCESS, NOT_FOUND (the scene is not present in the Scene Table) or INVALID_FIELD (the group is not present in the Group Table) as appropriate. The Group ID and Scene ID fields are set to the corresponding fields of the received Remove Scene command.

3.7.2.5.4 Remove All Scenes Response Command

3.7.2.5.4.1 Payload Format

The Remove All Scenes Response command payload shall be formatted as illustrated in Figure 3.28.

Octets	1	2
Data Type	8-bit enumeration	Unsigned 16-bit integer
Field Name	Status	Group ID

Figure 3.28 Format of the Remove All Scenes Response Command Payload

3.7.2.5.4.2 When Generated

This command is generated in response to a received Remove All Scenes command, see 3.7.2.4.4. The Status field is set to SUCCESS or INVALID_FIELD (the group is not present in the Group Table) as appropriate. The Group ID field is set to the corresponding field of the received Remove All Scenes command.

3.7.2.5.5 Store Scene Response Command

3.7.2.5.5.1 Payload Format

The Store Scene Response command payload shall be formatted as illustrated in Figure 3.29.

Octets	1	2	1
Data Type	8-bit enumeration	Unsigned 16-bit integer	Unsigned 8-bit integer
Field Name	Status	Group ID	Scene ID

Figure 3.29 Format of the Store Scene Response Command Payload

3.7.2.5.5.2 When Generated

This command is generated in response to a received Store Scene command 10.2.2.4. The Status field is set to SUCCESS, INSUFFICIENT_SPACE or INVALID_FIELD (the group is not present in the Group Table) as appropriate. The Group ID and Scene ID fields are set to the corresponding fields of the received Store Scene command.

3.7.2.5.6 Get Scene Membership Response Command

3.7.2.5.6.1 Payload Format

The Get Scene Membership Response command payload shall be formatted as illustrated in Figure 3.30.

Octets	1	1	2	0/1	Variable
Data Type	8-bit enumeration	Unsigned 8-bit integer	Unsigned 16-bit integer	Unsigned 8-bit integer	Unsigned 8-bit integer x N
Field Name	Status	Capacity	Group ID	Scene count	Scene list

Figure 3.30 Format of the Get Scene Membership Response Command Payload

The fields of the get scene membership response command have the following semantics:

- The Capacity field shall contain the remaining capacity of the scene table of the device. (for all groups). The following values apply:
  0No further scenes may be added.
  0 < Capacity < 0xfeCapacity holds the number of scenes that may be added
  0xfeAt least 1 further scene may be added (exact number is unknown)
  0xffIt is unknown if any further scenes may be added
- The Status field shall contain SUCCESS or INVALID_FIELD (the group is not present in the Group Table) as appropriate.
- The Group ID field shall be set to the corresponding field of the received Get Scene Membership command.
- If the status is not SUCCESS, then the Scene count and Scene list field are omitted, else
  - The Scene count field shall contain the number of scenes contained in the Scene list field.
  - The Scene list field shall contain the identifiers of all the scenes in the scene table with the corresponding Group ID.

3.7.2.5.6.2 When Generated

This command is generated in response to a received Get Scene Membership command, 3.7.2.4.7.

3.7.3 Client

3.7.3.1 Dependencies

None.

3.7.3.2 Attributes

The Client cluster has no attributes.

3.7.3.3 Commands Received

The client receives the cluster specific response commands detailed in 3.7.2.5.

3.7.3.4 Commands Generated

The client generates the cluster specific commands detailed in 3.7.2.4, as required by the application.

3.8 On/Off Cluster

3.8.1 Overview

Attributes and commands for switching devices between ‘On’ and ‘Off’ states.

3.8.2 Server

3.8.2.1 Dependencies

None

3.8.2.2 Attributes

The server supports the attributes shown in Table 3.39.

Table 3.39 Attributes of the On/Off Server Cluster

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0000	*OnOff*	Boolean	0x00 – 0x01	Read only	0x00	M

The *OnOff* attribute has the following values: 0 = Off, 1 = On

3.8.2.3 Commands Received

The command IDs for the *On/Off* cluster are listed in Table 3.40.

Table 3.40 Command IDs for the On/Off Cluster

Command Identifier Field Value	Description	Mandatory / Optional
0x00	Off	M
0x01	On	M
0x02	Toggle	M
0x03 – 0xff	Reserved	

3.8.2.3.1 Off Command

This command does not have a payload.

3.8.2.3.1.1 Effect on Receipt

On receipt of this command, a device shall enter its 'Off' state. This state is device dependent, but it is recommended that it is used for power off or similar functions.

3.8.2.3.2 On Command

This command does not have a payload.

3.8.2.3.2.1 Effect on Receipt

On receipt of this command, a device shall enter its 'On' state. This state is device dependent, but it is recommended that it is used for power on or similar functions.

3.8.2.3.3 Toggle Command

This command does not have a payload.

3.8.2.3.3.1 Effect on Receipt

On receipt of this command, if a device is in its ‘Off’ state it shall enter its 'On' state. Otherwise, if it is in its ‘On’ state it shall enter its 'Off' state.

3.8.2.4 Commands Generated

The server generates no commands.

3.8.2.5 Scene Table Extensions

If the Scenes server cluster (11) is implemented, the following extension field is added to the Scenes table:

OnOff

3.8.2.6 Attribute Reporting

This cluster shall support attribute reporting using the Report Attributes command and according to the minimum and maximum reporting interval settings described in the ZCL Foundation specification (see 2.4.7). The following attribute shall be reported:

OnOff

3.8.3 Client

3.8.3.1 Dependencies

None.

3.8.3.2 Attributes

The client has no attributes.

3.8.3.3 Commands Received

No cluster specific commands are received by the client.

3.8.3.4 Commands Generated

The client generates the cluster specific commands received by the server, as required by the application. See 3.8.2.3

3.9 On/Off Switch Configuration Cluster

3.9.1 Overview

Attributes and commands for configuring On/Off switching devices

3.9.2 Server

3.9.2.1 Dependencies

Any endpoint that implements this server cluster shall also implement the On/Off client cluster.

3.9.2.2 Attributes

For convenience, the attributes defined in this specification are arranged into sets of related attributes; each set can contain up to 16 attributes. Attribute identifiers are encoded such that the most significant three nibbles specify the attribute set and the least significant nibble specifies the attribute within the set. The currently defined attribute sets are listed in Table 3.41.

Table 3.41 On/Off Switch Configuration Attribute Sets

Attribute Set Identifier	Description
0x000	Switch Information
0x001	Switch Settings
0x002 – 0xfff	Reserved

3.9.2.2.1 Switch Information Attribute Set

The switch information attribute set contains the attributes summarized in Table 3.42.

Table 3.42 Attributes of the Switch Information Attribute Set

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0000	*SwitchType*	8-bit enumeration	0x00 – 0x01	Read only	-	M

3.9.2.2.2 *SwitchType* Attribute

The *SwitchType* attribute specifies the basic functionality of the On/Off switching device. This attribute shall be set to one of the non-reserved values listed in Table 3.43.

Table 3.43 Values of the *SwitchType* Attribute

Attribute Value	Description	Details
0x00	Toggle	A switch with two physical states. An action by the user (e.g. toggling a rocker switch) moves the switch from state 1 to state 2. The switch then remains in that state until another action from the user returns it to state 1.
0x01	Momentary	A switch with two physical states. An action by the user (e.g. pressing a button) moves the switch from state 1 to state 2. When the user ends his action (e.g. releases the button) the switch returns to state 1.
0x02 – 0xff	Reserved	-

3.9.2.2.3 Switch Settings Attribute Set

The switch settings attribute set contains the attributes summarized in Table 3.44.

Table 3.44 Attributes of the Switch Settings Attribute Set

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0010	*SwitchActions*	8-bit enumeration	0x00 – 0x02	Read/ write	0x00	M

3.9.2.2.3.1 *SwitchActions* Attribute

The *SwitchActions* attribute is 8 bits in length and specifies the commands of the On/Off cluster (12) to be generated when the switch moves between its two states, as detailed in Table 3.45.

Table 3.45 Values of the *SwitchActions* Attribute

Attribute Value	Command Generated When Arriving at State 2 From State 1	Command Generated When Arriving at State 1 From State 2
0x00	On	Off

Table 3.45 Values of the *SwitchActions* Attribute (Continued)

Attribute Value	Command Generated When Arriving at State 2 From State 1	Command Generated When Arriving at State 1 From State 2
0x01	Off	On
0x02	Toggle	Toggle
0x03 – 0xff	Reserved	

3.9.2.3 Commands Received

No commands are received by the server.

3.9.2.4 Commands Generated

The server generates no commands.

3.9.3 Client

3.9.3.1 Dependencies

None

3.9.3.2 Attributes

The client has no attributes.

3.9.3.3 Commands Received

No cluster specific commands are received by the client.

3.9.3.4 Commands Generated

No cluster specific commands are generated by the client.

3.10 Level Control Cluster

3.10.1 Overview

This cluster provides an interface for controlling a characteristic of a device that can be set to a level, for example the brightness of a light, the degree of closure of a door, or the power output of a heater.

3.10.2 Server

3.10.2.1 Dependencies

For many applications, a close relationship between this cluster and the OnOff cluster is needed. This section describes the dependencies that are required when an endpoint that implements the Level Control server cluster also implements the On/Off server cluster (12.2).

The *OnOff* attribute of the On/Off cluster and the *CurrentLevel* attribute of the Level Control cluster are intrinsically independent variables, as they are on different clusters. However, when both clusters are implemented on the same endpoint, dependencies may be introduced between them. Facilities are provided to introduce dependencies if required.

3.10.2.1.1 Effect of On/Off Commands on the *CurrentLevel* Attribute

The attribute *OnLevel* (see 3.10.2.2.4) determines whether commands of the On/Off cluster have a permanent effect on the *CurrentLevel* attribute or not. If this attribute is defined (i.e. implemented and not 0xff) they do have a permanent effect, otherwise they do not. There is always a temporary effect, due to fading up / down.

The effect on the Level Control cluster on receipt of the various commands of the On/Off cluster are as detailed in Table 3.46. In this table, and throughout this cluster specification, 'level' means the value of the *CurrentLevel* attribute (see 3.10.2.2.1).

Table 3.46 Actions on Receipt for On/Off Commands, when Associated with Level Control

Command	Action On Receipt
On	Temporarily store *CurrentLevel* Set *CurrentLevel* to the minimum level allowed for the device. Move *CurrentLevel* to *OnLevel*, or to the stored level if *OnLevel* is not defined, over the time period *OnOffTransitionTime*.
Off	Temporarily store *CurrentLevel* Move *CurrentLevel* to the minimum level allowed for the device over the time period *OnOffTransitionTime*. If *OnLevel* is not defined, set the *CurrentLevel* to the stored level.
Toggle	If the *OnOff* attribute has the value Off, proceed as for the On command. Otherwise proceed as for the Off command.

3.10.2.1.2 Effect of Level Control Commands on the *OnOff* Attribute

There are two sets of commands provided in the Level Control cluster. These are identical, except that the first set (Move to Level, Move and Step) shall not effect the *OnOff* attribute, whereas the second set ('with On/Off' variants) shall.

The first set is used to maintain independence between the *CurrentLevel* and *OnOff* attributes, so changing *CurrentLevel* has no effect on the *OnOff* attribute. As examples, this represents the behavior of a volume control with a mute button, or a 'turn to set level and press to turn on/off' light dimmer.

The second set is used to link the *CurrentLevel* and *OnOff* attributes. When the level is reduced to its minimum the *OnOff* attribute is automatically turned to Off, and when the level is increased above its minimum the *OnOff* attribute is automatically turned to On. As an example, this represents the behavior of a light dimmer with no independent on/off switch.

3.10.2.2 Attributes

The attributes of the Level Control server cluster are summarized in Table 3.47.

Table 3.47 Attributes of the Level Control Server Cluster

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0000	*CurrentLevel*	Unsigned 8-bit integer	0x00 – 0xff	Read only	0x00	M
0x0001	*RemainingTime*	Unsigned 16-bit integer	0x0000 – 0xffff	Read only	0x0000	O
0x0010	*OnOffTransitionTime*	Unsigned 16-bit integer	0x0000 – 0xffff	Read / Write	0x0000	O
0x0011	*OnLevel*	Unsigned 8-bit integer	0x00 – 0xfe	Read / Write	0xfe	O

3.10.2.2.1 *CurrentLevel* Attribute

The *CurrentLevel* attribute represents the current level of this device. The meaning of 'level' is device dependent.

3.10.2.2.2 *RemainingTime* Attribute

The *RemainingTime* attribute represents the time remaining until the current command is complete - it is specified in 1/10ths of a second.

3.10.2.2.3 *OnOffTransitionTime* Attribute

The *OnOffTransitionTime* attribute represents the time taken to move to or from the target level when On of Off commands are received by an On/Off cluster on the same endpoint. It is specified in 1/10ths of a second.

The actual time taken should be as close to *OnOffTransitionTime* as the device is able. N.B. If the device is not able to move at a variable rate, the *OnOffTransitionTime* attribute should not be implemented.

3.10.2.2.4 *OnLevel* Attribute

The *OnLevel* attribute determines the value that the *CurrentLevel* attribute is set to when the *OnOff* attribute of an On/Off cluster on the same endpoint is set to On. If the *OnLevel* attribute is not implemented, or is set to 0xff, it has no effect. For more details see 3.10.2.1.1.

3.10.2.3 Commands Received

The command IDs for the Level Control cluster are listed in Table 3.48.

Table 3.48 Command IDs for the Level Control Cluster

Command Identifier Field Value	Description	Mandatory / Optional
0x00	Move to Level	M
0x01	Move	M
0x02	Step	M
0x03	Stop	M
0x04	Move to Level (with On/Off)	M
0x05	Move (with On/Off)	M
0x06	Step (with On/Off)	M
0x07	Stop	M
0x08 – 0xff	Reserved	-

3.10.2.3.1 Move to Level Command

3.10.2.3.1.1 Payload Format

The Move to Level command payload shall be formatted as illustrated in Figure 3.31.

Octets	1	2
Data Type	Unsigned 8-bit integer	Unsigned 16-bit integer
Field Name	Level	Transition time

Figure 3.31 Format of the Move to Level Command Payload

3.10.2.3.1.2 Effect on Receipt

On receipt of this command, a device shall move from its current level to the value given in the Level field. The meaning of 'level' is device dependent – e.g. for a light it may mean brightness level.

The movement shall be as continuous as technically practical, i.e. not a step function, and the time taken to move to the new level shall 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.

If the Transition time field takes the value 0xffff then the time taken to move to the new level shall instead be determined by the *OnOffTransitionTime* attribute. If *OnOffTransitionTime* , which is an optional attribute, is not present, the device shall move to its new level as fast as it is able.

If the device is not able to move at a variable rate, the Transition time field may be disregarded.

3.10.2.3.2 Move Command

3.10.2.3.2.1 Payload Format

The Move command payload shall be formatted as illustrated in Figure 3.32.

Octets	1	1
Data Type	8-bit enumeration	Unsigned 8-bit integer
Field Name	Move mode	Rate

Figure 3.32 Format of the Move Command Payload

3.10.2.3.2.2 Move Mode Field

The Move mode field shall be one of the non-reserved values in Table 3.49.

Table 3.49 Values of the Move Mode Field

Fade Mode Value	Description
0x00	Up
0x01	Down
0x02 – 0xff	Reserved

3.10.2.3.2.3 Rate Field

The Rate field 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.

If the device is not able to move at a variable rate, this field may be disregarded.

3.10.2.3.2.4 Effect on Receipt

On receipt of this command, a device shall move from its current level in an up or down direction in a continuous fashion, as detailed in Table 3.50.

Table 3.50 Actions on Receipt for Move Command

Fade Mode	Action on Receipt
Up	Increase the device's level at the rate given in the Rate field. If the level reaches the maximum allowed for the device, stop.
Down	Decrease the device's level at the rate given in the Rate field. If the level reaches the minimum allowed for the device, stop.

3.10.2.3.3 Step Command

3.10.2.3.3.1 Payload Format

The Step command payload shall be formatted as illustrated in Figure 3.33.

Octets	1	1	2
Data Type	8-bit enumeration	Unsigned 8-bit integer	Unsigned 16-bit integer
Field Name	Step mode	Step size	Transition time

Figure 3.33 Format of the Step Command Payload

The Step mode field shall be one of the non-reserved values in Table 3.51.

Table 3.51 Values of the Step Mode Field

Fade Mode Value	Description
0x00	Up
0x01	Down
0x02 – 0xff	Reserved

The Transition time field specifies the time that shall be taken to perform the step, in tenths of a second. A step is a change in the *CurrentLevel* of 'Step size' units. The actual time taken should be as close to this as the device is able. If the Transition time field is 0xffff the device should move as fast as it is able.

If the device is not able to move at a variable rate, the Transition time field may be disregarded.

3.10.2.3.3.2 Effect on Receipt

On receipt of this command, a device shall move from its current level in an up or down direction as detailed in Table 3.52.

Table 3.52 Actions on Receipt for Step Command

Fade Mode	Action on Receipt
Up	Increase *CurrentLevel* by 'Step size' units, or until it reaches the maximum level allowed for the device if this reached in the process. In the latter case, the transition time shall be proportionally reduced.
Down	Decrease *CurrentLevel* by 'Step size' units, or until it reaches the minimum level allowed for the device if this reached in the process. In the latter case, the transition time shall be proportionally reduced.

3.10.2.3.4 Stop Command

This command has no payload. Upon receipt of this command, any Move to Level, Move or Step command (and their 'with On/Off' variants) currently in process shall be terminated. The value of *CurrentLevel* shall be left at its value upon receipt of the Stop command, and *RemainingTime* shall be set to zero.

This command has two entries in Table 3.1, one for the Move to Level, Move and Set commands, and one for their 'with On/Off' counterparts. This is solely for symmetry, to allow easy choice of one or other set of commands – the Stop commands are identical.

3.10.2.3.5 'With On/Off' Commands

The Move to Level (with On/Off), Move (with On/Off) and Step (with On/Off) commands have identical payloads to the Move to Level, Move and Step commands respectively. The also have the same effects, except for the following additions.

- Before commencing any command that has the effect of increasing *CurrentLevel*, the *OnOff* 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 *OnOff* attribute of the On/Off cluster on the same endpoint, if implemented, shall be set to Off.

3.10.2.4 Commands Generated

The server generates no commands.

3.10.2.5 Scene Table Extensions

If the Scenes server cluster (11) is implemented, the following extension field is added to the Scenes table:

CurrentLevel

3.10.2.6 Attribute Reporting

It is highly recommended that this cluster supports attribute reporting using the Report Attributes command and according to the minimum and maximum reporting interval and reportable change settings described in the ZCL Foundation Specification (see 2.4.7). If supported, the following attribute shall be reported:

CurrentLevel

3.10.3 Client

3.10.3.1 Dependencies

None.

3.10.3.2 Attributes

The client has no attributes.

3.10.3.3 Commands Received

No cluster specific commands are received by the client.

3.10.3.4 Commands Generated

The client generates the cluster specific commands received by the server (see 3.10.2.3), as required by the application.

3.11 Alarms Cluster

3.11.1 Overview

Attributes and commands for sending alarm notifications and configuring alarm functionality.

Alarm conditions and their respective alarm codes are described in individual clusters, along with an alarm mask field. Where not masked, alarm notifications are reported to subscribed targets using binding.

Where an alarm table is implemented, all alarms, masked or otherwise, are recorded and may be retrieved on demand.

Alarms may either reset automatically when the conditions that cause are no longer active, or may need to be explicitly reset.

3.11.2 Server

3.11.2.1 Dependencies

Any endpoint which implements time stamping shall also implement the Time server cluster.

3.11.2.2 Attributes

For convenience, the attributes defined in this specification are arranged into sets of related attributes; each set can contain up to 16 attributes. Attribute identifiers are encoded such that the most significant three nibbles specify the attribute set and the least significant nibble specifies the attribute within the set. The currently defined attribute sets are listed in Table 3.53.

Table 3.53 Alarms Cluster Attribute Sets

Attribute Set Identifier	Description
0x000	Alarm Information
0x001 – 0xfff	Reserved

3.11.2.2.1 Alarm Information Attribute Set

The Alarm Information attribute set contains the attributes summarized in Table 3.54.

Table 3.54 Attributes of the Alarm Information Attribute Set

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0000	*AlarmCount*	Unsigned 16-bit integer	0x00 – maximum defined in profile	Read only	0x00	O

3.11.2.2.1.1 *AlarmCount* Attribute

The *AlarmCount* attribute is 16 bits in length and specifies the number of entries currently in the alarm table. This attribute shall be specified in the range 0x00 to the maximum defined in the profile using this cluster.

If alarm logging is not implemented this attribute shall always take the value 0x00.

3.11.2.3 Alarm Table

The alarm table is used to store details of alarms generated within the devices. Alarms are requested by clusters which have alarm functionality, e.g. when attributes take on values that are outside ‘safe’ ranges.

The maximum number of entries in the table is device dependent.

When an alarm is generated, a corresponding entry is placed in the table. If the table is full, the entry with the earliest time stamp field is replaced by the new entry.

3.11.2.3.1 Alarm Table Format

The format of an alarm table entry is illustrated in Table 3.55.

Table 3.55 Format of the Alarm Table

Field	Type	Valid Range	Description
Alarm code	8-bit enumeration	0x00 – 0xff	Identifying code for the cause of the alarm, as given in the specification of the cluster whose attribute generated this alarm.
Cluster identifier	Cluster ID	0x0000 – 0xffff	The identifier of the cluster whose attribute generated this alarm.
Time stamp	Unsigned 32-bit integer	0x00000000 – 0xffffffff	The time at which the alarm occurred, or 0xffffffff if no time information is available. This time is taken from a Time server cluster, which must be present on the same endpoint.

3.11.2.4 Commands Received

The received command IDs for the Alarms cluster are listed in Table 3.56.

Table 3.56 Received Command IDs for the Alarms Cluster

Command Identifier Field Value	Description	Mandatory / Optional
0x00	Reset Alarm	M
0x01	Reset all alarms	M
0x02	Get Alarm	O
0x03	Reset alarm log	O
0x04 – 0xff	Reserved	

3.11.2.4.1 Reset Alarm Command

This command resets a specific alarm. This is needed for some alarms that do not reset automatically. If the alarm condition being reset was in fact still active then a new notification will be generated and, where implemented, a new record added to the alarm log.

3.11.2.4.1.1 Payload Format

The Reset Alarm command payload shall be formatted as illustrated in Figure 3.34.

Octets	1	2
Data Type	8-bit enumeration	Cluster ID
Field Name	Alarm code	Cluster identifier

Figure 3.34 Format of the Reset Alarm Command Payload

3.11.2.4.2 Reset All Alarms Command

This command resets all alarms. Any alarm conditions that were in fact still active will cause a new notification to be generated and, where implemented, a new record added to the alarm log.

3.11.2.4.3 Get Alarm Command

This command causes the alarm with the earliest timestamp in the alarm table to be reported in a get alarm response command 3.11.2.5.2. This command enables the reading of logged alarm conditions from the alarm table. Once an alarm condition has been reported the corresponding entry in the table is removed.

This command does not have a payload.

3.11.2.4.4 Reset Alarm Log Command

This command causes the alarm table to be cleared, and does not have a payload.

3.11.2.5 Commands Generated

The generated command IDs for the Alarms cluster are listed in Table 3.57.

Table 3.57 Generated Command IDs for the Alarms Cluster

Command Identifier Field Value	Description	Mandatory / Optional
0x00	Alarm	M
0x01	Get alarm response	O
0x02 – 0xff	Reserved	-

3.11.2.5.1 Alarm Command

The alarm command signals an alarm situation on the sending device.

An alarm command is generated when a cluster which has alarm functionality detects an alarm condition, e.g. an attribute has taken on a value that is outside a ‘safe’ range. The details are given by individual cluster specifications.

3.11.2.5.1.1 Payload Format

The alarm command payload shall be formatted as illustrated in Figure 3.35.

Octets	1	2
Data Type	8-bit enumeration	Cluster ID
Field Name	Alarm code	Cluster identifier

Figure 3.35 Format of the Alarm Command Payload

3.11.2.5.2 Get Alarm Response Command

The get alarm response command returns the results of a request to retrieve information from the alarm log, along with a time stamp indicating when the alarm situation was detected.

3.11.2.5.2.1 Payload Format

The get alarm response command payload shall be formatted as illustrated in Figure 3.36.

Octets	1	0/1	0/2	0/4
Data Type	8-bit enumeration	8-bit enumeration	Cluster ID	Unsigned 32-bit integer
Field Name	Status	Alarm code	Cluster identifier	Time stamp

Figure 3.36 Format of the Get Alarm Response Command Payload

If there is at least one alarm record in the alarm table then the status field is set to SUCCESS. The alarm code, cluster identifier and time stamp fields shall all be present and shall take their values from the item in the alarm table that they are reporting.

If there are no more alarms logged in the alarm table then the status field is set to NOT_FOUND and the alarm code, cluster identifier and time stamp fields shall be omitted.

3.11.3 Client

3.11.3.1 Dependencies

None

3.11.3.2 Attributes

The client has no attributes.

3.11.3.3 Commands Received

The client receives the cluster specific commands generated by the server (see 3.11.2.5).

3.11.3.4 Commands Generated

The client generates the cluster specific commands received by the server (see 3.11.2.4), as required by the application.

3.12 Time Cluster

3.12.1 Overview

This cluster provides a basic interface to a real-time clock. The clock time may be read and also written, in order to synchronize the clock (as close as practical) to a time standard. This time standard is the number of seconds since 0 hrs 0 mins 0 sec on 1st January 2000 UTC (Universal Coordinated Time).

The cluster also includes basic functionality for local time zone and daylight saving time.

3.12.2 Server

3.12.2.1 Dependencies

None

3.12.2.2 Attributes

The server supports the attributes shown in Table 3.58.

Table 3.58 Attributes of the On/Off Server Cluster

Identifier	Name	Type	Range	Access	Default	M/O
0x0000	*Time*	UTCTime	0x00000000 – 0xfffffffe	Read / Write	-	M
0x0001	*TimeStatus*	8-bit bitmap	0000 0xxx	Read / Write	0b00000000	M
0x0002	*TimeZone*	Signed 32-bit integer	-86400 - +86400	Read / Write	0x00000000	O
0x0003	*DstStart*	Unsigned 32-bit integer	0x00000000 – 0xfffffffe	Read / Write	-	O
0x0004	*DstEnd*	Unsigned 32-bit integer	0x00000000 – 0xfffffffe	Read / Write	-	O
0x0005	*DstShift*	Signed 32-bit integer	-86400 - +86400	Read / Write	0x00000000	O
0x0006	*StandardTime*	Unsigned 32-bit integer	0x00000000 – 0xfffffffe	Read only	-	O
0x0007	*LocalTime*	Unsigned 32-bit integer	0x00000000 – 0xfffffffe	Read only	-	O
0x0008	*LastSetTime*	UTCTime	0x00000000 – 0xfffffffe	Read only	0xffffffff	O
0x0009	*ValidUntilTime*	UTCTime	0x00000000 – 0xfffffffe	Read / Write	0xffffffff	O

3.12.2.2.1 *Time* Attribute

The *Time* attribute is 32 bits in length and holds the time value of a real time clock. This attribute has data type UTCTime, but note that it may not actually be synchronised to UTC - see discussion of the *TimeStatus* attribute below.

If the Master bit of the *TimeStatus* attribute has a value of 0, writing to this attribute shall set the real time clock to the written value, otherwise it cannot be written. The value 0xffffffff indicates an invalid time.

3.12.2.2.2 *TimeStatus* Attribute

The *TimeStatus* attribute holds a number of bit fields, as detailed in Table 3.59.

Table 3.59 Bit Values of the *TimeStatus* Attribute

Attribute Bit Number	Meaning	Values
0	Master	1 – master clock 0 – not master clock
1	Synchronized	1 – synchronized 0 – not synchronized
2	MasterZoneDst	1 – master for Time Zone and DST 0 – not master for Time Zone and DST
3[a]	Superseding	1 – time synchronization should be superseded 0 - time synchronization should not be superseded
other values	Reserved	-

a. CCB 1420

The Master and Synchronized bits together 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. This bit is not writeable – if a value is written to the *TimeStatus* attribute, this bit does not change.

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 (see 3.12.1). This bit must be explicitly written to indicate this – i.e. it is not set automatically on writing to the *Time* attribute. If the Master bit is 1, the value of this bit is 0.

If both the Master and Synchronized bits are 0, the real time clock has no defined relationship to the time standard (e.g. it may record the number of seconds since the device was initialized).

The MasterZoneDst bit specifies whether the *TimeZone*, *DstStart*, *DstEnd* and *DstShift* attributes are set internally to correct values for the location of the clock. If not, these attributes need to be set over the network. This bit is not writeable – if a value is written to the *TimeStatus* attribute, this bit does not change.

Devices shall synchronize to a Time server with the highest rank according to the following rules, listed in order of precedence:

1. A server with the Superseding bit set shall be chosen over a server without the bit set.
2. A server with the Master bit shall be chosen over a server without the bit set.
3. The server with the lower short address shall be chosen (note that this means a coordinator with the Superseding and Master bit set will always be chosen as the network time server).
4. A Time server with neither the Master nor Synchronized bits set should not be chosen as the network time server.[8]

3.12.2.2.3 *TimeZone* Attribute

The *TimeZone* attribute indicates the local time zone, as a signed offset in seconds from the *Time* attribute value. The value 0xffffffff indicates an invalid time zone.

The local Standard Time, i.e. the time adjusted for the time zone, but not adjusted for Daylight Saving Time (DST) is given by

Standard Time = *Time* + *TimeZone*

The range of this attribute is +/- one day. Note that the actual range of physical time zones on the globe is much smaller than this, so the manufacturer has the option to impose a smaller range.

If the MasterZoneDst bit of the *TimeStatus* attribute has a value of 1, this attribute cannot be written.

3.12.2.2.4 *DstStart* Attribute

The *DstStart* attribute indicates the DST start time in seconds. The value 0xffffffff indicates an invalid DST start time.

The Local Time, i.e. the time adjusted for both the time zone and DST, is given by

Local Time = Standard Time + *DstShift* (if *DstStart* <= *Time* <= *DstEnd*)

Local Time = Standard Time (if *Time* < *DstStart* or *Time* > *DstEnd*)

Note that the three attributes *DstStart*, *DstEnd* and *DstShift* are optional, but if any one of them is implemented the other two must also be implemented.

Note that this attribute should be set to a new value once every year.

If the MasterZoneDst bit of the *TimeStatus* attribute has a value of 1, this attribute cannot be written.

3.12.2.2.5 *DstEnd* Attribute

The *DstEnd* attribute indicates the DST end time in seconds. The value 0xffffffff indicates an invalid DST end time.

8. CCB 1420

Note that this attribute should be set to a new value once every year, and should be written synchronously with the *DstStart* attribute.

If the MasterZoneDst bit of the *TimeStatus* attribute has a value of 1, this attribute cannot be written.

3.12.2.2.6 *DstShift* Attribute

The *DstShift* attribute represents a signed offset in seconds from the standard time, to be applied between the times *DstStart* and *DstEnd* to calculate the Local Time (see 3.12.2.2.4). The value 0xffffffff indicates an invalid DST shift.

The range of this attribute is +/- one day. Note that the actual range of DST values employed by countries is much smaller than this, so the manufacturer has the option to impose a smaller range.

If the MasterZoneDst bit of the *TimeStatus* attribute has a value of 1, this attribute cannot be written.

3.12.2.2.7 *StandardTime* Attribute

The local Standard Time is given by the equation in 3.12.2.2.3. Another device on the network may calculate this time by reading the *Time* and *TimeZone* attributes and adding them together. If implemented however, the optional *StandardTime* attribute indicates this time directly. The value 0xffffffff indicates an invalid Standard Time.

3.12.2.2.8 *LocalTime* Attribute

The Local Time is given by the equation in 3.12.2.2.4. Another device on the network may calculate this time by reading the *Time*, *TimeZone*, *DstStart*, *DstEnd* and *DstShift* attributes and performing the calculation. If implemented however, the optional *LocalTime* attribute indicates this time directly. The value 0xffffffff indicates an invalid Local Time.

3.12.2.2.9 *LastSetTime* Attribute

The *LastSetTime* attribute indicates the most recent time that the *Time* attribute was set, either internally or over the ZigBee network (thus it holds a copy of the last value that *Time* was set to). This attribute is set automatically, so is read only. The value 0xffffffff indicates an invalid *LastSetTime*.

3.12.2.2.10 *ValidUntilTime* Attribute

The *ValidUntilTime* attribute indicates a time, later than *LastSetTime*, up to which the *Time* attribute may be trusted. ‘Trusted’ means that the difference between the *Time* attribute and the true UTC time is less than an acceptable error. The acceptable error is not defined by this cluster specification, but may be defined by the application profile in which devices that use this cluster are specified.

Note: The value that the *ValidUntilTime* attribute should be set to depends both on the acceptable error and the drift characteristics of the real time clock in the device that implements this cluster, which must therefore be known by the application entity that sets this value.

The value 0xffffffff indicates an invalid *ValidUntilTime*.

3.12.2.3 Commands Received

The server receives no commands except those to read and write attributes.

3.12.2.4 Commands Generated

The server generates no cluster specific commands.

3.12.3 Client

3.12.3.1 Dependencies

None.

3.12.3.2 Attributes

The client has no attributes.

3.12.3.3 Commands Received

No cluster specific commands are received by the client.

3.12.3.4 Commands Generated

The client generates no cluster specific commands.

3.13 RSSI Location Cluster

3.13.1 Overview

This cluster provides a means for

- Exchanging Received Signal Strength Indication (RSSI) based location information and channel parameters among devices
- Optionally, reporting RSSI data to a centralized device that collects RSSI data from devices in the network and calculates their positions from the set of collected data.

An example of the usage of this cluster with a centralized device is shown in Figure 3.4.

3.13.2 Server

3.13.2.1 Dependencies

None

3.13.2.2 Attributes

For convenience, the attributes defined in this specification are arranged into sets of related attributes; each set can contain up to 16 attributes. Attribute identifiers are encoded such that the most significant three nibbles specify the attribute set and the least significant nibble specifies the attribute within the set. The currently defined attribute sets are listed in Table 3.60.

Table 3.60 Location Attribute Sets

Attribute Set Identifier	Description
0x000	Location Information
0x001	Location Settings
0x002 – 0xfff	Reserved

3.13.2.2.1 Location Information Attribute Set

The Location Information attribute set contains the attributes summarized in Table 3.61.

Table 3.61 Attributes of the Location Information Attribute Set

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0000	*LocationType*	8-bit Data	0000xxxx	Read / Write	-	M
0x0001	*LocationMethod*	8-bit enumeration	0x00 – 0xff	Read / Write	-	M

Table 3.61 Attributes of the Location Information Attribute Set (Continued)

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0002	*LocationAge*	Unsigned 16-bit integer	0x0000 – 0xffff	Read only	-	O
0x0003	*QualityMeasure*	Unsigned 8-bit integer	0x00 – 0x64	Read only	-	O
0x0004	*NumberOfDevices*	Unsigned 8-bit integer	0x00 – 0xff	Read only	-	O

3.13.2.2.1.1 *LocationType* Attribute

The *LocationType* attribute is 8 bits long and is divided into bit fields. The meanings of the individual bit fields are detailed in Table 3.62.

Table 3.62 Bit Values of the *LocationType* Attribute

Bit Field (Bit Numbers)	Meaning	Values
0	Absolute	1 – Absolute location 0 – Measured location
1	2-D	1 – Two dimensional 0 – Three dimensional
2-3	Coordinate System	0 – Rectangular (installation-specific origin and orientation) 1-3 – Reserved
4-7	Reserved	-

The Absolute bit field indicates whether the location is a known absolute location or is calculated.

The 2-D bit field indicates whether the location information is two- or three-dimensional. If the location information is two-dimensional, Coordinate 3 is unknown and shall be set to 0x8000.

The Coordinate System bit field indicates the geometry of the system used to express the location coordinates. If the field is set to zero, the location coordinates are expressed using the rectangular coordinate system. All other values are reserved.

3.13.2.2.1.2 *LocationMethod* Attribute

The *LocationMethod* attribute shall be set to one of the non-reserved values in Table 3.63.

Table 3.63 Values of the *LocationMethod* Attribute

Value	Method	Description
0x00	Lateration	A method based on RSSI measurements from three or more sources.
0x01	Signposting	The location reported is the location of the neighboring device with the strongest received signal.
0x02	RF fingerprinting	RSSI signatures are collected into a database at commissioning time. The location reported is the location taken from the RSSI signature database that most closely matches the device's own RSSI signature.
0x03	Out of band	The location is obtained by accessing an out-of-band device (that is, the device providing the location is not part of the ZigBee network).
0x04	Centralized	The location is performed by a centralized device that collects RSSI measurements from devices on the network and calculates their positions from the set of collected data.
0x05 – 0x3f	-	Reserved
0x40 – 0xff	-	Reserved for manufacturer specific location methods.

3.13.2.2.1.3 *LocationAge* Attribute

The *LocationAge* attribute indicates the amount of time, measured in seconds, that has transpired since the location information was last calculated. This attribute is not valid if the Absolute bit of the *LocationType* attribute is set to one.

3.13.2.2.1.4 *QualityMeasure* Attribute

The *QualityMeasure* attribute is a measure of confidence in the corresponding location information. The higher the value, the more confident the transmitting device is in the location information. A value of 0x64 indicates complete (100%) confidence and a value of 0x00 indicates zero confidence. (Note: no fixed confidence metric is mandated – the metric may be application and manufacturer dependent.)

This field is not valid if the Absolute bit of the *LocationType* attribute is set to one.

3.13.2.2.1.5 *NumberOfDevices* Attribute

The *NumberOfDevices* attribute is the number of devices whose location data were used to calculate the last location value. This attribute is related to the *QualityMeasure* attribute.

3.13.2.2.2 Location Settings Attribute Set

The Location Settings attribute set contains the attributes summarized in Table 3.64.

Table 3.64 Attributes of the Location Settings Attribute Set

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0010	*Coordinate1*	Signed 16-bit integer	0x8000 – 0x7fff	Read / Write	-	M
0x0011	*Coordinate2*	Signed 16-bit integer	0x8000 – 0x7fff	Read / Write	-	M
0x0012	*Coordinate3*	Signed 16-bit integer	0x8000 – 0x7fff	Read / Write	-	O
0x0013	*Power*	Signed 16-bit integer	0x8000 – 0x7fff	Read / Write	-	M
0x0014	*PathLossExponent*	Unsigned 16-bit integer	0x0000 – 0xffff	Read / Write	-	M
0x0015	*ReportingPeriod*	Unsigned 16-bit integer	0x0000 – 0xffff	Read / Write	-	O
0x0016	*CalculationPeriod*	Unsigned 16-bit integer	0x0000 – 0xffff	Read / Write	-	O
0x0017	*NumberRSSIMeasurements*	Unsigned 8-bit integer	0x01 – 0xff	Read / Write	-	M

3.13.2.2.2.1 *Coordinate* 1,2,3 Attributes

The *Coordinate1, Coordinate2 and Coordinate3* attributes are signed 16-bit integers, and represent orthogonal linear coordinates x, y, z in meters as follows.

x = *Coordinate1* / 10, y = *Coordinate2* / 10, z = *Coordinate3* / 10

The range of x is -3276.7 to 3276.7 meters, corresponding to *Coordinate1* between 0x8001 and 0x7fff. The same range applies to y and z. A value of 0x8000 for any of the coordinates indicates that the coordinate is unknown.

3.13.2.2.2.2 *Power* Attribute

The *Power* attribute specifies the value of the average power P_0, measured in dBm, received at a reference distance of one meter from the transmitter.

P_0 = *Power* / 100

A value of 0x8000 indicates that *Power* is unknown.

3.13.2.2.2.3 *PathLossExponent* Attribute

The *PathLossExponent* attribute specifies the value of the Path Loss Exponent n, an exponent that describes the rate at which the signal power decays with increasing distance from the transmitter.

n = *PathLossExponent* / 100

A value of 0xffff indicates that *PathLossExponent* is unknown.

The signal strength in dBm at a distance d meters from the transmitter is given by

$P = P_0 – 10n \times \log_{10}(d)$

where

P is the power in dBm at the receiving device.

P_0 is the average power in dBm received at a reference distance of 1meter from the transmitter.

n is the path loss exponent.

d is the distance in meters between the transmitting device and the receiving device.

3.13.2.2.2.4 *ReportingPeriod* Attribute

The *ReportingPeriod* attribute specifies the time in seconds between successive reports of the device's location by means of the Location Data Notification command. The minimum value this attribute can take is specified by the profile in use. If *ReportingPeriod* is zero, the device does not automatically report its location. Note that location information can always be polled at any time.

3.13.2.2.2.5 *CalculationPeriod* Attribute

The *CalculationPeriod* attribute specifies the time in seconds between successive calculations of the device's location. If *CalculationPeriod* is less than the

physically possible minimum period that the calculation can be performed, the calculation will be repeated as frequently as possible.

In the case of centralized location (*LocationMethod* attribute equal to Centralized) the *CalculationPeriod* attribute specifies the period between successive RSSI Ping commands.

3.13.2.2.2.6 *NumberRSSIMeasurements* Attribute

The *NumberRSSIMeasurements* attribute specifies the number of RSSI measurements to be used to generate one location estimate. The measurements are averaged to improve accuracy. *NumberRSSIMeasurements* must be greater than or equal to 1.

In case of centralized location (*LocationMethod* attribute equal to Centralized) the *NumberRSSIMeasurements* attribute specifies the number of successive RSSI Ping commands to be sent.

3.13.2.3 Commands Received

The received command IDs for the Location cluster are listed in Table 3.65

Table 3.65 Received Command IDs for the Location Cluster

Command Identifier Field Value	Description	Mandatory / Optional
0x00	Set Absolute Location	M
0x01	Set Device Configuration	M
0x02	Get Device Configuration	M
0x03	Get Location Data	M
0x04	RSSI Response	O
0x05	Send Pings	O
0x06	Anchor Node Announce	O
0x07 – 0xff	Reserved	-

3.13.2.3.1 Set Absolute Location Command

This command is used to set a device's absolute (known, not calculated) location and the channel parameters corresponding to that location.

3.13.2.3.1.1 Payload Format

The Set Absolute Location command payload shall be formatted as illustrated in Figure 3.37.

Octets	2	2	2	2	2
Data Type	Signed Integer	Signed Integer	Signed Integer	Signed Integer	Unsigned Integer
Field Name	Coordinate 1	Coordinate 2	Coordinate 3	Power	Path Loss Exponent

Figure 3.37 Format of the Set Absolute Location Command Payload

The fields of the payload correspond directly to the attributes with the same names. For details of their meaning and ranges see the descriptions of the individual attributes.

The three coordinate fields shall contain the absolute location (known, not calculated) of the destination device. If any coordinate field(s) is not known, the value(s) shall be set to 0x8000.

3.13.2.3.1.2 Effect on Receipt

On receipt of this command, the device shall update the attributes corresponding to (i.e. with the same names as) the payload fields.

3.13.2.3.2 Set Device Configuration Command

This command is used to set a device's location parameters, which will be used for calculating and reporting measured location. This command is invalid unless the Absolute bit of the *LocationType* attribute has a value of 0.

3.13.2.3.2.1 Payload Format

The Set Device Configuration command payload shall be formatted as illustrated in Figure 3.38.

Octets	2	2	2	1	2
Data Type	Signed Integer	Unsigned Integer	Unsigned Integer	Unsigned Integer	Unsigned Integer
Field Name	Power	Path Loss Exponent	Calculation Period	Number RSSI Measurements	Reporting Period

Figure 3.38 Format of the Set Device Configuration Payload

The fields of the payload correspond directly to the attributes with the same names. For details of their meaning and ranges see the descriptions of the individual attributes.

3.13.2.3.2.2 Effect on Receipt

On receipt of this command, the device shall update the attributes corresponding to (i,.e. with the same names as) the payload fields.

3.13.2.3.3 Get Device Configuration Command

This command is used to request the location parameters of a device. The location parameters are used for calculating and reporting measured location.

3.13.2.3.3.1 Payload Format

The Get Device Configuration command payload shall be formatted as illustrated in Figure 3.39.

Octets	8
Data Type	IEEE Address
Field Name	Target Address

Figure 3.39 Format of the Get Device Configuration Payload

The Target Address field contains the 64-bit IEEE address of the device for which the location parameters are being requested. This field may contain the address of the sending device, the address of the receiving device or the address of a third device.

Note: one reason a device may request its own configuration is that there may be a designated device which holds the configurations of other devices for distribution at commissioning time. It is also possible that the device may lose its configuration settings for some other reason (loss of power, reset). In the case of a third device, that device may sleep a lot and not be easily accessible.

3.13.2.3.3.2 Effect on Receipt

On receipt of this command, the device shall generate a Device Configuration Response command (3.13.2.4.1).

3.13.2.3.4 Get Location Data Command

This command is used to request a device’s location information and channel parameters. It may be sent as a unicast, multicast or broadcast frame. When sent as a broadcast frame, care should be taken to minimize the risk of a broadcast 'storm' - in particular, it is recommended that the broadcast radius is set to 1.

(**Note:** devices may or may not acquire and store information on other devices' locations such that this information may be requested by another device. This is application dependent.)

3.13.2.3.4.1 Payload Format

The Get Location Data command payload shall be formatted as illustrated in Figure 3.40.

Bits	3	1	1	1	1	1	8	0 / 64
Data Type	8-bit bitmap						Unsigned integer	IEEE address
Field Name	Reserved	Compact Response	Broadcast Response	Broadcast Indicator	Re-calculate	Absolute Only	Number Responses	Target Address

Figure 3.40 Format of the Get Location Data Payload

The highest 3 bits of the first octet are reserved and shall be set to zero.

The Absolute Only field (bit 0 of the first octet) specifies the type of location information being requested. If the Absolute Only field is set to one, the device is only requesting absolute location information (a device may want to gather absolute node locations for use in its own location calculations, and may not be interested in neighbors with calculated values). Otherwise, if the field is set to zero, the device is requesting all location information (absolute and calculated).

The Recalculate field (bit 1 of the first octet) indicates whether the device is requesting that a new location calculation be performed. If the field is set to zero, the device is requesting the currently stored location information. Otherwise, if the field is set to one, the device is requesting that a new calculation be performed. This field is only valid if the Absolute Only field is set to zero.

The Broadcast Indicator field (bit 2 of the first octet) indicates whether the command is being sent as a unicast, multicast or broadcast frame. If the field is set to one, the command is sent as a broadcast or multicast, else it is sent as a unicast.

The Broadcast Response field (bit 3 of the first octet) indicates whether subsequent responses after the first (where the Number Responses field is greater than one) shall be unicast or broadcast. Broadcast responses can be used as a 'location beacon'.

The Compact Response field (bit 4 of the first octet) indicates whether subsequent responses after the first (where the Number Responses field is greater than one) shall be sent using the Location Data Notification or the Compact Location Data Notification command.

The Number Responses field indicates the number of location responses to be returned. The information to be returned is evaluated this number of times, with a period equal to the value of the *ReportingPeriod* attribute, and a separate response is sent for each evaluation. This field shall have a minimum value of one. Values greater than one are typically used for situations where locations are changing.

The Target Address field contains the 64-bit IEEE address of the device for which the location information and channel parameters are being requested. If the Broadcast Indicator field is set to zero (i.e. the command is sent as a unicast) this field may contain the address of the receiving device, the address of the sending device or the address of any other device. If the Broadcast Indicator field is set to one (i.e. the command is sent as a broadcast or multicast) the target address is implicitly that of the receiving device, so this field shall be omitted.

3.13.2.3.4.2 Effect on Receipt

On receipt of this command, if the Location Type field is set to zero, only a receiving device(s) that knows its absolute location shall respond by generating a Location Data Response command. If the Location Type field is set to one, all devices receiving this command shall respond by generating a Location Data Response command.

If the command is sent as a unicast, information for the device specified in the Target Address field shall be returned, if the receiving device has or can obtain the information for that device. If the information is not available, the Status field of the Location Data Response command shall be set to NOT_FOUND.

If the command is sent as a broadcast or multicast, receiving devices shall send back their own information (there is no IEEE target address in this case).

If the Number Responses field is greater than one, the subsequent location readings/calculations shall be sent using the Location Data Notification or the Compact Location Data Notification command, depending on the value of the Reduced Response field.

3.13.2.3.5 RSSI Response Command

This command is sent by a device in response to an RSSI Request command.

3.13.2.3.5.1 Payload Format

The command payload shall be formatted as illustrated in Figure 3.41.

Octets	8	2	2	2	1	1
Data Type	IEEE Address	Signed 16-bit integer	Signed 16-bit integer	Signed 16-bit integer	Signed 8-bit integer	Unsigned 8-bit integer
Field Name	Replying Device	Coordinate1	Coordinate2	Coordinate3	RSSI	Number RSSI Measurements

Figure 3.41 Format of the RSSI Response Command Payload

The fields of the payload have the following meanings:

Replying Device: The IEEE address of the node that is has sent this command (in reply to an RSSI request command).

Coordinate1, Coordinate1, Coordinate3: The coordinates of the replying node.

RSSI: The mean RSSI value registered by the replying node, referring to the radio link between the node that sent the RSSI request command and the replying node, expressed in dBm.

Number RSSI Measurements: The number of packets measured in order to calculate the RSSI value (1 indicates that ‘mean value’ is not supported).

3.13.2.3.6 Send Pings Command

This command instructs the receiving node to start sending multiple RSSI Ping commands so that all its one hop neighbors can calculate the mean RSSI value of the radio link.

3.13.2.3.6.1 Payload Format

The command payload shall be formatted as illustrated in Figure 3.42.

Octets	8	1	2
Data Type	IEEE Address	Unsigned 8-bit integer	Unsigned 16-bit integer
Field Name	Target Address	Number RSSI Measurements	Calculation Period

Figure 3.42 Format of the Send Pings Command Payload

The Target Address field contains the IEEE address of the intended target node. This is included because there can be cases when the sender does not definitely know the short address of the intended target (see below for effect on receipt). The other fields of the payload correspond directly to the attributes with the same names. For details of their meaning and ranges see the descriptions of the individual attributes.

3.13.2.3.6.2 Effect on Receipt

On receipt of this command, the device shall compare the Target Address field with its actual IEEE address. If the addresses match, it shall update the attributes corresponding to (i.e. with the same names as) the payload fields and generate a number of RSSI Ping commands equal to *NumberRSSIMeasurements*, waiting for *CalculationPeriod* time between successive RSSI Ping commands.

If the addresses do not match, then if the command was broadcast, it shall ignore it, else if the command was unicast it shall return a Default Response command (2.4.12) with status code FAILURE.

3.13.2.3.7 Anchor Node Announce Command

This command is sent by an anchor node (a node that has been commissioned with a set of absolute coordinates) when it joins the network, to announce itself. The command is used when *LocationMethod* = Centralized, to let the central device knows the exact position of that anchor node. This message should be either unicast to the central node or broadcast in the case of unknown destination address.

3.13.2.3.7.1 Payload Format

The command payload shall be formatted as illustrated in Figure 3.43.

Octets	8	2	2	2
Data Type	IEEE Address	Signed 16-bit integer	Signed 16-bit integer	Signed 16-bit integer
Field Name	Anchor Node Address	Coordinate1	Coordinate2	Coordinate3

Figure 3.43 Format of the Anchor Node Announce Command Payload

The Anchor Node Address field contains the IEEE address of the anchor node. The other fields of the payload correspond directly to the attributes with the same names. For details of their meaning and ranges see the descriptions of the individual attributes. If any coordinate is unknown, it should be set to 0x8000.

3.13.2.4 Commands Generated

Table 3.66 Generated Command IDs for the RSSI Location Cluster

Command Identifier Field Value	Description	Mandatory / Optional
0x00	Device configuration response	M
0x01	Location data response	M
0x02	Location data notification	M
0x03	Compact location data notification	M
0x04	RSSI Ping	M
0x05	RSSI Request	O
0x06	Report RSSI Measurements	O
0x07	Request Own Location	O
0x08 – 0xff	Reserved	-

3.13.2.4.1 Device Configuration Response Command

This command is sent by a device in response to a Get Device Configuration command (3.13.2.3.3).

3.13.2.4.1.1 Payload Format

The Device Configuration Response command payload shall be formatted as illustrated in Figure 3.44. All payload fields are relevant to the device for which the location parameters have been requested.

Octets	1	0 / 2	0 / 2	0 / 2	0 / 1	0 / 2
Data Type	Enumeration	Signed Integer	Unsigned Integer	Unsigned Integer	Unsigned Integer	Unsigned Integer
Field Name	Status	Power	Path Loss Exponent	Calculation Period	Number RSSI Measurements	Reporting Period

Figure 3.44 Format of the Device Configuration Response Payload

The fields of the payload (other than Status) correspond directly to the attributes with the same names. For details of their meaning and ranges see the descriptions of the individual attributes.

The Status field indicates whether the response to the request was successful or not. If the field is set to SUCCESS, the response was successful. If the field is set

to NOT_FOUND, the receiving device was unable to provide the location parameters of the device for which the location parameters were requested. If the field is set to NOT_FOUND, all other payload fields shall not be sent. See 2.5.3 for status codes.

3.13.2.4.2 Location Data Response Command

This command is sent by a device in response to a request for location information and channel parameters.

3.13.2.4.2.1 Payload Format

The Location Data Response command payload shall be formatted as illustrated in Figure 3.45. All payload fields are relevant to the device for which the location parameters have been requested.

Octets	1	0 / 1	0 / 2	0 / 2	0 / 2	0 / 2	0 / 2	0 / 1	0 / 1	0 / 2
Data Type	Enumeration	Data	Signed Integer	Signed Integer	Signed Integer	Signed Integer	Unsigned Integer	Enumeration	Unsigned Integer	Unsigned Integer
Field Name	Status	Location Type	Coordinate 1	Coordinate 2	Coordinate 3	Power	Path Loss Exponent	Location Method	Quality Measure	Location Age

Figure 3.45 Format of the Location Data Response Payload

The fields of the payload correspond directly to the attributes with the same names. For details of their meaning and ranges see the descriptions of the individual attributes.

If the Absolute bit of the Location Type field is set to 1, the Location Method, Quality Measure and Location Age fields are not applicable and shall not be sent.

If the 2-D bit of the Location Type field is set to 1, the Coordinate 3 field shall not be sent.

The Status field indicates whether the response to the request was successful or not. If the field is set to SUCCESS, the response was successful. If the field is set to NOT_FOUND, the receiving device was unable to provide the location parameters of the device for which the location parameters were requested. If the field is set to NOT_FOUND, all other payload fields shall not be sent. See 2.5.3 for status codes.

3.13.2.4.3 Location Data Notification Command

This command is sent periodically by a device to announce its location information and channel parameters. The period is equal to the value of the *ReportingPeriod* attribute.

The location data notification command may be sent as a unicast or as a broadcast frame. When sent as a broadcast frame, it is recommended that the broadcast radius is set to 1.

3.13.2.4.3.1 Payload Format

The Location Data Notification command payload shall be formatted as illustrated in Figure 3.46.

Octets	**1**	**2**	**2**	**0 / 2**	**2**	**2**	**0 / 1**	**0 / 1**	**0 / 2**
Data Type	Data	Signed Integer	Signed Integer	Signed Integer	Signed Integer	Unsigned Integer	Enumeration	Unsigned Integer	Unsigned Integer
Field Name	Location Type	Coordinate 1	Coordinate 2	Coordinate 3	Power	Path Loss Exponent	Location Method	Quality Measure	Location Age

Figure 3.46 Format of the Location Data Notification Payload

The fields of the payload correspond directly to the attributes with the same names. For details of their meaning and ranges see the descriptions of the individual attributes.

If the 2-D bit of the Location Type field is set to 1, the Coordinate 3 field shall not be sent.

If the Absolute bit of the Location Type field is set to 1, the Location Method, Quality Measure and Location Age fields are not applicable and shall not be sent.

3.13.2.4.4 Compact Location Data Notification Command

This command is identical in format and use to the Location Data Notification command, except that the Power, Path Loss Exponent and Location Method fields are not included.

3.13.2.4.5 RSSI Ping Command

This command is sent periodically by a device to enable listening devices to measure the received signal strength in the absence of other transmissions from that device. The period is given by the *ReportingPeriod* attribute.

The RSSI Ping command may be sent as a unicast or as a broadcast frame. When sent as a broadcast frame, it is recommended that the broadcast radius is set to 1.

3.13.2.4.5.1 Payload Format

The RSSI Ping command payload shall be formatted as illustrated in Figure 3.47.

Octets	1
Data Type	8-bit Data
Field Name	Location Type

Figure 3.47 Format of the RSSI Ping Command Payload

The Location Type field holds the value of the *LocationType* attribute.

3.13.2.4.6 RSSI Request Command

A device uses this command to ask one, more or all its one hop neighbors for the mean RSSI value they hear from itself. The command may be broadcast (typical usage is broadcast with radius equal to one).

3.13.2.4.6.1 Payload Format

The payload is empty.

3.13.2.4.6.2 Effect on Receipt

On receipt of this command, the device shall respond by generating an RSSI Response command back to the sender of this request.

3.13.2.4.7 Report RSSI Measurements Command

This command is sent by a device to report its measurements of the link between itself and one or more neighbors. In a centralized location scenario, the device that sends this command is the device that needs to be localized.

3.13.2.4.7.1 Payload Format

The command payload shall be formatted as illustrated in Figure 3.48.

Octets	8	1	NumberOfNeighbors x 16
Data Type	IEEE Address	Unsigned 8-bit integer	(see Figure 3.49)
Field Name	Reporting Address	NumberOfNeighbors	Neighbors Information

Figure 3.48 Format of the Report RSSI Measurements Command Payload

The Neighbors Information field consists of NumberOfNeighbors sets of sub-fields. Each such set shall be formatted as illustrated in Figure 3.49.

Octets	8	2	2	2	1	1
Data Type	IEEE Address	Signed 16-bit integer	Signed 16-bit integer	Signed 16-bit integer	Signed 8-bit integer	Unsigned 8-bit integer
Field Name	Neighbor Address	Coordinate1	Coordinate2	Coordinate3	RSSI	Number RSSI Measurements

Figure 3.49 Format of Each Set of Subfields of the Neighbors Information Field

The format of each set of subfields is identical to that of the RSSI Response command payload (Figure 3.41), and relays the information sent to the reporting device from one of its neighbors. The subfields have the meaning given in 3.13.2.3.5.1.

3.13.2.4.8 Request Own Location Command

In the *LocationMethod* = Centralized scenario, this command is sent to the 'centralized device' by a node wishing to know its own location.

3.13.2.4.8.1 Payload Format

The Request Own Location command payload shall be formatted as illustrated in Figure 3.50.

Octets	8
Data Type	IEEE Address
Field Name	Requesting Address

Figure 3.50 Format of the Request Own Location Command Payload

The Requesting Address field contains the IEEE address of the node that sent the command.

3.13.2.4.8.2 Effect On Receipt

The node receiving the Request Own Location command shall reply as follows. If it has knowledge of the location of the requesting entity, it shall reply with a Set Absolute Location command, telling the requesting entity its location. If it does not have knowledge of the location of the requesting entity, it shall reply with a Default Response command (2.4.12) with status code FAILURE.

3.13.3 Client

3.13.3.1 Dependencies

None

3.13.3.2 Attributes

None

3.13.3.3 Commands Received

The client receives the cluster specific commands generated by the server (see 3.13.2.4).

3.13.3.4 Commands Generated

The client generates the cluster specific commands received by the server (see 3.13.2.3), as required by the application.

3.14 Input, Output and Value Clusters

This section specifies a number of clusters which are based on ‘Basic’ properties of the Input, Output and Value objects specified by BACnet (see [B8]). See also section 9.4.

The clusters specified herein are for use typically in ZigBee Commercial Building applications, but may be used in any application domain.

For these clusters, the Access field for each attribute specification may contain one of the following symbols:

R Readable, but not writeable

R/W Readable and writable

R*W Readable and Optionally Writable The ability to write to this attribute is not mandatory but is determined by the vendor supplying the product. If not writable, a READ_ONLY error is returned for any write attempt.

3.14.1 Analog Input (Basic) Cluster

The Analog Input (Basic) cluster provides an interface for reading the value of an analog measurement and accessing various characteristics of that measurement. The cluster is typically used to implement a sensor that measures an analog physical quantity.

3.14.1.1 Server

3.14.1.1.1 Dependencies

None.

3.14.1.1.2 Attributes

The attributes of this cluster are detailed in Table 3.67.

Table 3.67 Attributes of the Analog Input (Basic) Server Cluster

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x001C	*Description*	Character string	-	R*W	Null string	O
0x0041	*MaxPresentValue*	Single precision	-	R*W	-	O
0x0045	*MinPresentValue*	Single precision	-	R*W	-	O
0x0051	*OutOfService*	Boolean	False (0) or True (1)	R*W	False (0)	M
0x0055	*PresentValue*	Single precision	-	R/W	-	M
0x0067	*Reliability*	8-bit enumeration	-	R*W	0x00	O
0x006A	*Resolution*	Single precision	-	R*W	-	O
0x006F	*StatusFlags*	8-bit bitmap	0x00 - 0x0f	R	0	M
0x0075	*EngineeringUnits*	16-bit enumeration	See section 3.14.10.10	R*W	-	O
0x0100	*ApplicationType*	Unsigned 32-bit integer	0 - 0xffffffff	R	-	O
All others < 0x0400	Reserved					
0x0400 - 0xFFFF	Reserved for vendor specific attributes					

For an explanation of the attributes, see section 3.14.10.

3.14.1.1.2.1 Commands

No cluster specific commands are received or generated.

3.14.1.1.2.2 Attribute Reporting

This cluster shall support attribute reporting using the Report Attributes and Configure Reporting commands, according to the minimum and maximum reporting interval, value and timeout settings.

The following attributes shall be reported: *StatusFlags, PresentValue*

3.14.1.1.3 Client

The client has no dependencies, no attributes, and receives or generates no cluster specific commands.

3.14.2 Analog Output (Basic) Cluster

The Analog Output (Basic) cluster provides an interface for setting the value of an analog output (typically to the environment) and accessing various characteristics of that value.

3.14.2.1 Server

3.14.2.1.1 Dependencies

None.

3.14.2.1.2 Attributes

The attributes of this cluster are detailed in Table 3.68.

Table 3.68 Attributes of the Analog Output (Basic) Server Cluster

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x001C	*Description*	Character string	-	R*W	Null string	O
0x0041	*MaxPresentValue*	Single precision	-	R*W	-	O
0x0045	*MinPresentValue*	Single precision	-	R*W	-	O
0x0051	*OutOfService*	Boolean	False (0) or True (1)	R*W	False (0)	M
0x0055	*PresentValue*	Single precision	-	R/W	-	M

Table 3.68 Attributes of the Analog Output (Basic) Server Cluster (Continued)

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0057	*PriorityArray*	Array of 16 structures of (boolean, single precision)	-	R/W	16 x (0, 0.0)	O
0x0067	*Reliability*	8-bit enumeration	-	R*W	0x00	O
0x0068	*RelinquishDefault*	Single precision	-	R*W	-	O
0x006A	*Resolution*	Single precision	-	R*W	-	O
0x006F	*StatusFlags*	8-bit bitmap	0x00 - 0x0f	R	0	M
0x0075	*EngineeringUnits*	16-bit enumeration	See section 3.14.10.10	R*W	-	O
0x0100	*ApplicationType*	Unsigned 32-bit integer	0 - 0xffffffff	R	-	O
All others < 0x0400	Reserved					
0x0400 - 0xFFFF	Reserved for vendor specific attributes					

For an explanation of the attributes, see section 3.14.10.

3.14.2.1.3 Commands

No cluster specific commands are received or generated.

3.14.2.1.4 Attribute Reporting

This cluster shall support attribute reporting using the Report Attributes and Configure Reporting commands, according to the minimum and maximum reporting interval, value and timeout settings.

The following attributes shall be reported: *StatusFlags*, *PresentValue*

3.14.2.2 Client

The client has no dependencies, no attributes, and receives or generates no cluster specific commands.

3.14.3 Analog Value (Basic) Cluster

The Analog Value (Basic) cluster provides an interface for setting an analog value, typically used as a control system parameter, and accessing various characteristics of that value.

3.14.3.1 Server

3.14.3.1.1 Dependencies

None.

3.14.3.1.2 Attributes

The attributes of this cluster are detailed in Table 3.69.

Table 3.69 Attributes of the Analog Value (Basic) Server Cluster

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x001C	*Description*	Character string	-	R*W	Null string	O
0x0051	*OutOfService*	Boolean	False (0) or True (1)	R*W	False (0)	M
0x0055	*PresentValue*	Single precision	-	R/W	-	M
0x0057	*PriorityArray*	Array of 16 structures of (boolean, single precision)	-	R/W	16 x (0, 0.0)	O
0x0067	*Reliability*	8-bit enumeration	-	R*W	0x00	O
0x0068	*RelinquishDefault*	Single precision	-	R*W	-	O
0x006F	*StatusFlags*	8-bit bitmap	0x00 - 0x0f	R	0	M
0x0075	*EngineeringUnits*	16-bit enumeration	See section 3.14.10.10	R*W	-	O

Table 3.69 Attributes of the Analog Value (Basic) Server Cluster (Continued)

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0100	*ApplicationType*	Unsigned 32-bit integer	0 - 0xffffffff	R	-	O
All others < 0x0400	*Reserved*					
0x0400 - 0xFFFF	*Reserved for vendor specific attributes*					

For an explanation of the attributes, see section 3.14.10.

3.14.3.1.3 Commands

No cluster specific commands are received or generated.

3.14.3.1.4 Attribute Reporting

This cluster shall support attribute reporting using the Report Attributes and Configure Reporting commands, according to the minimum and maximum reporting interval, value and timeout settings.

The following attributes shall be reported: *StatusFlags, PresentValue*

3.14.3.2 Client

The client has no dependencies, no attributes, and receives or generates no cluster specific commands.

3.14.4 Binary Input (Basic) Cluster

The Binary Input (Basic) cluster provides an interface for reading the value of a binary measurement and accessing various characteristics of that measurement. The cluster is typically used to implement a sensor that measures a two-state physical quantity.

3.14.4.1 Server

3.14.4.1.1 Dependencies

None.

3.14.4.1.2 Attributes

The attributes of this cluster are detailed in Table 3.70.

Table 3.70 Attributes of the Binary Input (Basic) Server Cluster

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0004	*ActiveText*	Character string	-	R*W	Null string	O
0x001C	*Description*	Character string	-	R*W	Null string	O
0x002E	*InactiveText*	Character string	-	R*W	Null string	O
0x0051	*OutOfService*	Boolean	False (0) or True (1)	R*W	False (0)	M
0x0054	*Polarity*	8-bit enumeration	-	R	0	O
0x0055	*PresentValue*	Boolean	-	R*W	-	M
0x0067	*Reliability*	8-bit enumeration	-	R*W	0x00	O
0x006F	*StatusFlags*	8-bit bitmap	0x00 - 0x0f	R	0	M
0x0100	*ApplicationType*	Unsigned 32-bit integer	0 - 0xffffffff	R	-	O
All others < 0x0400	Reserved					
0x0400 - 0xFFFF	Reserved for vendor specific attributes					

For an explanation of the attributes, see section 3.14.10.

3.14.4.1.3 Commands

No cluster specific commands are received or generated.

3.14.4.1.4 Attribute Reporting

This cluster shall support attribute reporting using the Report Attributes and Configure Reporting commands, according to the minimum and maximum reporting interval, value and timeout settings.

The following attributes shall be reported: *StatusFlags*, *PresentValue*

3.14.4.2 Client

The client has no dependencies, no attributes, and receives or generates no cluster specific commands.

3.14.5 Binary Output (Basic) Cluster

The Binary Output (Basic) cluster provides an interface for setting the value of a binary output, and accessing various characteristics of that value.

3.14.5.1 Server

3.14.5.1.1 Dependencies

None.

3.14.5.1.2 Attributes

The attributes of this cluster are detailed in Table 3.71.

Table 3.71 Attributes of the Binary Output (Basic) Server Cluster

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0004	*ActiveText*	Character string	-	R*W	Null string	O
0x001C	*Description*	Character string	-	R*W	Null string	O
0x002E	*InactiveText*	Character string	-	R*W	Null string	O
0x0042	*MinimumOffTime*	Unsigned 32-bit integer	-	R*W	0xffffffff	O
0x0043	*MinimumOnTime*	Unsigned 32-bit integer	-	R*W	0xffffffff	O
0x0051	*OutOfService*	Boolean	False (0) or True (1)	R*W	False (0)	M
0x0054	*Polarity*	8-bit enumeration	-	R	0	O
0x0055	*PresentValue*	Boolean	-	R*W	-	M
0x0057	*PriorityArray*	Array of 16 structures of (boolean, boolean)	-	R/W	16 x (0, 0)	O

Table 3.71 Attributes of the Binary Output (Basic) Server Cluster (Continued)

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0067	*Reliability*	8-bit enumeration	-	R*W	0x00	O
0x0068	*RelinquishDefault*	Boolean	-	R*W	-	O
0x006F	*StatusFlags*	8-bit bitmap	0x00 - 0x0f	R	0	M
0x0100	*ApplicationType*	Unsigned 32-bit integer	0 - 0xffffffff	R	-	O
All others < 0x0400	*Reserved*					
0x0400 - 0xFFFF	*Reserved for vendor specific attributes*					

For an explanation of the attributes, see section 3.14.10.

3.14.5.1.3 Commands

No cluster specific commands are received or generated.

3.14.5.1.4 Attribute Reporting

This cluster shall support attribute reporting using the Report Attributes and Configure Reporting commands, according to the minimum and maximum reporting interval, value and timeout settings.

The following attributes shall be reported: *StatusFlags, PresentValue*

3.14.5.2 Client

The client has no dependencies, no attributes, and receives or generates no cluster specific commands.

3.14.6 Binary Value (Basic) Cluster

The Binary Value (Basic) cluster provides an interface for setting a binary value, typically used as a control system parameter, and accessing various characteristics of that value.

3.14.6.1 Server

3.14.6.1.1 Dependencies

None.

3.14.6.1.2 Attributes

The attributes of this cluster are detailed in Table 3.72.

Table 3.72 Attributes of the Binary Value (Basic) Server Cluster

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0004	*ActiveText*	Character string	-	R*W	Null string	O
0x001C	*Description*	Character string	-	R*W	Null string	O
0x002E	*InactiveText*	Character string	-	R*W	Null string	O
0x0042	*MinimumOffTime*	Unsigned 32-bit integer	-	R*W	0xffffffff	O
0x0043	*MinimumOnTime*	Unsigned 32-bit integer	-	R*W	0xffffffff	O
0x0051	*OutOfService*	Boolean	False (0) or True (1)	R*W	False (0)	M
0x0055	*PresentValue*	Boolean	-	R*W	-	M
0x0057	*PriorityArray*	Array of 16 structures of (boolean, boolean)	-	R/W	16 x (0, 0)	O
0x0067	*Reliability*	8-bit enumeration	-	R*W	0x00	O
0x0068	*RelinquishDefault*	Boolean	-	R*W	-	O
0x006F	*StatusFlags*	8-bit bitmap	0x00 - 0x0f	R	0	M
0x0100	*ApplicationType*	Unsigned 32-bit integer	0 - 0xffffffff	R	-	O
All others < 0x0400	*Reserved*					
0x0400 - 0xFFFF	*Reserved for vendor specific attributes*					

For an explanation of the attributes, see section 3.14.10.

3.14.6.1.3 Commands

No cluster specific commands are received or generated.

3.14.6.1.4 Attribute Reporting

This cluster shall support attribute reporting using the Report Attributes and Configure Reporting commands, according to the minimum and maximum reporting interval, value and timeout settings.

The following attributes shall be reported: *StatusFlags, PresentValue*

3.14.6.2 Client

The client has no dependencies, no attributes, and receives or generates no cluster specific commands.

3.14.7 Multistate Input (Basic) Cluster

The Multistate Input (Basic) cluster provides an interface for reading the value of a multistate measurement and accessing various characteristics of that measurement. The cluster is typically used to implement a sensor that measures a physical quantity that can take on one of a number of discrete states.

3.14.7.1 Server

3.14.7.1.1 Dependencies

None.

3.14.7.1.2 Attributes

The attributes of this cluster are detailed in Table 3.73.

Table 3.73 Attributes of the Multistate Input (Basic) Server Cluster

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x000E	*StateText*	Array of character string	-	R*W	Null	O
0x001C	*Description*	Character string	-	R*W	Null string	O
0x004A	*NumberOfStates*	Unsigned 16-bit integer	1 - 0xffff	R*W	0	M
0x0051	*OutOfService*	Boolean	False (0) or True (1)	R*W	False (0)	M
0x0055	*PresentValue*	Unsigned 16-bit integer	-	R*W	-	M

Table 3.73 Attributes of the Multistate Input (Basic) Server Cluster (Continued)

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0067	*Reliability*	8-bit enumeration	-	R*W	0x00	O
0x006F	*StatusFlags*	8-bit bitmap	0x00 - 0x0f	R	0	M
0x0100	*ApplicationType*	Unsigned 32-bit integer	0 - 0xffffffff	R	-	O
All others < 0x0400	*Reserved*					
0x0400 - 0xFFFF	*Reserved for vendor specific attributes*					

For an explanation of the attributes, see section 3.14.10.

3.14.7.1.3 Commands

No cluster specific commands are received or generated.

3.14.7.1.4 Attribute Reporting

This cluster shall support attribute reporting using the Report Attributes and Configure Reporting commands, according to the minimum and maximum reporting interval, value and timeout settings.

The following attributes shall be reported: StatusFlags, PresentValue

3.14.7.2 Client

The client has no dependencies, no attributes, and receives or generates no cluster specific commands.

3.14.8 Multistate Output (Basic) Cluster

The Multistate Output (Basic) cluster provides an interface for setting the value of an output that can take one of a number of discrete values, and accessing characteristics of that value.

3.14.8.1 Server

3.14.8.1.1 Dependencies

None.

3.14.8.1.2 Attributes

The attributes of this cluster are detailed in Table 3.74.

Table 3.74 Attributes of the Multistate Output (Basic) Server Cluster

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x000E	*StateText*	Array of character string	-	R*W	Null	O
0x001C	*Description*	Character string	-	R*W	Null string	O
0x004A	*NumberOfStates*	Unsigned 16-bit integer	1 - 0xffff	R*W	0	M
0x0051	*OutOfService*	Boolean	False (0) or True (1)	R*W	False (0)	M
0x0055	*PresentValue*	Unsigned 16-bit integer	-	R/W	-	M
0x0057	*PriorityArray*	Array of 16 structures of (boolean, unsigned 16-bit integer)	-	R/W	16 x (0, 0)	O
0x0067	*Reliability*	8-bit enumeration	-	R*W	0x00	O
0x0068	*RelinquishDefault*	Unsigned 16-bit integer	-	R*W	-	O
0x006F	*StatusFlags*	8-bit bitmap	0x00 - 0x0f	R	0	M
0x0100	*ApplicationType*	Unsigned 32-bit integer	0 - 0xffffffff	R	-	O
All others < 0x0400	Reserved					
0x0400 - 0xFFFF	Reserved for vendor specific attributes					

For an explanation of the attributes, see section 3.14.10.

3.14.8.1.3 Commands

No cluster specific commands are received or generated.

3.14.8.1.4 Attribute Reporting

This cluster shall support attribute reporting using the Report Attributes and Configure Reporting commands, according to the minimum and maximum reporting interval, value and timeout settings.

The following attributes shall be reported: *StatusFlags, PresentValue*

3.14.8.2 Client

The client has no dependencies, no attributes, and receives or generates no cluster specific commands.

3.14.9 Multistate Value (Basic) Cluster

The Multistate Value (Basic) cluster provides an interface for setting a multistate value, typically used as a control system parameter, and accessing characteristics of that value.

3.14.9.1 Server

3.14.9.1.1 Dependencies

None.

3.14.9.1.2 Attributes

The attributes of this cluster are detailed in Table 3.75.

Table 3.75 Attributes of the Multistate Value (Basic) Server Cluster

Identifier	*Name*	Type	Range	Access	Default	Mandatory / Optional
0x000E	*StateText*	Array of character string	-	R*W	Null	O
0x001C	*Description*	Character string	-	R*W	Null string	O
0x004A	*NumberOfStates*	Unsigned 16-bit integer	1 - 0xffff	R*W	0	M
0x0051	*OutOfService*	Boolean	False (0) or True (1)	R*W	False (0)	M
0x0055	*PresentValue*	Unsigned 16-bit integer	-	R/W	-	M

Table 3.75 Attributes of the Multistate Value (Basic) Server Cluster (Continued)

Identifier	*Name*	Type	Range	Access	Default	Mandatory / Optional
0x0057	*PriorityArray*	Array of 16 structures of (boolean, unsigned 16-bit integer)	-	R/W	16 x (0, 0)	O
0x0067	*Reliability*	8-bit enumeration	-	R*W	0x00	O
0x0068	*RelinquishDefault*	Unsigned 16-bit integer	-	R*W	-	O
0x006F	*StatusFlags*	8-bit bitmap	0x00 - 0x0f	R	0	M
0x0100	*ApplicationType*	Unsigned 32-bit integer	0 - 0xffffffff	R	-	O
All others < 0x0400	*Reserved*					
0x0400 - 0xFFFF	*Reserved for vendor specific attributes*					

For an explanation of the attributes, see section 3.14.10.

3.14.9.1.3 Commands

No cluster specific commands are received or generated.

3.14.9.1.4 Attribute Reporting

This cluster shall support attribute reporting using the Report Attributes and Configure Reporting commands, according to the minimum and maximum reporting interval, value and timeout settings.

The following attributes shall be reported: *StatusFlags*, *PresentValue*

3.14.9.2 Client

The client has no dependencies, no attributes, and receives or generates no cluster specific commands.

3.14.10 Attribute Descriptions

Note: These attributes are based on BACnet properties with the same names. For more information, refer to the BACnet reference manual [B8].

3.14.10.1 *OutOfService* Attribute

The *OutOfService* attribute, of type Boolean, indicates whether (TRUE) or not (FALSE) the physical input, output or value that the cluster represents is not in service. For an Input cluster, when *OutOfService* is TRUE the *PresentValue* attribute is decoupled from the physical input and will not track changes to the physical input. For an Output cluster, when *OutOfService* is TRUE the *PresentValue* attribute is decoupled from the physical output, so changes to *PresentValue* will not affect the physical output. For a Value cluster, when *OutOfService* is TRUE the *PresentValue* attribute may be written to freely by software local to the device that the cluster resides on.

3.14.10.2 *PresentValue* Attribute

The *PresentValue* attribute indicates the current value of the input, output or value, as appropriate for the cluster. For Analog clusters it is of type single precision, for Binary clusters it is of type Boolean, and for multistate clusters it is of type Unsigned 16-bit integer.

The PresentValue attribute of an input cluster shall be writable when *OutOfService* is TRUE.

When the *PriorityArray* attribute is implemented, writing to *PresentValue* shall be equivalent to writing to element 16 of *PriorityArray*, i.e. with a priority of 16.

3.14.10.3 *StatusFlags* Attribute

This attribute, of type bitmap, represents four Boolean flags that indicate the general “health” of the analog sensor. Three of the flags are associated with the values of other optional attributes of this cluster. A more detailed status could be determined by reading the optional attributes (if supported) that are linked to these flags. The relationship between individual flags is not defined. The four flags are

Bit 0 = IN ALARM, Bit 1 = FAULT, Bit 2 = OVERRIDDEN, Bit 3 = OUT OF SERVICE

where:

IN ALARM - Logical FALSE (0) if the *EventState* attribute has a value of NORMAL, otherwise logical TRUE (1). This bit is always 0 unless the cluster implementing the *EventState* attribute is implemented on the same endpoint.

FAULT - Logical TRUE (1) if the *Reliability* attribute is present and does not have a value of NO FAULT DETECTED, otherwise logical FALSE (0).

OVERRIDDEN- Logical TRUE (1) if the cluster has been overridden by some mechanism local to the device. Otherwise, the value is logical FALSE (0).

In this context, for an input cluster, “overridden” is taken to mean that the *PresentValue* and *Reliability* (optional) attributes are no longer tracking changes

to the physical input. For an Output cluster, “overridden” is taken to mean that the physical output is no longer tracking changes to the *PresentValue* attribute and the *Reliability* attribute is no longer a reflection of the physical output. For a Value cluster, “overridden” is taken to mean that the *PresentValue* attribute is not writeable.

OUT OF SERVICE - Logical TRUE (1) if the *OutOfService* attribute has a value of TRUE, otherwise logical FALSE (0).

3.14.10.4 *Description* Attribute

The *Description* attribute, of type Character string, may be used to hold a description of the usage of the input, output or value, as appropriate to the cluster. The character set used shall be ASCII, and the attribute shall contain a maximum of 16 characters, which shall be printable but are otherwise unrestricted.

3.14.10.5 *MaxPresentValue* Attribute

The *MaxPresentValue* attribute, of type Single precision, indicates the highest value that can be reliably obtained for the *PresentValue* attribute of an Analog Input cluster, or which can reliably be used for the *PresentValue* attribute of an Analog Output or Analog Value cluster.

3.14.10.6 *PriorityArray* Attribute

The *PriorityArray* attribute is an array of 16 structures. The first element of each structure is a boolean, and the second element is of the same type as the *PresentValue* attribute of the corresponding cluster.

PriorityArray holds potential values for the *PresentValue* attribute of the corresponding cluster, in order of decreasing priority. The first value in the array corresponds to priority 1 (highest), the second value corresponds to priority 2, and so on, to the sixteenth value that corresponds to priority 16 (lowest).

The boolean value in each element of the array indicates whether (TRUE) or not (FALSE) there is a valid value at that priority. All entries within the priority table are continuously monitored in order to locate the entry with the highest priority valid value, and *PresentValue* is set to this value.

When *PriorityArray* is supported, *PresentValue* may be written to indirectly by writing to the *PriorityArray*, as described above. If *PresentValue* is written to directly, a default priority of 16 (the lowest priority) shall be assumed, and the value is entered into the 16th element of *PriorityArray.*

When a value at a given priority is marked as invalid, by writing FALSE to its corresponding boolean value, it is said to be relinquished.

(**Informative note:** In BACnet, each element of PriorityArray consists of a single value, which may be either of the same type as *PresentValue* or may be of type

NULL to indicate that a value is not present. In ZigBee an attribute cannot have a variable data type, thus an extra boolean value is associated with each element of the array to indicate whether or not it is null).

3.14.10.7 *RelinquishDefault* Attribute

The *RelinquishDefault* attribute is the default value to be used for the *PresentValue* attribute when all elements of the *PriorityArray* attribute are marked as invalid.

3.14.10.8 *MinPresentValue* Attribute

The *MinPresentValue* attribute, of type Single precision, indicates the lowest value that can be reliably obtained for the *PresentValue* attribute of an Analog Input cluster, or which can reliably be used for the *PresentValue* attribute of an Analog Output or Analog Value cluster.

3.14.10.9 *Reliability* Attribute

The *Reliability* attribute, of type 8-bit enumeration, provides an indication of whether the *PresentValue* or the operation of the physical input, output or value in question (as appropriate for the cluster) is “reliable” as far as can be determined and, if not, why not. The *Reliability* attribute may have any of the following values:

NO-FAULT-DETECTED (0)

NO-SENSOR (1) - for input clusters only

OVER-RANGE (2)

UNDER-RANGE (3)

OPEN-LOOP (4)

SHORTED-LOOP (5)

NO-OUTPUT (6) - for input clusters only

UNRELIABLE-OTHER (7)

PROCESS-ERROR (8)

MULTI-STATE-FAULT (9) - for multistate clusters only

CONFIGURATION-ERROR (10)

3.14.10.10 *EngineeringUnits* Attribute

The *EngineeringUnits* attribute indicates the physical units associated with the value of the *PresentValue* attribute of an Analog cluster.

Values 0x0000 to 0x00fe are reserved for the list of engineering units with corresponding values specified in Clause 21 of the BACnet standard [B8]. 0x00ff represents 'other'. Values 0x0100 to 0xffff are available for proprietary use.

If the *ApplicationType* attribute is implemented, and is set to a value with a defined physical unit, the physical unit defined in *ApplicationType* takes priority over *EngineeringUnits*.

This attribute is defined to be read only, but a vendor can decide to allow this to be written to if *ApplicationType* is also supported. If this attribute is written to, how the device handles invalid units (e.g. changing Deg F to Cubic Feet per Minute), any local display or other vendor-specific operation (upon the change) is a local matter.

3.14.10.11 *Resolution* Attribute

This attribute, of type Single precision, indicates the smallest recognizable change to PresentValue.

3.14.10.12 *ActiveText* Attribute

This attribute, of type Character string, may be used to hold a human readable description of the ACTIVE state of a binary *PresentValue*. For example, for a Binary Input cluster, if the physical input is a switch contact, then the *ActiveText* attribute might be assigned a value such as “Fan 1 On”. If either the *ActiveText* attribute or the *InactiveText* attribute are present, then both of them shall be present.

The character set used shall be ASCII, and the attribute shall contain a maximum of 16 characters, which shall be printable but are otherwise unrestricted.

3.14.10.13 *InactiveText* Attribute

This attribute, of type Character string, may be used to hold a human readable description of the INACTIVE state of a binary *PresentValue*. For example, for a Binary Input cluster, if the physical input is a switch contact, then the *InactiveText* attribute might be assigned a value such as “Fan 1 Off”. If either the *InactiveText* attribute or the *ActiveText* attribute are present, then both of them shall be present.

The character set used shall be ASCII, and the attribute shall contain a maximum of 16 characters, which shall be printable but are otherwise unrestricted.

3.14.10.14 *MinimumOffTime* Attribute

This property, of type 32-bit unsigned integer, represents the minimum number of seconds that a binary *PresentValue* shall remain in the INACTIVE (0) state after a write to *PresentValue* causes it to assume the INACTIVE state.

3.14.10.15 *MinimumOnTime* Attribute

This property, of type 32-bit unsigned integer, represents the minimum number of seconds that a binary *PresentValue* shall remain in the ACTIVE (1) state after a write to *PresentValue* causes it to assume the ACTIVE state.

3.14.10.16 *Polarity* Attribute

This attribute, of type enumeration, indicates the relationship between the physical state of the input (or output as appropriate for the cluster) and the logical state represented by a binary *PresentValue* attribute, when *OutOfService* is FALSE. If the *Polarity* attribute is NORMAL (0), then the ACTIVE (1) state of the *PresentValue* attribute is also the ACTIVE or ON state of the physical input (or output). If the *Polarity* attribute is REVERSE (1), then the ACTIVE (1) state of the *PresentValue* attribute is the INACTIVE or OFF state of the physical input (or output).

Thus, when *OutOfService* is FALSE, for a constant physical input state a change in the *Polarity* attribute shall produce a change in the *PresentValue* attribute. If *OutOfService* is TRUE, then the *Polarity* attribute shall have no effect on the *PresentValue* attribute.

3.14.10.17 *NumberOfStates* Attribute

This attribute, of type Unsigned 16-bit integer, defines the number of states that a multistate *PresentValue* may have. The *NumberOfStates* property shall always have a value greater than zero. If the value of this property is changed, the size of the *StateText* array, if present, shall also be changed to the same value. The states are numbered consecutively, starting with 1.

3.14.10.18 *StateText* Attribute

This attribute, of type Array of Character strings, holds descriptions of all possible states of a multistate *PresentValue*. The number of descriptions matches the number of states defined in the *NumberOfStates* property. The *PresentValue*, interpreted as an integer, serves as an index into the array. If the size of this array is changed, the *NumberOfStates* property shall also be changed to the same value.

The character set used shall be ASCII, and the attribute shall contain a maximum of 16 characters, which shall be printable but are otherwise unrestricted.

3.14.10.19 *ApplicationType* Attribute

The *ApplicationType* attribute is an unsigned 32 bit integer that indicates the specific application usage for this cluster. (Note, this attribute has no BACnet equivalent.)

ApplicationType is subdivided into Group, Type and an Index number, as follows.

Group = Bits 24 - 31
An indication of the cluster this attribute is part of.

Type = Bits 16 - 23
For Analog clusters, the physical quantity that the Present Value attribute of the cluster represents.
For Binary and Multistate clusters, the application usage domain.

Index = Bits 0 - 15
The specific application usage of the cluster

3.14.10.19.1 Analog Input (AI) Types

Group = 0x00.

The following sub-clauses describe the values when Type = 0x00 - 0x0E. Types 0x0F to 0xFE are reserved, Type = 0xFF indicates other.

3.14.10.19.1.1 Type = 0x00: Temperature in degrees C

Table 3.76 AI Types, Type = 0x00: Temperature in Degrees C

Index	Application Usage
0x0000	2 Pipe Entering Water Temperature AI
0x0001	2 Pipe Leaving Water Temperature AI
0x0002	Boiler Entering Temperature AI
0x0003	Boiler Leaving Temperature AI
0x0004	Chiller Chilled Water Entering Temp AI
0x0005	Chiller Chilled Water Leaving Temp AI
0x0006	Chiller Condenser Water Entering Temp AI
0x0007	Chiller Condenser Water Leaving Temp AI
0x0008	Cold Deck Temperature AI
0x0009	Cooling Coil Discharge Temperature AI
0x000A	Cooling Entering Water Temperature AI
0x000B	Cooling Leaving Water Temperature AI
0x000C	Condenser Water Return Temperature AI
0x000D	Condenser Water Supply Temperature AI
0x000E	Decouple Loop Temperature AI
0x000F	Building Load AI
0x0010	Decouple Loop Temperature AI
0x0011	Dew Point Temperature AI
0x0012	Discharge Air Temperature AI
0x0013	Discharge Temperature AI
0x0014	Exhaust Air Temperature After Heat Recovery AI
0x0015	Exhaust Air Temperature AI
0x0016	Glycol Temperature AI
0x0017	Heat Recovery Air Temperature AI
0x0018	Hot Deck Temperature AI
0x0019	Heat Exchanger Bypass Temp AI
0x001A	Heat Exchanger Entering Temp AI
0x001B	Heat Exchanger Leaving Temp AI
0x001C	Mechanical Room Temperature AI

Table 3.76 AI Types, Type = 0x00: Temperature in Degrees C (Continued)

Index	Application Usage
0x001D	Mixed Air Temperature AI
0x001E	Mixed Air Temperature AI
0x001F	Outdoor Air Dewpoint Temp AI
0x0020	Outdoor Air Temperature AI
0x0021	Preheat Air Temperature AI
0x0022	Preheat Entering Water Temperature AI
0x0023	Preheat Leaving Water Temperature AI
0x0024	Primary Chilled Water Return Temp AI
0x0025	Primary Chilled Water Supply Temp AI
0x0026	Primary Hot Water Return Temp AI
0x0027	Primary Hot Water Supply Temp AI
0x0028	Reheat Coil Discharge Temperature AI
0x0029	Reheat Entering Water Temperature AI
0x002A	Reheat Leaving Water Temperature AI
0x002B	Return Air Temperature AI
0x002C	Secondary Chilled Water Return Temp AI
0x002D	Secondary Chilled Water Supply Temp AI
0x002E	Secondary HW Return Temp AI
0x002F	Secondary HW Supply Temp AI
0x0030	Sideloop Reset Temperature AI
0x0031	Sideloop Temperature Setpoint AI
0x0032	Sideloop Temperature AI
0x0033	Source Temperature
0x0034	Supply Air Temperature AI
0x0035	Supply Low Limit Temperature AI
0x0036	Tower Basin Temp AI
0x0037	Two Pipe Leaving Water Temp AI
0x0038	Reserved
0x0039	Zone Dewpoint Temperature AI
0x003A	Zone Sensor Setpoint AI

Table 3.76 AI Types, Type = 0x00: Temperature in Degrees C (Continued)

Index	Application Usage
0x003B	Zone Sensor Setpoint Offset AI
0x003C	Zone Temperature AI
0x003D - 0x01FF	Reserved
0x0200 - 0xFFFE	Vendor defined
0xFFFF	Other

3.14.10.19.1.2 Type = 0x01: Relative Humidity in %

Table 3.77 AI Types, Type = 0x01: Relative Humidity in %

Index	Application Usage
0x0000	Discharge Humidity AI
0x0001	Exhaust Humidity AI
0x0002	Hot Deck Humidity AI
0x0003	Mixed Air Humidity AI
0x0004	Outdoor Air Humidity AI
0x0005	Return Humidity AI
0x0006	Sideloop Humidity AI
0x0007	Space Humidity AI
0x0008	Zone Humidity AI
0x0009 - 0x01FF	Reserved
0x0200 - 0xFFFE	Vendor defined
0xFFFF	Other

3.14.10.19.1.3 Type = 0x02: Pressure in Pascal

Table 3.78 AI Types, Type = 0x02: Pressure in Pascal

Index	Application Usage
0x0000	Boiler Pump Differential Pressure AI
0x0001	Building Static Pressure AI
0x0002	Cold Deck Differential Pressure Sensor AI
0x0003	Chilled Water Building Differential Pressure AI
0x0004	Cold Deck Differential Pressure AI
0x0005	Cold Deck Static Pressure AI
0x0006	Condenser Water Pump Differential Pressure AI
0x0007	Discharge Differential Pressure AI
0x0008	Discharge Static Pressure 1 AI
0x0009	Discharge Static Pressure 2 AI
0x000A	Exhaust Air Differential Pressure AI
0x000B	Exhaust Air Static Pressure AI
0x000C	Exhaust Differential Pressure AI

Table 3.78 AI Types, Type = 0x02: Pressure in Pascal (Continued)

Index	Application Usage
0x000D	Exhaust Differential Pressure AI
0x000E	Hot Deck Differential Pressure AI
0x000F	Hot Deck Differential Pressure AI
0x0010	Hot Deck Static Pressure AI
0x0011	Hot Water Bldg Diff Pressure AI
0x0012	Heat Exchanger Steam Pressure AI
0x0013	Minimum Outdoor Air Differential Pressure AI
0x0014	Outdoor Air Differential Pressure AI
0x0015	Primary Chilled water Pump Differential Pressure AI
0x0016	Primary Hot water Pump Differential Pressure AI
0x0017	Relief Differential Pressure AI
0x0018	Return Air Static Pressure AI
0x0019	Return Differential Pressure AI
0x001A	Secondary Chilled Water Pump Differential Pressure AI
0x001B	Secondary Hot water Pump Differential Pressure AI
0x001C	Sideloop Pressure AI
0x001D	Steam Pressure AI
0x001E	Supply Differential Pressure Sensor AI
0x001F - 0x01FF	Reserved
0x0200 - 0xFFFE	Vendor defined
0xFFFF	Other

3.14.10.19.1.4 Type = 0x03: Flow in Liters/Second

Table 3.79 AI Types, Type = 0x03: Flow in Liters/second

Index	Application Usage
0x0000	Chilled Water Flow AI
0x0001	Chiller Chilled Water Flow AI
0x0002	Chiller Condenser Water Flow AI
0x0003	Cold Deck Flow AI
0x0004	Decouple Loop Flow AI
0x0005	Discharge Flow AI
0x0006	Exhaust Fan Flow AI
0x0007	Exhaust Flow AI
0x0008	Fan Flow AI
0x0009	Hot Deck Flow AI
0x000A	Hot Water Flow AI
0x000B	Minimum Outdoor Air Fan Flow AI
0x000C	Minimum Outdoor Air Flow AI
0x000D	Outdoor Air Flow AI
0x000E	Primary Chilled Water Flow AI
0X000F	Relief Fan Flow AI
0x0010	Relief Flow AI
0x0011	Return Fan Flow AI
0x0012	Return Flow AI
0x0013	Secondary Chilled Water Flow AI
0x0014	Supply Fan Flow AI
0x0015	Tower Fan Flow AI
0x0016 - 0x01FF	Reserved
0x0200 - 0xFFFE	Vendor defined
0xFFFF	Other

3.14.10.19.1.5 Type = 0x04: Percentage %

Table 3.80 AI Types, Type = 0x04: Percentage %

Index	Application Usage
0x0000	Chiller % Full Load Amperage AI
0x0001 - 0x01FF	Reserved
0x0200 - 0xFFFE	Vendor defined
0xFFFF	Other

3.14.10.19.1.6 Type = 0x05: Parts per Million PPM

Table 3.81 AI types, Type = 0x05: Parts per Million PPM

Index	Application Usage
0x0000	Return Carbon Dioxide AI
0x0001	Zone Carbon Dioxide AI
0x0002 - 0x01FF	Reserved
0x0200 - 0xFFFE	Vendor defined
0xFFFF	Other

3.14.10.19.1.7 Type = 0x06: Rotational Speed in RPM

Table 3.82 AI Types, Type = 0x06: Rotational Speed in RPM

Index	Application Usage
0x0000	Exhaust Fan Remote Speed AI
0x0001	Heat Recovery Wheel Remote Speed AI
0x0002	Min Outdoor Air Fan Remote Speed AI
0x0003	Relief Fan Remote Speed AI
0x0004	Return Fan Remote Speed AI
0x0005	Supply Fan Remote Speed AI
0x0006	Variable Speed Drive Motor Speed AI
0x0007	Variable Speed Drive Speed Setpoint AI
0x0008 - 0x01FF	Reserved
0x0200- 0xFFFE	Vendor defined
0xFFFF	Other

3.14.10.19.1.8 Type = 0x07: Current in Amps

Table 3.83 AI Types, Type = 0x07: Current in Amps

Index	Application Usage
0x0000	Chiller Amps AI
0x0001 - 0x01FF	Reserved
0x0200 - 0xFFFE	Vendor defined
0xFFFF	Other

3.14.10.19.1.9 Type = 0x08: Frequency in Hz

Table 3.84 AI Types, Type = 0x08: Frequency in Hz

Index	Application Usage
0x0000	Variable Speed Drive Output Frequency AI
0x0001 - 0x01FF	Reserved
0x0200- 0xFFFE	Vendor defined
0xFFFF	Other

3.14.10.19.1.10 Type = 0x09: Power in Watts

Table 3.85 AI Types, Type = 0x09: Power in Watts

Index	Application Usage
0x0000	Power Consumption AI
0x0001 - 01FF	Reserved
0x0200- FFFE	Vendor defined
0xFFFF	Other

3.14.10.19.1.11 Type = 0x0A: Power in kW

Table 3.86 AI Types, Type = 0x0A: Power in kW

Index	Application Usage
0x0000	Absolute Power AI
0x0001	Power Consumption AI
0x0002 - 0x01FF	Reserved
0x0200 - 0xFFFE	Vendor defined
0xFFFF	Other

3.14.10.19.1.12 Type = 0x0B: Energy in kWH

Table 3.87 AI Types, Type = 0x0B: Energy in kWH

Index	Application Usage
0x0000	Variable Speed Drive Kilowatt Hours AI
0x0001 - 0x01FF	Reserved
0x0200- FFFE	Vendor defined
0xFFFF	Other

3.14.10.19.1.13 Type = 0x0C: Count - Unitless

Table 3.88 AI Types, Type = 0x0C: Count - Unitless

Index	Application Usage
0x0000	Count
0x0001 - 0x01FF	Reserved
0x0200 - 0xFFFE	Vendor defined
0xFFFF	Other

3.14.10.19.1.14 Type = 0x0D: Enthalpy in KJoules/Kg

Table 3.89 AI Types, Type = 0x0D: Enthalpy in KJoules/Kg

Index	Application Usage
0x0000	Outdoor Air Enthalpy AI
0x0001	Return Air Enthalpy AI
0x0002	Space Enthalpy
0x0003 - 0x01FF	Reserved
0x0200 - 0xFFFE	Vendor defined
0xFFFF	Other

3.14.10.19.1.15 Type = 0x0E: Time in Seconds

Table 3.90 AI types, Type = 0x0E: Time in Seconds

Index	Application Usage
0x0000	Relative time AI
0x0001 - 0x01FF	Reserved
0x0200 - 0xFFFE	Vendor defined
0xFFFF	Other

3.14.10.19.2 Analog Output (AO) types

Group = 0x01.

The following sub-clauses describe the values when Type = 0x00 - 0x0E. Types 0x0F to 0xFE are reserved, Type = 0xFF indicates other.

3.14.10.19.2.1 Type = 0x00: Temperature in Degrees C

Table 3.91 AO Types, Type = 0x00: Temperature in Degrees C

Index	Application Usage
0x0000	Boiler AO
0x0001	Boiler Setpoint AO
0x0002	Cold Deck AO
0x0003	Chiller Setpoint AO
0x0004	Chiller Setpoint AO
0x0005	Hot Deck AO
0x0006	Cooling Valve AO
0x0007	Zone Temperature Setpoint AO
0x0008	Setpoint Offset AO
0x0009	Setpoint Shift AO
0x000A - 0x01FF	Reserved
0x0200 - 0xFFFE	Vendor defined
0xFFFF	Other

3.14.10.19.2.2 Type = 0x01: Relative Humidity in %

Table 3.92 AO Types, Type = 0x01: Relative Humidity in %

Index	Application Usage
0x0000	Humidification AO
0x0001	Zone Relative Humidity Setpoint AO
0x0002 - 0x01FF	Reserved
0x0200 - 0xFFFE	Vendor defined
0xFFFF	Other

3.14.10.19.2.3 Type = 0x02: Pressure Pascal

Table 3.93 AO Types, Type = 0x02: Pressure Pascal

Index	Application Usage
0x0000 - 01FF	Reserved
0x0200 - 0xFFFE	Vendor defined
0xFFFF	Other

3.14.10.19.2.4 Type = 0x03: Flow in Liters/Second

Table 3.94 AO Types, Type = 0x03: Flow in Liters/Second

Index	Application Usage
0x0000 - 0x01FF	Reserved
0x0200 - 0xFFFE	Vendor defined
0xFFFF	Other

3.14.10.19.2.5 Type = 0x04: Percentage %

Table 3.95 AO Types, Type = 0x04: Percentage %

Index	Application Usage
0x0000	Face & Bypass Damper AO
0x0001	Heat Recovery Valve AO
0x0002	Heat Recovery Wheel AO
0x0003	Heating Valve AO
0x0004	Hot Deck Damper AO
0x0005	2 Pipe Damper AO
0x0006	2 Pipe Valve AO
0x0007	Boiler Mixing Valve AO
0x0008	Box Cooling Valve AO
0x0009	Box Heating Valve AO
0x000A	Chilled Water Bypass Valve AO
0x000B	Cold Deck Damper AO
0x000C	Cooling Damper AO
0x000D	Cooling Valve AO
0x000E	Damper AO
0x000F	Exhaust Air Damper AO
0x0010	Exhaust Damper AO
0x0011	Hot Water Bypass Valve AO
0x0012	Hot Water Mixing Valve AO
0x0013	Minimum Outside Air Damper AO
0x0014	Minimum Outside Air Fan AO
0x0015	Mixed Air Damper AO

Table 3.95 AO Types, Type = 0x04: Percentage % (Continued)

Index	Application Usage
0x0016	Mixing Valve AO
0x0017	Outside Air Damper AO
0x0018	Primary Chilled Water Pump AO
0x0019	Primary Hot Water Pump AO
0x001A	Primary Heat Exchange Pump AO
0x001B	Preheat Damper AO
0x001C	Preheat Valve AO
0x001D	Reheat Valve 1 AO
0x001E	Reheat Valve AO
0x001F	Return Air Damper AO
0x0020	Secondary Chilled Water Pump AO
0x0021	Sequenced Valves AO
0x0022	Secondary Hot Water Pump AO
0x0023	Secondary Heat Exchange Pump AO
0x0024	Sideloop AO
0x0025	Supply Heating Valve AO
0x0026	Supply Damper AO
0x0027	Tower Bypass Valve AO
0x0028	Tower Fan AO
0x0029	Valve AO
0x002A	Zone 1 Damper AO
0x002B	Zone 1 Heating Valve AO
0x002C	Heat Recovery Exhaust Bypass Damper AO
0x002D	Heat Recovery Outside Air Bypass Damper AO
0x002E - 0x01FF	Reserved
0x0200 - 0xFFFE	Vendor defined
0xFFFF	Other

3.14.10.19.2.6 Type = 0x05: Parts per Million PPM

Table 3.96 AO Types, Type = 0x05: Parts per Million PPM

Index	Application Usage
0x0000	Space Carbon Dioxide limit AO
0x0001 - 0x01FF	Reserved
0x0200 - 0xFFFE	Vendor defined
0xFFFF	Other

3.14.10.19.2.7 Type = 0x06: Rotational Speed RPM

Table 3.97 AO Types, Type = 0x06: Rotational Speed RPM

Index	Application Usage
0x0000	Exhaust Fan Speed AO
0x0001	Fan Speed AO
0x0002	Relief Fan Speed AO
0x0003	Return Fan Speed AO
0x0004	Supply Fan Speed AO
0x0005 - 0x01FF	Reserved
0x0200- 0xFFFE	Vendor defined
0xFFFF	Other

3.14.10.19.2.8 Type = 0x07: Current in Amps

Table 3.98 AO Types, Type = 0x07: Current in Amps

Index	Application Usage
0x0000 - 0x01FF	Reserved
0x0200- 0xFFFE	Vendor defined
0xFFFF	Other

3.14.10.19.2.9 Type = 0x08: Frequency in Hz

Table 3.99 AO Types, Type = 0x08: Frequency in Hz

Index	Application Usage
0x0000 - 0x01FF	Reserved
0x0200- 0xFFFE	Vendor defined
0xFFFF	Other

3.14.10.19.2.10 Type = 0x09: Power in Watts

Table 3.100 AO Types, Type = 0x09: Power in Watts

Index	Application Usage
0x0000 - 0x01FF	Reserved
0x0200- 0xFFFE	Vendor defined
0xFFFF	Other

3.14.10.19.2.11 Type = 0x0A: Power in kW

Table 3.101 AO Types, Type = 0x0A: Power in kW

Index	Application Usage
0x0000 - 0x01FF	Reserved
0x0200- 0xFFFE	Vendor defined
0xFFFF	Other

3.14.10.19.2.12 Type = 0x0B: Energy in kWh

Table 3.102 AO Types, Type = 0x0B: Energy in kWh

Index	Application Usage
0x0000 - 0x01FF	Reserved
0x0200- 0xFFFE	Vendor defined
0xFFFF	Other

3.14.10.19.2.13 Type = 0x0C: Count - Unitless

Table 3.103 AO Types, Type = 0x0C: Count - Unitless

Index	Application Usage
0x0000 - 0x01FF	Reserved
0x0200- 0xFFFE	Vendor defined
0xFFFF	Other

3.14.10.19.2.14 Type = 0x0D: Enthalpy in KJoules/Kg

Table 3.104 AO Types, Type = 0x0D: Enthalpy in KJoules/Kg

Index	Application Usage
0x0000 - 0x01FF	Reserved
0x0200- 0xFFFE	Vendor defined
0xFFFF	Other

3.14.10.19.2.15 Type = 0x0E: Time in Seconds

Table 3.105 AO Types, Type = 0x0E: Time in Seconds

Index	Application Usage
0x0000	Relative time AO
0x0001 - 0x01FF	Reserved
0x0200- 0xFFFE	Vendor defined
0xFFFF	Other

3.14.10.19.3 Analog Value (AV) Types

Group = 0x02.

The following sub-clauses describe the values when Type = 0x00 - 0x03. Types 0x04 to 0xFE are reserved, Type = 0xFF indicates other.

3.14.10.19.3.1 Type = 0x00: Temperature in Degrees C

Table 3.106 AV Types, Type = 0x00: Temperature in Degrees C

Index	Application Usage
0x0000	Setpoint Offset AV
0x0001	Temp Deadband AV
0x0002	Occupied Heating Setpoint AV
0x0003	Unoccupied Heating Setpoint AV
0x0004	Occupied Cooling Setpoint AV
0x0005	Unoccupied Cooling Setpoint AV
0x0006	Standby Heat Setpoint AV
0x0007	Standby Cooling Setpoint AV
0x0008	Effective Occupied Heating Setpoint AV
0x0009	Effective Unoccupied Heating Setpoint AV
0x000A	Effective Occupied Cooling Setpoint AV
0x000B	Effective Unoccupied Cooling Setpoint AV
0x000C	Effective Standby Heat Setpoint AV
0x000D	Effective Standby Cooling Setpoint AV
0x000E	Setpoint Offset AV
0x000F	Setpoint Shift AV
0x0010 - 01FF	Reserved
0x0200 - FFFE	Vendor defined
0xFFFF	Other

3.14.10.19.3.2 Type = 0x01: Area in Square Metres

Table 3.107 AV Types, Type = 0x01: Area in Square Metres

Index	Application Usage
0x0000	Duct Area AV
0x0001 - 0x01FF	Reserved
0x0200- 0xFFFE	Vendor defined
0xFFFF	Other

3.14.10.19.3.3 Type = 0x02: Multiplier - Number

Table 3.108 AV Types, Type = 0x02: Multiplier - Number

Index	Application Usage
0x0000	Gain multiplier AV
0x0001 - 0x01FF	Reserved
0x0200- 0xFFFE	Vendor defined
0xFFFF	Other

3.14.10.19.3.4 Type 0x03: Flow in Litres/Second

Table 3.109 AV Types, Type = 0x03: Flow in Litres/Second

Index	Application Usage
0x0000	Minimum Air Flow AV
0x0001	Maximum Air Flow AV
0x0002	Heating Minimum Air Flow AV
0x0003	Heating Maximum Air Flow AV
0x0004	Standby Minimum Air Flow AV
0x0005	Standby Maximum Air Flow AV
0x0006- 01FF	Reserved
0x0200 - FFFE	Vendor defined
0xFFFF	Other

3.14.10.19.4 Binary Inputs (BI) Types

Group = 0x03.

The following sub-clauses describe the values when Type = 0x00 - 0x01. Types 0x02 to 0xFE are reserved, Type = 0xFF indicates other.

Present Value = 0 represents False, Off, Normal

Present Value = 1 represents True, On, Alarm

3.14.10.19.4.1 Type = 0x00: Application Domain HVAC

Table 3.110 BI Types, Type = 0x00: Application Domain HVAC

Index	Application Usage
0x0000	2 Pipe Pump Status BI
0x0001	Air Proving Switch BI
0x0002	Alarm Reset BI
0x0003	Boiler Status BI
0x0004	Boiler Flow Status BI
0x0005	Boiler General Alarm BI
0x0006	Boiler High Temperature Alarm BI
0x0007	Boiler Isolation Valve Status BI
0x0008	Boiler Maintenance Switch BI
0x0009	Boiler Pump Overload BI
0x000A	Boiler Pump Status BI
0x000B	Boiler Status BI
0x000C	Box Heating Alarm BI
0x000D	Chiller Alarm BI
0x000E	Chiller Chilled Water Flow Status BI
0x000F	Chiller Chilled Water Isolation Valve Status BI
0x0010	Chiller Condenser Water Flow Status BI
0x0011	Chiller Condenser Water Isolation Valve Status BI
0x0012	Chiller Maintenance Switch BI
0x0013	Chiller Status BI
0x0014	Chilled Water Expansion Tank Alarm BI
0x0015	Chilled Water Expansion Tank High Pressure Alarm BI
0x0016	Chilled Water Expansion Tank Low Pressure Alarm BI
0x0017	Chilled Water Expansion Tank Status BI
0x0018	Combustion Damper Status BI
0x0019	Cooling Alarm BI
0x001A	Cooling Pump Maintenance Switch BI
0x001B	Cooling Pump Overload BI
0x001C	Cooling Pump Status BI

Table 3.110 BI Types, Type = 0x00: Application Domain HVAC (Continued)

Index	Application Usage
0x001D	Condenser Water Expansion Tank Alarm BI
0x001E	Condenser Water Expansion Tank High Pressure Alarm BI
0x001F	Condenser Water Expansion Tank Low Pressure Alarm BI
0x0020	Condenser Water Expansion Tank Status BI
0x0021	Condenser Water Pump Maintenance Switch BI
0x0022	Condenser Water Pump Overload BI
0x0023	Condenser Water Pump Status BI
0x0024	Decouple Loop Flow Direction BI
0x0025	Discharge Smoke BI
0x0026	Door Status BI
0x0027	Economizer Command BI
0x0028	Emergency Shutdown BI
0x0029	Equipment Tamper BI
0x002A	Energy Hold Off BI
0x002B	Exhaust Fan Maintenance Switch BI
0x002C	Exhaust Fan Overload BI
0x002D	Exhaust Fan Status BI
0x002E	Exhaust Filter Status BI
0x002F	Exhaust Smoke BI
0x0030	Expansion Tank Alarm BI
0x0031	Expansion Tank High Pressure Alarm BI
0x0032	Expansion Tank Low Pressure Alarm BI
0x0033	Expansion Tank Status BI
0x0034	Fan Control By Others BI
0x0035	Fan Overload BI
0x0036	Filter Monitoring BI
0x0037	Final Filter Status BI
0x0038	Free Cooling Availability BI
0x0039	Heat Recovery Pump Status BI
0x003A	Heat Recovery Wheel Alarm BI

Table 3.110 BI Types, Type = 0x00: Application Domain HVAC (Continued)

Index	Application Usage
0x003B	Heat Recovery Wheel Maintenance Switch BI
0x003C	Heat Recovery Wheel Overload BI
0x003D	Heat Recovery Wheel Status BI
0x003E	Heating Alarm BI
0x003F	Heating/Cooling Pump Maintenance Switch BI
0x0040	Heating/Cooling Pump Overload BI
0x0041	High Humidity Limit BI
0x0042	High Static Pressure Fault BI
0x0043	High Temperature Limit Fault BI
0x0044	Humidifier Alarm BI
0x0045	Humidifier Maintenance Switch BI
0x0046	Humidifier Overload BI
0x0047	Humidifier Status BI
0x0048	Heat Exchanger Alarm BI
0x0049	Heat Exchanger Isolation Valve Status BI
0x004A	Heat Exchanger Maintenance Switch BI
0x004B	Lighting Status BI
0x004C	Low Static Pressure Fault BI
0x004D	Low Temperature Limit Fault BI
0x004E	Minimum Outdoor Air Damper End Switch BI
0x004F	Minimum Outdoor Air Fan Maintenance Switch BI
0x0050	Minimum Outdoor Air Fan Overload BI
0x0051	Minimum Outdoor Air Fan Status BI
0x0052	Minimum Outdoor Air Fan Variable Frequency Drive Fault BI
0x0053	Occupancy BI
0x0054	Occupancy Sensor BI
0x0055	Primary Chilled Water Pump Maintenance Switch BI
0x0056	Primary Chilled Water Pump Overload BI
0x0057	Primary Chilled Water Pump Status BI
0x0058	Primary Chilled Water Pump Maintenance Switch BI

Table 3.110 BI Types, Type = 0x00: Application Domain HVAC (Continued)

Index	Application Usage
0x0059	Primary Chilled Water Pump Overload BI
0x005A	Primary Chilled Water Pump Status BI
0x005B	Pre-Filter Status BI
0x005C	Preheat Alarm BI
0x005D	Preheat Bonnet Switch BI
0x005E	Preheat Pump Maintenance Switch BI
0x005F	Preheat Pump Overload BI
0x0060	Preheat Pump Status BI
0x0061	Refrigerant Alarm BI
0x0062	Reheat Alarm BI
0x0063	Reheat Bonnet Switch BI
0x0064	Reheat Pump Maintenance Switch BI
0x0065	Reheat Pump Overload BI
0x0066	Reheat Pump Status BI
0x0067	Relief Fan Maintenance Switch BI
0x0068	Relief Fan Overload BI
0x0069	Relief Fan Status BI
0x006A	Relief Fan Variable Frequency Drive Fault BI
0x006B	Return Air Smoke BI
0x006C	Return Fan Maintenance Switch BI
0x006D	Return Fan Overload BI
0x006E	Return Fan Status BI
0x006F	Return Fan VFD Fault BI
0x0070	Return Smoke BI
0x0071	Secondary Chilled Water Pump 1 Maintenance Switch BI
0x0072	Secondary Chilled Water Pump 1 Overload BI
0x0073	Secondary Chilled Water Pump 1 Status BI
0x0074	Secondary Chilled Water Pump 1 Maintenance Switch BI
0x0075	Secondary Chilled Water Pump 1 Overload BI
0x0076	Secondary Chilled Water Pump 1 Status BI

Table 3.110 BI Types, Type = 0x00: Application Domain HVAC (Continued)

Index	Application Usage
0x0077	Sideloop BI
0x0078	Generic Status BI
0x0079	Summer Winter BI
0x007A	Supplemental Heating Alarm BI
0x007B	Supplemental Heating Pump Maintenance Switch BI
0x007C	Supplemental Heating Pump Overload BI
0x007D	Supplemental Heating Pump Status BI
0x007E	Supply Fan Maintenance Switch BI
0x007F	Supply Fan Overload BI
0x0080	Supply Fan Status BI
0x0081	Supply Fan Variable Frequency Drive Fault BI
0x0082	Temporary Occupancy BI
0x0083	Tower Level Alarm BI
0x0084	Tower Level Status BI
0x0085	Tower Temp BI
0x0086	Tower Vibration Alarm Status BI
0x0087	Tower Level Alarm BI
0x0088	Tower Level Switch BI
0x0089	Tower Temp Switch BI
0x008A	Tower Fan Isolation Valve Status BI
0x008B	Tower Fan Maintenance Switch BI
0x008C	Tower Fan Overload BI
0x008D	Tower Fan Status BI
0x008E	Unit Enable BI
0x008F	Unit Reset BI
0x0090	Window Status BI
0x0091	Zone Sensor Temporary Occupancy BI
0x0092	Air Proving Switch BI
0x0093	Primary Heating Status BI
0x0094	Primary Cooling Status BI

Table 3.110 BI Types, Type = 0x00: Application Domain HVAC (Continued)

Index	Application Usage
0x0095 -0x01FF	Reserved
0x0200 -0xFFFE	Vendor defined
0xFFFF	Other

3.14.10.19.4.2 Type = 0x01: Application Domain Security

Table 3.111 BI Types, Type = 0x01: Application Domain Security

Index	Application Usage
0x0000	Glass Breakage Detection
0x0001	Intrusion Detection
0x0002	Motion Detection
0x0003	Glass Breakage Detection
0x0004	Zone Armed
0x0005	Glass Breakage Detection
0x0006	Smoke Detection
0x0007	Carbon Dioxide Detection
0x0008	Heat Detection
0x0009 - 0x01FF	Reserved
0x0200 - 0xFFFE	Vendor defined
0xFFFF	Other

3.14.10.19.5 Binary Output (BO) Types

Group = 0x04.

The following sub-clauses describe the values when Type = 0x00 - 0x01. Types 0x02 to 0xFE are reserved, Type = 0xFF indicates other.

Present Value = 0 represents False, Off, Normal

Present Value = 1 represents True, On, Alarm

3.14.10.19.5.1 Type = 0x00: Application Domain HVAC

Table 3.112 BO Types, Type = 0x00: Application Domain HVAC

Index	Application Usage
0x0000	2 Pipe Circulation Pump BO
0x0001	2 Pipe Valve BO
0x0002	2 Pipe Valve Command BO
0x0003	Boiler BO
0x0004	Boiler Isolation Valve BO
0x0005	Boiler Pump BO
0x0006	Box Cooling 2 Position BO
0x0007	Box Heating 2 Position BO
0x0008	Box Heating Enable BO
0x0009	Box Heating Stage 1 BO
0x000A	Box Heating Stage 2 BO
0x000B	Box Heating Stage 3 BO
0x000C	Chiller 1 Isolation Valve BO
0x000D	Chiller BO
0x000E	Chiller Chilled Water Isolation Valve BO
0x000F	Chiller Condenser Water Isolation Valve BO
0x0010	Combustion Damper BO
0x0011	Compressor Stage 1 BO
0x0012	Compressor Stage 2 BO
0x0013	Cooling Circulation Pump BO
0x0014	Cooling Stage 1 BO
0x0015	Cooling Stage 2 BO
0x0016	Cooling Stage 3 BO
0x0017	Cooling Stage 4 BO
0x0018	Cooling Stage 5 BO
0x0019	Cooling Stage 6 BO
0x001A	Cooling Stage 7 BO
0x001B	Cooling Stage 8 BO
0x001C	Cooling Valve BO

Table 3.112 BO Types, Type = 0x00: Application Domain HVAC (Continued)

Index	Application Usage
0x001D	Cooling Valve Command BO
0x001E	Chilled Water Pump BO
0x001F	Economizer Enable BO
0x0020	Exhaust Air Damper BO
0x0021	Exhaust Fan BO
0x0022	Fan BO
0x0023	Fan Speed 1 BO
0x0024	Fan Speed 2 BO
0x0025	Fan Speed 3 BO
0x0026	Heat Recovery Pump BO
0x0027	Heat Recovery Valve BO
0x0028	Heat Recovery Wheel BO
0x0029	Heating Stage 1 BO
0x002A	Heating Stage 2 BO
0x002B	Heating Stage 3 BO
0x002C	Heating Valve BO
0x002D	Heating Valve Command BO
0x002E	Hot Gas Bypass Valve BO
0x002F	Humidification Stage 1 BO
0x0030	Humidification Stage 2 BO
0x0031	Humidification Stage 3 BO
0x0032	Humidification Stage 4 BO
0x0033	Humidifier Enable BO
0x0034	Heat Exchanger Isolation Valve BO
0x0035	Lighting BO
0x0036	Minimum Outside Air Damper BO
0x0037	Minimum Outside Air Fan BO
0x0038	Outside Air Damper BO
0x0039	Primary Chilled Water Pump 1 BO
0x003A	Plate-and-Frame Heat Exchanger Isolation Valve BO

Table 3.112 BO Types, Type = 0x00: Application Domain HVAC (Continued)

Index	Application Usage
0x003B	Primary Hot Water Pump BO
0x003C	Primary Heat Exchange Pump BO
0x003D	Preheat Circulation Pump BO
0x003E	Preheat Enable BO
0x003F	Preheat Stage 1 BO
0x0040	Preheat Stage 2 BO
0x0041	Preheat Stage 3 BO
0x0042	Preheat Stage 4 BO
0x0043	Preheat Stage 5 BO
0x0044	Preheat Stage 6 BO
0x0045	Preheat Stage 7 BO
0x0046	Preheat Stage 8 BO
0x0047	Preheat Valve BO
0x0048	Reheat Circulation Pump BO
0x0049	Reheat Enable BO
0x004A	Reheat Stage 1 BO
0x004B	Reheat Stage 2 BO
0x004C	Reheat Stage 3 BO
0x004D	Reheat Stage 4 BO
0x004E	Reheat Stage 5 BO
0x004F	Reheat Stage 6 BO
0x0050	Reheat Stage 7 BO
0x0051	Reheat Stage 8 BO
0x0052	Relief Fan BO
0x0053	Return Fan BO
0x0054	Reversing Valve 1 BO
0x0055	Reversing Valve 2 BO
0x0056	Secondary Chilled Water Pump BO
0x0057	Secondary Hot Water Pump BO
0x0058	Secondary Heat Exchange Pump BO

Table 3.112 BO Types, Type = 0x00: Application Domain HVAC (Continued)

Index	Application Usage
0x0059	Sideloop BO
0x005A	Sideloop Stage 1 BO
0x005B	Sideloop Stage 2 BO
0x005C	Sideloop Stage 3 BO
0x005D	Sideloop Stage 4 BO
0x005E	Sideloop Stage 5 BO
0x005F	Sideloop Stage 6 BO
0x0060	Sideloop Stage 7 BO
0x0061	Sideloop Stage 8 BO
0x0062	Steam Isolation Valve BO
0x0063	Supplemental Heating 2 Position BO
0x0064	Supplemental Heating Stage 1 BO
0x0065	Supplemental Heating Valve BO
0x0066	Supplemental Heating Enable BO
0x0067	Supplemental Heating Pump BO
0x0068	Supply Fan BO
0x0069	Tower Basin Heater BO
0x006A	Tower Basin Makeup BO
0x006B	Tower Basin Heater BO
0x006C	Tower Basin Makeup BO
0x006D	Tower Isolation Valve BO
0x006E	Tower Fan BO
0x006F	Tower Fan Speed 1 BO
0x0070	Tower Fan Speed 2 BO
0x0071	Tower Fan Speed 3 BO
0x0072	Zone Heating Stage 1 BO
0x0073	Zone Heating Stage 2 BO
0x0074	Zone Heating Stage 3 BO
0x0075	Zone Heating Valve BO
0x0076	2 Pipe Circulation Pump BO

Table 3.112 BO Types, Type = 0x00: Application Domain HVAC (Continued)

Index	Application Usage
0x0077 - 0x01FF	Reserved
0x0200 - 0xFFFE	Vendor defined
0xFFFF	Other

3.14.10.19.5.2 Type = 0x02: Application Domain Security

Table 3.113 BO Types, Type = 0x02: Application Domain Security

Index	Application Usage
0x0000	Arm Disarm Command BO
0x0001	Occupancy Control BO
0x0002	Enable Control BO
0x0003	Access Control BO
0x0004 - 0x01FF	Reserved
0x0200 - 0xFFFE	Vendor defined
0xFFFF	Other

3.14.10.19.6 Binary Value (BV) Types

Group = 0x05.

The following sub-clauses describe the values when Type = 0x00. Types 0x01 to 0xFE are reserved, Type = 0xFF indicates other.

Present Value = 0 represents False, Off, Normal

Present Value = 1 represents True, On, Alarm

3.14.10.19.6.1 Type = 0x00

Table 3.114 BV Types, Type = 0x00

Index	Application Usage
0x0000 - 0x01FF	Reserved
0x0200- 0xFFFE	Vendor defined
0xFFFF	Other

3.14.10.19.7 Multistate Input (MI) Types

Group = 0x0D.

The following sub-clauses describe the values when Type = 0x00. Types 0x01 to 0xFE are reserved, Type = 0xFF indicates other.

3.14.10.19.7.1 Type = 0x00: Application Domain HVAC

Table 3.115 MI Types, Type = 0x00: Application Domain HVAC

Index	Application Usage [Number of States] States
0x0000	[3] Off, On, Auto
0x0001	[4] Off, Low, Medium, High
0x0002	[7] Auto, Heat, Cool, Off, Emergency Heat, Fan Only, Max Heat
0x0003	[4] Occupied, Unoccupied, Standby, Bypass
0x0004	[3] Inactive, Active, Hold
0x0005	[8] Auto, Warm-up, Water Flush, Autocalibration, Shutdown Open, Shutdown Closed, Low Limit, Test and Balance
0x0006	[6] Off, Auto, Heat Cool, Heat Only, Cool Only, Fan Only
0x0007	[3] High, Normal, Low
0x0008	[4] Occupied, Unoccupied, Startup, Shutdown
0x0009	[3] Night, Day, Hold
0x000A	[5] Off, Cool, Heat, Auto, Emergency Heat
0x000B	[7] Shutdown Closed, Shutdown Open, Satisfied, Mixing, Cooling, Heating, Supplemental Heat
0x000C - 0x01FF	Reserved
0x0200- 0xFFFE	Vendor defined
0xFFFF	Other

3.14.10.19.8 Multistate Output (MO) Types

Group = 0x0E.

The following sub-clauses describe the values when Type = 0x00. Types 0x01 to 0xFE are reserved, Type = 0xFF indicates other.

3.14.10.19.8.1 Type = 0x00: Application Domain HVAC

Table 3.116 MO Types, Type = 0x00: Application Domain HVAC

Index	Application Usage [Number of States] States
0x0000	[3] Off, On, Auto
0x0001	[4] Off, Low, Medium, High
0x0002	[7] Auto, Heat, Cool, Off, Emerg Heat, Fan Only, Max Heat
0x0003	[4] Occupied, Unoccupied, Standby, Bypass
0x0004	[3] Inactive, Active, Hold
0x0005	[8] Auto, Warm-up, Water Flush, Autocalibration, Shutdown Open, Shutdown Closed, Low Limit, Test and Balance
0x0006	[6] Off, Auto, Heat Cool, Heat Only, Cool Only, Fan Only
0x0007	[3] High, Normal, Low
0x0008	[4] Occupied, Unoccupied, Startup, Shutdown
0x0009	[3] Night, Day, Hold
0x000A	[5] Off, Cool, Heat, Auto, Emergency Heat
0x000B	[7] Shutdown Closed, Shutdown Open, Satisfied, Mixing, Cooling, Heating, Suppl Heat
0x000C - 0x01FF	Reserved
0x0200- 0xFFFE	Vendor defined
0xFFFF	Other

3.14.10.19.9 Multistate Value (MV) Types

Group = 0x13.

The following sub-clauses describe the values when Type = 0x00. Types 0x01 to 0xFE are reserved, Type = 0xFF indicates other.

3.14.10.19.9.1 Type = 0x00: Application Domain HVAC

Table 3.117 MV Types, Type = 0x00: Application Domain HVAC

Index	Application Usage [Number of States] States
0x0000	[3] Off, On, Auto
0x0001	[4] Off, Low, Medium, High
0x0002	[7] Auto, Heat, Cool, Off, Emerg Heat, Fan Only, Max Heat
0x0003	[4] Occupied, Unoccupied, Standby, Bypass
0x0004	[3] Inactive, Active, Hold
0x0005	[8] Auto, Warm-up, Water Flush, Autocalibration, Shutdown Open, Shutdown Closed, Low Limit, Test and Balance
0x0006	[6] Off, Auto, Heat Cool, Heat Only, Cool Only, Fan Only
0x0007	[3] High, Normal, Low
0x0008	[4] Occupied, Unoccupied, Startup, Shutdown
0x0009	[3] Night, Day, Hold
0x000A	[5] Off, Cool, Heat, Auto, Emergency Heat
0x000B	[7] Shutdown Closed, Shutdown Open, Satisfied, Mixing, Cooling, Heating, Suppl Heat
0x000C - 0x01FF	Reserved
0x0200- 0xFFFE	Vendor defined
0xFFFF	Other

All other group values are currently reserved

3.15 Commissioning Cluster

3.15.1 Overview

This cluster provides attributes and commands pertaining to the commissioning and management of ZigBee devices operating in a network.

As shown in Figure 3.4, a device will typically be commissioned using a “Commissioning Tool”. But, depending on the particular application and installation scenario, this tool may take many forms. For purposes of this document, any device that implements the client side of this cluster may be considered a commissioning tool.

As with all clusters defined in the Cluster Library and intended for use in ZigBee application profiles, a device may operate as many instances of this cluster as needed and may place them on any addressable endpoint with the exception of endpoint 0, which is reserved for the ZigBee Device Objects (ZDO) and ZigBee Device Profile (ZDP).

This cluster is exclusively used for commissioning the ZigBee stack and defining device behavior with respect to the ZigBee network. It does not apply to applications operating on those devices.

3.15.1.1 Security and Authorization

The attributes and commands covered in this cluster specification are critical to the operation of a ZigBee device. An application entity that receives a request to access the attributes of this cluster or to execute one of the commands described in sub-clause 3.15.2.3 shall determine whether the originator is authorized to make that request and whether the security processing applied to the received frame was appropriate. The method or methods whereby this is accomplished are out of the scope of this document but it is strongly recommended that Entity Authentication, as described in [B1], be used. This, and any other methods used to authorize commissioning tools and other devices acting as a client for this cluster, shall be detailed in any Application Profile documents that use it.

Similarly, it is strongly recommended that the cluster specified here be deployed only on a single device endpoint or that, at very least, all deployments of this cluster be managed by a single application object with a unitary set of security requirements etc.

3.15.2 Server

The attributes accessible on the server side of this cluster are typically attributes of the ZigBee stack, which are either described in the layer Information Base for

some stack layer, or are ZDO configuration attributes. The function of the server is to provide read/write access to these attributes and to manage changes of certain critical attributes in a way that prevents the device from getting into an inconsistent and unrecoverable state.

Thus, for example, the application entity that receives and processes commands to set attributes in the Startup Parameters attribute set shall check whether the *StartupControl* attribute has been set to a value that is inconsistent with the value of the *ExtendedPanID* attribute (see). If such a condition arises, e.g. a request is made to set the *StartupControl* attribute to 0x02, indicating network rejoin, and simultaneously to clear the *ExtendedPanID* attribute indicating an unspecified network, then an error (INCONSISTENT_STARTUP_STATE) shall be reported.

3.15.2.1 Dependencies

None.

3.15.2.2 Attributes

For convenience, the attributes defined in this specification are arranged into sets of related attributes; each set can contain up to 16 attributes. Attribute identifiers are encoded such that the most significant three nibbles specify the attribute set and the least significant nibble specifies the attribute within the set. The currently defined attribute sets are listed in .

Table 3.118 Commissioning Attribute Sets

Attribute Set Identifier	Description
0x000, 0x001	Startup Parameters
0x002	Join Parameters
0x003	End Device Parameters
0x004	Concentrator Parameters
0x001 – 0xfff	Reserved

For each of these sets, each attribute is mandatory unless specifically specified as optional in the relevant sub-clause defining it. Similarly, any default values are specified in these sub-clauses.

3.15.2.2.1 Startup Parameters Attribute Set

The Startup Parameters attribute set contains the attributes summarized in Table 3.119. Since this is a relatively large attribute set, it is actually divided, as shown in , into two subsets with different identifiers, 0x001 and 0x002, one of which contains parameters that are strictly related to the behavior of the APS and NWK

layers and the other of which contains parameters that are related to security. Taken together these are known as the Startup Parameters attribute set.

These are application attributes and, as such, are sent, received and managed by application entities. However, except where otherwise noted, each of them corresponds to, and is intended to provide a value for a particular stack attribute that controls the startup behavior of the stack. The ZigBee specification describes a schematic startup procedure (see [B1]), which governs the order and manner in which these stack attributes must be used in order to gain access to a network or form a new network. This procedure should run when a device starts up, but may also run without an actual restart as part of the ongoing operation of the device.

The Restart Device command (see) provides a means whereby a set of Startup Parameters - the “current” Startup Parameters attribute set - stored at the application layer, can be installed in the stack and put into force by executing the startup procedure described above and in the specification. A change to one of the attributes contained in this set, e.g. the *ShortAddress* attribute, does not immediately result in a change to the underlying stack attribute. The attribute set will be installed on receipt of a Restart Device command.

Note that the attributes in this set are mutually interdependent and must be taken as a whole. One consequence of this is that, while there are no explicit requirements with regard to storage class for these attributes, implementers must carefully consider whether to make a particular attribute non-volatile or static in order to prevent inconsistencies in the attribute set after an unintentional processor restart. Another consequence is that, wherever possible, startup attributes should be written atomically using a single write attributes command frame.

Table 3.119 Attributes of the Startup Parameters Attribute Set

Identifier	Name	Type	Range
0x0000	*ShortAddress*	Unsigned 16-bit integer	0x0000 – 0xfff7
0x0001	*ExtendedPANId*	IEEE Address	0x0000000000000000 - 0xfffffffffffffffe
0x0002	*PANId*	Unsigned 16-bit integer	0x0000 - 0xffff
0x0003	*Channelmask*	32-bit bitmap	Any valid IEEE 802.15.4 channel mask (see [B6]).
0x0004	*ProtocolVersion*	Unsigned 8-bit integer	0x02
0x0005	*StackProfile*	Unsigned 8-bit integer	0x01 - 0x02
0x0006	*StartupControl*	8-bit enumeration	0x00 - 0x03
0x0010	*TrustCenterAddress*	IEEE Address	Any valid IEEE Address

Table 3.119 Attributes of the Startup Parameters Attribute Set (Continued)

Identifier	Name	Type	Range
0x0011	*TrustCenterMasterKey*	128-bit Security Key	Any 128-bit value
0x0012	*NetworkKey*	128-bit Security Key	Any 128-bit value
0x0013	*UseInsecureJoin*	Boolean	FALSE/TRUE
0x0014	*PreconfiguredLinkKey*	128-bit Security Key	Any 128-bit value
0x0015	*NetworkKeySeqNum*	Unsigned 8-bit integer	0x00 - 0xff
0x0016	*NetworkKeyType*	8-bit enumeration	Any valid key type value
0x0017	*NetworkManagerAddress*	Unsigned 16-bit integer	Any valid network address

Except where specifically noted, an implementer of this cluster shall provide read access to all attributes of the Startup Parameters attribute set. However, if an attempt is made to read an attribute that may not be read, a WRITE_ONLY status value shall be returned (see Table 2.17).

Even in cases where the commissioning cluster is a mandatory part of a given application profile, an implementer is not required to provide write access for all attributes. If write access is not provided, it is assumed that the implementer has some other preferred, generally out-of-band, method for setting the value of the underlying stack attribute, and that the value returned on read reflects the actual value in use. If an attempt is made to write to such an attribute, a DEFINED_OUT_OF_BAND status value shall be returned (see Table 2.17).

3.15.2.2.1.1 *ShortAddress* Attribute

The *ShortAddress* attribute contains the intended 16-bit network address of the device. This attribute corresponds to the *nwkShortAddress* attribute of the NIB (see [B1]).

The default value is the value stored in the *nwkShortAddress* attribute of the NIB. When this attribute is not set as part of the Restart Device Request command, this default value ensures that the previous short address is preserved. This makes it possible for a device to preserve its short address after being commissioned.

Stack profile compatibility for this attribute is described in Table .

Table 3.120 Stack Profile Compatibility for the *ShortAddress* Attribute

StackProfile **Value**	**Supported**	**Comment**
0x01	No	Under the ZigBee stack profile a ZigBee router or device shall obtain a network address from its parent at network formation time.
0x02	Yes	Under the ZigBee PRO stack profile and stochastic addressing a device may, under certain circumstances, generate its own network address and keep it through the joining process (see [B1]). In this case, it may make sense for that address to be provided by a tool if, for example, this will reduce the likelihood of address conflicts.

3.15.2.2.1.2 *ExtendedPANId* Attribute

The *ExtendedPANId* attribute holds the extended PAN Id of the network of which the device should be a member. See 3.15.2.2.1.7 for usage details.

The default value of 0xffffffffffffffff indicates an unspecified value. In the case where a device is required to join a commissioning network on startup, this attribute may be set, under application control, to the global commissioning EPID (see 3.15.4.1).

Depending in the value of the *StartupControl* attribute, this attribute may correspond to the *nwkExtendedPANID* attribute of the NIB (see [B1]) or the *apsUseExtendedPANID* attribute of the AIB (see [B1]).

3.15.2.2.1.3 *PANId* Attribute

The *PANId* attribute holds the PAN Id of the network of which the device should be a member. This attribute corresponds to the *macPANId* attribute of the MAC PIB (see [B6]).

The default value of 0xffff indicates that the device has not joined a network.

Stack profile compatibility for this attribute is described in .

Table 3.121 Stack Profile Compatibility for the *PANId* Attribute

***StackProfile* Value**	**Comment**
0x01	Under the ZigBee stack profile, The ZigBee coordinator shall select an appropriate PANId at network formation time. In this case the value of the PANId attribute may be used. A ZigBee router or ZigBee end device shall obtain a PANId from its parent at network join time. In this case, the value of the PANId attribute shall be ignored.
0x02	Under the ZigBee PRO stack profile a ZigBee router or end device that has the StartupControl attribute equal to 0x00, must have the PANId attribute set to the correct value since it has no other way of obtaining it.

3.15.2.2.1.4 *ChannelMask* Attribute

The *ChannelMask* attribute is an IEEE802.15.4 channel mask, see [B6], containing the set of channels the device should scan as part of the network join or formation procedures. This attribute corresponds to the *apsChannelMask* attribute of the AIB (see [B1]).

The default value corresponds to all channels supported by the device.

3.15.2.2.1.5 *ProtocolVersion* Attribute

The *ProtocolVersion* attribute is used to select the current protocol version for a device that supports multiple versions of the ZigBee specification.

This attribute is optional. A device may support a single protocol version or multiple protocol versions at the option of the implementer.

Currently only one value, 0x02 denoting ZigBee 2006 and later, is supported. The default value shall be the protocol version supported by the application if only one protocol version is supported. Should more than one protocol version be supported, the default value may be any of the protocol versions supported.

The *ProtocolVersion* attribute corresponds to a NWK layer constant, *nwkcProtocolVersion*, which is defined as a constant because most implementations will support only a single ZigBee protocol version. In this case, the attribute will be read-only. However, there is nothing to prevent a device with sufficient resources from supporting more than one ZigBee protocol version under control of the commissioning cluster.

3.15.2.2.1.6 *StackProfile* Attribute

The *StackProfile* attribute is used to select the stack profile for the device.

This attribute is optional. A device may only support one stack profile.

Supported values include:

- 0x01: ZigBee Stack profile
- 0x02: ZigBee PRO Stack Profile

This attribute corresponds to the *nwkStackProfile* attribute of the NIB (see [B1]). The default value shall be the stack profile supported by the application if only one stack profile is supported. Should more than one stack profile be supported, the default value may be any of the stack profiles supported.

3.15.2.2.1.7 *StartupControl* Attribute

The *StartupControl* attribute is an enumerated type that determines how certain other parameters are to be used. Values for this attribute and interaction with other attributes are shown in . If an attribute appears in the “required attributes” column this indicates that this attribute must be set to a value that is valid for the intended operational network in order for this StartupControl attribute value to be used. Note that in some cases the default value may be sufficient.

If an attribute appears in the “optional attributes” column it means that the attribute value will affect startup or operation under the given attribute set but that any value, including the default, is a valid value. If an attribute appears in the “ignored attributes” column it means that the value of this attribute has no affect

on device startup when the StartupControl attribute value in the “value” column is in force.

Table 3.122 *StartupControl* Attribute Usage

Value	Description	Required Attributes	Optional Attributes	Ignored Attributes
0x00	Indicates that the device should consider itself part of the network indicated by the *ExtendedPANId* attribute. In this case it will not perform any explicit join or rejoin operation.	*ShortAddress, ExtendedPANId, PANId, TrustCenterAddress, NetworkKey, NetworkKeySeqNum, NetworkKeyType*	*ChannelMask, UseInsecureJoin, NetworkManagerAddress, TrustCenterMasterKey* (required for Stack Profile 2, optional for Stack Profile 1), *PreconfiguredLinkKey*	-
0x01	Indicates that the device should form a network with extended PAN ID given by the *ExtendedPANId* attribute. The AIB attribute *apsDesignatedCoordinator* (see [B1]) shall be set to TRUE in this case.	*ExtendedPANId*	*PANId, ChannelMask, NetworkManagerAddress, NetworkKey, NetworkKeyType, TrustCenterAddress*	*ShortAddress, UseInsecureJoin NetworkKeySeqNum, TrustCenterMasterKey, PreconfiguredLinkKey*
0x02	Indicates that the device should rejoin the network with extended PAN ID given by the *ExtendedPANId* attribute. The AIB attribute apsDesignatedCoordinator (see [B1]) shall be set to FALSE in this case.	*ExtendedPANId*	*ShortAddress, ChannelMask, UseInsecureJoin, NetworkKey, NetworkKeySeqNum, NetworkKeyType TrustCenterAddress, TrustCenterMasterKey, NetworkManagerAddress, PreconfiguredLinkKey*	*PANId*
0x03	Indicates that the device should start “from scratch” and join the network using (unsecured) MAC association. The AIB attribute apsDesignatedCoordinator (see [B1]) shall be set to FALSE in this case.	-	*ExtendedPANId, ChannelMask, PreconfiguredLinkKey*[a]	*ShortAddress, UseInsecureJoin, PANId, TrustCenterAddress, NetworkKey, NetworkKeySeqNum, NetworkKeyType, NetworkManagerAddress , TrustCenterMasterKey*

a. CCB 1377

Note that these values control the execution of the device startup procedure as specified in [B1], sub-clause 2.5.5.5.6.2. See this sub-clause for a detailed description of the operation of this procedure.

The default value of the StartupControl attribute for an un-commissioned device is 0x03.

Stack profile compatibility for this attribute is shown in Table 3.123.

Table 3.123 Stack Profile Compatibility for the *StartupControl* Attribute

StackProfile Value	StartupControl Value		Comment
	Mandatory	Optional	
0x01	0x01 0x03 [a]	0x02	ZigBee networks use tree-structured address assignment and must form, at startup, from the ZigBee coordinator. The "mode" implied by *StartupControl* = 0 in which a device is essentially preconfigured to run on a network without having to explicitly join in order to get an address or PAN Id is not supported.
0x02	0x01 0x03	0x00 0x02	StartupControl = 0 is supported under the ZigBee Pro stack profile.

a. CCB 1377

Note: An implementation shall return an error code of INVALID_VALUE when a client attempts to write an unsupported *StartupContol* value.[9]

3.15.2.2.1.8 TrustCenterAddress Attribute

The trust center address to use when performing security operations on the network whose extended PAN ID is given by the *ExtendedPANId* attribute is, in turn, given by the *TrustCenterAddress* attribute.

This attribute corresponds to the *apsTrustCenterAddress* attribute of the AIB (see [B1].

The default value of 0x0000000000000000 indicates unspecified.

9. CCB 1377

3.15.2.2.1.9 *TrustCenterMasterKey* Attribute

This attribute holds the trust center master key to use during key establishment with the TC of the network with the extended PAN ID given by the *ExtendedPANId* attribute.

The default value, i.e. a 128-bit value containing all zeros, indicates that the key is unspecified.

This attribute corresponds to the MasterKey element of the key-pair set from the *apsDeviceKeyPairSet* attribute of the AIB for which the DeviceAddress element corresponds to the value of the *TrustCenterAddress* attribute. (see [B1]).

3.15.2.2.1.10 *NetworkKey* Attribute

This attribute supplies the NWK key to use when communicating with the network specified by the *ExtendedPANId* attribute. The default value, i.e. a 128-bit value containing all zeros, indicates that the key is unspecified.

This attribute corresponds to the active key from the *nwkSecurityMaterialSet* attribute of the NIB (see [B1].

3.15.2.2.1.11 *UseInsecureJoin* Attribute

This attribute is a Boolean flag that enables the use of unsecured join as a fallback case at startup time. It corresponds to the Boolean AIB attribute *apsUseInsecureJoin* (see [B1]. The default value is TRUE.

3.15.2.2.1.12 *PreconfiguredLinkKey* Attribute

The preconfigured link key is the key between the device and the trust center. The default value, i.e. a 128-bit value containing all zeros, indicates that the key is unspecified.

This attribute corresponds to the LinkKey element of the Key-Pair descriptor contained in the *apsDeviceKeyPairSet* attribute of the AIB (see [B1]).

3.15.2.2.1.13 *NetworkKeySeqNum* Attribute

This attribute sets the network key's sequence number. The default value is 0x00.

This attribute corresponds to the value of the *nwkActiveKeySeqNumber* attribute of the NIB (see [B1]).

3.15.2.2.1.14 *NetworkKeyType* Attribute

This attribute sets the network key's type. It corresponds to the value of the KeyType element of the current security material descriptor corresponding

to the Trust Center found in the *nwkSecurityMaterialSet* attribute of the NIB (see [B1]).

The default value is 0x01 when the StackProfile is 0x01 and 0x05 when the StackProfile is 0x02.

3.15.2.2.1.15 *NetworkManagerAddress* Attribute

This attribute sets the address of the Network Manager. It corresponds to the value of the *nwkManagerAddr* attribute of the NIB (see [B1]).

The default value is 0x0000 indicating that, by default, the Network Manager is on the ZigBee coordinator.

3.15.2.2.2 Join Parameters Attribute Set

The Join Parameters attribute set contains the attributes summarized in .

These attributes control the details of the network joining process. Each of them, as described below, corresponds to a ZDO configuration attribute, the function and use of which is described in the ZigBee specification (see [B1]).

Table 3.124 Attributes of the Join Parameters Attribute Set

Identifier	Name	Type	Range
0x0020	*ScanAttempts*	Unsigned 8-bit integer	0x001 – 0xff
0x0021	*TimeBetweenScans*	Unsigned 16-bit integer	0x0001 - 0xffff
0x0022	*RejoinInterval*	Unsigned 16-bit integer	0x0001 - *MaxRejoinInterval*
0x0023	*MaxRejoinInterval*	Unsigned 16-bit integer	0x0001 - 0xffff

As with the attributes in Table 3.119, an implementer of this cluster shall provide read access to all attributes in . The implementer may provide write access. If write access is not provided, it is assumed that the implementer has some other preferred method for setting the value of the underlying stack attribute, and that the value returned on read reflects the actual value in use.

3.15.2.2.2.1 *ScanAttempts* Attribute

The *ScanAttempts* attribute determines how many scan attempts to make before selecting the ZigBee Coordinator or Router to join.

This attribute corresponds to the *:Config_NWK_Scan_Attempts* configuration attribute of the ZDO (see [B1]).

The default value for this attribute is 0x05.

3.15.2.2.2.2 *TimeBetweenScans* Attribute

The *TimeBetweenScans* attribute determines the time between each scan attempt.

This attribute corresponds to the *:Config_NWK_Time_btwn_Scans* configuration attribute of the ZDO (see [B1]).

The units of this attribute are milliseconds and the default value is 0x64.

3.15.2.2.2.3 *RejoinInterval* Attribute

The *RejoinInterval* determines the interval between attempts to rejoin the network if an end device finds itself disconnected.

This attribute corresponds to the *:Config_Rejoin_Interval* configuration attribute of the ZDO (see [B1]).

The units of this attribute are seconds and the default value is 0x3c.

3.15.2.2.2.4 *MaxRejoinInterval* Attribute

The MaxRejoinInterval attribute imposes an upper bound on the RejoinInterval parameter.

This attribute corresponds to the *:Config_Max_Rejoin_Interval* configuration attribute of the ZDO (see [B1]).

The units of this attribute are seconds and the default value is 0x0e10.

3.15.2.2.3 End Device Parameters Attribute Set

The End Device Parameters attribute set contains the attributes summarized in .

Table 3.125 Attributes of the End Device Parameters Attribute Set

Identifier	Name	Type	Range
0x0030	*IndirectPollRate*	Unsigned 16-bit integer	0x0000 – 0xffff
0x0031	*ParentRetryThreshold*	Unsigned 8-bit integer	0x00 - 0xff

As with the attributes in Table 3.119 and , an implementer of this cluster shall provide read access to all attributes in . The implementer may provide write access. If write access is not provided, it is assumed that the implementer has some other preferred method for setting the value of the underlying stack attribute, and that the value returned on read reflects the actual value in use.

3.15.2.2.3.1 *IndirectPollRate* Attribute

The *IndirectPollRate* attribute determines the rate at which a device, usually an end device, where the *macRxOnWhenIdle* attribute of the PIB has a value of FALSE, will poll for messages from its parent.

This attribute corresponds to the *:Config_NWK_IndirectPollRate* configuration attribute of the ZDO (see [B1]).

The units for this attribute are milliseconds and the default value, broad limits for which are given in [B2] and [B3], shall be determined by the relevant Application Profile document. Values assigned using this cluster should be within the given limits in order to promote correct network operation.

3.15.2.2.3.2 *ParentRetryThreshold* Attribute

The *ParentRetryThreshold* attribute determines how many times a ZigBee end device should attempt to contact its parent before initiating the rejoin process. ZigBee routers and ZigBee coordinators should return a value of 0xff for this attribute on read, and should return an error on any attempt to write it.

This attribute corresponds to the *:Config_Parent_Link_Retry_Threshold* configuration attribute of the ZDO (see [B1]).

3.15.2.2.4 Concentrator Parameters Attribute Set

The Concentrator Parameters attribute set contains the attributes summarized in .

Table 3.126 Attributes of the Concentrator Parameters Attribute Set

Identifier	Name	Type	Range
0x0040	*ConcentratorFlag*	Boolean	FALSE/TRUE
0x0041	*ConcentratorRadius*	Unsigned 8-bit integer	0x00 - 0xff
0x0042	*ConcentratorDiscoveryTime*	Unsigned 8-bit integer	0x00 - 0xff

As with the other attribute sets in this cluster, an implementer shall provide read access to all attributes in . The implementer may provide write access. If write access is not provided, it is assumed that the implementer has some other preferred method for setting the value of the underlying stack attribute, and that the value returned on read reflects the actual value in use.

3.15.2.2.4.1 *ConcentratorFlag* Attribute

The *ConcentratorFlag* attribute will configure the device to be a concentrator for the purpose of many-to-one routing. This attribute corresponds to the *nwkIsConcentrator* attribute of the NIB (see [B1]).

The default value for this attribute is FALSE.

3.15.2.2.4.2 *ConcentratorRadius* Attribute

The *ConcentratorRadius* attribute determines the hop count radius for concentrator route discoveries. This attribute corresponds to the *nwkConcentratorRadius* attribute of the NIB (see [B1]).

The default value for this attribute is 0x0f.

3.15.2.2.4.3 *ConcentratorDiscoveryTime* Attribute

Routes to the concentrator are known as inbound routes. These routes are created after the receipt of a command from the concentrator. The *ConcentratorDiscoveryTime* attribute determines the period for triggering such route creation.

This attribute corresponds to the *nwkConcentratorDiscoveryTime* attribute of the NIB (see [B1]).

The units of this attribute are seconds and the default value is 0x0000, which indicates that the discovery time is unknown and must be performed by the application.

3.15.2.3 Commands Received

The received command IDs for the commissioning cluster server are listed in Table 3.127. These commands may, in principle, be received as unicasts or as broadcasts, but application developers should be aware that, since these commands require a response, broadcasting them to a large number of devices may not be advisable.

Table 3.127 Commands Received by the Commissioning Cluster Server

Command Identifier Field Value	Description	Mandatory / Optional
0x00	Restart Device	M
0x01	Save Startup Parameters	O

Table 3.127 Commands Received by the Commissioning Cluster Server

Command Identifier Field Value	Description	Mandatory / Optional
0x02	Restore Startup Parameters	O
0x03	Reset Startup Parameters	M
0x04 – 0xff	Reserved	

In Table 3.127, if the actions associated with an optional command are not implemented, at least the relevant response command (see Table 3.129) must be returned with status UNSUP_CLUSTER_COMMAND.

3.15.2.3.1 Restart Device Command

The Restart Device command is used to optionally install a set of startup parameters in a device and run the startup procedure so as to put the new values into effect. The new values may take effect immediately or after an optional delay with optional jitter. The server will send a Restart Device Response command back to the client device before executing the procedure or starting the countdown timer required to time the delay.

3.15.2.3.1.1 Payload Format

The Restart Device command is formatted as shown in Figure 3.51.

Octets	1	1	1
Data Type	8-bit bitmap	Unsigned 8-bit integer	Unsigned 8-bit integer
Field Name	Options	Delay	Jitter

Figure 3.51 Format of the Restart Device Command Payload

Bits: 0...2	3	4...7
Startup Mode	Immediate	Reserved

Figure 3.52 Format of the Options Field

The Startup Mode sub-field of the options field is 3 bits in length and shall take one of the non-reserved values from .

Table 3.128 Startup Mode Sub-field Values

Field value	Description
0b000	Restart the device using, i.e. installing, the current set of startup parameters.
0b001	Restart the device using, and not replacing, the current state of the device, i.e. the current set of stack attributes.
0b010...0b111	Reserved

The Immediate sub-field of the options field is 1 bit in length. If this sub-field has a value of 1 then the device is to execute the restart either immediately on receipt of the Restart Device Request frame, if the value of the delay field is 0, or immediately after the prescribed delay and jitter has transpired if not. If the immediate sub-field has a value of 0, then the device may wait to restart until after the prescribed delay and jitter, if any, have transpired but may also wait for a “convenient” moment, e.g. until pending frames have been transmitted, to actually perform the restart.

The delay field is one octet in length and gives a delay in seconds, in the range [0…255], after which the startup procedure is to be invoked.

The jitter field is one octet in length and specifies a random jitter range. While possible field values fall in the interval [0…255], the actual jitter, in milliseconds, that should be added to the delay, given in seconds, in the delay field should be:

RAND(<jitter field contents> * 80) ms.

Where RAND(X) returns a random number in the interval [0…X].

3.15.2.3.1.2 Effect on Receipt

On receipt of the Restart Device command, the application checks the current startup attribute set for consistency. If the attribute set is incorrect or inconsistent, processing of the command is terminated and a Restart Device Response command is returned to the sender of the request with a status value of INCONSISTENT_STARTUP_STATE. Otherwise, the application sends a Restart Device Response command to the sender of the request with a status value of SUCCESS, then leaves the current network, installs the current startup attribute set, if the startup mode sub-field of the options field has a value of 0b00, and runs the restart procedure after the given delay and jitter have transpired.

3.15.2.3.2 Save Startup Parameters Command

In addition to the current set of startup parameters, which every device implementing the commissioning cluster must maintain, a device may store and maintain up to 256 sets of startup attributes. The Save Startup Parameters Request command allows for the current attribute set to be stored under a given index. Note that while the startup attribute set index is 8 bits, allowing for as many as 256 attribute sets, the actual number of attribute sets will typically be much smaller.

While storage of additional startup attribute sets is optional, a device that chooses to store additional startup attribute sets must store them in such a way that they are non-volatile.

3.15.2.3.2.1 Payload Format

The Save Startup Parameters command is formatted as shown in Figure 3.53.

Octets	1	1
Data Type	8-bit bitmap	Unsigned 8-bit integer
Field Name	Options (Reserved)	Index

Figure 3.53 Format of Save Startup Parameters Command Payload

The Options field is one octet in length and is reserved.

The Index field is one octet in length and gives an index under which the current startup parameter attribute set is to be saved.

3.15.2.3.2.2 Effect on Receipt

On receipt of the Save Startup Parameters command, the application shall check the value of the index field of the command payload. If the index field has a value that is equal to an index under which a set of startup parameters has already been saved then the current startup parameters attribute set is simply saved in place of the previously saved set and a Save Startup Parameters Response command is sent back to the sender of the request with a status value of SUCCESS.

If the value of the index field is such that no startup parameters attribute set has been saved under that index then the application shall check that there is storage capacity to save another attribute set. If there is capacity then the current startup parameters attribute set shall be stored under the index given in the index field such that it may be restored at a future time in response to the receipt of a Restore

Startup Parameters Request command carrying the same index. A Save Startup Parameters Response command with status value of SUCCESS is then sent as described above.

If there is not storage capacity, then a save Startup Parameters Response command is sent back to the sender of the request with a status INSUFFICIENT_SPACE.

3.15.2.3.3 Restore Startup Parameters Command

A device that implements the optional Save Startup Parameters command shall also implement the Restore Startup Parameters Request command (and vice-versa). This command allows a saved startup parameters attribute set to be restored to current status overwriting whatever was there previously.

3.15.2.3.3.1 Payload Format

The Restore Startup Parameters command is formatted as shown in Figure 3.54.

Octets	1	1
Data Type	8-bit bitmap	Unsigned 8-bit integer
Field Name	Options (Reserved)	Index

Figure 3.54 Restore Startup Parameters Command Payload

The options field is one octet in length and is reserved.

The index field is one octet in length and gives the index of the saved startup parameter attribute set to be restored to current status.

3.15.2.3.3.2 Effect on Receipt

On receipt of the Restore Startup Parameters command, the application shall check the value of the index field of the command payload. If the index field has a value that is equal to an index under which a startup parameters attribute set has been saved then that attribute set is copied into the current startup parameters attribute set overwriting whatever was there and a Restore Startup Parameters Response command is sent back to the sender of the request with a status value of SUCCESS. If the value of the index field is such that no startup parameters attribute set has been saved under that index then a Restore Startup Parameters Response command is sent back to the sender of the request with a status value of INVALID_FIELD.

3.15.2.3.4 Reset Startup Parameters Command

This command allows current startup parameters attribute set and one or all of the saved attribute sets to be set to default values. There is also an option for erasing the index under which an attribute set is saved thereby freeing up storage capacity.

3.15.2.3.4.1 Payload Format

The Reset Startup Parameters command is formatted as shown in Figure 3.55.

Octets	1	1
Data Type	8-bit bitmap	Unsigned 8-bit integer
Field Name	Options	Index

Figure 3.55 Format of Reset Startup Parameters Command Payload

The Options field is formatted as shown in Figure 3.56.

Bits: 0	1	2	3...7
Reset Current	Reset All	Erase Index	Reserved

Figure 3.56 Format of the Options Field

The Reset Current sub-field of the options field is 1 bit in length. If it has a value of 1 then all attributes in the current startup parameters attribute set shall be reset to their default values. Otherwise the current startup parameters attribute set shall remain unchanged.

The Reset All sub-field of the options field is 1 bit in length. If it has a value of 1 then all attributes of all saved startup parameter attribute sets shall be reset to their default values. Otherwise, all attributes of the saved attribute set with an index given by the value of the index field shall be set to their default values

The Erase Index sub-field of the options field is 1 bit in length. If it has a value of 1 then the index under which a saved attribute set has been saved shall be cleared as well, essentially freeing the storage associated with that index.

The Index field is one octet in length and gives the index of a saved startup parameter attribute set. The value of this field is ignored if either the reset all sub-field or the reset current sub-field of the options field have a value of 1.

3.15.2.3.4.2 Effect on Receipt

On receipt of the Reset Startup Parameters Request command the application interprets the options field and index field as described in sub-clause 3.15.2.3.4.1 and acts accordingly. The Reset Startup Parameters Response command sent back to the sender of the request shall always have a status value of SUCCESS

3.15.2.4 Commands Generated

The command IDs for the commands generated by the commissioning cluster server are listed in Table 3.129.

Table 3.129 Commands Generated by the Commissioning Cluster Server

Command Identifier Field Value	Description	Mandatory / Optional
0x00	Restart Device Response Response	M
0x01	Save Startup Parameters Response	M
0x02	Restore Startup Parameters Response	M
0x03	Reset Startup Parameters Response	M
0x04 – 0xff	Reserved	

These commands should always be issued as unicasts.

3.15.2.4.1 Payload Format

All response commands emitted by the server have the same payload format as shown in Figure 3.57.

Octets	1
Data Type	8-bit enumeration
Field Name	Status

Figure 3.57 Format of Reset Startup Parameters Command Payload

Status values are chosen from the set of non-reserved values shown in Table 2.17.

3.15.2.4.2 Effect on Receipt

On receipt of one of the response commands shown in Table 3.129, the client is made aware that the server has received the corresponding request and is informed of the status of the request.

3.15.3 Client

The commissioning cluster client (e.g. implemented on a commissioning tool) manages the attributes described above on a remote device and sends the Restart Device command as necessary.

3.15.3.1 Dependencies

None.

3.15.3.2 Attributes

The client cluster has no attributes.

3.15.3.3 Commands Received

The client receives the cluster specific commands generated by the server (see 3.15.2.4).

3.15.3.4 Commands Generated

The client generates the cluster specific commands received by the server, as required by the application. See 3.15.2.3.

3.15.4 ZigBee Alliance EUI-64s

In order to assist in ensuring that commissioning can be achieved in an interoperable environment while minimizing the possibility of interference from existing or future ZigBee and 802.15.4 networks and devices, the ZigBee Alliance has reserved a range of IEEE-defined 64-bit extended unique identifiers (EUI-64s) for use as Extended PAN IDs. The reserved range is as follows:

- 00-50-C2-77-10-00-00-00 is the global commissioning EPID
- 00-50-C2-77-10-00-00-01 to 00-50-C2-77-10-00-FF-FF are EUI-64s reserved for other commissioning use

3.15.4.1 Global Commissioning EPID

The global commissioning EPID is intended to serve as a single EUI-64 to be used by any ZigBee application for the purpose of commissioning. It is recommended that profile and application developers that require interoperability between products offered by different OEMs incorporate this global commissioning EPID within their respective application profiles as the EPID that devices attempt to join when they are first turned on straight “out-of-the-box”.

This global commissioning EPID provides a guarantee that devices will join a specific network for commissioning purposes. As part of commissioning, devices are then provided with a startup attribute set (SAS) that ensures that they join a network other than this global commissioning network. These SASs may be provided over-the-air using the commissioning cluster or some other out-of-band method.

It is also recommended that this global commissioning EPID be used only for commissioning, and especially not for ongoing operational use. Commissioning networks formed using the global commissioning EPID should be temporary and such networks should be stopped upon completion of commissioning to minimize the possibility of such networks interfering with other attempts at forming commissioning networks.

3.15.4.2 EUI-64s Reserved for Other Uses

Additional EUI-64s have been reserved for other use by the Alliance. At this point, their intended usage has not been specified. These identifiers should not be used without prior agreement with the ZigBee Alliance. It is recommended that if a profile or application developer requires the use of these additional EUI-64s, they should contact the Core Stack Group (CSG) within the ZigBee Alliance.

CHAPTER

4

MEASUREMENT AND SENSING SPECIFICATION

4.1 General Description

4.1.1 Introduction

The clusters specified in this document are generic measurement and sensing interfaces that are sufficiently general to be of use across a wide range of application domains.

4.1.2 Cluster List

This section lists the clusters specified in this document, and gives examples of typical usage for the purpose of clarification.

The clusters specified in the Measurement and sensing functional domain are listed in Table 4.1 to Table 4.3.

4.1.2.1 Illuminance Measurement and Level Sensing

Table 4.1 Illuminance Measurement and Level Sensing Clusters

Cluster Name	Description
Illuminance measurement	Attributes and commands for configuring the measurement of illuminance, and reporting illuminance measurements
Illuminance level sensing	Attributes and commands for configuring the sensing of illuminance levels, and reporting whether illuminance is above, below, or on target.

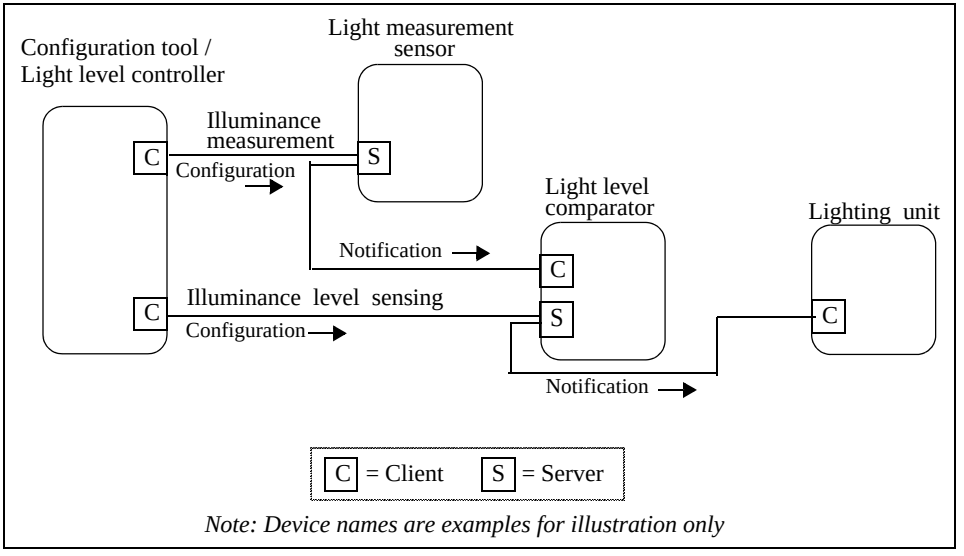

Figure 4.1 Typical Usage of Illuminance Measurement and Level Sensing Clusters

4.1.2.2 Temperature, Pressure and Flow Measurement

Table 4.2 Pressure and Flow Measurement Clusters

Cluster Name	Description
Temperature measurement	Attributes and commands for configuring the measurement of temperature, and reporting temperature measurements
Pressure measurement	Attributes and commands for configuring the measurement of pressure, and reporting pressure measurements
Flow measurement	Attributes and commands for configuring the measurement of flow, and reporting flow rates
Relative Humidity measurement	Attributes and commands for configuring the measurement of relative humidity, and reporting relative humidity measurements

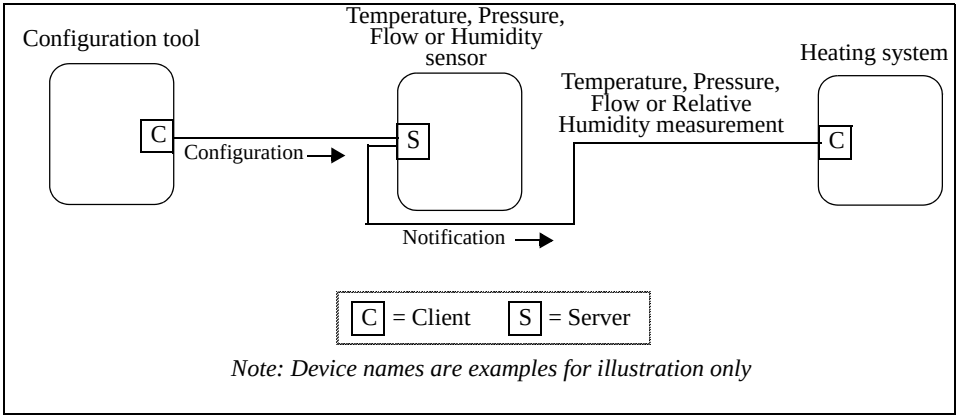

Figure 4.2 Typical Usage of Temperature, Pressure and Flow Measurement Clusters

4.1.2.3 Occupancy Sensing

Table 4.3 Occupancy Sensing Clusters

Cluster Name	Description
Occupancy sensing	Attributes and commands for configuring occupancy sensing, and reporting occupancy status

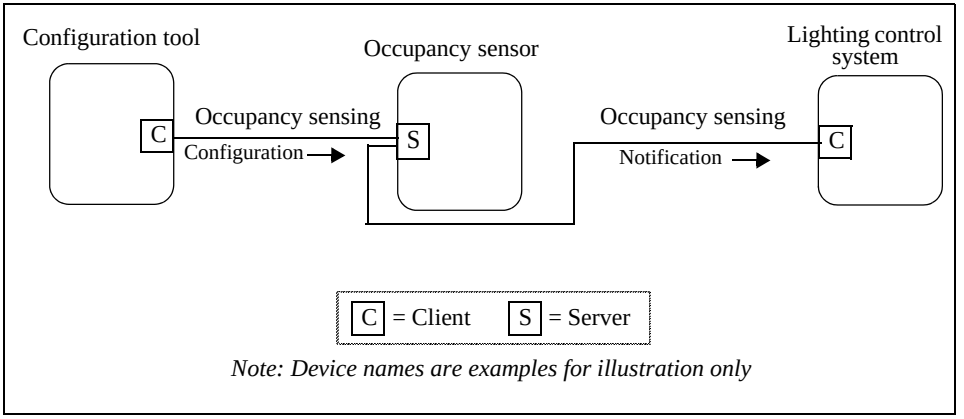

Figure 4.3 Typical Usage of Occupancy Sensing Cluster

4.2 Illuminance Measurement Cluster

4.2.1 Overview

The server cluster provides an interface to illuminance measurement functionality, including configuration and provision of notifications of illuminance measurements.

4.2.2 Server

4.2.2.1 Dependencies

None

4.2.2.2 Attributes

For convenience, the attributes defined in this specification are arranged into sets of related attributes; each set can contain up to 16 attributes. Attribute identifiers

are encoded such that the most significant nibble specifies the attribute set and the least significant nibble specifies the attribute within the set. The currently defined attribute sets are listed in Table 4.4.

Table 4.4 Illuminance Measurement Attribute Sets

Attribute Set Identifier	Description
0x000	Illuminance Measurement Information
0x001 – 0xfff	Reserved

4.2.2.2.1 Illuminance Measurement Information Attribute Set

The Illuminance Measurement Information attribute set contains the attributes summarized in Table 4.5.

Table 4.5 Illuminance Measurement Information Attribute Set

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0000	*MeasuredValue*	16-bit unsigned integer	*MinMeasuredValue* to *MaxMeasuredValue*	Read only	0	M
0x0001	*MinMeasuredValue*	16-bit unsigned integer	0x0002 – 0xfffd	Read only	-	M
0x0002	*MaxMeasuredValue*	16-bit unsigned integer	0x0001 – 0xfffe	Read only	-	M
0x0003	*Tolerance*	16-bit unsigned integer	0x0000 – 0x0800	Read only	-	O
0x0004	*LightSensorType*	8-bit enumeration	0x00 – 0xff	Read only	-	O

4.2.2.2.1.1 *MeasuredValue* Attribute

MeasuredValue represents the Illuminance in Lux (symbol lx) as follows:

$MeasuredValue = 10{,}000 \times \log_{10} \text{Illuminance} + 1$

Where 1 lx <= Illuminance <=3.576 Mlx, corresponding to a *MeasuredValue* in the range 1 to 0xfffe.

The following special values of *MeasuredValue* apply.

0x0000 indicates a value of Illuminance that is too low to be measured.

0xffff indicates that the Illuminance measurement is invalid.

MeasuredValue is updated continuously as new measurements are made.

4.2.2.2.1.2 *MinMeasuredValue* Attribute

The *MinMeasuredValue* attribute indicates the minimum value of *MeasuredValue* that can be measured. A value of 0xffff indicates that this attribute is not defined.

4.2.2.2.1.3 *MaxMeasuredValue* Attribute

The *MaxMeasuredValue* attribute indicates the maximum value of *MeasuredValue* that can be measured. A value of 0xffff indicates that this attribute is not defined.

MaxMeasuredValue shall be greater than *MinMeasuredValue.*

MinMeasuredValue and *MaxMeasuredValue* define the range of the sensor.

4.2.2.2.1.4 *Tolerance* Attribute

The *Tolerance* attribute indicates the magnitude of the possible error that is associated with *MeasuredValue* . The true value is located in the range (*MeasuredValue – Tolerance*) to (*MeasuredValue + Tolerance*).

4.2.2.2.1.5 *LightSensorType* Attribute

The *LightSensorType* attribute specifies the electronic type of the light sensor. This attribute shall be set to one of the non-reserved values listed in Table 4.6.

Table 4.6 Values of the *LightSensorType* Attribute

Attribute Value	Description
0x00	Photodiode
0x01	CMOS
0x02 – 0x3f	Reserved
0x40 – 0xfe	Reserved for manufacturer specific light sensor types
0xff	Unknown

4.2.2.3 Commands Received

No cluster specific commands are received by the server cluster.

4.2.2.4 Commands Generated

No cluster specific commands are generated by the server cluster.

4.2.2.5 Attribute Reporting

This cluster shall support attribute reporting using the Report Attributes command and according to the minimum and maximum reporting intervals and reportable change settings described in the ZCL Foundation specification (see 2.4.7). The following attributes shall be reported:

MeasuredValue

Tolerance

4.2.3 Client

4.2.3.1 Dependencies

None.

4.2.3.2 Attributes

The Client cluster has no attributes.

4.2.3.3 Commands Received

No cluster specific commands are received by the client cluster.

4.2.3.4 Commands Generated

No cluster specific commands are generated by the client cluster.

4.3 Illuminance Level Sensing Cluster

4.3.1 Overview

The server cluster provides an interface to illuminance level sensing functionality, including configuration and provision of notifications of whether the illuminance is within, above or below a target band.

4.3.2 Server

4.3.2.1 Dependencies

None.

4.3.2.2 Attributes

For convenience, the attributes defined in this specification are arranged into sets of related attributes; each set can contain up to 16 attributes. Attribute identifiers are encoded such that the most significant three nibbles specify the attribute set and the least significant nibble specifies the attribute within the set. The currently defined attribute sets are listed in Table 4.7.

Table 4.7 Illuminance Level Sensing Attribute Sets

Attribute Set Identifier	Description
0x000	Illuminance Level Sensing Information
0x001	Illuminance Level Sensing Settings
0x002 – 0xfff	Reserved

4.3.2.3 Illuminance Level Sensing Information Attribute Set

The light sensor configuration attribute set contains the attributes summarized in Table 4.8.

Table 4.8 Illuminance Level Sensing Information Attribute Set

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0000	*LevelStatus*	8-bit enumeration	0x00 – 0xfe	Read only	-	M
0x0001	*LightSensorType*	8-bit enumeration	0x00 – 0xfe	Read only	-	O

4.3.2.3.1 *LevelStatus* Attribute

The *LevelStatus* attribute indicates whether the measured illuminance is above, below, or within a band around *IlluminanceTargetLevel* (see 4.3.2.4.1). It may have any non-reserved value shown in Table 4.9.

Table 4.9 Values of the *LevelStatus* Attribute

Attribute Value	Description
0x00	Illuminance on target
0x01	Illuminance below target
0x02	Illuminance above target
0x03 – 0xff	Reserved

4.3.2.3.2 *LightSensorType* Attribute

The *LightSensorType* attribute specifies the electronic type of the light sensor. This attribute shall be set to one of the non-reserved values listed in Table 4.10.

Table 4.10 Values of the *LightSensorType* Attribute

Attribute Value	Description
0x00	Photodiode
0x01	CMOS
0x02 – 0x3f	Reserved
0x40 – 0xfe	Reserved for manufacturer specific light sensor types
0xff	Unknown

4.3.2.4 Illuminance Level Sensing Settings Attribute Set

The light sensor configuration attribute set contains the attributes summarized in Table 4.11.

Table 4.11 Illuminance Level Sensing Settings Attribute Set

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0010	*IlluminanceTargetLevel*	Unsigned 16-bit integer	0x0000 – 0xfffe	Read/ Write	-	M

4.3.2.4.1 *IlluminanceTargetLevel* Attribute

The *IlluminanceTargetLevel* attribute specifies the target illuminance level. This target level is taken as the centre of a 'dead band', which must be sufficient in width, with hysteresis bands at both top and bottom, to provide reliable notifications without 'chatter'. Such a dead band and hysteresis bands must be provided by any implementation of this cluster. (N.B. Manufacturer specific attributes may be provided to configure these).

IlluminanceTargetLevel represents illuminance in Lux (symbol lx) as follows:

IlluminanceTargetLevel = 10,000 x $\log_{10}$ Illuminance

Where 1 lx <= Illuminance <=3.576 Mlx, corresponding to a *MeasuredValue* in the range 0 to 0xfffe.

A value of 0xffff indicates that this attribute is not valid.

4.3.2.5 Commands Received

No cluster specific commands are received by the server.

4.3.2.6 Commands Generated

No cluster specific commands are generated by the server cluster.

4.3.2.7 Attribute Reporting

This cluster shall support attribute reporting using the Report Attributes command and according to the minimum and maximum reporting interval settings described in the ZCL Foundation Specification (see 2.4.7). The following attribute shall be reported:

LevelStatus

4.3.3 Client

4.3.3.1 Dependencies

None.

4.3.3.2 Attributes

The client cluster has no attributes.

4.3.3.3 Commands Received

No cluster specific commands are received by the client cluster.

4.3.3.4 Commands Generated

No cluster specific commands are generated by the client cluster.

4.4 Temperature Measurement Cluster

4.4.1 Overview

The server cluster provides an interface to temperature measurement functionality, including configuration and provision of notifications of temperature measurements.

4.4.2 Server

4.4.2.1 Dependencies

None.

4.4.2.2 Attributes

For convenience, the attributes defined in this specification are arranged into sets of related attributes; each set can contain up to 16 attributes. Attribute identifiers are encoded such that the most significant nibble specifies the attribute set and the least significant nibble specifies the attribute within the set. The currently defined attribute sets are listed in Table 4.12.

Table 4.12 Temperature Measurement Attribute Sets

Attribute Set Identifier	Description
0x000	Temperature Measurement Information
0x001 – 0xfff	Reserved

4.4.2.2.1 Temperature Measurement Information Attribute Set

The Temperature Measurement Information attribute set contains the attributes summarized in Table 4.13

Table 4.13 Temperature Measurement Information Attribute Set

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0000	*MeasuredValue*	Signed 16-bit integer	*MinMeasuredValue* to *MaxMeasuredValue*	Read only	0	M
0x0001	*MinMeasuredValue*	Signed 16-bit integer	0x954d – 0x7ffe	Read only	-	M
0x0002	*MaxMeasuredValue*	Signed 16-bit integer	0x954e – 0x7fff	Read only	-	M
0x0003	*Tolerance*	Unsigned 16-bit integer	0x0000 – 0x0800	Read only	-	O

4.4.2.2.1.1 *MeasuredValue* Attribute

MeasuredValue represents the temperature in degrees Celsius as follows:

MeasuredValue = 100 x temperature in degrees Celsius.

Where -273.15°C <= temperature <= 327.67 ºC, corresponding to a *MeasuredValue* in the range 0x954d to 0x7fff. The maximum resolution this format allows is 0.01 ºC.

A *MeasuredValue* of 0x8000 indicates that the temperature measurement is invalid.

MeasuredValue is updated continuously as new measurements are made.

4.4.2.2.1.2 *MinMeasuredValue* Attribute

The *MinMeasuredValue* attribute indicates the minimum value of *MeasuredValue* that is capable of being measured. A *MinMeasuredValue* of 0x8000 indicates that the minimum value is unknown.

4.4.2.2.1.3 *MaxMeasuredValue* Attribute

The *MaxMeasuredValue* attribute indicates the maximum value of *MeasuredValue* that is capable of being measured.

MaxMeasuredValue shall be greater than *MinMeasuredValue.*

MinMeasuredValue and *MaxMeasuredValue* define the range of the sensor.

A *MaxMeasuredValue* of 0x8000 indicates that the maximum value is unknown.

4.4.2.2.1.4 *Tolerance* Attribute

The *Tolerance* attribute indicates the magnitude of the possible error that is associated with *MeasuredValue* . The true value is located in the range (*MeasuredValue – Tolerance*) to (*MeasuredValue + Tolerance*).

4.4.2.3 Commands Received

No cluster specific commands are received by the server cluster.

4.4.2.4 Commands Generated

No cluster specific commands are generated by the server cluster.

4.4.2.5 Attribute Reporting

This cluster shall support attribute reporting using the Report Attributes command and according to the minimum and maximum reporting interval and reportable change settings described in the ZCL Foundation specification (see 2.4.7). The following attributes shall be reported:

M*easuredValue*

Tolerance

4.4.3 Client

4.4.3.1 Dependencies

None.

4.4.3.2 Attributes

The Client cluster has no attributes.

4.4.3.3 Commands Received

No cluster specific commands are received by the client cluster.

4.4.3.4 Commands Generated

No cluster specific commands are generated by the client cluster.

4.5 Pressure Measurement Cluster

4.5.1 Overview

The server cluster provides an interface to pressure measurement functionality, including configuration and provision of notifications of pressure measurements.

4.5.2 Server

4.5.2.1 Dependencies

None

4.5.2.2 Attributes

For convenience, the attributes defined in this specification are arranged into sets of related attributes; each set can contain up to 16 attributes. Attribute identifiers are encoded such that the most significant three nibbles specify the attribute set

and the least significant nibble specifies the attribute within the set. The currently defined attribute sets are listed in Table 4.14.

Table 4.14 Pressure Measurement Attribute Sets

Attribute Set Identifier	Description
0x000	Pressure Measurement Information
0x001	Extended Pressure Measurement Information
0x002 – 0xfff	Reserved

4.5.2.2.1 Pressure Measurement Information Attribute Set

The Pressure Measurement Information attribute set contains the attributes summarized in Table 4.15.

Table 4.15 Pressure Measurement Information Attribute Set

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0000	*MeasuredValue*	Signed 16-bit integer	*MinMeasuredValue* to *MaxMeasuredValue*	Read only	0	M
0x0001	*MinMeasuredValue*	Signed 16-bit integer	0x8001-0x7ffe	Read only	-	M
0x0002	*MaxMeasuredValue*	Signed 16-bit integer	0x8002-0x7fff	Read only	-	M
0x0003	*Tolerance*	Unsigned 16-bit integer	0x0000 – 0x0800	Read only	-	O

This set provides for measurements with a fixed maximum resolution of 0.1 kPa.

4.5.2.2.1.1 *MeasuredValue* Attribute

MeasuredValue represents the pressure in kPa as follows:

MeasuredValue = 10 x Pressure

Where -3276.7 kPa <= Pressure <= 3276.7 kPa, corresponding to a *MeasuredValue* in the range 0x8001 to 0x7fff.

A *MeasuredValue* of 0x8000 indicates that the pressure measurement is invalid.

MeasuredValue is updated continuously as new measurements are made.

4.5.2.2.1.2 *MinMeasuredValue* Attribute

The *MinMeasuredValue* attribute indicates the minimum value of *MeasuredValue* that can be measured. A value of 0x8000 means this attribute is not defined

4.5.2.2.1.3 *MaxMeasuredValue* Attribute

The *MaxMeasuredValue* attribute indicates the maximum value of *MeasuredValue* that can be measured. A value of 0x8000 means this attribute is not defined.

MaxMeasuredValue shall be greater than *MinMeasuredValue.*

MinMeasuredValue and *MaxMeasuredValue* define the range of the sensor.

4.5.2.2.1.4 *Tolerance* Attribute

The *Tolerance* attribute indicates the magnitude of the possible error that is associated with *MeasuredValue*. The true value is located in the range (*MeasuredValue – Tolerance*) to (*MeasuredValue + Tolerance*).

4.5.2.2.2 Extended Pressure Measurement Information Attribute Set

The Extended Pressure Measurement Information attribute set contains the attributes summarized in Table 4.16.

Table 4.16 Extended Pressure Measurement Information Attribute Set

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0010	*ScaledValue*	Signed 16-bit integer	*MinScaledValue* to *MaxScaledValue*	Read only	0	O Note 1
0x0011	*MinScaledValue*	Signed 16-bit integer	0x8001-0x7ffe	Read only	-	O Note 1
0x0012	*MaxScaledValue*	Signed 16-bit integer	0x8002-0x7fff	Read only	-	O Note 1
0x0013	*ScaledTolerance*	Unsigned 16-bit integer	0x0000 – 0x0800	Read only	-	O Note 2
0x0014	*Scale*	Signed 8-bit integer	0x81 - 0x7f	Read only	-	O Note 1

Note 1: If any one of these attributes is supported, all four shall be supported.

Note 2: If this attribute is supported, all attributes in this set shall be supported.

This attribute set is optional, and allows the range and resolution of measured pressures to be extended beyond those catered for by the Pressure Measurement

Information Attribute Set, in a way fully backward compatible with devices that implement (or can read) only that attribute set.

4.5.2.2.2.1 *ScaledValue* Attribute

ScaledValue represents the pressure in Pascals as follows:

ScaledValue = 10^{Scale} x Pressure in Pa

Where -3276.7x10^{Scale} Pa <= Pressure <= 3276.7x10^{Scale} Pa corresponding to a *ScaledValue* in the range 0x8001 to 0x7fff.

A *ScaledValue* of 0x8000 indicates that the pressure measurement is invalid.

ScaledValue is updated continuously as new measurements are made.

4.5.2.2.2.2 *MinScaledValue* Attribute

The *MinScaledValue* attribute indicates the minimum value of *ScaledValue* that can be measured. A value of 0x8000 means this attribute is not defined

4.5.2.2.2.3 *MaxScaledValue* Attribute

The *MaxScaledValue* attribute indicates the maximum value of *ScaledValue* that can be measured. A value of 0x8000 means this attribute is not defined.

MaxScaledValue shall be greater than *MinScaledValue.*

MinScaledValue and *MaxScaledValue* define the range of the sensor.

4.5.2.2.2.4 *ScaledTolerance* Attribute

The *ScaledTolerance* attribute indicates the magnitude of the possible error that is associated with *ScaledValue*. The true value is located in the range (*ScaledValue – ScaledTolerance*) to (*ScaledValue + ScaledTolerance*).

4.5.2.2.2.5 *Scale* Attribute

The *Scale* attribute indicates the base 10 exponent used to obtain *ScaledValue* (see 4.5.2.2.2.1).

4.5.2.3 Commands Received

No cluster specific commands are received by the server cluster.

4.5.2.4 Commands Generated

No cluster specific commands are generated by the server cluster.

4.5.2.5 Attribute Reporting

This cluster shall support attribute reporting using the Report Attributes command and according to the minimum and maximum reporting interval and reportable change settings described in the ZCL Foundation specification (see 2.4.7). The following attributes shall be reportable:

MeasuredValue

Tolerance

If the Extended Pressure Measurement Information attribute set is implemented, it is recommended that the following attributes are also reportable:

ScaledValue

ScaledTolerance

4.5.3 Client

4.5.3.1 Dependencies

None.

4.5.3.2 Attributes

The Client cluster has no attributes.

4.5.3.3 Commands Received

No commands are received by the client cluster.

4.5.3.4 Commands Generated

No cluster specific commands are generated by the client cluster.

4.6 Flow Measurement Cluster

4.6.1 Overview

The server cluster provides an interface to flow measurement functionality, including configuration and provision of notifications of flow measurements.

4.6.2 Server

4.6.2.1 Dependencies

None

4.6.2.2 Attributes

For convenience, the attributes defined in this specification are arranged into sets of related attributes; each set can contain up to 16 attributes. Attribute identifiers are encoded such that the most significant three nibbles specify the attribute set and the least significant nibble specifies the attribute within the set. The currently defined attribute sets for are listed in Table 4.17.

Table 4.17 Flow Measurement Attribute Sets

Attribute Set Identifier	Description
0x000	Flow Measurement Information
0x001 – 0xfff	Reserved

4.6.2.2.1 Flow Measurement Information Attribute Set

The Flow Measurement Information attribute set contains the attributes summarized in Table 4.18.

Table 4.18 Flow Measurement Information Attribute Set

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0000	*MeasuredValue*	Unsigned 16-bit integer	*MinMeasuredValue* to *MaxMeasuredValue*	Read only	0	M
0x0001	*MinMeasuredValue*	Unsigned 16-bit integer	0x0000 – 0xfffd	Read only	-	M
0x0002	*MaxMeasuredValue*	Unsigned 16-bit integer	0x0001 – 0xfffe	Read only	-	M
0x0003	*Tolerance*	Unsigned 16-bit integer	0x0000 – 0x0800	Read only	-	O

4.6.2.2.1.1 *MeasuredValue* Attribute

MeasuredValue represents the flow in m^3/h as follows:

MeasuredValue = 10 x Flow

Where 0 m^3/h <= Flow <= 6,553.4 m^3/h, corresponding to a *MeasuredValue* in the range 0 to 0xfffe.

The maximum resolution this format allows is 0.1 m^3/h.

A *MeasuredValue* of 0xffff indicates that the pressure measurement is invalid.

MeasuredValue is updated continuously as new measurements are made.

4.6.2.2.1.2 *MinMeasuredValue* Attribute

The *MinMeasuredValue* attribute indicates the minimum value of *MeasuredValue* that can be measured. A value of 0xffff means this attribute is not defined

4.6.2.2.1.3 *MaxMeasuredValue* Attribute

The *MaxMeasuredValue* attribute indicates the maximum value of *MeasuredValue* that can be measured. A value of 0xffff means this attribute is not defined

MaxMeasuredValue shall be greater than *MinMeasuredValue.*

MinMeasuredValue and *MaxMeasuredValue* define the range of the sensor

4.6.2.2.1.4 *Tolerance* Attribute

The *Tolerance* attribute indicates the magnitude of the possible error that is associated with *MeasuredValue* . The true value is located in the range (*MeasuredValue – Tolerance*) to (*MeasuredValue + Tolerance*).

4.6.2.3 Commands Received

No cluster specific commands are received by the server cluster.

4.6.2.4 Commands Generated

No cluster specific commands are generated by the server cluster.

4.6.2.5 Attribute Reporting

This cluster shall support attribute reporting using the Report Attributes command and according to the minimum and maximum reporting interval and reportable change settings described in the ZCL Foundation specification (see 2.4.7). The following attributes shall be reported:

MeasuredValue
Tolerance

4.6.3 Client

4.6.3.1 Dependencies

None.

4.6.3.2 Attributes

The Client cluster has no attributes.

4.6.3.3 Commands Received

No cluster specific commands are received by the client cluster.

4.6.3.4 Commands Generated

No cluster specific commands are generated by the client cluster.

4.7 Relative Humidity Measurement Cluster

4.7.1 Overview

The server cluster provides an interface to relative humidity measurement functionality, including configuration and provision of notifications of relative humidity measurements.

4.7.2 Server

4.7.2.1 Dependencies

None

4.7.2.2 Attributes

For convenience, the attributes defined in this specification are arranged into sets of related attributes; each set can contain up to 16 attributes. Attribute identifiers are encoded such that the most significant three nibbles specify the attribute set

and the least significant nibble specifies the attribute within the set. The currently defined attribute sets are listed in Table 4.19.

Table 4.19 Relative Humidity Measurement Attribute Sets

Attribute Set Identifier	Description
0x000	Relative Humidity Measurement Information
0x001 – 0xfff	Reserved

4.7.2.2.1 Relative Humidity Measurement Information Attribute Set

The Relative Humidity Measurement Information attribute set contains the attributes summarized in Table 4.20.

Table 4.20 Attributes of the Relative Humidity Measurement Information Attribute Set

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0000	*MeasuredValue*	Unsigned 16-bit integer	*MinMeasuredValue* to *MaxMeasuredValue*	Read only	-	M
0x0001	*MinMeasuredValue*	Unsigned 16-bit integer	0x0000 – 0x270f	Read only	-	M
0x0002	*MaxMeasuredValue*	Unsigned 16-bit integer	0x0001 – 0x2710	Read only	-	M
0x0003	*Tolerance*	Unsigned 16-bit integer	0x0000 – 0x0800	Read only	-	O

4.7.2.2.1.1 *MeasuredValue* Attribute

MeasuredValue represents the relative humidity in % as follows:

MeasuredValue = 100 x Relative humidity

Where 0% <= Relative humidity <= 100%, corresponding to a *MeasuredValue* in the range 0 to 0x2710.

The maximum resolution this format allows is 0.01%.

A *MeasuredValue* of 0xffff indicates that the measurement is invalid.

MeasuredValue is updated continuously as new measurements are made.

4.7.2.2.1.2 *MinMeasuredValue* Attribute

The *MinMeasuredValue* attribute indicates the minimum value of *MeasuredValue* that can be measured. A value of 0xffff means this attribute is not defined

4.7.2.2.1.3 *MaxMeasuredValue* Attribute

The *MaxMeasuredValue* attribute indicates the maximum value of *MeasuredValue* that can be measured. A value of 0xffff means this attribute is not defined

MaxMeasuredValue shall be greater than *MinMeasuredValue.*

MinMeasuredValue and *MaxMeasuredValue* define the range of the sensor

4.7.2.2.1.4 *Tolerance* Attribute

The *Tolerance* attribute indicates the magnitude of the possible error that is associated with *MeasuredValue*. The true value is located in the range (*MeasuredValue – Tolerance*) to (*MeasuredValue + Tolerance*).

4.7.2.3 Commands Received

No cluster specific commands are received by the server cluster.

4.7.2.4 Commands Generated

No cluster specific commands are generated by the server cluster.

4.7.2.5 Attribute Reporting

This cluster shall support attribute reporting using the Report Attributes command and according to the minimum and maximum reporting interval and reportable change settings described in the ZCL Foundation specification (see 2.4.7). The following attributes shall be reported:

MeasuredValue
Tolerance

4.7.3 Client

4.7.3.1 Dependencies

None.

4.7.3.2 Attributes

The Client cluster has no attributes.

4.7.3.3 Commands Received

No cluster specific commands are received by the client cluster.

4.7.3.4 Commands Generated

No cluster specific commands are generated by the client cluster.

4.8 Occupancy Sensing Cluster

4.8.1 Overview

The server cluster provides an interface to occupancy sensing functionality, including configuration and provision of notifications of occupancy status.

4.8.2 Server

4.8.2.1 Dependencies

None.

4.8.2.2 Attributes

For convenience, the attributes defined in this specification are arranged into sets of related attributes; each set can contain up to 16 attributes. Attribute identifiers are encoded such that the most significant three nibbles specify the attribute set and the least significant nibble specifies the attribute within the set. The currently defined attribute sets are listed in Table 4.21.

Table 4.21 Occupancy Sensor Attribute Sets

Attribute Set Identifier	Description
0x000	Occupancy sensor information
0x001	PIR configuration
0x002	Ultrasonic configuration
0x003 – 0xfff	Reserved

4.8.2.2.1 Occupancy Sensor Information Set

The occupancy sensor information attribute set contains the attributes summarized in Table 4.22.

Table 4.22 Occupancy Sensor Information Attribute Set

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0000	*Occupancy*	8-bit bitmap	0000 000x	Read only	-	M
0x0001	*OccupancySensorType*	8-bit enumeration	0x00 – 0xfe	Read only	-	M

4.8.2.2.1.1 *Occupancy* Attribute

The *Occupancy* attribute is a bitmap.

Bit 0 specifies the sensed occupancy as follows: 1 = occupied, 0 = unoccupied.

All other bits are reserved.

4.8.2.2.1.2 *OccupancySensorType* Attribute

The *OccupancySensorType* attribute specifies the type of the occupancy sensor. This attribute shall be set to one of the non-reserved values listed in Table 4.23.

Table 4.23 Values of the *OccupancySensorType* Attribute

Attribute Value	Description
0x00	PIR
0x01	Ultrasonic
0x02	PIR and ultrasonic
0x03 – 0xff	Reserved

4.8.2.2.2 PIR Configuration Set

The PIR sensor configuration attribute set contains the attributes summarized in Table 4.24.

Table 4.24 Attributes of the PIR Configuration Attribute Set

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0010	*PIROccupiedTo UnoccupiedDelay*	Unsigned 16-bit integer	0x00 – 0xfffe	Read/ Write	0x00	O
0x0011	*PIRUnoccupiedTo OccupiedDelay*	Unsigned 16-bit integer	0x00 – 0xfffe	Read/ write	0x00	O
0x0012	*PIRUnoccupiedTo OccupiedThreshold*	Unsigned 8-bit integer	0x01 – 0xfe	Read/ write	0x01	O

4.8.2.2.2.1 *PIROccupiedToUnoccupiedDelay* Attribute

The *PIROccupiedToUnoccupiedDelay* attribute is 16 bits in length and specifies the time delay, in seconds, before the PIR sensor changes to its unoccupied state after the last detection of movement in the sensed area.

4.8.2.2.2.2 *PIRUnoccupiedToOccupiedDelay* Attribute

The *PIRUnoccupiedToOccupiedDelay* attribute is 16 bits in length and specifies the time delay, in seconds, before the PIR sensor changes to its occupied state after the detection of movement in the sensed area. This attribute is mandatory if the *PIRUnoccupiedToOccupiedThreshold* attribute is implemented.

4.8.2.2.2.3 *PIRUnoccupiedToOccupiedThreshold* Attribute

The *PIRUnoccupiedToOccupiedThreshold* attribute is 8 bits in length and specifies the number of movement detection events that must occur in the period *PIRUnoccupiedToOccupiedDelay*, before the PIR sensor changes to its occupied state. This attribute is mandatory if the *PIRUnoccupiedToOccupiedDelay* attribute is implemented.

4.8.2.2.3 Ultrasonic Configuration Set

The ultrasonic sensor configuration attribute set contains the attributes summarized in Table 4.25.

Table 4.25 Attributes of the Ultrasonic Configuration Attribute Set

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0020	*UltrasonicOccupiedToUnoccupiedDelay*	Unsigned 16-bit integer	0x0000 – 0xfffe	Read/ Write	0x00	O
0x0021	*UltrasonicUnoccupiedToOccupiedDelay*	Unsigned 16-bit integer	0x0000 – 0xfffe	Read/ write	0x00	O
0x0022	*UltrasonicUnoccupiedToOccupiedThreshold*	Unsigned 8-bit integer	0x01 – 0xfe	Read/ write	0x01	O

4.8.2.2.3.1 *UltrasonicOccupiedToUnoccupiedDelay* Attribute

The *UltrasonicOccupiedToUnoccupiedDelay* attribute is 16 bits in length and specifies the time delay, in seconds, before the Ultrasonic sensor changes to its unoccupied state after the last detection of movement in the sensed area.

4.8.2.2.3.2 *UltrasonicUnoccupiedToOccupiedDelay* Attribute

The *UltrasonicUnoccupiedToOccupiedDelay* attribute is 16 bits in length and specifies the time delay, in seconds, before the Ultrasonic sensor changes to its occupied state after the detection of movement in the sensed area. This attribute is mandatory if the *UltrasonicUnoccupiedToOccupiedThreshold* attribute is implemented.

4.8.2.2.3.3 *UltrasonicUnoccupiedToOccupiedThreshold* Attribute

The *UltrasonicUnoccupiedToOccupiedThreshold* attribute is 8 bits in length and specifies the number of movement detection events that must occur in the period *UltrasonicUnoccupiedToOccupiedDelay*, before the Ultrasonic sensor changes to its occupied state. This attribute is mandatory if the *UltrasonicUnoccupiedToOccupiedDelay* attribute is implemented.

4.8.2.3 Commands Received

No cluster specific commands are received by the server cluster.

4.8.2.4 Commands Generated

No cluster specific commands are generated by the server cluster.

4.8.2.5 Attribute Reporting

This cluster shall support attribute reporting using the Report Attributes command and according to the minimum and maximum reporting interval settings described in the ZCL Foundation specification (see 2.4.7). The following attribute shall be reported:

Occupancy

4.8.3 Client

4.8.3.1 Dependencies

None.

4.8.3.2 Attributes

The client cluster has no attributes.

4.8.3.3 Commands Received

No cluster specific commands are received by the client cluster.

4.8.3.4 Commands Generated

No cluster specific commands are generated by the client cluster.

This page left intentionally blank

CHAPTER

5

LIGHTING SPECIFICATION

5.1 General Description

5.1.1 Introduction

The clusters specified in this document are for use typically in ZigBee lighting applications, but may be used in any application domain.

5.1.2 Cluster List

This section lists the clusters specified in this document, and gives examples of typical usage for the purpose of clarification.

The clusters specified in this document are listed in Table 5.1.

Table 5.1 Clusters Specified for the Lighting Functional Domain

Cluster Name	Description
Color Control	Attributes and commands for controlling the color of a color-capable light.
Ballast Configuration	Attributes and commands for configuring a lighting ballast

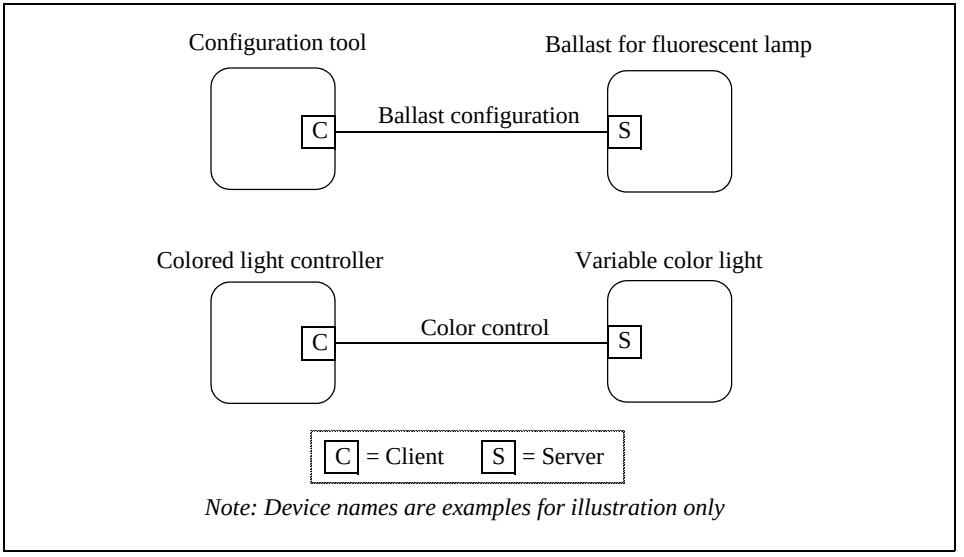

Figure 5.1 Typical Usage of Ballast Configuration and Color Control Clusters

5.2 Color Control Cluster

5.2.1 Overview

This cluster provides an interface for changing the color of a light. Color is specified according to the Commission Internationale de l'Éclairage (CIE) specification CIE 1931 Color Space, [B4]. Color control is carried out in terms of x,y values, as defined by this specification.

Additionally, color may optionally be controlled in terms of color temperature, or as hue and saturation values based on optionally variable RGB and W color points. It is recommended that the hue and saturation are interpreted according to the HSV (aka HSB) color model.

Control over luminance is not included, as this is provided by means of the Level Control cluster of the General library (see 3.10). It is recommended that the level provided by this cluster be interpreted as representing a proportion of the maximum intensity achievable at the current color.

5.2.2 Server

5.2.2.1 Dependencies

None

5.2.2.2 Attributes

For convenience, the attributes defined in this specification are arranged into sets of related attributes; each set can contain up to 16 attributes. Attribute identifiers are encoded such that the most significant three nibbles specify the attribute set and the least significant nibble specifies the attribute within the set. The currently defined attribute sets are listed in Table 5.2.

Table 5.2 Hue Control Attribute Sets

Attribute Set Identifier	Description
0x000	Color Information
0x001	Defined Primaries Information
0x002	Additional Defined Primaries Information
0x003	Defined Color Point Settings
0x004 – 0xfff	Reserved

5.2.2.2.1 Color Information Attribute Set

The Color Information attribute set contains the attributes summarized in Table 5.3.

Table 5.3 Attributes of the Color Information Attribute Set

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0000	*CurrentHue*	Unsigned 8-bit integer	0x00 – 0xfe	Read only	0x00	O
0x0001	*CurrentSaturation*	Unsigned 8-bit integer	0x00 – 0xfe	Read only	0x00	O
0x0002	*RemainingTime*	Unsigned 16-bit integer	0x0000 – 0xfffe	Read only	0x00	O
0x0003	*CurrentX*	Unsigned 16-bit integer	0x0000 - 0xfeff	Read only	0x616b (0.381)	M
0x0004	*CurrentY*	Unsigned 16-bit integer	0x0000 - 0xfeff	Read only	0x607d (0.377)	M

Table 5.3 Attributes of the Color Information Attribute Set (Continued)

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0005	*DriftCompensation*	8-bit enumeration	0x00 – 0x04	Read only	-	O
0x0006	*CompensationText*	Character string	0 to 255 chars	Read only	-	O
0x0007	*ColorTemperature*	Unsigned 16-bit integer	0x0000 - 0xfeff	Read only	0x00fa (4000K)	O
0x0008	*ColorMode*	8-bit enumeration	0x00 – 0x02	Read only	0x01	O (see 5.2.2.2.1.9)

5.2.2.2.1.1 *CurrentHue* Attribute

The *CurrentHue* attribute contains the current hue value of the light. It is updated as fast as practical during commands that change the hue.

The hue in degrees shall be related to the CurrentHue attribute by the relationship
Hue = CurrentHue x 360 / 254 (CurrentHue in the range 0 - 254 inclusive)

If this attribute is implemented then the *CurrentSaturation* and *ColorMode* attributes shall also be implemented.

5.2.2.2.1.2 *CurrentSaturation* Attribute

The *CurrentSaturation* attribute holds the current saturation value of the light. It is updated as fast as practical during commands that change the saturation.

The saturation shall be related to the CurrentSaturation attribute by the relationship
Saturation = CurrentSaturation/254 (CurrentSaturation in the range 0 - 254 inclusive)

If this attribute is implemented then the *CurrentHue* and *ColorMode* attributes shall also be implemented.

5.2.2.2.1.3 *RemainingTime* Attribute

The *RemainingTime* attribute holds the time remaining, in 1/10ths of a second, until the currently active command will be complete.

5.2.2.2.1.4 *CurrentX* Attribute

The *CurrentX* attribute contains the current value of the normalized chromaticity value x, as defined in the CIE xyY Color Space. It is updated as fast as practical during commands that change the color.

The value of x shall be related to the CurrentX attribute by the relationship

x = CurrentX / 65536 (CurrentX in the range 0 to 65279 inclusive)

5.2.2.2.1.5 *CurrentY* Attribute

The *CurrentY* attribute contains the current value of the normalized chromaticity value y, as defined in the CIE xyY Color Space. It is updated as fast as practical during commands that change the color.

The value of y shall be related to the CurrentY attribute by the relationship

y = CurrentY / 65536 (CurrentY in the range 0 to 65279 inclusive)

5.2.2.2.1.6 *DriftCompensation* Attribute

The *DriftCompensation* attribute indicates what mechanism, if any, is in use for compensation for color/intensity drift over time. It shall be one of the non-reserved values in .

Table 5.4 Values of the *DriftCompensation* Attribute

Attribute Value	Description
0x00	None
0x01	Other / Unknown
0x02	Temperature monitoring
0x03	Optical luminance monitoring and feedback
0x04	Optical color monitoring and feedback
0x05 – 0xff	Reserved

5.2.2.2.1.7 *CompensationText* Attribute

The *CompensationText* attribute holds a textual indication of what mechanism, if any, is in use to compensate for color/intensity drift over time.

5.2.2.2.1.8 *ColorTemperature* Attribute

The *ColorTemperature* attribute contains a scaled inverse of the current value of the color temperature. It is updated as fast as practical during commands that change the color.

The color temperature value in Kelvins shall be related to the *ColorTemperature* attribute by the relationship

Color temperature = 1,000,000 / *ColorTemperature* in the range 1 to 65279 inclusive, giving a color temperature range from 1,000,000 Kelvins to 15.32 Kelvins).

The value *ColorTemperature* = 0 indicates an undefined value. The value ColorTemperature = 65535 indicates an invalid value.

If this attribute is implemented then the *ColorMode* attribute shall also be implemented.

5.2.2.2.1.9 *ColorMode* Attribute

The *ColorMode* attribute indicates which attributes are currently determining the color of the device, as detailed in . If either the *CurrentHue* or *CurrentSaturation* attribute is implemented, this attribute shall also be implemented, otherwise it is optional.

The value of the *ColorMode* attribute cannot be written directly - it is set upon reception of any command in section 5.2.2.3 to the appropriate mode for that command.

Table 5.5 Values of the *ColorMode* Attribute

Attribute Value	Attributes that Determine the Color
0x00	CurrentHue and CurrentSaturation
0x01	CurrentX and CurrentY
0x02	ColorTemperature
0x03 – 0xff	Reserved

5.2.2.2.2 Defined Primaries Information Attribute Set

The Defined Primaries Information attribute set contains the attributes summarized in Table 5.6.

Table 5.6 Defined Primaries Information Attribute Set

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0010	*NumberOfPrimaries*	Unsigned 8-bit integer	0x00 – 0x06	Read only	-	O
0x0011	*Primary1X*	Unsigned 16-bit integer	0x0000 – 0xfeff	Read only	-	O
0x0012	*Primary1Y*	Unsigned 16-bit integer	0x0000 – 0xfeff	Read only	-	O

Table 5.6 Defined Primaries Information Attribute Set (Continued)

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0013	*Primary1Intensity*	Unsigned 8-bit integer	0x00 - 0xff	Read only	-	O
0x0014	*Reserved*	-	-	-	-	-
0x0015	*Primary2X*	Unsigned 16-bit integer	0x0000 – 0xfeff	Read only	-	O
0x0016	*Primary2Y*	Unsigned 16-bit integer	0x0000 – 0xfeff	Read only	-	O
0x0017	*Primary2Intensity*	Unsigned 8-bit integer	0x0000- 0xff	Read only	-	O
0x0018	*Reserved*	-	-	-	-	-
0x0019	*Primary3X*	Unsigned 16-bit integer	0x0000 – 0xfeff	Read only	-	O
0x001a	*Primary3Y*	Unsigned 16-bit integer	0x0000 – 0xfeff	Read only	-	O
0x001b	*Primary3Intensity*	Unsigned 8-bit integer	0x00 - 0xff	Read only	-	O

5.2.2.2.2.1 *NumberOfPrimaries* Attribute

The *NumberOfPrimaries* attribute contains the number of color primaries implemented on this device. A value of 0xff shall indicate that the number of primaries is unknown.

Where this attribute is implemented, the attributes below for indicating the “x” and “y” color values of the primaries shall also be implemented for each of the primaries from 1 to NumberOfPrimaries, without leaving gaps. Implementation of the Primary1Intensity attribute and subsequent intensity attributes is optional.

5.2.2.2.2.2 *Primary1X* Attribute

The *Primary1X* attribute contains the normalized chromaticity value x for this primary, as defined in the CIE xyY Color Space.

The value of x shall be related to the Primary1X attribute by the relationship

x = Primary1X / 65536 (Primary1X in the range 0 to 65279 inclusive)

5.2.2.2.2.3 *Primary1Y* Attribute

The *Primary1Y* attribute contains the normalized chromaticity value y for this primary, as defined in the CIE xyY Color Space.

The value of y shall be related to the Primary1Y attribute by the relationship

y = Primary1Y / 65536 (Primary1Y in the range 0 to 65279 inclusive)

5.2.2.2.2.4 *Primary1Intensity* Attribute

The Primary1intensity attribute contains a representation of the maximum intensity of this primary as defined in the Dimming Light Curve in the Ballast Configuration cluster (see 5.3), normalized such that the primary with the highest maximum intensity contains the value 0xfe.

A value of 0xff shall indicate that this primary is not available.

5.2.2.2.2.5 Remaining Attributes

The *Primary2X, Primary2Y, Primary2Intensity, Primary3X, Primary3Y and Primary3Intensity* attributes are used to represent the capabilities of the 2nd and 3rd primaries, where present, in the same way as for the *Primary1X, Primary1Y and Primary1Intensity* attributes.

5.2.2.2.3 Additional Defined Primaries Information Attribute Set

The Additional Defined Primaries Information attribute set contains the attributes summarized in Table .

Table 5.7 Additional Defined Primaries Information Attribute Set

Identifier	Name	Type	Range	Access	Default	Mandatory /Optional
0x0020	*Primary4X*	Unsigned 16-bit integer	0x0000 – 0xfeff	Read only	-	O
0x0021	*Primary4Y*	Unsigned 16-bit integer	0x0000 – 0xfeff	Read only	-	O
0x0022	*Primary4Intensity*	Unsigned 8-bit integer	0x00 – 0xff	Read only	-	O
0x0023	*Reserved*	-	-	-	-	-
0x0024	*Primary5X*	Unsigned 16-bit integer	0x0000 – 0xfeff	Read only	-	O
0x0025	*Primary5Y*	Unsigned 16-bit integer	0x0000 – 0xfeff	Read only	-	O

Table 5.7 Additional Defined Primaries Information Attribute Set (Continued)

Identifier	Name	Type	Range	Access	Default	Mandatory /Optional
0x0026	*Primary5Intensity*	Unsigned 8-bit integer	0x00 – 0xff	Read only	-	O
0x0027	*Reserved*	-	-	-	-	-
0x0028	*Primary6X*	Unsigned 16-bit integer	0x0000 – 0xfeff	Read only	-	O
0x0029	*Primary6Y*	Unsigned 16-bit integer	0x0000 – 0xfeff	Read only	-	O
0x002a	*Primary6Intensity*	Unsigned 8-bit integer	0x00 – 0xff	Read only	-	O

5.2.2.2.3.1 Attributes

The *Primary4X*, *Primary4Y*, *Primary4Intensity*, *Primary5X*, *Primary5Y*, *Primary5Intensity*, *Primary6X*, *Primary6Y* and *Primary6Intensity* attributes represent the capabilities of the 4th, 5th and 6th primaries, where present, in the same way as the *Primary1X*, *Primary1Y* and *Primary1Intensity* attributes.

5.2.2.2.4 Defined Color Points Settings Attribute Set

The Defined Color Points Settings attribute set contains the attributes summarized in Table .

Table 5.8 Defined Color Points Settings Attribute Set

Identifier	Name	Type	Range	Access	Default	Mandatory /Optional
0x0030	*WhitePointX*	Unsigned 16-bit integer	0x0000 – 0xfeff	Read/Write	-	O
0x0031	*WhitePointY*	Unsigned 16-bit integer	0x0000 – 0xfeff	Read/Write	-	O
0x0032	*ColorPointRX*	Unsigned 16-bit integer	0x0000 – 0xfeff	Read/Write	-	O
0x0033	*ColorPointRY*	Unsigned 16-bit integer	0x0000 – 0xfeff	Read/Write	-	O
0x0034	*ColorPointRIntensity*	Unsigned 8-bit integer	0x00 – 0xff	Read/Write	-	O
0x0035	*Reserved*	-	-	-	-	-

Table 5.8 Defined Color Points Settings Attribute Set (Continued)

Identifier	Name	Type	Range	Access	Default	Mandatory /Optional
0x0036	*ColorPointGX*	Unsigned 16-bit integer	0x0000 – 0xfeff	Read/Write	-	O
0x0037	*ColorPointGY*	Unsigned 16-bit integer	0x0000 – 0xfeff	Read/Write	-	O
0x0038	*ColorPointGIntensity*	Unsigned 8-bit integer	0x00 – 0xff	Read/Write	-	O
0x0039	*Reserved*	-	-	-	-	-
0x003a	*ColorPointBX*	Unsigned 16-bit integer	0x0000 – 0xfeff	Read/Write	-	O
0x003b	*ColorPointBY*	Unsigned 16-bit integer	0x0000 – 0xfeff	Read/Write		O
0x003c	*ColorPointBIntensity*	Unsigned 8-bit integer	0x00 – 0xff	Read/Write		O
0x003d	*Reserved*	-	-	-	-	-

5.2.2.2.4.1 *WhitePointX* Attribute

The *WhitePointX* attribute contains the normalized chromaticity value x, as defined in the CIE xyY Color Space, of the current white point of the device.

The value of x shall be related to the WhitePointX attribute by the relationship

x = WhitePointX / 65536 (WhitePointX in the range 0 to 65279 inclusive)

5.2.2.2.4.2 *WhitePointY* Attribute

The *WhitePointY* attribute contains the normalized chromaticity value y, as defined in the CIE xyY Color Space, of the current white point of the device.

The value of y shall be related to the WhitePointY attribute by the relationship

y = WhitePointY / 65536 (WhitePointY in the range 0 to 65279 inclusive)

5.2.2.2.4.3 *ColorPointRX* Attribute

The *ColorPointRX* attribute contains the normalized chromaticity value x, as defined in the CIE xyY Color Space, of the red color point of the device.

The value of x shall be related to the ColorPointRX attribute by the relationship

x = ColorPointRX / 65536 (ColorPointRX in the range 0 to 65279 inclusive)

5.2.2.2.4.4 *ColorPointRY* Attribute

The *ColorPointRY* attribute contains the normalized chromaticity value y, as defined in the CIE xyY Color Space, of the red color point of the device.

The value of y shall be related to the ColorPointRY attribute by the relationship

y = ColorPointRY / 65536 (ColorPointRY in the range 0 to 65279 inclusive)

5.2.2.2.4.5 *ColorPointRIntensity* Attribute

The *ColorPointRIntensity* attribute contains a representation of the relative intensity of the red color point as defined in the Dimming Light Curve in the Ballast Configuration cluster (see 5.3), normalized such that the color point with the highest relative intensity contains the value 0xfe.

A value of 0xff shall indicate an invalid value.

5.2.2.2.4.6 Remaining Attributes

The *ColorPointGX, ColorPointGY, ColorPointGIntensity, ColorPointBX, ColorPointBY* and, *ColorPointBIntensity* attributes are used to represent the chromaticity values and intensities of the green and blue color points, in the same way as for the *ColorPointRX, ColorPointRY* and *ColorPointRIntensity* attributes.

If any one of these red, green or blue color point attributes is implemented then they shall all be implemented.

5.2.2.3 Commands Received

The command IDs for the Color Control cluster are listed in Table 5.9.

Table 5.9 Command IDs for the Color Control Cluster

Command Identifier Field Value	Description	Mandatory / Optional
0x00	Move to Hue	O
0x01	Move Hue	O
0x02	Step Hue	O
0x03	Move to Saturation	O
0x04	Move Saturation	O
0x05	Step Saturation	O
0x06	Move to Hue and Saturation	O
0x07	Move to Color	M

Table 5.9 Command IDs for the Color Control Cluster (Continued)

Command Identifier Field Value	Description	Mandatory / Optional
0x08	Move Color	M
0x09	Step Color	M
0x0a	Move to Color Temperature	O
0x0b – 0xff	Reserved	-

5.2.2.3.1 Note on change of *ColorMode*

The first action taken when any one of these commands is received is to change the *ColorMode* attribute to the appropriate value for the command (see individual commands). Note that, when moving from one color mode to another (e.g. CurrentX/CurrentY to CurrentHue/CurrentSaturation), the starting color for the command is formed by calculating the values of the new attributes (in this case CurrentHue, CurrentSaturation) from those of the old attributes (in this case CurrentX and CurrentY).

When moving from a mode to another mode that has a more restricted color range (e.g. CurrentX/CurrentY to CurrentHue/CurrentSaturation, or CurrentHue/ CurrentSaturation to ColorTemperature) it is possible for the current color value to have no equivalent in the new mode. The behavior in such cases is manufacturer dependent, and therefore it is recommended to avoid color mode changes of this kind during usage.

5.2.2.3.2 Move to Hue Command

5.2.2.3.2.1 Payload Format

The Move to Hue command payload shall be formatted as illustrated in Figure 5.2.

Bits	8	8	16
Data Type	Unsigned 8-bit integer	8-bit enumeration	Unsigned 16-bit integer
Field Name	Hue	Direction	Transition time

Figure 5.2 Format of the Move to Hue Command Payload

5.2.2.3.2.2 Hue Field

The Hue field specifies the hue to be moved to.

5.2.2.3.2.3 Direction Field

The Direction field shall be one of the non-reserved values in Table 5.10.

Table 5.10 Values of the Direction Field

Fade Mode Value	Description
0x00	Shortest distance
0x01	Longest distance
0x02	Up
0x03	Down
0x04 – 0xff	Reserved

5.2.2.3.2.4 Transition Time Field

The Transition time field specifies, in 1/10ths of a second, the time that shall be taken to move to the new hue

5.2.2.3.2.5 Effect on Receipt

On receipt of this command, a device shall also set the ColorMode attribute to the value 0x00 and then shall move from its current hue to the value given in the Hue field.

The movement shall be continuous, i.e. not a step function, and the time taken to move to the new hue shall be equal to the Transition time field.

As hue is effectively measured on a circle, the new hue may be moved to in either direction. The direction of hue change is given by the Direction field. If Direction is 'Shortest distance', the direction is taken that involves the shortest path round the circle. This case corresponds to expected normal usage. If Direction is 'Longest distance', the direction is taken that involves the longest path round the circle. This case can be used for 'rainbow effects'. In both cases, if both distances are the same, the Up direction shall be taken.

Note that if the target color specified is not achievable by the hardware then the command shall not be carried out, and a ZCL default response command shall be generated, where not disabled, with status code equal to INVALID_VALUE.

5.2.2.3.3 Move Hue Command

5.2.2.3.3.1 Payload Format

The Move Hue command payload shall be formatted as illustrated in Figure 5.3.

Bits	8	8
Data Type	8-bit enumeration	Unsigned 8-bit integer
Field Name	Move mode	Rate

Figure 5.3 Format of the Move Hue Command Payload

5.2.2.3.3.2 Move Mode Field

The Move mode field shall be one of the non-reserved values in Table 5.11.

Table 5.11 Values of the Move Mode Field

Fade Mode Value	Description
0x00	Stop
0x01	Up
0x02	Reserved
0x03	Down
0x04 – 0xff	Reserved

5.2.2.3.3.3 Rate Field

The Rate field specifies the rate of movement in steps per second. A step is a change in the device's hue of one unit. If the Rate field has a value of zero, the command has no effect and a default response command (see 2.4.12) is sent in response, with the status code set to INVALID_FIELD.

5.2.2.3.3.4 Effect on Receipt

On receipt of this command, a device shall set the ColorMode attribute to the value 0x00 and shall then move from its current hue in an up or down direction in a continuous fashion, as detailed in Table 5.12.

Table 5.12 Actions on Receipt for Move Hue Command

Fade Mode	Action on Receipt
Stop	If moving, stop, else ignore the command (i.e. the command is accepted but has no effect). NB This may also be used to stop a Move to Hue command, a Move to Saturation command, or a Move to Hue and Saturation command.

Table 5.12 Actions on Receipt for Move Hue Command (Continued)

Fade Mode	Action on Receipt
Up	Increase the device's hue at the rate given in the Rate field. If the hue reaches the maximum allowed for the device, then proceed to its minimum allowed value.
Down	Decrease the device's hue at the rate given in the Rate field. If the hue reaches the minimum allowed for the device, then proceed to its maximum allowed value.

5.2.2.3.4 Step Hue Command

5.2.2.3.4.1 Payload Format

The Step Hue command payload shall be formatted as illustrated in Figure 5.4.

Bits	8	8	8
Data Type	8-bit enumeration	Unsigned 8-bit integer	Unsigned 8-bit integer
Field Name	Step mode	Step size	Transition time

Figure 5.4 Format of the Step Hue Command Payload

5.2.2.3.4.2 Step Mode Field

The Step mode field shall be one of the non-reserved values in Table 5.13.

Table 5.13 Values of the Step Mode Field

Fade Mode Value	Description
0x00	Reserved
0x01	Up
0x02	Reserved
0x03	Down
0x04 – 0xff	Reserved

5.2.2.3.4.3 Step Size Field

The change to be added to (or subtracted from) the current value of the device's hue.

5.2.2.3.4.4 Transition Time Field

The Transition time field specifies, in 1/10ths of a second, the time that shall be taken to perform the step. A step is a change in the device's hue of 'Step size' units.

5.2.2.3.4.5 Effect on Receipt

On receipt of this command, a device shall set the ColorMode attribute to the value 0x00 and shall then move from its current hue in an up or down direction by one step, as detailed in Table 5.14.

Table 5.14 Actions on Receipt for Step Hue Command

Fade Mode	Action on Receipt
Up	Increase the device's hue by one step, in a continuous fashion. If the hue value reaches the maximum value then proceed to the minimum allowed value.
Down	Decrease the device's hue by one step, in a continuous fashion. If the hue value reaches the minimum value then proceed to the maximum allowed value.

5.2.2.3.5 Move to Saturation Command

5.2.2.3.5.1 Payload Format

The Move to Saturation command payload shall be formatted as illustrated in Figure 5.5.

Bits	8	16
Data Type	Unsigned 8-bit integer	Unsigned 16-bit integer
Field Name	Saturation	Transition time

Figure 5.5 Format of the Move to Saturation Command Payload

5.2.2.3.5.2 Effect on Receipt

On receipt of this command, a device set the ColorMode attribute to the value 0x00 and shall then move from its current saturation to the value given in the Saturation field.

The movement shall be continuous, i.e. not a step function, and the time taken to move to the new saturation shall be equal to the Transition time field, in 1/10ths of a second.

Note that if the target color specified is not achievable by the hardware then the command shall not be carried out and a ZCL default response command shall be generated, where not disabled, with status code equal to INVALID_VALUE.

5.2.2.3.6 Move Saturation Command

5.2.2.3.6.1 Payload Format

The Move Saturation command payload shall be formatted as illustrated in Figure 5.6.

Bits	8	8
Data Type	8-bit enumeration	Unsigned 8-bit integer
Field Name	Move mode	Rate

Figure 5.6 Format of the Move Saturation Command Payload

5.2.2.3.6.2 Move Mode Field

The Move Mode field shall be one of the non-reserved values in Table 5.15.

Table 5.15 Values of the Move Mode Field

Fade Mode Value	Description
0x00	Stop
0x01	Up
0x02	Reserved
0x03	Down
0x04 – 0xff	Reserved

5.2.2.3.6.3 Rate Field

The Rate field specifies the rate of movement in steps per second. A step is a change in the device's saturation of one unit. If the Rate field has a value of zero, the command has no effect and a default response command (see 2.4.12) is sent in response, with the status code set to INVALID_FIELD.

5.2.2.3.6.4 Effect on Receipt

On receipt of this command, a device shall set the ColorMode attribute to the value 0x00 and shall then move from its current saturation in an up or down direction in a continuous fashion, as detailed in Table 5.16.

Table 5.16 Actions on Receipt for Move Saturation Command

Fade Mode	Action on Receipt
Stop	If moving, stop, else ignore the command (i.e. the command is accepted but has no affect). NB This may also be used to stop a Move to Saturation command, a Move to Hue command, or a Move to Hue and Saturation command.
Up	Increase the device's saturation at the rate given in the Rate field. If the saturation reaches the maximum allowed for the device, stop.
Down	Decrease the device's saturation at the rate given in the Rate field. If the saturation reaches the minimum allowed for the device, stop.

Note that if the target color specified is not achievable by the hardware then the command shall not be carried out and a ZCL default response command shall be generated, where not disabled, with status code equal to INVALID_VALUE

5.2.2.3.7 Step Saturation Command

5.2.2.3.7.1 Payload Format

The Step Saturation command payload shall be formatted as illustrated in Figure 5.7.

Bits	8	8	8
Data Type	8-bit enumeration	Unsigned 8-bit integer	Unsigned 8-bit integer
Field Name	Step mode	Step size	Transition time

Figure 5.7 Format of the Step Saturation Command Payload

5.2.2.3.7.2 Step Mode Field

The Step mode field shall be one of the non-reserved values in Table 5.17.

Table 5.17 Values of the Step Mode Field

Step Mode Value	Description
0x00	Reserved
0x01	Up
0x02	Reserved
0x03	Down
0x04 – 0xff	Reserved

5.2.2.3.7.3 Step Size Field

The change to be added to (or subtracted from) the current value of the device's saturation.

5.2.2.3.7.4 Transition Time Field

The Transition time field specifies, in 1/10ths of a second, the time that shall be taken to perform the step. A step is a change in the device's saturation of 'Step size' units.

5.2.2.3.7.5 Effect on Receipt

On receipt of this command, a device shall set the ColorMode attribute to the value 0x00 and shall then move from its current saturation in an up or down direction by one step, as detailed in Table 5.18.

Table 5.18 Actions on Receipt for Step Saturation Command

Step Mode	Action on Receipt
Up	Increase the device's saturation by one step, in a continuous fashion. However, if the saturation value is already the maximum value then do nothing.
Down	Decrease the device's saturation by one step, in a continuous fashion. However, if the saturation value is already the minimum value then do nothing.

Note that if the target color specified is not achievable by the hardware then the command shall not be carried out and a ZCL default response command shall be generated, where not disabled, with status code equal to INVALID_VALUE

5.2.2.3.8 Move to Hue and Saturation Command

5.2.2.3.8.1 Payload Format

The Move to Hue and Saturation command payload shall be formatted as illustrated in Figure 5.8.

Bits	8	8	16
Data Type	Unsigned 8-bit integer	Unsigned 8-bit integer	Unsigned 16-bit integer
Field Name	Hue	Saturation	Transition time

Figure 5.8 Move to Hue and Saturation Command Payload

5.2.2.3.8.2 Effect on Receipt

On receipt of this command, a device shall set the ColorMode attribute to the value 0x00 and shall then move from its current hue and saturation to the values given in the Hue and Saturation fields.

The movement shall be continuous, i.e. not a step function, and the time taken to move to the new color shall be equal to the Transition time field, in 1/10ths of a second.

The path through color space taken during the transition is not specified, but it is recommended that the shortest path is taken though hue/saturation space, i.e. movement is 'in a straight line' across the hue/saturation disk.

Note that if the target color specified is not achievable by the hardware then the command shall not be carried out and a ZCL default response command shall be generated, where not disabled, with status code equal to INVALID_VALUE

5.2.2.3.9 Move to Color Command

5.2.2.3.9.1 Payload Format

The Move to Color command payload shall be formatted as illustrated in Table 5.9.

Bits	16	16	16
Data Type	Unsigned 16-bit integer	Unsigned 16-bit integer	Unsigned 16-bit integer
Field Name	ColorX	ColorY	Transition time

Figure 5.9 Format of the Move to Color Command Payload

5.2.2.3.9.2 Effect on Receipt

On receipt of this command, a device shall set the value of the *ColorMode* attribute, where implemented, to 0x01, and shall then move from its current color to the color given in the ColorX and ColorY fields.

The movement shall be continuous, i.e. not a step function, and the time taken to move to the new color shall be equal to the Transition time field, in 1/10ths of a second.

The path through color space taken during the transition is not specified, but it is recommended that the shortest path is taken though color space, i.e. movement is 'in a straight line' across the CIE xyY Color Space.

Note that if the target color specified is not achievable by the hardware then the command shall not be carried out, and a ZCL default response command shall be generated, where not disabled, with status code equal to INVALID_VALUE.

5.2.2.3.10 Move Color Command

5.2.2.3.10.1 Payload Format

The Move Color command payload shall be formatted as illustrated in Table 5.10.

Bits	16	16
Data Type	Signed 16-bit integer	Signed 16-bit integer
Field Name	RateX	RateY

Figure 5.10 Format of the Move Color Command Payload

5.2.2.3.10.2 RateX Field

The RateX field specifies the rate of movement in steps per second. A step is a change in the device's CurrentX attribute of one unit.

5.2.2.3.10.3 RateY Field

The RateY field specifies the rate of movement in steps per second. A step is a change in the device's CurrentY attribute of one unit.

5.2.2.3.10.4 Effect on Receipt

On receipt of this command, a device shall set the value of the *ColorMode* attribute, where implemented, to 0x01, and shall then move from its current color in a continuous fashion according to the rates specified. This movement shall continue until the target color for the next step cannot be implemented on this device.

If both the RateX and RateY fields contain a value of zero, no movement shall be carried out, and the command execution shall have no effect other than stopping the operation of any previously received command of this cluster. This command can thus be used to stop the operation of any other command of this cluster.

5.2.2.3.11 Step Color Command

5.2.2.3.11.1 Payload Format

The Step Color command payload shall be formatted as illustrated in Figure 5.11.

Bits	16	16	16
Data Type	Signed 16-bit integer	Signed 16-bit integer	Unsigned 16-bit integer
Field Name	StepX	StepY	Transition time

Figure 5.11 Format of the Move Color Command Payload

5.2.2.3.11.2 StepX and StepY Fields

The StepX and StepY fields specify the change to be added to the device's CurrentX attribute and CurrentY attribute respectively.

5.2.2.3.11.3 Transition Time Field

The Transition time field specifies, in 1/10ths of a second, the time that shall be taken to perform the color change.

5.2.2.3.11.4 Effect on Receipt

On receipt of this command, a device shall set the value of the *ColorMode* attribute, where implemented, to 0x01, and shall then move from its current color by the color step indicated.

The movement shall be continuous, i.e. not a step function, and the time taken to move to the new color shall be equal to the Transition time field, in 1/10ths of a second.

The path through color space taken during the transition is not specified, but it is recommended that the shortest path is taken though color space, i.e. movement is 'in a straight line' across the CIE xyY Color Space.

Note that if the target color specified is not achievable by this hardware then command shall not be carried out, and a ZCL default response command shall be generated, where not disabled, with status code equal to INVALID_VALUE.

Note also that if the required step is larger than can be represented by signed 16-bit integers then more than one step command should be issued.

5.2.2.3.12 Move to Color Temperature Command

5.2.2.3.12.1 Payload Format

The Move to Color Temperature command payload shall be formatted as illustrated in Figure 5.12.

Bits	16	16
Data Type	Unsigned 16-bit integer	Unsigned 16-bit integer
Field Name	Color Temperature	Transition time

Figure 5.12 Move to Color Temperature Command Payload

5.2.2.3.12.2 Effect on Receipt

On receipt of this command, a device shall set the value of the *ColorMode* attribute, where implemented, to 0x02, and shall then move from its current color to the color given by the Color Temperature field.

The movement shall be continuous, i.e. not a step function, and the time taken to move to the new color shall be equal to the Transition time field, in 1/10ths of a second.

By definition of this color mode, the path through color space taken during the transition is along the ‘Black Body Line'.

Note that if the target color specified is not achievable by the hardware then the command shall not be carried out, and a ZCL default response command shall be generated, where not disabled, with status code equal to INVALID_VALUE.

5.2.2.4 Commands Generated

The server generates no cluster specific commands

5.2.2.5 Scene Table Extensions

If the Scenes server cluster (see 3.7) is implemented, the following extension fields are added to the Scenes table:

CurrentX, CurrentY

5.2.2.6 Attribute Reporting

This cluster shall support attribute reporting using the Report Attributes command and according to the minimum and maximum reporting interval and reportable change settings described in the ZCL Foundation specification (see 2.4.7). The following attributes shall be reportable:

CurrentX, CurrentY

CurrentHue (if implemented), *CurrentSaturation* (if implemented), *ColorTemperature* (if implemented)

5.2.3 Client

5.2.3.1 Dependencies

None

5.2.3.2 Attributes

None

5.2.3.3 Commands Received

No cluster specific commands are received by the client.

5.2.3.4 Commands Generated

The client generates the cluster specific commands detailed in 5.2.2.3, as required by the application.

5.3 Ballast Configuration Cluster

5.3.1 Overview

Attributes and commands for configuring a lighting ballast.

5.3.2 Server

5.3.2.1 Dependencies

For the alarm functionality specified by this cluster to be operational, the Alarms server cluster shall be implemented on the same endpoint.

5.3.2.2 Attributes

For convenience, the attributes defined in this specification are arranged into sets of related attributes; each set can contain up to 16 attributes. Attribute identifiers are encoded such that the most significant three nibbles specify the attribute set and the least significant nibble specifies the attribute within the set. The currently defined attribute sets are listed in Table 5.19.

Table 5.19 Ballast Configuration Attribute Sets

Attribute Set Identifier	Description
0x000	Ballast information
0x001	Ballast settings
0x002	Lamp information
0x003	Lamp settings
0x004 – 0xfff	Reserved

5.3.2.2.1 Ballast Information Attribute Set

The Ballast Information attribute set contains the attributes summarized in Table 5.20.

Table 5.20 Attributes of the Ballast Information Attribute Set

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0000	*PhysicalMinLevel*	Unsigned 8-bit integer	0x01 – 0xfe	Read only	0x01	O
0x0001	*PhysicalMaxLevel*	Unsigned 8-bit integer	0x01 – 0xfe	Read only	0xfe	O
0x0002	*BallastStatus*	8-bit bitmap	0000 00xx	Read only	0000 0000	M

5.3.2.2.1.1 *PhysicalMinLevel* Attribute

The *PhysicalMinLevel* attribute is 8 bits in length and specifies the minimum light level the ballast can achieve. This attribute shall be specified in the range 0x01 to 0xfe, and specifies the light output of the ballast according to the dimming light curve (see 5.3.4).

5.3.2.2.1.2 *PhysicalMaxLevel* Attribute

The *PhysicalMaxLevel* attribute is 8 bits in length and specifies the maximum light level the ballast can achieve. This attribute shall be specified in the range 0x01 to 0xfe, and specifies the light output of the ballast according to the dimming light curve (see 5.3.4).

5.3.2.2.1.3 *BallastStatus* Attribute

The *BallastStatus* attribute is 8 bits in length and specifies the activity status of the ballast functions. The usage of the bits is specified in Table 5.21. Where a function is active, the corresponding bit shall be set to 1. Where a function is not active, the corresponding bit shall be set to 0.

Table 5.21 Bit Usage of the *BallastStatus* Attribute

Attribute Bit Number	Ballast Function	Details
0	Non-operational	0 = The ballast is fully operational 1 = The ballast is not fully operational
1	Lamp not in socket	0 = All lamps are in their sockets 1 = One or more lamp is not in its socket
2 – 7	Reserved	-

5.3.2.2.2 Ballast Settings Attribute Set

The Ballast Settings attribute set contains the attributes summarized in Table 5.22.

Table 5.22 Attributes of the Ballast Settings Attribute Set

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0010	*MinLevel*	Unsigned 8-bit integer	0x01 – 0xfe	Read/ write	Physical MinLevel	O
0x0011	*MaxLevel*	Unsigned 8-bit integer	0x01 – 0xfe	Read/ write	Physical MaxLevel	O
0x0012	*PowerOnLevel*	Unsigned 8-bit integer	0x00 – 0xfe	Read/ write	Physical MaxLevel	O
0x0013	*PowerOn FadeTime*	Unsigned 16-bit integer	0x0000 – 0xfffe	Read/ write	0x0000	O
0x0014	*Intrinsic BallastFactor*	Unsigned 8-bit integer	0x00 – 0xfe	Read/ write	-	O

Table 5.22 Attributes of the Ballast Settings Attribute Set (Continued)

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0015	*BallastFactor Adjustment*	Unsigned 8-bit integer	0x64 – manufacturer dependent	Read/ write	0xff	O

5.3.2.2.2.1 *MinLevel* Attribute

The *MinLevel* attribute is 8 bits in length and specifies the minimum light level the ballast is permitted to use. This attribute shall be specified in the range 0x01 to 0xfe, and specifies the light output of the ballast according to the dimming light curve (see 7.4).

The value of this attribute shall be both greater than or equal to *PhysicalMinLevel* and less than or equal to *MaxLevel*. If an attempt is made to set this attribute to a level where these conditions are not met, a default response command shall be returned with status code set to INVALID_VALUE, and the level shall not be set.

5.3.2.2.2.2 *MaxLevel* Attribute

The *MaxLevel* attribute is 8 bits in length and specifies the maximum light level the ballast is permitted to use. This attribute shall be specified in the range 0x01 to 0xfe, and specifies the light output of the ballast according to the dimming light curve (see 5.3.4).

The value of this attribute shall be both less than or equal to *PhysicalMaxLevel* and greater than or equal to *MinLevel*. If an attempt is made to set this attribute to a level where these conditions are not met, a default response command shall be returned with status code set to INVALID_VALUE, and the level shall not be set.

5.3.2.2.2.3 *PowerOnLevel* Attribute

The *PowerOnLevel* attribute is 8 bits in length and specifies the light level to which the ballast will go when power is applied (e.g. when mains power is re-established after a power failure). This attribute shall be set to one of the values listed in Table 5.23.

Table 5.23 Values of the *PowerOnLevel* Attribute

Attribute Value	Description
0x00 – 0xfe	Set to this specific light level, according to the dimming light curve (see 5.3.4).
0xff	Restore the light level being used prior to power failure.

The value of this attribute shall be both less than or equal to *PhysicalMaxLevel* and greater than or equal to *MinLevel*. If an attempt is made to set this attribute to a level where these conditions are not met, a default response command shall be returned with status code set to INVALID_VALUE, and the level shall not be set.

5.3.2.2.2.4 *PowerOnFadeTime* Attribute

The *PowerOnFadeTime* attribute is 16 bits in length and specifies the length of time, in tenths of a second, that the ballast takes to move to the light level specified in *PowerOnLevel* when power is applied (e.g. when mains power is re-established after a power failure).

5.3.2.2.2.5 *IntrinsicBallastFactor* Attribute

The *IntrinsicBallastFactor* attribute is 8 bits in length and specifies as a percentage the ballast factor of the ballast/lamp combination (see also clause 5.3), prior to any adjustment.

A value of 0xff indicates in invalid value.

5.3.2.2.2.6 *BallastFactorAdjustment* Attribute

The *BallastFactorAdjustment* attribute is 8 bits in length and specifies the multiplication factor, as a percentage, to be applied to the configured light output of the lamps (see also clause 5.3). A typical usage of this mechanism is to compensate for reduction in efficiency over the lifetime of a lamp.

The light output is given by

Actual light output = configured light output x *BallastFactorAdjustment* / 100%

The range for this attribute is manufacturer dependent. If an attempt is made to set this attribute to a level that cannot be supported, a default response command shall be returned with status code set to INVALID_VALUE, and the level shall not be set. The value 0xff indicates that ballast factor scaling is not in use.

5.3.2.2.3 Lamp Information Attribute Set

The lamp information attribute set contains the attributes summarized in Table 5.24.

Table 5.24 Attributes of the Lamp Information Attribute Set

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0020	*LampQuantity*	Unsigned 8-bit integer	0x00 – 0xfe	Read only	-	O

5.3.2.2.3.1 *LampQuantity* Attribute

The *LampQuantity* attribute is 8 bits in length and specifies the number of lamps connected to this ballast. (Note 1: this number does not take into account whether lamps are actually in their sockets or not).

5.3.2.2.4 Lamp Settings Attribute Set

The Lamp Settings attribute set contains the attributes summarized in Table 5.25. If *LampQuantity* is greater than one, each of these attributes is taken to apply to the lamps as a set. For example, all lamps are taken to be of the same *LampType* with the same *LampBurnHours.*

Table 5.25 Attributes of the Lamp Settings Attribute Set

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0030	*LampType*	Character string	-	Read/write	Empty string	O
0x0031	*LampManufacturer*	Character string	-	Read/write	Empty string	O
0x0032	*LampRatedHours*	Unsigned 24-bit integer	0x000000 – 0xffffe	Read/write	0xffffff	O
0x0033	*LampBurnHours*	Unsigned 24-bit integer	0x000000 – 0xffffe	Read/write	0x000000	O
0x0034	*LampAlarmMode*	8-bit bitmap	0000 000x	Read/write	0000 0000	O
0x0035	*LampBurnHours TripPoint*	Unsigned 24-bit integer	0x000000 – 0xffffe	Read/write	0xffffff	O

5.3.2.2.4.1 *LampType* Attribute

The *LampType* attribute is an character string of up to 16 bytes in length. It specifies the type of lamps (including their wattage) connected to the ballast.

5.3.2.2.4.2 *LampManufacturer* Attribute

The *LampManufacturer* attribute is an character string of up to 16 bytes in length. It specifies the name of the manufacturer of the currently connected lamps.

5.3.2.2.4.3 *LampRatedHours* Attribute

The *LampRatedHours* attribute is 24 bits in length and specifies the number of hours of use the lamps are rated for by the manufacturer.

A value of 0xffffff indicates an invalid or unknown time.

5.3.2.2.4.4 *LampBurnHours* Attribute

The *LampBurnHours* attribute is 24 bits in length and specifies the length of time, in hours, the currently connected lamps have been operated, cumulative since the last re-lamping. Burn hours shall not be accumulated if the lamps are off.

This attribute should be reset to zero (e.g. remotely) when the lamp(s) are changed. If partially used lamps are connected, *LampBurnHours* should be updated to reflect the burn hours of the lamps.

A value of 0xffffff indicates an invalid or unknown time.

5.3.2.2.4.5 *LampAlarmMode* Attribute

The *LampsAlarmMode* attribute is 8 bits in length and specifies which attributes may cause an alarm notification to be generated, as listed in Table 5.26. A ‘1’ in each bit position causes its associated attribute to be able to generate an alarm. (Note: All alarms are also logged in the alarm table – see Alarms cluster 3.11).

Table 5.26 Values of the *MainsAlarmMode* Attribute

Attribute Bit Number	Attribute
0	LampBurnHours
1 – 7	Reserved

5.3.2.2.4.6 *LampBurnHoursTripPoint* Attribute

The *LampBurnHoursTripPoint* attribute is 24 bits in length and specifies the number of hours the *LampBurnHours* attribute may reach before an alarm is generated.

If the Alarms cluster is not present on the same device this attribute is not used and thus may be omitted (see 5.3.2.1).

The Alarm Code field included in the generated alarm shall be 0x01.

If this attribute takes the value 0xffffff then this alarm shall not be generated.

5.3.2.3 Commands Received

No cluster specific commands are received by the server.

5.3.2.4 Commands Generated

The server generates no cluster specific commands.

5.3.3 Client

5.3.3.1 Dependencies

None

5.3.3.2 Attributes

The client has no attributes.

5.3.3.3 Commands Received

No cluster specific commands are received by the client.

5.3.3.4 Commands Generated

No cluster specific commands are generated by the client.

5.3.4 The Dimming Light Curve

The dimming curve is recommended to be logarithmic, as defined by the following equation:

$$\%Light = 10^{\left(\frac{Level-1}{\left(\frac{253}{3}\right)}\right)-1}$$

Where:%Light is the percent light output of the ballast
Level is an 8-bit integer between 1 (0.1% light output) and 254 (100% output).

255 is reserved - the exact meaning of this value depends on the specific attribute or command.

Note: The light output is determined by this curve together with the *IntrinsicBallastFactor* and *BallastFactorAdjustment* Attributes.

This page left intentionally blank

CHAPTER

6

HVAC SPECIFICATION

6.1 General Description

6.1.1 Introduction

The clusters specified in this document are for use typically in ZigBee HVAC applications, but may be used in any application domain.

6.1.2 Cluster List

This section lists the clusters specified in this document, and gives examples of typical usage for the purpose of clarification.

The clusters defined in this document are listed in Table 6.1.

Table 6.1 Clusters Specified in the HVAC Functional Domain

Cluster Name	Description
Pump Configuration and Control	An interface for configuring and controlling pumps.
Thermostat	An interface for configuring and controlling the functionality of a thermostat
Fan Control	An interface for controlling a fan in a heating / cooling system
Dehumidification Control	An interface for controlling dehumidification
Thermostat User Interface Configuration	An interface for configuring the user interface of a thermostat (which may be remote from the thermostat)

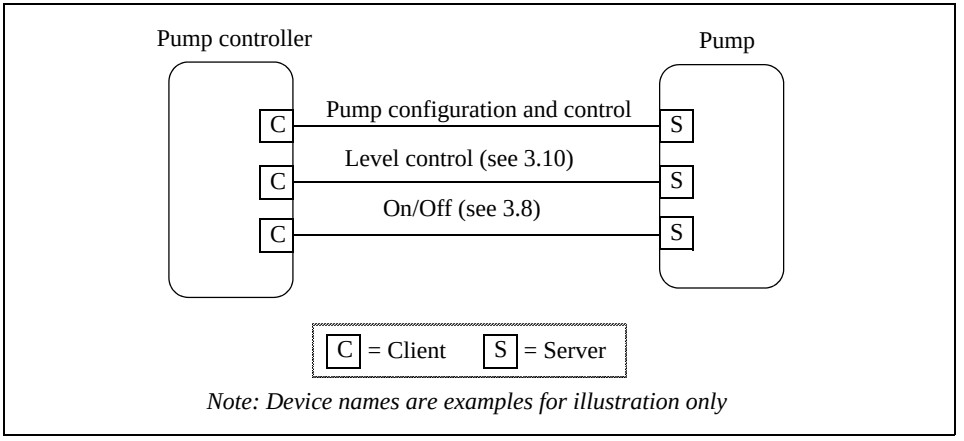

Figure 6.1 Typical Usage of Pump Configuration and Control Cluster

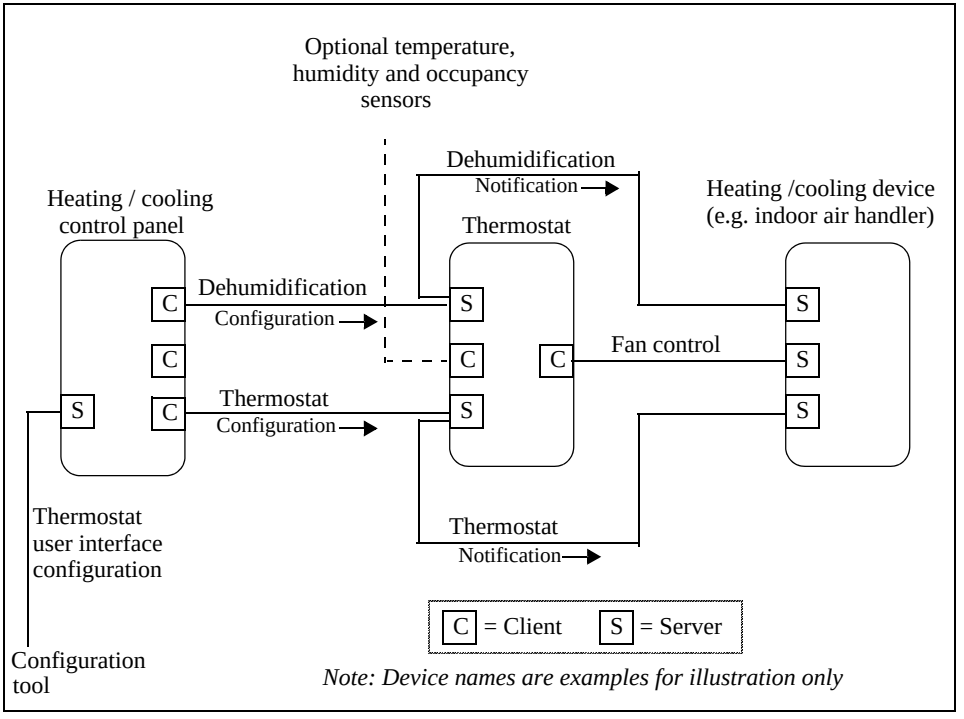

Figure 6.2 Example Usage of the Thermostat and Related Clusters

6.2 Pump Configuration and Control Cluster

6.2.1 Overview

The Pump Configuration and Control cluster provides an interface for the setup and control of pump devices, and the automatic reporting of pump status information. Note that control of pump speed is not included – speed is controlled by the On/Off and Level Control clusters (see Figure 6.1).

6.2.2 Server

6.2.2.1 Dependencies

Where external pressure, flow and temperature measurements are processed by this cluster (see Table 6.8), these are provided by a Pressure Measurement cluster (4.5), a Flow Measurement cluster (4.6) and a Temperature Measurement client cluster (4.4) respectively. These 3 client clusters are used for connection to a remote sensor device. The pump is able to use the sensor measurement provided by a remote sensor for regulation of the pump speed.

For the alarms described in Table 6.9 to be operational, the Alarms server cluster (3.11) shall be implemented on the same endpoint.

Note that control of the pump setpoint is not included in this cluster – the On/Off and Level Control clusters (see Figure 6.1) may be used by a pump device to turn it on and off and control its setpoint. Note that the Pump Configuration and Control Cluster may override on/off/setpoint settings for specific operation modes (See section 6.2.2.2.3.1 for detailed description of the operation and control of the pump.).

6.2.2.2 Attributes

For convenience, the attributes defined in this specification are arranged into sets of related attributes; each set can contain up to 16 attributes. Attribute identifiers are encoded such that the most significant three nibbles specify the attribute set

and the least significant nibble specifies the attribute within the set. The currently defined attribute sets are listed in Table 6.2.

Table 6.2 Pump Configuration Attribute Sets

Attribute Set Identifier	Description
0x000	Pump Information
0x001	Pump Dynamic Information
0x002	Pump Settings
0x003 – 0xfff	Reserved

6.2.2.2.1 Pump Information Attribute Set

The pump information attribute set contains the attributes summarized in Table 6.3.

Table 6.3 Attributes of the Pump Information Attribute Set

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0000	*MaxPressure*	Signed 16-bit integer	0x8001-0x7fff	Read only	-	M
0x0001	*MaxSpeed*	Unsigned 16-bit integer	0x0000 – 0xfffe	Read only	-	M
0x0002	*MaxFlow*	Unsigned 16-bit integer	0x0000 – 0xfffe	Read only	-	M
0x0003	*MinConstPressure*	Signed 16-bit integer	0x8001-0x7fff	Read only	-	O
0x0004	*MaxConstPressure*	Signed 16-bit integer	0x8001-0x7fff	Read only	-	O
0x0005	*MinCompPressure*	Signed 16-bit integer	0x8001-0x7fff	Read only	-	O
0x0006	*MaxCompPressure*	Signed 16-bit integer	0x8001-0x7fff	Read only	-	O
0x0007	*MinConstSpeed*	Unsigned 16-bit integer	0x0000 – 0xfffe	Read only	-	O
0x0008	*MaxConstSpeed*	Unsigned 16-bit integer	0x0000 – 0xfffe	Read only	-	O

Table 6.3 Attributes of the Pump Information Attribute Set (Continued)

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0009	*MinConstFlow*	Unsigned 16-bit integer	0x0000 – 0xfffe	Read only	-	O
0x000a	*MaxConstFlow*	Unsigned 16-bit integer	0x0000 – 0xfffe	Read only	-	O
0x000b	*MinConstTemp*	Signed 16-bit integer	0x954d – 0x7fff	Read only	-	O
0x000c	*MaxConstTemp*	Signed 16-bit integer	0x954d – 0x7fff	Read only	-	O

6.2.2.2.1.1 *MaxPressure* Attribute

The *MaxPressure* attribute specifies the maximum pressure the pump can achieve. It is a physical limit, and does not apply to any specific control mode or operation mode.

This attribute is read only, and can only be set by the manufacturer. If the value is not available, this attribute will display the invalid value.

Valid range is -3,276.7 kPa to 3,276.7 kPa (steps of 0.1 kPa)
The value -3,276.8 kPa (0x8000) indicates that this value is invalid.

6.2.2.2.1.2 *MaxSpeed* Attribute

The *MaxSpeed* attribute specifies the maximum speed the pump can achieve. It is a physical limit, and does not apply to any specific control mode or operation mode.

This attribute is read only, and can only be set by the manufacturer. If the value is not available, this attribute will display the invalid value.

Valid range is 0 to 65,534 RPM (steps of 1 RPM)
The value 65,535 RPM (0xffff) indicates that this value is invalid.

6.2.2.2.1.3 *MaxFlow* Attribute

The *MaxFlow* attribute specifies the maximum flow the pump can achieve. It is a physical limit, and does not apply to any specific control mode or operation mode.

This attribute is read only, and can only be set by the manufacturer. If the value is not available, this attribute will display the invalid value.

Valid range is 0 m^3/h to 6,553.4 m^3/h (steps of 0.1 m^3/h)
The value 6,553.5 m^3/h (0xffff) indicates that this value is invalid.

6.2.2.2.1.4 *MinConstPressure* Attribute

The *MinConstPressure* attribute specifies the minimum pressure the pump can achieve when it is running and working in control mode constant pressure (*ControlMode* attribute of the Pump settings attribute set is set to Constant pressure).

This attribute is read only, and can only be set by the manufacturer. If the value is not available, this attribute will display the invalid value.

Valid range is –3,276.7 kPa to 3,276.7 kPa (steps of 0.1 kPa)
The value -3,276.8 kPa (0x8000) indicates that this value is invalid.

6.2.2.2.1.5 *MaxConstPressure* Attribute

The *MaxConstPressure* attribute specifies the maximum pressure the pump can achieve when it is working in control mode constant pressure (*ControlMode* attribute of the Pump settings attribute set is set to Constant pressure).

This attribute is read only, and can only be set by the manufacturer. If the value is not available, this attribute will display the invalid value.

Valid range is –3,276.7 kPa to 3,276.7 kPa (steps of 0.1 kPa)
The value -3,276.8 kPa (0x8000) indicates that this value is invalid.

6.2.2.2.1.6 *MinCompPressure* Attribute

The *MinCompPressure* attribute specifies the minimum compensated pressure the pump can achieve when it is running and working in control mode Proportional pressure (*ControlMode* attribute of the Pump settings attribute set is set to Proportional pressure).

This attribute is read only, and can only be set by the manufacturer. If the value is not available, this attribute will display the invalid value.

Valid range is –3,276.7 kPa to 3,276.7 kPa (steps of 0.1 kPa)
The value -3,276.8 kPa (0x8000) indicates that this value is invalid.

6.2.2.2.1.7 *MaxCompPressure* Attribute

The *MaxCompPressure* attribute specifies the maximum compensated pressure the pump can achieve when it is working in control mode Proportional pressure (*ControlMode* attribute of the Pump settings attribute set is set to Proportional pressure).

This attribute is read only, and can only be set by the manufacturer. If the value is not available, this attribute will display the invalid value.

Valid range is –3,276.7 kPa to 3,276.7 kPa (steps of 0.1 kPa)
The value -3,276.8 kPa (0x8000) indicates that this value is invalid.

6.2.2.2.1.8 *MinConstSpeed* Attribute

The *MinConstSpeed* attribute specifies the minimum speed the pump can achieve when it is running and working in control mode Constant speed (*ControlMode* attribute of the Pump settings attribute set is set to Constant speed).

This attribute is read only, and can only be set by the manufacturer. If the value is not available, this attribute will display the invalid value.

Valid range is 0 to 65,534 RPM (steps of 1 RPM)
The value 65,535 RPM (0xffff) indicates that this value is invalid.

6.2.2.2.1.9 *MaxConstSpeed* Attribute

The *MaxConstSpeed* attribute specifies the maximum speed the pump can achieve when it is working in control mode Constant speed (*ControlMode* attribute of the Pump settings attribute set is set to Constant speed).

This attribute is read only, and can only be set by the manufacturer. If the value is not available, this attribute will display the invalid value.

Valid range is 0 to 65,534 RPM (steps of 1 RPM)
The value 65,535 RPM (0xffff) indicates that this value is invalid.

6.2.2.2.1.10 *MinConstFlow* Attribute

The *MinConstFlow* attribute specifies the minimum flow the pump can achieve when it is running and working in control mode Constant flow (*ControlMode* attribute of the Pump settings attribute set is set to Constant flow).

This attribute is read only, and can only be set by the manufacturer. If the value is not available, this attribute will display the invalid value.

Valid range is 0 m^3/h to 6,553.4 m^3/h (steps of 0.1 m^3/h)
The value 6,553.5 m^3/h (0xffff) indicates that this value is invalid.

6.2.2.2.1.11 *MaxConstFlow* Attribute

The *MaxConstFlow* attribute specifies the maximum flow the pump can achieve when it is running and working in control mode Constant flow (*ControlMode* attribute of the Pump settings attribute set is set to Constant flow).

This attribute is read only, and can only be set by the manufacturer. If the value is not available, this attribute will display the invalid value.

Valid range is 0 m^3/h to 6,553.4 m^3/h (steps of 0.1 m^3/h).
The value 6,553.5 m^3/h (0xffff) indicates that this value is invalid.

6.2.2.2.1.12 *MinConstTemp* Attribute

The *MinConstTemp* attribute specifies the minimum temperature the pump can maintain in the system when it is running and working in control mode Constant temperature (*ControlMode* attribute of the Pump settings attribute set is set to Constant temperature).

This attribute is read only, and can only be set by the manufacturer. If the value is not available, this attribute will display the invalid value.

Valid range is –273.15 °C to 327.67 °C (steps of 0.01 °C).
The value -327.68°C (0x8000) indicates that this value is invalid.

6.2.2.2.1.13 *MaxConstTemp* Attribute

The *MaxConstTemp* attribute specifies the maximum temperature the pump can maintain in the system when it is running and working in control mode Constant temperature (*ControlMode* attribute of the Pump settings attribute set is set to Constant temperature).

This attribute is read only, and can only be set by the manufacturer. If the value is not available, this attribute will display the invalid value. *MaxConstTemp* shall be greater than or equal to *MinConstTemp*

Valid range is –273.15 °C to 327.67 °C (steps of 0.01 °C).
The value -327.68°C (0x8000) indicates that this value is invalid.

6.2.2.2.2 Pump Dynamic Information Attribute Set

The pump dynamic information attribute set contains the attributes summarized in Table 6.4.

Table 6.4 Attributes of the Pump Dynamic Information Attribute Set

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0010	*PumpStatus*	16-bit bitmap	-	Read only	-	O
0x0011	*EffectiveOperationMode*	8-bit enumeration	0x00 – 0xfe	Read only	-	M
0x0012	*EffectiveControlMode*	8-bit enumeration	0x00 – 0xfe	Read only	-	M

Table 6.4 Attributes of the Pump Dynamic Information Attribute Set (Continued)

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0013	*Capacity*	Signed 16-bit integer	0x0000-0x7fff	Read only	-	M
0x0014	*Speed*	Unsigned 16-bit integer	0x0000 - 0xfffe	Read only	-	O
0x0015	*LifetimeRunningHours*	Unsigned 24-bit integer	0x000000 - 0xffffe	Read / Write	0	O
0x0016	*Power*	Unsigned 24-bit integer	0x000000 - 0xffffe	Read / Write	-	O
0x0017	*LifetimeEnergyConsumed*	Unsigned 32-bit integer	0x00000000 - 0xfffffffe	Read only	0	O

6.2.2.2.2.1 *PumpStatus* Attribute

The *PumpStatus* attribute specifies the activity status of the pump functions listed in Table 6.5. Where a pump controller function is active, the corresponding bit shall be set to 1. Where a pump controller function is not active, the corresponding bit shall be set to 0.

Table 6.5 Values of the *PumpStatus* Attribute

Attribute Bit Number	Pump Function	Remarks
0	Device fault	A fault related to the pump device is detected (Corresponds to a Alarm code in the range 6-13, see Table 6.9)
1	Supply fault	A fault related to the supply to the pump is detected (Corresponds to a Alarm code in the range 0-5 or 13, see Table 6.9)
2	Speed low	Setpoint is too low to achieve
3	Speed high	Setpoint is too high to achieve
4	Local override	The pump is overridden by local control
5	Running	Pump is currently running

Table 6.5 Values of the *PumpStatus* Attribute (Continued)

Attribute Bit Number	Pump Function	Remarks
6	Remote Pressure	A remote pressure sensor is used as the sensor for the regulation of the pump. *EffectiveControlMode* is Constant pressure, and the setpoint for the pump is interpreted as a percentage of the range of the remote sensor ([*MinMeasuredValue* – *MaxMeasuredValue*])
7	Remote Flow	A remote flow sensor is used as the sensor for the regulation of the pump. *EffectiveControlModeI* is Constant flow, and the setpoint for the pump is interpreted as a percentage of the range of the remote sensor ([*MinMeasuredValue* – *MaxMeasuredValue*])
8	Remote Temperature	A remote temperature sensor is used as the sensor for the regulation of the pump. *EffectiveControlModeI* is Constant temperature, and setpoint is interpreted as a percentage of the range of the remote sensor ([*MinMeasuredValue* – *MaxMeasuredValue*])
9 – 15	Reserved	-

6.2.2.2.2.2 *EffectiveOperationMode* Attribute

The *EffectiveOperationMode* attribute specifies current effective operation mode of the pump. The value of the *EffectiveOperationMode* attribute is the same as the *OperationMode* attribute of the Pump settings attribute set, except when it is overridden locally. See section 6.2.2.2.3.1 for a detailed description of the operation and control of the pump.

This attribute is read only.

Valid range is defined by the operation modes listed in Table 6.1.

6.2.2.2.2.3 *EffectiveControlMode* Attribute

The *EffectiveControlMode* attribute specifies the current effective control mode of the pump.

The *EffectiveControlMode* attribute contains the control mode that currently applies to the pump. It will have the value of the *ControlMode* attribute, unless a remote sensor is used as the sensor for regulation of the pump. In this case, *EffectiveControlMode* will display Constant pressure, Constant flow or Constant temperature if the remote sensor is a pressure sensor, a flow sensor or a temperature sensor respectively, regardless of the value of the *ControlMode* attribute.

See section 6.2.2.2.3.1 for detailed description of the operation and control of the pump. This attribute is read only.

Valid range is defined by the control modes listed in Table 6.8.

6.2.2.2.2.4 *Capacity* Attribute

The *Capacity* attribute specifies the actual capacity of the pump as a percentage of the effective maximum setpoint value. It is updated dynamically as the speed of the pump changes.

This attribute is read only. If the value is not available (the measurement or estimation of the speed is done in the pump), this attribute will contain the invalid value.

Valid range is 0 % to 163.835% (0.005 % granularity). Although the *Capacity* attribute is a signed value, values of capacity less than zero have no physical meaning.
The value -163.840 % (0x8000) indicates that this value is invalid.

6.2.2.2.2.5 *Speed* Attribute

The *Speed* attribute specifies the actual speed of the pump measured in RPM. It is updated dynamically as the speed of the pump changes.

This attribute is read only. If the value is not available (the measurement or estimation of the speed is done in the pump), this attribute will contain the invalid value.

Valid range is 0 to 65.534 RPM
The value 65.535 RPM (0xffff) indicates that this value is invalid.

6.2.2.2.2.6 *LifetimeRunningHours* Attribute

The *LifetimeRunningHours* attribute specifies the accumulated number of hours, that the pump has been powered and the motor has been running. It is updated dynamically as it increases. It is preserved over power cycles of the pump. if *LifeTimeRunningHours* rises above maximum value it “rolls over” and starts at 0 (zero).

This attribute is writeable, in order to allow setting to an appropriate value after maintenance. If the value is not available, this attribute will contain the invalid value.

Valid range is 0 to 16,777,214 hrs.
The value 16,777,215 (0xffffff) indicates that this value is unknown.

6.2.2.2.2.7 *Power* Attribute

The *Power* attribute specifies the actual power consumption of the pump in Watts. The value of the *Power* attribute is updated dynamically as the power consumption of the pump changes.

This attribute is read only. If the value is not available (the measurement of power consumption is not done in the pump), this attribute will display the invalid value.

Valid range is 0 to 16,777,214 Watts.
The value 16,777,215 (0xffffff) indicates that this value is unknown.

6.2.2.2.2.8 *LifetimeEnergyConsumed* Attribute

The *LifetimeEnergyConsumed* attribute specifies the accumulated energy consumption of the pump through the entire lifetime of the pump in kWh. The value of the *LifetimeEnergyConsumed* attribute is updated dynamically as the energy consumption of the pump increases. If *LifetimeEnergyConsumed* rises above maximum value it "rolls over" and starts at 0 (zero).

This attribute is writeable, in order to allow setting to an appropriate value after maintenance.

Valid range is 0 kWh to 4,294,967,294 kWh.
The value 4,294,967,295 (0xffffffff) indicates that this value is unknown.

6.2.2.2.3 Pump Settings Attribute Set

The pump settings attribute set contains the attributes summarized in Table 6.6.

Table 6.6 Attributes of the Pump Settings Attribute Set

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0020	*OperationMode*	8-bit enumeration	0x00 – 0xfe	Read / Write	0x0	M
0x0021	*ControlMode*	8-bit enumeration	0x00 – 0xfe	Read / Write	0x0	O
0x0022	*AlarmMask*	16-bit bitmap	-	Read only	-	O

6.2.2.2.3.1 *OperationMode* Attribute

The *OperationMode* attribute specifies the operation mode of the pump. This attribute shall have one of the values listed in Table 6.7.

The actual operating mode of the pump is a result of the setting of the attributes *OperationMode*, *ControlMode* and the optional connection of a remote sensor. The operation and control is prioritized as shown in the scheme in Figure 6.3:

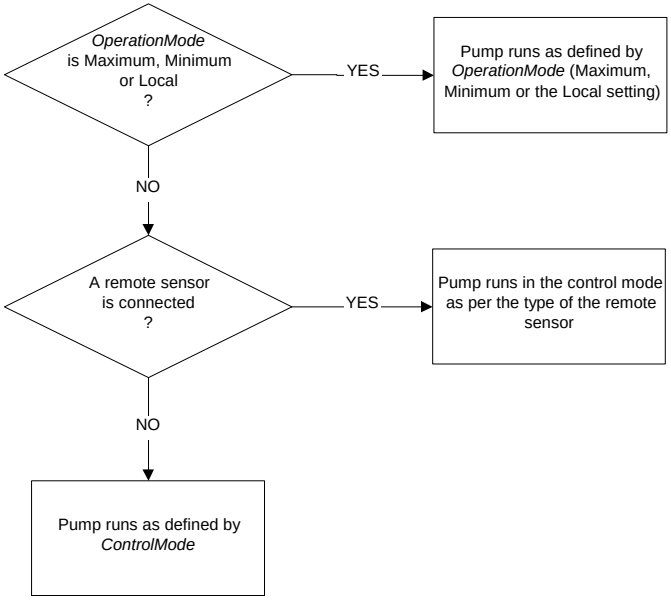

Figure 6.3 Priority Scheme of Pump Operation and Control

If the *OperationMode* attribute is Maximum, Minimum or Local, the *OperationMode* attribute decides how the pump is operated.

If the *OperationMode* attribute is Normal and a remote sensor is connected to the pump, the type of the remote sensor decides the control mode of the pump. A connected remote pressure sensor will make the pump run in control mode Constant pressure and vice versa for flow and temperature type sensors. This is regardless of the setting of the *ControlMode* attribute.

If the *OperationMode* attribute is Normal and no remote sensor is connected, the control mode of the pump is decided by the *ControlMode* attribute.

OperationMode may be changed at any time, even when the pump is running. The behavior of the pump at the point of changing the value of the *OperationMode* attribute is vendor specific.

Table 6.7 Values of the *OperationMode* Attribute

Attribute Value	Name	Explanation
0	Normal	The pump is controlled by a setpoint, as defined by a connected remote sensor or by the *ControlMode* attribute. (N.B. The setpoint is an internal variable which may be controlled between 0% and 100%, e.g. by means of the Level Control cluster 3.10)
1	Minimum	This value sets the pump to run at the minimum possible speed it can without being stopped
2	Maximum	This value sets the pump to run at its maximum possible speed
3	Local	This value sets the pump to run with the local settings of the pump, regardless of what these are
4-254	Reserved	Reserved for future use

6.2.2.2.3.2 *ControlMode* Attribute

The *ControlMode* attribute specifies the control mode of the pump. This attribute shall have one of the values listed in Table 6.8.

See section 6.2.2.2.3.1 for detailed description of the operation and control of the pump.

ControlMode may be changed at any time, even when the pump is running. The behavior of the pump at the point of changing is vendor-specific.

Table 6.8 Values of the *ControlMode* Attribute

Attribute Value	Name	Explanation
0	Constant speed	The pump is running at a constant speed. The setpoint is interpreted as a percentage of the *MaxSpeed* attribute
1	Constant pressure	The pump will regulate its speed to maintain a constant differential pressure over its flanges. The setpoint is interpreted as a percentage of the range of the sensor used for this control mode. In case of the internal pressure sensor, this will be the range derived from the [*MinConstPressure* - *MaxConstPressure*] attributes. In case of a remote pressure sensor, this will be the range derived from the [*MinMeasuredValue* – *MaxMeasuredValue*] attributes of the remote pressure sensor.
2	Proportional pressure	The pump will regulate its speed to maintain a constant differential pressure over its flanges. The setpoint is interpreted as a percentage of the range derived of the [*MinCompPressure* - *MaxCompPressure*] attributes. The internal setpoint will be lowered (compensated) dependant on the flow in the pump (lower flow => lower internal setpoint)
3	Constant flow	The pump will regulate its speed to maintain a constant flow through the pump. The setpoint is interpreted as a percentage of the range of the sensor used for this control mode. In case of the internal flow sensor, this will be the range derived from the [*MinConstFlow* - *MaxConstFlow*] attributes. In case of a remote flow sensor, this will be the range derived from the [*MinMeasuredValue* – *MaxMeasuredValue*] attributes of the remote flow sensor.
4	Reserved	-
5	Constant temperature	The pump will regulate its speed to maintain a constant temperature. The setpoint is interpreted as a percentage of the range of the sensor used for this control mode. In case of the internal temperature sensor, this will be the range derived from the [*MinConstTemp* - *MaxConstTemp*] attributes. In case of a remote temperature sensor, this will be the range derived from the [*MinMeasuredValue* – *MaxMeasuredValue*] attributes of the remote temperature sensor.

Table 6.8 Values of the *ControlMode* Attribute (Continued)

Attribute Value	Name	Explanation
6	Reserved	-
7	Automatic	The operation of the pump is automatically optimized to provide the most suitable performance with respect to comfort and energy savings. This behavior is manufacturer defined. The pump can be stopped by setting the setpoint of the level control cluster to 0 of by using the On/Off cluster. If the pump is started (at any setpoint), the speed of the pump is entirely determined by the pump.
8-254	Reserved	-

6.2.2.2.3.3 *AlarmMask* Attribute

The *AlarmMask* attribute specifies whether each of the alarms listed in Table 6.9 is enabled. When the bit number corresponding to the alarm code is set to 1, the alarm is enabled, else it is disabled. Bits not corresponding to a code in the table (bits 14, 15) are reserved.

When the Alarms cluster is implemented on a device, and one of the alarm conditions included in this table occurs, an alarm notification is generated, with the alarm code field set as listed in the table.

Table 6.9 Alarm Codes

Alarm Code	Alarm Condition
0	Supply voltage too low
1	Supply voltage too high
2	Power missing phase
3	System pressure too low
4	System pressure too high
5	Dry running
6	Motor temperature too high
7	Pump motor has fatal failure
8	Electronic temperature too high
9	Pump blocked
10	Sensor failure

Table 6.9 Alarm Codes (Continued)

Alarm Code	Alarm Condition
11	Electronic non fatal failure
12	Electronic fatal failure
13	General fault

6.2.2.3 Commands Received

None.

6.2.2.4 Commands Generated

None.

6.2.2.5 Attribute Reporting

This cluster shall support attribute reporting using the Report Attributes command, according to the minimum and maximum reporting interval, reportable change, and timeout period settings described in the ZCL Foundation Specification (see 2.4.7).

The following attributes shall be reported:

PumpStatus
Capacity

6.2.3 Client

6.2.3.1 Dependencies

None

6.2.3.2 Attributes

The client supports no attributes.

6.2.3.3 Commands Received

The client receives no cluster specific commands.

6.2.3.4 Commands Generated

The client generates no cluster specific commands.

6.3 Thermostat Cluster

6.3.1 Overview

This cluster provides an interface to the functionality of a thermostat.

6.3.2 Server

6.3.2.1 Dependencies

For alarms to be generated by this cluster, the Alarms server cluster (see 3.11) shall be included on the same endpoint. For remote temperature sensing, the Temperature Measurement client cluster (see 4.4) may be included on the same endpoint. For occupancy sensing, the Occupancy Sensing client cluster (see 4.8) may be included on the same endpoint.

6.3.2.2 Attributes

For convenience, the attributes defined in this specification are arranged into sets of related attributes; each set can contain up to 16 attributes. Attribute identifiers are encoded such that the most significant three nibbles specify the attribute set and the least significant nibble specifies the attribute within the set. The currently defined attribute sets for Thermostat are listed in Table 6.10.

Table 6.10 Currently Defined Thermostat Attribute Sets

Attribute Set Identifier	Description
0x000	Thermostat Information
0x001	Thermostat Settings
0x002 – 0x3ff	Reserved
0x400 – 0xfff	Reserved for vendor specific attributes

6.3.2.2.1 Thermostat Information Attribute Set

The Thermostat Information attribute set contains the attributes summarized in Table 6.11.

Table 6.11 Attributes of the Thermostat Information Attribute Set

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0000	*LocalTemperature*	Signed 16-bit integer	0x954d – 0x7fff	Read	-	M
0x0001	*OutdoorTemperature*	Signed 16-bit integer	0x954d – 0x7fff	Read	-	O
0x0002	*Ocupancy*	8-bit bitmap	0000000x	Read	00000000	O
0x0003	*AbsMinHeatSetpointLimit*	Signed 16-bit integer	0x954d – 0x7fff	Read	0x02bc (7°C)	O
0x0004	*AbsMaxHeatSetpointLimit*	Signed 16-bit integer	0x954d – 0x7fff	Read	0x0bb8 (30°C)	O
0x0005	*AbsMinCoolSetpointLimit*	Signed 16-bit integer	0x954d – 0x7fff	Read	0x0640 (16°C)	O
0x0006	*AbsMaxCoolSetpointLimit*	Signed 16-bit integer	0x954d – 0x7fff	Read	0x0c80 (32°C)	O
0x0007	*PICoolingDemand*	Unsigned 8-bit integer	0x00 – 0x64	Read	-	O
0x0008	*PIHeatingDemand*	Unsigned 8-bit integer	0x00 – 0x64	Read	-	O

6.3.2.2.1.1 *LocalTemperature* Attribute

LocalTemperature represents the temperature in degrees Celsius, as measured locally or remotely (over the network) as follows:

LocalTemperature = 100 x temperature in degrees Celsius.

Where -273.15°C <= temperature <= 327.67 ºC, corresponding to a *LocalTemperature* in the range 0x954d to 0x7fff.

The maximum resolution this format allows is 0.01 ºC.

A *LocalTemperature* of 0x8000 indicates that the temperature measurement is invalid.

6.3.2.2.1.2 *OutdoorTemperature* Attribute

OutdoorTemperature represents the outdoor temperature in degrees Celsius, as measured locally or remotely (over the network). It is measured as described for *LocalTemperature*.

6.3.2.2.1.3 *Occupancy* Attribute

Occupancy specifies whether the heated/cooled space is occupied or not, as measured locally or remotely (over the network). If bit 0 = 1, the space is occupied, else it is unoccupied. All other bits are reserved.

6.3.2.2.1.4 *AbsMinHeatSetpointLimit* Attribute

The *MinHeatSetpointLimit* attribute specifies the absolute minimum level that the heating setpoint may be set to. This is a limitation imposed by the manufacturer. The value is calculated as described in the LocalTemperature attribute.

6.3.2.2.1.5 *AbsMaxHeatSetpointLimit* Attribute

The *MaxHeatSetpointLimit* attribute specifies the absolute maximum level that the heating setpoint may be set to. This is a limitation imposed by the manufacturer. The value is calculated as described in the LocalTemperature attribute.

6.3.2.2.1.6 *AbsMinCoolSetpointLimit* Attribute

The *MinCoolSetpointLimit* attribute specifies the absolute minimum level that the cooling setpoint may be set to. This is a limitation imposed by the manufacturer. The value is calculated as described in the LocalTemperature attribute.

6.3.2.2.1.7 *AbsMaxCoolSetpointLimit* Attribute

The *MaxCoolSetpointLimit* attribute specifies the absolute maximum level that the cooling setpoint may be set to. This is a limitation imposed by the manufacturer. The value is calculated as described in the LocalTemperature attribute.

6.3.2.2.1.8 *PICoolingDemand* Attribute

The *PICoolingDemand* attribute is 8 bits in length and specifies the level of cooling demanded by the PI (proportional integral) control loop in use by the thermostat (if any), in percent. This value is 0 when the thermostat is in “off” or “heating” mode.

This attribute is reported regularly and may be used to control a heating device.

6.3.2.2.1.9 *PIHeatingDemand* Attribute

The *PIHeatingDemand* attribute is 8 bits in length and specifies the level of heating demanded by the PI loop in percent. This value is 0 when the thermostat is in “off” or “cooling” mode.

This attribute is reported regularly and may be used to control a cooling device.

6.3.2.2.2 Thermostat Settings Attribute Set

The Thermostat settings attribute set contains the attributes summarized in Table 6.12.

Table 6.12 Attributes of the Thermostat Settings Attribute Set

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0010	*LocalTemperature Calibration*	Signed 8-bit integer	0xE7 – 0x19	Read / Write	0x00 (0°C)	O
0x0011	*OccupiedCooling Setpoint*	Signed 16-bit integer	*MinCoolSetpoint Limit* – *MaxCoolSetpoint Limit*	Read / Write	0x0a28 (26°C)	M
0x0012	*OccupiedHeating Setpoint*	Signed 16-bit integer	*MinHeatSetpoint Limit* – *MaxHeatSetpoint Limit*	Read / Write	0x07d0 (20°C)	M
0x0013	*UnoccupiedCooling Setpoint*	Signed 16-bit integer	*MinCoolSetpoint Limit* – *MaxCoolSetpoint Limit*	Read / Write	0x0a28 (26°C)	O
0x0014	*UnoccupiedHeating Setpoint*	Signed 16-bit integer	*MinHeatSetpoint Limit* – *MaxHeatSetpoint Limit*	Read / Write	0x07d0 (20°C)	O
0x0015	*MinHeatSetpoint Limit*	Signed 16-bit integer	0x954d – 0x7fff	Read / Write	0x02bc (7°C)	O
0x0016	*MaxHeatSetpoint Limit*	Signed 16-bit integer	0x954d – 0x7fff	Read / Write	0x0bb8 (30°C)	O
0x0017	*MinCoolSetpoint Limit*	Signed 16-bit integer	0x954d – 0x7fff	Read / Write	0x02bc (7°C)	O
0x0018	*MaxCoolSetpoint Limit*	Signed 16-bit integer	0x954d – 0x7fff	Read / Write	0x0bb8 (30°C)	O

Table 6.12 Attributes of the Thermostat Settings Attribute Set (Continued)

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0019	*MinSetpointDead Band*	Signed 8-bit integer	0x0a – 0x19	Read / Write	0x19 (2.5°C)	O
0x001a	*RemoteSensing*	8-bit bitmap	00000xxx	Read / Write	0	O
0x001b	*ControlSequenceOf Operation*	8-bit enumeration	0x00 – 0x05	Read / Write	0x04	M
0x001c	*SystemMode*	8-bit enumeration	See Table 6.15	Read / Write	0x01	M
0x001d	*AlarmMask*	8-bit bitmap	00000xxx	Read only	0	O

6.3.2.2.2.1 *LocalTemperatureCalibration* Attribute

The *LocalTemperatureCalibration* attribute specifies the offset that can be added/ subtracted to the actual displayed room temperature, in steps of 0.1°C. The range of this offset is –2.5 °C to +2.5 °C).

6.3.2.2.2.2 *OccupiedCoolingSetpoint* Attribute

The *OccupiedCoolingSetpoint* attribute is 16 bits in length and specifies the cooling mode setpoint when the room is occupied. It shall be set to a value in the range defined by the MinCoolSetpointLimit and MaxCoolSetpointLimit attributes. The value is calculated as described in the LocalTemperature attribute.

The OccupiedHeatingSetpoint attribute shall always be below the value specified in the OccupiedCoolingSetpoint by at least SetpointDeadband. If an attempt is made to set it such that this condition is violated, a default response command with the status code INVALID_VALUE (see 2.5.3) shall be returned. This shall apply to all attempts to set values of attributes which violate similar conditions.

If it is unknown if the room is occupied or not, this attribute shall be used as the cooling mode setpoint.

6.3.2.2.2.3 *OccupiedHeatingSetpoint* Attribute

The *OccupiedHeatingSetpoint* attribute is 16 bits in length and specifies the heating mode setpoint when the room is occupied. It shall be set to a value in the range defined by the *MinHeatSetpointLimit* and *MaxHeatSetpointLimit* attributes. The value is calculated as described in the *LocalTemperature* attribute. The

OccupiedCoolingSetpoint attribute shall always be above the value specified in the *OccupiedHeatingSetpoint* by at least *SetpointDeadband*.

If it is unknown if the room is occupied or not, this attribute shall be used as the cooling mode setpoint.

6.3.2.2.2.4 *UnoccupiedCoolingSetpoint* Attribute

The *UnoccupiedCoolingSetpoint* attribute is 16 bits in length and specifies the cooling mode setpoint when the room is unoccupied. It shall be set to a value in the range defined by the *MinCoolSetpointLimit* and *MaxCoolSetpointLimit* attributes. The value is calculated as described in the *LocalTemperature* attribute. The *UnoccupiedHeatingSetpoint* attribute shall always be below the value specified in the *UnoccupiedCoolingSetpoint* by at least *SetpointDeadband*.

If it is unknown if the room is occupied or not, this attribute shall not be used.

6.3.2.2.2.5 *UnoccupiedHeatingSetpoint* Attribute

The *UnoccupiedHeatingSetpoint* attribute is 16 bits in length and specifies the heating mode setpoint when the room is unoccupied. It shall be set to a value in the range defined by the *MinHeatSetpointLimit* and *MaxHeatSetpointLimit* attributes. The value is calculated as described in the *LocalTemperature* attribute. The *UnoccupiedCoolingSetpoint* attribute shall always be below the value specified in the *UnoccupiedHeatingSetpoint* by at least *SetpointDeadband*.

If it is unknown if the room is occupied or not, this attribute shall not be used.

6.3.2.2.2.6 *MinHeatSetpointLimit* Attribute

The *MinHeatSetpointLimit* attribute specifies the minimum level that the heating setpoint may be set to. The value is calculated as described in the LocalTemperature attribute. It must be greater than or equal to *AbsMinHeatSetpointLimit.* If this attribute is not present, it shall be taken as equal to *AbsMinHeatSetpointLimit.*

This attribute, and the following three attributes, allow the user to define setpoint limits more constrictive than the manufacturer imposed ones. Limiting users (e.g., in a commercial building) to such setpoint limits can help conserve power.

6.3.2.2.2.7 *MaxHeatSetpointLimit* Attribute

The *MaxHeatSetpointLimit* attribute specifies the maximum level that the heating setpoint may be set to. The value is calculated as described in the LocalTemperature attribute. It must be less than or equal to *AbsMaxHeatSetpointLimit.* If this attribute is not present, it shall be taken as equal to *AbsMaxHeatSetpointLimit.*

6.3.2.2.2.8 *MinCoolSetpointLimit* Attribute

The *MinCoolSetpointLimit* attribute specifies the minimum level that the cooling setpoint may be set to. The value is calculated as described in the LocalTemperature attribute. It must be greater than or equal to *AbsMinCoolSetpointLimit.* If this attribute is not present, it shall be taken as equal to *AbsMinCoolSetpointLimit.*

6.3.2.2.2.9 *MaxCoolSetpointLimit* Attribute

The *MaxCoolSetpointLimit* attribute specifies the maximum level that the cooling setpoint may be set to. The value is calculated as described in the LocalTemperature attribute. It must be less than or equal to *AbsMaxCoolSetpointLimit.* If this attribute is not present, it shall be taken as equal to *AbsMaxCoolSetpointLimit.*

6.3.2.2.2.10 *MinSetpointDeadBand* Attribute

The *MinSetpointDeadBand* attribute specifies the minimum difference between the Heat Setpoint and the Cool SetPoint, in steps of 0.1°C. Its range is 0x0a to 0x19 (1°C to 2.5°C).

6.3.2.2.2.11 *RemoteSensing* Attribute

The RemoteSensing attribute is an 8-bit bitmap that specifies whether the local temperature, outdoor temperature and occupancy are being sensed by internal sensors or remote networked sensors. The meanings of individual bits are detailed in Table 6.13.

Table 6.13 *RemoteSensing* Attribute Bit Values

Bit Number	Description
0	0 – local temperature sensed internally 1 – local temperature sensed remotely
1	0 – outdoor temperature sensed internally 1 – outdoor temperature sensed remotely
2	0 – occupancy sensed internally 1 – occupancy sensed remotely
3 - 7	Reserved

6.3.2.2.2.12 *ControlSequenceOfOperation* Attribute

The *ControlSequenceOfOperation* attribute specifies the overall operating environment of the thermostat, and thus the possible system modes that the

thermostat can operate in. It shall be set to one of the non-reserved values in Table 6.14. (Note - it is not mandatory to support all values).

Table 6.14 *ControlSequenceOfOperation* Attribute Values

Attribute Value	Description	Possible Values of *SystemMode*
0x00	Cooling Only	Heat and Emergency are not possible
0x01	Cooling With Reheat	Heat and Emergency are not possible
0x02	Heating Only	Cool and precooling (see 1.3.2) are not possible
0x03	Heating With Reheat	Cool and precooling are not possible
0x04	Cooling and Heating 4-pipes (see 1.3.2)	All modes are possible
0x05	Cooling and Heating 4-pipes with Reheat	All modes are possible
0x06 – 0xfe	Reserved	-

6.3.2.2.3 *SystemMode* Attribute

The *SystemMode* attribute specifies the current operating mode of the thermostat,. It shall be set to one of the non-reserved values in Table 6.15, as limited by Table 6.16. (Note - it is not mandatory to support all values).

Table 6.15 *SystemMode* Attribute Values

Attribute Value	Description
0x00	Off
0x01	Auto
0x03	Cool
0x04	Heat
0x05	Emergency heating
0x06	Precooling (see 1.3.2)
0x07	Fan only
0x02, 0x08 – 0xfe	Reserved

The interpretation of the Heat, Cool and Auto values of *SystemMode* is shown in Table 6.16.

Table 6.16 Interpretation of *SystemMode* Values

Attribute Values	Temperature Below Heat Setpoint	Temperature Between Heat Setpoint and Cool Setpoint	Temperature Above Cool Setpoint
Heat	Temperature below target	Temperature on target	Temperature on target
Cool	Temperature on target	Temperature on target	Temperature above target
Auto	Temperature below target	Temperature on target	Temperature above target

6.3.2.2.4 *AlarmMask* Attribute

The *AlarmMask* attribute specifies whether each of the alarms listed in Table 6.17 is enabled. When the bit number corresponding to the alarm code is set to 1, the alarm is enabled, else it is disabled. Bits not corresponding to a code in the table are reserved.

When the Alarms cluster is implemented on a device, and one of the alarm conditions included in this table occurs, an alarm notification is generated, with the alarm code field set as listed in the table.

Table 6.17 Alarm Codes

Alarm Code	Alarm Condition
0	Initialization failure. The device failed to complete initialization at power-up.
1	Hardware failure
2	Self-calibration failure

6.3.2.3 Commands Received

The command IDs for the Thermostat cluster are listed in Table 6.18.

Table 6.18 Command IDs for the Thermostat Cluster

Command Identifier Field Value	Description
0x00	Setpoint Raise/Lower
0x01 – 0xff	Reserved

6.3.2.3.1 Setpoint Raise/Lower Command

6.3.2.3.1.1 Payload Format

The Setpoint Raise/Lower command payload shall be formatted as illustrated in Figure 6.4.

Bits	8	8
Data Type	8-bit enumeration	Signed 8-bit integer
Field Name	Mode	Amount

Figure 6.4 Format of the Setpoint Raise/Lower Command Payload

6.3.2.3.1.2 Mode Field

The mode field shall be set to one of the non-reserved values in Table 6.19. It specifies which setpoint is to be configured. If it is set to auto, then both setpoints shall be adjusted.

Table 6.19 Mode Field Values for Setpoint Raise/Lower Command

Mode Field Value	**Description**
0x00	Heat (adjust Heat Setpoint)
0x01	Cool (adjust Cool Setpoint)
0x02	Both (adjust Heat Setpoint and Cool Setpoint)
0x03 – 0xff	Reserved

6.3.2.3.1.3 Amount Field

The amount field is a signed 8-bit integer that specifies the amount the setpoint(s) are to be a increased (or decreased) by, in steps of 0.1°C.

6.3.2.3.2 Effect on Receipt

The attributes for the indicated setpoint(s) shall be increased by the amount specified in the Amount field.

6.3.2.4 Commands Generated

No commands are generated by the server cluster.

6.3.2.5 Attribute Reporting

This cluster shall support attribute reporting using the Report Attributes command and according to the minimum and maximum reporting interval and reportable change settings described in the ZCL Foundation specification (see 2.4.7) and whenever they change. The following attributes shall be reported:

- *LocalTemperature*
- *PICoolingDemand*
- *PIHeatingDemand*

Other attributes may optionally be reported.

6.3.3 Client

6.3.3.1 Dependencies

None.

6.3.3.2 Attributes

The Client cluster has no attributes.

6.3.3.3 Commands Received

The client receives no cluster specific commands.

6.3.3.4 Commands Generated

The client cluster generates the commands received by the server cluster, i.e. those detailed in 6.3.2.3, as required by the application.

6.4 Fan Control

6.4.1 Overview

This cluster specifies an interface to control the speed of a fan as part of a heating / cooling system.

6.4.2 Server

6.4.2.1 Dependencies

None.

6.4.2.2 Attributes

The Fan Control Status attribute set contains the attributes summarized in Table 6.20.

Table 6.20 Attributes of the Fan Control Cluster

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0000	*FanMode*	8-bit enumeration	0x00 – 0x06	Read/ Write	0x05 (auto)	M
0x0001	*FanModeSequence*	8-bit enumeration	0x00 – 0x04	Read/ Write	0x02	M

6.4.2.2.1 *FanMode* Attribute

The *FanMode* attribute is an 8-bit value that specifies the current speed of the fan. It shall be set to one of the non-reserved values in Table 6.21.

Table 6.21 *FanMode* Attribute Values

Value	Description
0x00	Off
0x01	Low
0x02	Medium
0x03	High
0x04	On
0x05	Auto (the fan speed is self-regulated)
0x06	Smart (when the heated/cooled space is occupied, the fan is always on)
0x07– 0xfe	Reserved

Note that for Smart mode, information must be available as to whether the heated/ cooled space is occupied. This may be accomplished by use of the Occupancy Sensing cluster (see 4.8).

6.4.2.2.2 *FanModeSequence* Attribute

The *FanModeSequence* attribute is an 8-bit value that specifies the possible fan speeds that the thermostat can set. It shall be set to one of the non-reserved values

in Table 6.22. (**Note:** 'Smart' is not in this table, as this mode resolves to one of the other modes depending on occupancy).

Table 6.22 *FanSequenceOperation* Attribute Values

Attribute Value	Description
0x00	Low/Med/High
0x01	Low/High
0x02	Low/Med/High/Auto
0x03	Low/High/Auto
0x04	On/Auto
0x05 – 0xfe	Reserved

6.4.2.3 Commands Received

No cluster specific commands are received by the server.

6.4.2.4 Commands Generated

No cluster specific commands are generated by the server.

6.4.3 Client

6.4.3.1 Dependencies

None.

6.4.3.2 Attributes

The Client cluster has no attributes.

6.4.3.3 Commands Received

No cluster specific commands are received by the server.

6.4.3.4 Commands Generated

No cluster specific commands are generated by the server.

6.5 Dehumidification Control

6.5.1 Overview

This cluster provides an interface to dehumidification functionality.

6.5.2 Server

6.5.2.1 Dependencies

None.

6.5.2.2 Attributes

For convenience, the attributes defined in this specification are arranged into sets of related attributes; each set can contain up to 16 attributes. Attribute identifiers are encoded such that the most significant nibble specifies the attribute set and the least significant nibble specifies the attribute within the set. The currently defined attribute set for the dehumidification control cluster is listed in Table 6.23.

Table 6.23 Dehumidification Control Attribute Sets

Attribute Set Identifier	Description
0x000	Dehumidification Information
0x001	Dehumidification Settings
0x002 – 0xfff	Reserved

6.5.2.2.1 Dehumidification Information Attribute Set

The Dehumidification Information attribute set contains the attributes summarized in Table 6.24.

Table 6.24 Dehumidification Information Attribute Set

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0000	*RelativeHumidity*	Unsigned 8-bit integer	0x00 – 0x64	Read only	-	O
0x0001	*DehumidificationCooling*	Unsigned 8-bit integer	0 - *DehumidificationMaxCool*	Read only	-	M

6.5.2.2.1.1 *RelativeHumidity* Attribute

The *RelativeHumidity* attribute is an 8-bit value that represents the current relative humidity (in %) measured by a local or remote sensor. The valid range ix 0x00 – 0x64 (0% to 100%).

6.5.2.2.1.2 *DehumidificationCooling* Attribute

The *DehumidificationCooling* attribute is an 8-bit value that specifies the current dehumidification cooling output (in %). The valid range is 0 to *DehumidificationMaxCool*.

6.5.2.2.2 Dehumidification Settings Attribute Set

The Dehumidification Settings attribute set contains the attributes summarized in Table 6.25.

Table 6.25 Dehumidification Settings Attribute Set

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0010	*RHDehumidificationSetpoint*	Unsigned 8-bit integer	0x1E – 0x64	Read/ Write	0x32	M
0x0011	*RelativeHumidityMode*	8-bit enumeration	0x00 – 0x01	Read/ Write	0x00	O
0x0012	*DehumidificationLockout*	8-bit enumeration	0x00 – 0x01	Read/ Write	0x01	O
0x0013	*DehumidificationHysteresis*	Unsigned 8-bit integer	0x02 – 0x14	Read/ Write	0x02	M
0x0014	*DehumidificationMaxCool*	Unsigned 8-bit integer	0x14 – 0x64	Read/ Write	0x14	M
0x0015	*RelativeHumidityDisplay*	8-bit enumeration	0x00 – 0x01	Read/ Write	0x00	O

6.5.2.2.2.1 *RHDehumidificationSetpoint* Attribute

The *RHDehumidificationSetpoint* attribute is an 8-bit value that represents the relative humidity (in %) at which dehumidification occurs. The valid range ix 0x1E – 0x64 (30% to 100%).

6.5.2.2.2.2 *RelativeHumidityMode* Attribute

The *RelativeHumidityMode* attribute is an 8-bit value that specifies how the *RelativeHumidity* value is being updated. It shall be set to one of the non-reserved values in Table 6.26.

Table 6.26 *RelativeHumidityMode* Attribute Values

Attribute Value	Description
0x00	*RelativeHumidity* measured locally
0x01	*RelativeHumidity* updated over the network
0x02 – 0xff	Reserved

6.5.2.2.2.3 *DehumidificationLockout* Attribute

The *DehumidificationLockout* attribute is an 8-bit value that specifies whether dehumidification is allowed or not. It shall be set to one of the non-reserved values in Table 6.27.

Table 6.27 *DehumidificationLockout* Attribute Values

Attribute Value	Description
0x00	Dehumidification is not allowed.
0x01	Dehumidification is allowed.
0x02 – 0xff	Reserved

6.5.2.2.2.4 *DehumidificationHysteresis* Attribute

The *DehumidificationHysteresis* attribute is an 8-bit value that specifies the hysteresis (in %) associated with *RelativeHumidity* value. The valid range ix 0x02 – 0x14 (2% to 20%).

6.5.2.2.2.5 *DehumidificationMaxCool* Attribute

The *DehumidificationMaxCool* attribute is an 8-bit value that specifies the maximum dehumidification cooling output (in %). The valid range ix 0x14 – 0x64 (20% to 100%).

6.5.2.2.2.6 *RelativeHumidityDisplay* Attribute

The *RelativeHumidityDisplay* attribute is an 8-bit value that specifies whether the *RelativeHumidity* value is displayed to the user or not. It shall be set to one of the non-reserved values in Table 6.28.

Table 6.28 *RelativeHumidityMode* Attribute Values

Attribute Value	Description
0x00	*RelativeHumidity* is not displayed
0x01	*RelativeHumidity* is displayed
0x02 – 0xff	Reserved

6.5.2.3 Commands Received

No commands are received by the server cluster except those to read / write attributes

6.5.2.4 Commands Generated

No commands are generated by the server cluster except responses to commands to read/write attributes, and attribute reports.

6.5.2.5 Attribute Reporting

This cluster shall support attribute reporting using the Report Attributes command and according to the minimum and maximum reporting interval settings described in the ZCL Foundation specification (see 2.4.7).

The following attribute shall be reported: *DehumidificationCooling*

This attribute shall also be reported whenever it changes (a minimum change is 1%).

Reports of this attribute may be used to control a remote dehumidifier device.

6.5.3 Client

6.5.3.1 Dependencies

None

6.5.3.2 Attributes

The client cluster has no attributes.

6.5.3.3 Commands Received

No commands are received by the server cluster except responses to commands to read/write attributes, and attribute reports.

6.5.3.4 Commands Generated

No commands are generated by the server cluster except those to read / write attributes, as required by the application

6.6 Thermostat User Interface Configuration Cluster

6.6.1 Overview

This cluster provides an interface to allow configuration of the user interface for a thermostat, or a thermostat controller device, that supports a keypad and LCD screen.

6.6.2 Server

6.6.2.1 Dependencies

None.

6.6.2.2 Attributes

The attributes of this cluster are summarized in Table 6.29.

Table 6.29 Thermostat User Interface Configuration Cluster

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0000	*TemperatureDisplayMode*	8-bit enumeration	0x00 – 0x01	Read/ Write	0x00 (Celsius)	M
0x0001	*KeypadLockout*	8-bit enumeration	0x00 – 0x05	Read/ Write	0x00 (no lockout)	M

6.6.2.2.1 *TemperatureDisplayMode* Attribute

The *TemperatureDisplayMode* attribute specifies the units of the temperature displayed on the thermostat screen. This attribute shall be set to one of the non-reserved values in Table 6.30.

Table 6.30 *DisplayMode* Attribute Values

Attribute Value	Description
0x00	Temperature in ^{o}C
0x01	Temperature in ^{o}F
0x02 – 0xff	Reserved

6.6.2.2.2 *KeypadLockout* Attribute

The *KeypadLockout* attribute specifies the level of functionality that is available to the user via the keypad. This attribute shall be set to one of the non-reserved values Table 6.31.

Table 6.31 *KeypadLockout* Attribute Values

Attribute Value	Description
0x00	No lockout
0x01	Level 1 lockout
0x02	Level 2 lockout
0x03	Level 3 lockout
0x04	Level 4 lockout
0x05	Level 5 lockout (least functionality available to the user)
0x06– 0xff	Reserved

The interpretation of the various levels is device dependent.

6.6.2.3 Commands Received

No commands are received by the server except those to read and write the attributes of the server.

6.6.2.4 Commands Generated

No commands are generated by the server except responses to commands to read and write the attributes of the server.

6.6.3 Client

6.6.3.1 Dependencies

None.

6.6.3.2 Attributes

The Client cluster has no attributes.

6.6.3.3 Commands Received

No commands are received by the server except responses to commands to read and write the attributes of the server.

6.6.3.4 Commands Generated

No commands are generated by the client except those to read and write the attributes of the server, as required by the application.

CHAPTER

7

CLOSURES SPECIFICATION

7.1 General Description

7.1.1 Introduction

The clusters specified in this document are for use typically in ZigBee applications involving closures (e.g. shades, windows doors), but may be used in any application domain.

7.1.2 Cluster List

This section lists the clusters specified in this document, and gives examples of typical usage for the purpose of clarification.

The clusters defined in this document are listed in Table 7.1.

Table 7.1 Clusters Specified in the Closures Functional Domain

Cluster Name	Description
Shade Configuration	Attributes and commands for configuring a shade
Door Lock	An interface to a generic way to secure a door

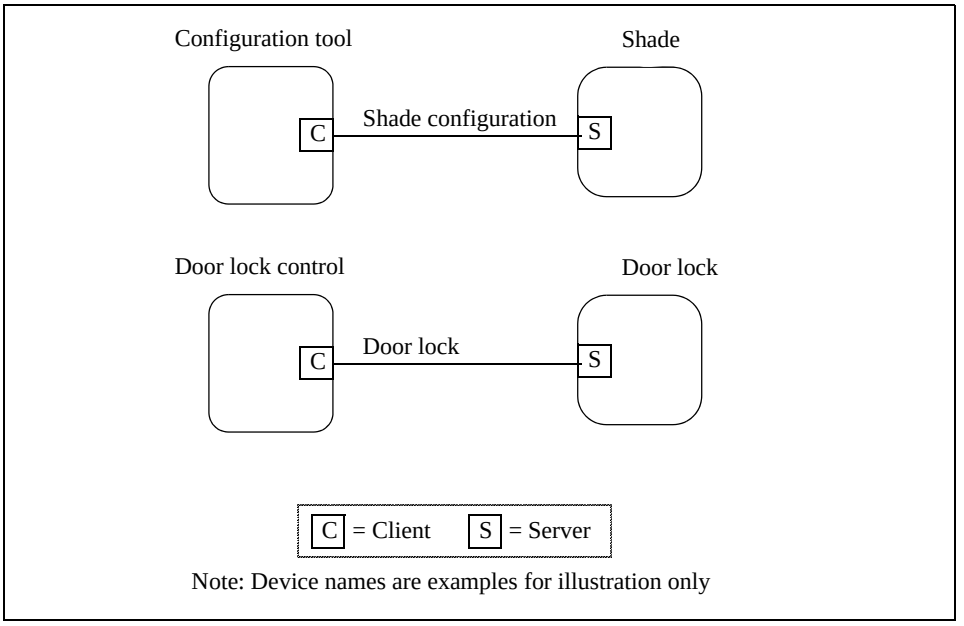

Figure 7.1 Typical Usage of the Closures Clusters

7.2 Shade Configuration Cluster

7.2.1 Overview

This cluster provides an interface for reading information about a shade, and configuring its open and closed limits.

7.2.2 Server

7.2.2.1 Dependencies

None

7.2.2.2 Attributes

For convenience, the attributes defined in this specification are arranged into sets of related attributes; each set can contain up to 16 attributes. Attribute identifiers are encoded such that the most significant three nibbles specify the attribute set

and the least significant nibble specifies the attribute within the set. The currently defined attribute sets are listed in Table 7.2.

Table 7.2 Shade Configuration Attribute Sets

Attribute Set Identifier	Description
0x000	Shade information
0x001	Shade settings
0x002 – 0xfff	Reserved

7.2.2.2.1 Shade Information Attribute Set

The Shade Information attribute set contains the attributes summarized in Table 7.3.

Table 7.3 Attributes of the Shade Information Attribute Set

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0000	*PhysicalClosedLimit*	Unsigned 16-bit integer	0x0001 – 0xfffe	Read only	-	O
0x0001	*MotorStepSize*	Unsigned 8-bit integer	0x00 – 0xfe	Read only	-	O
0x0002	*Status*	8-bit bitmap	0000 xxxx	Read / write	0000 0000	M

7.2.2.2.1.1 *PhysicalClosedLimit* Attribute

The *PhysicalClosedLimit* attribute indicates the most closed (numerically lowest) position that the shade can physically move to. This position is measured in terms of steps of the motor, taking the physical most open position of the shade as zero.

This attribute is for installation informational purposes only.

The value 0xffff indicates an invalid or unknown *PhysicalClosedLimit*.

7.2.2.2.1.2 *MotorStepSize* Attribute

The *MotorStepSize* attribute indicates the angle the shade motor moves for one step, measured in 1/10ths of a degree.

This attribute is for installation informational purposes only.

The value 0xff indicates an invalid or unknown step size.

7.2.2.2.1.3 *Status* Attribute

The *Status* attribute indicates the status of a number of shade functions, as shown in Table 7.4 Writing a value to this attribute only affects those bits with Read / Write access.

Table 7.4 Bit Values for the *Status* Attribute

Bit Number	Meaning	Access
0	Shade operational 0 = no 1 = yes	Read only
1	Shade adjusting 0 = no 1 = yes	Read only
2	Shade direction 0 = closing 1 = opening	Read only
3	Direction corresponding to forward direction of motor 0 = closing 1 = opening	Read / write
4 – 7	Reserved	-

7.2.2.2.2 Shade Settings Attribute Set

The Shade Settings attribute set contains the attributes summarized in Table 7.5.

Table 7.5 Attributes of the Shade Settings Attribute Set

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0010	*ClosedLimit*	Unsigned 16-bit integer	0x0001 – 0xfffe	Read/write	0x0001	M
0x0011	*Mode*	8-bit enumeration	0x00 – 0xfe	Read/write	0x00	M

7.2.2.2.2.1 *ClosedLimit* Attribute

The *ClosedLimit* attribute indicates the most closed position that the shade can move to. This position is measured in terms of steps of the motor, taking the physical most open position of the shade as zero. This attribute is set either by directly writing it, or by the following method.

When the Mode attribute is set to Configure, the shade is opening, and either the shade is stopped or it reaches its physical most open limit (if there is one – the

motor may continue to turn at the top), the zero point for the motor-step measurement system is set to the current position of the shade.

When the Mode attribute is set to Configure, the shade is closing, and either the shade is stopped or it reaches its physical closed limit, the *ClosedLimit* attribute is set to the current position of the shade, relative to the zero point set as described above.

7.2.2.2.2.2 *Mode* Attribute

The *Mode* attribute indicates the current operating mode of the shade, as shown in Table 7.6.

The value 0xff indicates an invalid or unknown mode.

Table 7.6 Values of the *Mode* Attribute

Attribute Value	Meaning
0x00	Normal
0x01	Configure
0x02 – 0xfe	Reserved

In configure mode, the *ClosedLimit* attribute may be set as described above.

7.2.2.3 Commands Received

No cluster specific commands are received by the server.

7.2.2.4 Commands Generated

No cluster specific commands are generated by the server.

7.2.3 Client

7.2.3.1 Dependencies

None

7.2.3.2 Attributes

The client has no attributes.

7.2.3.3 Commands Received

No cluster specific commands are received by the client.

7.2.3.4 Commands Generated

No cluster specific commands are generated by the client.

7.3 Door Lock Cluster

7.3.1 Overview

The door lock cluster provides an interface to a generic way to secure a door. The physical object that provides the locking functionality is abstracted from the cluster. The cluster has a small list of mandatory attributes and functions and a list of optional features.

7.3.2 Server

7.3.2.1 Dependencies

None

7.3.2.2 Attributes

The Door Lock cluster contains the attributes summarized in Table 7.7.

Table 7.7 Attributes of the Door Lock Cluster

Identifier	Name	Type	Access	Default	Mandatory / Optional	Reportable
0x0000	*LockState*	8-bit enumeration	Read only	-	M	Yes
0x0001	*LockType*	8-bit enumeration	Read only	-	M	No
0x0002	*ActuatorEnabled*	boolean	Read only	-	M	No
0x0003	*DoorState*	8-bit enumeration	Read only	-	O	Yes

Table 7.7 Attributes of the Door Lock Cluster (Continued)

Identifier	Name	Type	Access	Default	Mandatory / Optional	Reportable
0x0004	*DoorOpenEvents*	unsigned 32-bit integer	Read / write	-	O	No
0x0005	*DoorClosedEvents*	unsigned 32-bit integer	Read / write	-	O	No
0x006	*OpenPeriod*	unsigned 16-bit integer	Read / write	-	O	No

7.3.2.2.1 *LockState* Attribute

This attribute has the following possible values

0x00: Not fully locked
0x01: Locked
0x02: Unlocked
0x03 - 0xfe: Reserved (0xff = not defined)

7.3.2.2.2 *LockType* Attribute

This attribute has the following possible values

0x00: Deadbolt
0x01: Magnetic
0x02: Other
0x03 - 0xfe: Reserved (0xff = not defined)

7.3.2.2.3 *ActuatorEnabled* Attribute

This attribute has the following possible values

0b0: Disabled
0b1: Enabled

7.3.2.2.4 *DoorState* Attribute

This attribute has the following possible values

0x00: Open
0x01: Closed
0x02: Error (jammed)
0x02: Error (forced open)
0x02: Error (unspecified)
0x05- 0xfe: Reserved (0xff = not defined)

7.3.2.2.5 *DoorOpenEvents* Attribute

This attribute holds the number of door open events that have occurred since it was last zeroed.

7.3.2.2.6 *DoorClosedEvents* Attribute

This attribute holds the number of door closed events that have occurred since it was last zeroed.

7.3.2.2.7 *OpenPeriod* Attribute

This attribute holds the number of minutes the door has been open since the last time it transitioned from closed to open.

7.3.2.3 Commands Received

The commands received by the server are listed in Table 7.8.

Table 7.8 Commands Received by the Server

Command ID	Description	Mandatory / Optional
0x00	Lock Door	M
0x01	Unlock Door	M

7.3.2.3.1 Lock Door command

This command causes the lock device to lock the door. It has no payload.

7.3.2.3.2 Unlock Door command

This command causes the lock device to unlock the door. It has no payload.

7.3.2.4 Commands Generated

The commands generated by the server are listed in Table 7.9.

Table 7.9 Commands Generated by the Server

Command ID	Description	Mandatory / Optional
0x00	Lock Door Response	M
0x01	Unlock Door Response	M

7.3.2.4.1 Lock Door Response Command

This command is sent in response to a Lock Door command. Its payload shall be formatted as illustrated in Figure 7.2.

Bits	8
Data Type	8-bit enumeration
Field Name	Status

Figure 7.2 Format of the Lock Door Response Command Payload

The Status field shall be set to SUCCESS or FAILURE (see Table 2.17) depending on the result of the Lock Door command.

7.3.2.4.2 Unlock Door Response Command

This command is sent in response to an Unlock Door command. Its payload shall be formatted as illustrated in Figure 7.3.

Bits	8
Data Type	8-bit enumeration
Field Name	Status

Figure 7.3 Format of the Unlock Door Response Command Payload

The Status field shall be set to SUCCESS or FAILURE (see Table 2.17) depending on the result of the Unlock Door command.

7.3.2.5 Scene Table Extensions

If the Scene server cluster is implemented, the following extension field is added to the Scene table:

- ***LockState***

  When the *LockState* attribute is part of a Scene table, the attribute has an associated action; that is, setting the *LockState* to lock will command the lock to lock, and setting the *LockState* to unlocked will command the lock to unlock. Setting the *LockState* attribute to “not fully locked” is not supported.

  The transition time field in the Scene table will be treated as a delay before setting the *LockState* attribute, that is it is possible to activate a scene with the lock actuation some seconds later.

Locks that do not have an actuation mechanism should not support the Scene table extension.

7.3.3 Client

7.3.3.1 Dependencies

None

7.3.3.2 Attributes

The client has no attributes.

7.3.3.3 Commands Received

The client receives the cluster specific commands shown in Table 7.9.

7.3.3.4 Commands Generated

The client generates the cluster specific commands shown in Table 7.8.

CHAPTER

8

SECURITY AND SAFETY SPECIFICATION

8.1 General Description

8.1.1 Introduction

The clusters specified in this document are for use in ZigBee security and safety related applications.

The clusters currently defined are those that are used by wireless Intruder Alarm Systems (IAS). Intruder Alarm systems include functions for the detection of intruders and/or triggering, processing of information, notification of alarms and the means to operate the IAS.

Functions additional to those may be included in IAS providing they do not influence the correct operation of the mandatory functions. Components of other applications may be combined or integrated with a IAS, providing the performance of the IAS components is not adversely influenced.

8.1.2 Cluster List

This section lists the clusters specified in this document, and gives examples of typical usage for the purpose of clarification.

The clusters defined in this document are listed in Table 8.1.

Table 8.1 Clusters of the Security and Safety Functional Domain

Cluster Name	Description
IAS Zone	Attributes and commands for IAS security zone devices.
IAS ACE	Attributes and commands for IAS Ancillary Control Equipment.
IAS WD	Attributes and commands for IAS Warning Devices

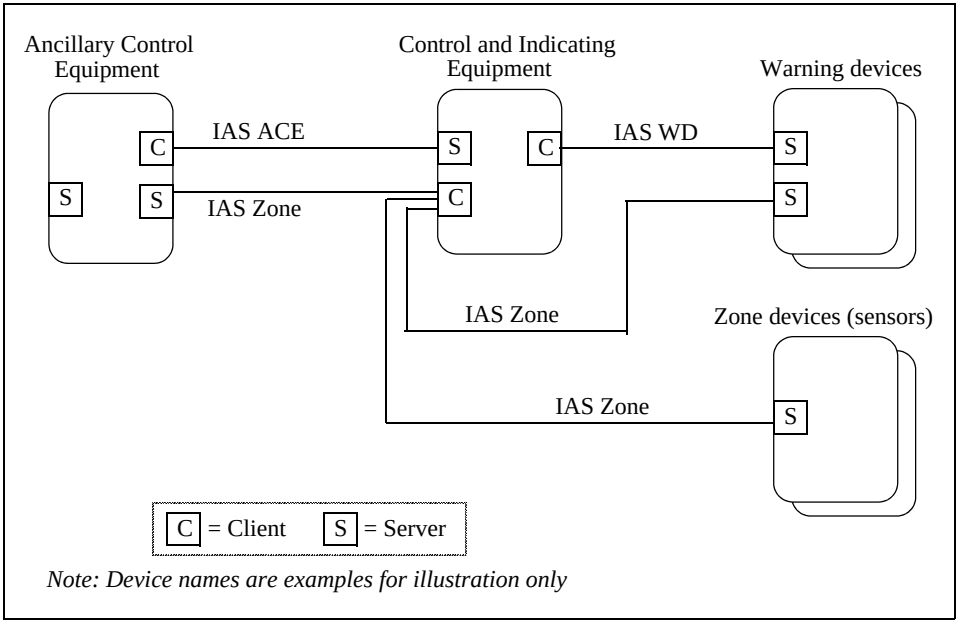

Figure 8.1 Typical Usage of the IAS Clusters

8.2 IAS Zone Cluster

8.2.1 Overview

The IAS Zone cluster defines an interface to the functionality of an IAS security zone device. IAS Zone supports up to two alarm types per zone, low battery reports and supervision of the IAS network.

8.2.2 Server

8.2.2.1 Dependencies

None.

8.2.2.2 Attributes

For convenience, the attributes defined in this specification are arranged into sets of related attributes; each set can contain up to 16 attributes. Attribute identifiers are encoded such that the most significant three nibbles specify the attribute set

and the least significant nibble specifies the attribute within the set. The currently defined attribute sets are listed in Table 8.2.

Table 8.2 Attribute Sets for the IAS Zone Cluster

Attribute Set Identifier	Description
0x000	Zone information
0x001	Zone settings
0x002 – 0xfff	Reserved

8.2.2.2.1 Zone Information Attribute Set

The Zone Information attribute set contains the attributes summarized in Table 8.3.

Table 8.3 Attributes of the Zone Information Attribute Set

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0000	*ZoneState*	8-bit enumeration	All	Read only	0x00	M
0x0001	*ZoneType*	16-bit enumeration	All	Read only	-	M
0x0002	*ZoneStatus*	16-bit bitmap	All	Read only	0x00	M

8.2.2.2.1.1 *ZoneState* Attribute

The *ZoneState* attribute contains the values summarized in Table 8.4.

Table 8.4 Values of the *ZoneState* Attribute

Attribute Value	Meaning
0x00	Not enrolled
0x01	Enrolled (the client will react to Zone State Change Notification commands from the server)
0x02-0xff	Reserved

8.2.2.2.1.2 *ZoneType* Attribute

The *ZoneType* attribute values are summarized in Table 8.5. The Zone Type dictates the meaning of Alarm1 and Alarm2 bits of the *ZoneStatus* attribute, as also indicated in this table.

Table 8.5 Values of the *ZoneType* Attribute

Attribute Value	Zone Type	Alarm1	Alarm2
0x0000	Standard CIE	System Alarm	-
0x000d	Motion sensor	Intrusion indication	Presence indication
0x0015	Contact switch	1st portal Open-Close	2nd portal Open-Close
0x0028	Fire sensor	Fire indication	-
0x002a	Water sensor	Water overflow indication	-
0x002b	Gas sensor	CO indication	Cooking indication
0x002c	Personal emergency device	Fall / Concussion	Emergency button
0x002d	Vibration / Movement sensor	Movement indication	Vibration
0x010f	Remote Control	Panic	Emergency
0x0115	Key fob	Panic	Emergency
0x021d	Keypad	Panic	Emergency
0x0225	Standard Warning Device (see [B5] part 4)	-	-
Other values < 0x7fff	Reserved	-	-
0x8000-0xfffe	Reserved for manufacturer specific types	-	-
0xffff	Invalid Zone Type	-	-

8.2.2.2.1.3 *ZoneStatus* Attribute

The *ZoneStatus* attribute is a bit map. The meaning of each bit is summarized in Table 8.6.

Table 8.6 Values of the *ZoneStatus* Attribute

Attribute Bit Number	Meaning	Values
0	Alarm1	1 – opened or alarmed 0 – closed or not alarmed
1	Alarm2	1 – opened or alarmed 0 – closed or not alarmed
2	Tamper	1 – Tampered 0 – Not tampered
3	Battery	1 – Low battery 0 – Battery OK
4	Supervision reports (Note 1)	1 – Reports 0 – Does not report
5	Restore reports (Note 2)	1 – Reports restore 0 – Does not report restore
6	Trouble	1 – Trouble/Failure 0 – OK
7	AC (mains)	1 – AC/Mains fault 0 – AC/Mains OK
8-15	Reserved	-

Note 1: This bit indicates whether the Zone issues periodic Zone Status Change Notification commands. The CIE device may use these periodic reports as an indication that a zone is operational. Zones that do not implement the periodic reporting are required to set this bit to zero (the CIE will know not to interpret the lack of reports as a problem).

Note 2: This bit indicates whether or not a Zone Status Change Notification command will be sent to indicate that an alarm is no longer present. Some Zones do not have the ability to detect that alarm condition is no longer present, they only can tell that an alarm has occurred. These Zones must set the “Restore” bit to zero, indicating to the CIE not to look for alarm-restore notifications.

8.2.2.2.2 Zone Settings Attribute Set

The Zone Settings attribute set contains the attributes summarized in Table 8.7.

Table 8.7 Attributes of the Zone Settings Attribute Set

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0010	*IAS_CIE_Address*	IEEE address	Valid 64bit IEEE address	Read/ Write	-	M

8.2.2.2.2.1 *IAS_CIE_Address* Attribute

The *IAS_CIE_Address* attribute specifies the address that commands generated by the server shall be sent to. All commands received by the server must also come from this address.

It is up to the zone's specific implementation to permit or deny change (write) of this attribute at specific times. Also, it is up to the zone's specific implementation to implement some auto-detect for the CIE (example: by requesting the ZigBee cluster discovery service to locate a Zone Server cluster.) or require the intervention of a CT in order to configure this attribute during installation.

8.2.2.3 Commands Received

The command IDs received by the IAS Zone server cluster are listed in Table 8.8.

Table 8.8 Received Command IDs for the IAS Zone Cluster

Command Identifier Field Value	Description	Mandatory / Optional
0x00	Zone Enroll Response	M
0x01 – 0xff	Reserved	

8.2.2.3.1 Zone Enroll Response Command

8.2.2.3.1.1 Payload Format

The Zone Enroll Response command payload shall be formatted as illustrated in Figure 8.2.

Bits	8	8
Data Type	8-bit enumeration	Unsigned 8-bit integer
Field Name	Enroll response code	Zone ID

Figure 8.2 Format of the Zone Enroll Response Command Payload

The permitted values of the Enroll Response Code are shown in Table 8.9.

Table 8.9 Values of the Enroll Response Code

Code	Meaning	Details
0x00	Success	Success
0x01	Not supported	This specific Zone type is not known to the CIE and is not supported.
0x02	No enroll permit	CIE does not permit new zones to enroll at this time.
0x03	Too many zones	CIE reached its limit of number of enrolled zones
0x04-0xfe	Reserved	-

The Zone ID field is the index into the zone table of the CIE (Table 8.11). This field is only relevant if the response code is success.

8.2.2.3.1.2 Effect on Receipt

On receipt, the device embodying the Zone server is notified that it is now enrolled as an active alarm device

The device embodying the Zone server must authenticate received messages by checking the address of their sender against IAS_CIE_Address. This is to ensure that only messages from the correct CIE are accepted.

8.2.2.4 Commands Generated

The generated command IDs for the IAS Zone server cluster are listed in Table 8.10.

Table 8.10 Generated Command IDs for the IAS Zone Cluster

Command Identifier Field Value	Description	Mandatory / Optional
0x00	Zone Status Change Notification	M
0x01	Zone Enroll Request	M
0x02 – 0xff	Reserved	

8.2.2.4.1 Zone Status Change Notification Command

8.2.2.4.1.1 Payload Format

The Zone Status Change Notification command payload shall be formatted as illustrated in Figure 8.3.

Bits	16	8
Data Type	16-bit enumeration	8-bit enumeration
Field Name	Zone Status	Extended Status

Figure 8.3 Format of the Zone Status Change Notification Command Payload

The Zone Status field shall be the current value of the *ZoneStatus* attribute.

The Extended Status field is reserved for additional status information and shall be set to zero.

8.2.2.4.1.2 When Generated

The Zone Status Change Notification command is generated when a change takes place in one or more bits of the *ZoneStatus* attribute.

8.2.2.4.2 Zone Enroll Request Command

8.2.2.4.2.1 Payload Format

The Zone Enroll Request command payload shall be formatted as illustrated in Figure 8.4.

Bits	16	16
Data Type	16-bit enumeration	Unsigned 16-bit integer
Field Name	Zone Type	Manufacturer Code

Figure 8.4 Format of the Zone Enroll Request Command Payload

The Zone Type field shall be the current value of the *ZoneType* attribute.

The Manufacturer Code field shall be the manufacturer code as held in the node descriptor for the device. Manufacturer Codes are allocated by the ZigBee Alliance.

8.2.2.4.2.2 When Generated

The Zone Enroll Request command is generated when a device embodying the Zone server cluster wishes to be enrolled as an active alarm device. It must do this immediately it has joined the network (during commissioning).

8.2.3 Client

8.2.3.1 Dependencies

None.

8.2.3.2 Attributes

No attributes are currently defined for this cluster.

8.2.3.3 Commands Received

The client receives the cluster specific commands detailed in 8.2.2.4.

8.2.3.4 Commands Generated

The client generates the cluster specific commands detailed in 8.2.2.3, as required by the application.

8.3 IAS ACE Cluster

8.3.1 Overview

The IAS ACE cluster defines an interface to the functionality of any Ancillary Control Equipment of the IAS system. Using this cluster, a ZigBee enabled ACE device can access a IAS CIE device and manipulate the IAS system, on behalf of a level-2 user (see [B5]).

The client is usually implemented by the IAS ACE device. It allows the IAS ACE device to control the IAS CIE device, which typically implements the server side.

8.3.2 Server

8.3.2.1 Dependencies

None.

8.3.2.2 Attributes

No attributes are currently defined for this cluster.

8.3.2.3 Zone Table

The Zone Table is used to store information for each Zone enrolled by the CIE. The maximum number of entries in the table is 256.

The format of a group table entry is illustrated in Table 8.11.

Table 8.11 Format of the Zone Table

Field	Type	Valid Range	Description
Zone ID	Unsigned 8-bit integer	0x00 – 0xfe	The unique identifier of the zone
Zone Type	16-bit enumeration	0x0000 – 0xfffe	See Table 8.5.
Zone Address	IEEE Address	Valid 64bit IEEE address	Device address

The Zone ID is a unique reference number allocated by the CIE at zone enrollment time.

The Zone ID is used by IAS devices to reference specific zones when communicating with the CIE. The Zone ID of each zone stays fixed until that zone is un-enrolled.

8.3.2.4 Commands Received

The received command IDs for the IAS ACE server cluster are listed in Table 8.12

Table 8.12 Received Command IDs for the IAS ACE Cluster

Command Identifier Field Value	Description	Mandatory / Optional
0x00	Arm	M
0x01	Bypass	M
0x02	Emergency	M
0x03	Fire	M
0x04	Panic	M
0x05	Get Zone ID Map	M
0x06	Get Zone Information	M
0x07 – 0xff	Reserved	-

8.3.2.4.1 Arm Command

8.3.2.4.1.1 Payload Format

The Arm command payload shall be formatted as illustrated in Figure 8.5.

Bits	8
Data Type	8-bit enumeration
Field Name	Arm Mode

Figure 8.5 Format of the Arm Command Payload

8.3.2.4.1.2 Arm Mode Field

The Arm Mode field shall have one of the values shown in Table 8.13.

Table 8.13 Arm Mode Field Values

Arm Mode Field Value	Meaning
0x00	Disarm
0x01	Arm Day/Home Zones Only
0x02	Arm Night/Sleep Zones Only
0x03	Arm All Zones
0x08-0xff	Reserved

8.3.2.4.1.3 Effect on Receipt

On receipt of this command, the receiving device sets its arm mode according to the value of the Arm Mode field, as detailed in Table 8.13. It is not guaranteed that an Arm command will succeed. Based on the current state of the IAS CIE, and its related devices, the command can be rejected. The device shall generate an Arm Response command (see 8.3.2.5.1) to indicate the resulting armed state.

8.3.2.4.2 Bypass Command

8.3.2.4.2.1 Payload Format

The Bypass command payload shall be formatted as illustrated in Figure 8.6.

Bits	8	8	- - - - -	8
Data Type	Unsigned 8-bit integer	Unsigned 8-bit integer	- - - - -	Unsigned 8-bit integer
Field Name	Number of Zones	Zone ID	- - - - -	Zone ID

Figure 8.6 Format of the Bypass Command Payload

8.3.2.4.2.2 Number of Zones Parameter

This is the number of Zone IDs included in the payload.

8.3.2.4.2.3 Zone ID Parameter

Zone ID is the index of the Zone in the CIE's zone table (Table 8.11).

8.3.2.4.3 Emergency, Fire and Panic Commands

These commands indicate the emergency situations inherent in their names. They have no payload.

8.3.2.4.4 Get Zone ID Map Command

8.3.2.4.4.1 Payload Format

This command has no payload.

8.3.2.4.4.2 Effect on Receipt

On receipt of this command, the device shall generate a Get Zone ID Map Response command. See 7.2.5.2

8.3.2.4.5 Get Zone Information Command

8.3.2.4.5.1 Payload Format

The Get Zone Information command payload shall be formatted as illustrated in Figure 8.7.

Bits	8
Data Type	Unsigned 8-bit integer
Field Name	Zone ID

Figure 8.7 Format of the Get Zone Information Command Payload

8.3.2.4.5.2 Effect on Receipt

On receipt of this command, the device shall generate a Get Zone Information Response command. See 7.2.5.3.

8.3.2.5 Commands Generated

The generated command IDs for the IAS ACE server cluster are listed in Table 8.14.

Table 8.14 Generated Command IDs for the IAS ACE Cluster

Command Identifier Field Value	Description	Mandatory / Optional
0x00	Arm Response	M
0x01	Get Zone ID Map Response	M
0x02	Get Zone Information Response	M
0x03 – 0xff	Reserved	

8.3.2.5.1 Arm Response Command

8.3.2.5.1.1 Payload Format

The Arm Response command payload shall be formatted as illustrated in Figure 8.8.

Bits	8
Data Type	8-bit enumeration
Field Name	Arm Notification

Figure 8.8 Format of the Arm Response Command Payload

8.3.2.5.1.2 Arm Notification Field

The Arm Notification field shall have one of the values shown in Table 8.15.

Table 8.15 Arm Notification Values

Arm Mode Attribute Value	Meaning
0x00	All Zones Disarmed
0x01	Only Day/Home Zones Armed
0x02	Only Night/Sleep Zones Armed
0x03	All Zones Armed
0x04 – 0xfe	Reserved

8.3.2.5.2 Get Zone ID Map Response Command

8.3.2.5.2.1 Payload Format

The Get Zone ID Map Response command payload shall be formatted as illustrated in Figure 8.9.

Bits	16	. . . . .	16
Data Type	16-bit bitmap	. . . . .	16-bit bitmap
Field Name	Zone ID Map section 0	. . . . .	Zone ID Map section 15

Figure 8.9 Get Zone ID Map Response Command Payload

The 16 fields of the payload indicate whether each of the Zone IDs from 0 to 0xff is allocated or not. If bit n of Zone ID Map section N is set to 1, then Zone ID (16 x N + n) is allocated, else it is not allocated.

8.3.2.5.3 Get Zone Information Response Command

8.3.2.5.3.1 Payload Format

The Get Zone Information Response command payload shall be formatted as illustrated in Figure 8.10.

Bits	8	16	64
Data Type	Unsigned 8-bit integer	16-bit enumeration	IEEE address
Field Name	Zone ID	Zone Type	IEEE address

Figure 8.10 Format of the Get Zone Information Response Command Payload

The fields of the payload are equal to the fields of the Group Table entry corresponding to the ZoneID field of the Get Zone Information command to which this command is a response.

If the Zone ID is unallocated, this shall be indicated by setting the Zone Type and IEEE Address fields to 0xffff (see Table 8.5) and 0xffffffffffffffff respectively.

8.3.3 Client

8.3.3.1 Dependencies

None.

8.3.3.2 Attributes

No attributes are currently defined for this cluster.

8.3.3.3 Commands Received

The client receives the cluster specific commands detailed in 8.3.2.5.

8.3.3.4 Commands Generated

The client cluster generates the commands detailed in 8.3.2.4, as required by the application.

8.4 IAS WD Cluster

8.4.1 Overview

The IAS WD cluster provides an interface to the functionality of any Warning Device equipment of the IAS system. Using this cluster, a ZigBee enabled CIE device can access a ZigBee enabled IAS WD device and issue alarm warning indications (siren, strobe lighting, etc.) when a system alarm condition is detected (according to [B5]).

8.4.2 Server

8.4.2.1 Dependencies

None.

8.4.2.2 Attributes

The attributes defined for the server cluster are detailed in Table 8.16.

Table 8.16 Attributes of the IAS WD (Server) Cluster

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0000	MaxDuration	Unsigned 16-bit integer	0x0000 – 0fffe	Read/Write	240	M
0x0001-0xffff	Reserved	-	-	-	-	-

8.4.2.2.1 *MaxDuration* Attribute

The *MaxDuration* attribute specifies the maximum time in seconds that the siren will sound continuously, regardless of start/stop commands.

8.4.2.3 Commands Received

The received command IDs are listed in Table 8.17.

Table 8.17 Received Command IDs for the IAS WD Server Cluster

Command Identifier Field Value	Description	Mandatory / Optional
0x00	Start warning	M
0x01	Squawk	M
0x02 – 0xff	Reserved	

8.4.2.3.1 Start Warning Command

This command starts the WD operation. The WD alerts the surrounding area by audible (siren) and visual (strobe) signals.

A Start Warning command shall always terminate the effect of any previous command that is still current.

8.4.2.3.1.1 Payload Format

The Start Warning command payload shall be formatted as illustrated in Figure 8.11.

Bits	4	2	2	16
Data Type	8-bit data			Unsigned 16-bit integer
Field Name	Warning mode	Strobe	Reserved	Warning duration

Figure 8.11 Format of the Start Siren Command Payload

The Warning mode and Strobe subfields are concatenated together to a single 8-bit bitmap field. The groups of bits these subfields occupy are used as follows.

8.4.2.3.1.2 Warning Mode Field

The Warning Mode field is used as an 4-bit enumeration, can have one of the values set in Table 8.18. The exact behavior of the WD device in each mode is according to the relevant security standards.

Table 8.18 Warning Modes

Warning Mode	Meaning
0	Stop (no warning)
1	Burglar
2	Fire
3	Emergency
4-15	Reserved

8.4.2.3.1.3 Strobe Field

The Strobe field is used as a 2-bit enumeration, and determines if the visual indication is required in addition to the audible siren, as indicated in Table 8.19. If the strobe field is “1” and the Warning Mode is “0” (“Stop”) then only the strobe is activated.

Table 8.19 Values of the Strobe Field

Value	Meaning
0	No strobe
1	Use strobe in parallel to warning
2-3	Reserved

8.4.2.3.1.4 Warning Duration Field

Requested duration of warning, in seconds. If both Strobe and Warning Mode are "0" this field shall be ignored.

8.4.2.3.2 Squawk Command

This command uses the WD capabilities to emit a quick audible/visible pulse called a "squawk". The squawk command has no effect if the WD is currently active (warning in progress).

8.4.2.3.2.1 Payload Format

The Squawk command payload shall be formatted as illustrated in Figure 8.12.

Bits	4	1	1	2
Data Type	8-bit data			
Field Name	Squawk mode	Strobe	Reserved	Squawk level

Figure 8.12 Format of the Start Siren Command Payload

8.4.2.3.2.2 Squawk Mode Field

The Squawk Mode field is used as a 4-bit enumeration, and can have one of the values shown in Table 8.20. The exact operation of each mode (how the WD “squawks”) is implementation specific.

Table 8.20 Squawk Mode Field

Warning Mode	Meaning
0	Notification sound for “System is armed”
1	Notification sound for "System is disarmed"
2-15	Reserved

8.4.2.3.2.3 Strobe Field

The strobe field is used as a boolean, and determines if the visual indication is also required in addition to the audible squawk., as shown in Table 8.21.

Table 8.21 Strobe Bit

Value	Meaning
0	No strobe
1	Use strobe blink in parallel to squawk

8.4.2.3.2.4 Squawk Level Field

The squawk level field is used as a 2-bit enumeration, and determines the intensity of audible squawk sound as shown in Table 8.22.

Table 8.22 Squawk Level Field Values

Value	Meaning
0	Low level sound
1	Medium level sound
2	High level sound
3	Very High level sound

8.4.2.4 Commands Generated

No cluster specific commands are generated by the server cluster.

8.4.3 Client

The client side is implemented by the CIE. The CIE is a client of the warning service provided by this cluster. Usually a WD would implement an IAS WD cluster server and an IAS Zone cluster server.

8.4.3.1 Dependencies

None.

8.4.3.2 Attributes

No attributes are currently defined for the client cluster.

8.4.3.3 Received Commands

The client receives no cluster specific commands.

8.4.3.4 Commands Generated

The client cluster generates the cluster specific commands detailed in 8.4.2.3, as required by the application.

This page left intentionally blank

CHAPTER

9

PROTOCOL INTERFACES

9.1 General Description

9.1.1 Introduction

The clusters specified in this document are for use in applications which interface to external protocols.

9.1.2 Cluster List

This section lists the clusters specified in this document, and gives examples of typical usage for the purpose of clarification.

The clusters defined in this document are listed in Table 9.1.

Table 9.1 Clusters of the Protocol Interfaces Functional Domain

Cluster Name	Description
Generic tunnel	The minimum common commands and attributes required to tunnel any protocol.
BACnet protocol tunnel	Commands and attributes required to tunnel the BACnet protocol.
Analog input (BACnet regular)	An interface for accessing a number of commonly used BACnet based attributes of an analog measurement.
Analog input (BACnet extended)	An interface for accessing a number of BACnet based attributes of an analog measurement.
Analog output (BACnet regular)	An interface for accessing a number of commonly used BACnet based attributes of an analog output.

Table 9.1 Clusters of the Protocol Interfaces Functional Domain (Continued)

Cluster Name	Description
Analog output (BACnet extended)	An interface for accessing a number of BACnet based attributes of an analog output.
Analog value(BACnet regular)	An interface for accessing a number of commonly used BACnet based attributes of an analog value, typically used as a control system parameter.
Analog value(BACnet extended)	An interface for accessing a number of BACnet based attributes of an analog value, typically used as a control system parameter.
Binary input (BACnet regular)	An interface for accessing a number of commonly used BACnet based attributes of a binary measurement.
Binary input (BACnet extended)	An interface for accessing a number of BACnet based attributes of a binary measurement.
Binary output (BACnet regular)	An interface for accessing a number of commonly used BACnet based attributes of a binary output.
Binary output (BACnet extended)	An interface for accessing a number of BACnet based attributes of a binary output.
Binary value (BACnet regular)	An interface for accessing a number of commonly used BACnet based attributes of a binary value, typically used as a control system parameter.
Binary value (BACnet extended)	An interface for accessing a number of BACnet based attributes of a binary value, typically used as a control system parameter.
Multistate input (BACnet regular)	An interface for accessing a number of commonly used BACnet based attributes of a multistate measurement.
Multistate input (BACnet extended)	An interface for accessing a number of BACnet based attributes of a multistate measurement.
Multistate output (BACnet regular)	An interface for accessing a number of commonly used BACnet based attributes of a multistate output.
Multistate output (BACnet extended)	An interface for accessing a number of BACnet based attributes of a multistate output.
Multistate value (BACnet regular)	An interface for accessing a number of commonly used BACnet based attributes of a multistate value, typically used as a control system parameter.
Multistate value (BACnet extended)	An interface for accessing a number of BACnet based attributes of a multistate value, typically used as a control system parameter.

9.2 Generic Tunnel Cluster

9.2.1 Overview

The generic cluster provides the minimum common commands and attributes required to discover protocol tunnelling devices. A protocol cluster specific to the protocol being tunnelled shall be implemented on the same endpoint as the Generic Tunnel cluster.

Note: The reverse is not true, as there may be tunnel clusters that do not require the Generic Tunnel cluster.[10]

9.2.2 Server

9.2.2.1 Dependencies

The maximum size of the *ProtocolAddress* attribute is dependent on the protocol supported by any associated specific protocol tunnel cluster supported on the same endpoint (see sub-clause 9.2.2.2.3).

9.2.2.2 Attributes

The Generic Tunnel contains the attributes summarized in Table 9.2.

Table 9.2 Attributes of the Generic Tunnel Cluster

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0001	*Maximum Incoming TransferSize*	Unsigned 16-bit integer	0x0000 - 0xffff	Read only	0x0000	M
0x0002	*Maximum Outgoing TransferSize*	Unsigned 16-bit integer	0x0000 - 0xffff	Read only	0x0000	M
0x0003	*ProtocolAddress*	Octet string	0 - 255 octets	Read/ Write	Null string	M

9.2.2.2.1 *MaximumIncomingTransferSize* Attribute

The *MaximumIncomingTransferSize* attribute specifies the maximum size, in octets, of the application service data unit (ASDU), that can be transferred to this node in one single message transfer. The ASDU referred to is the ZCL frame,

10. CCB 1260

including header and payload, of any command received by a protocol specific tunnel cluster on the same endpoint.

This value can not exceed the Maximum Incoming Transfer Size field of the node descriptor on the device supporting this cluster.

9.2.2.2.2 *MaximumOutgoingTransferSize* Attribute

The *MaximumOutgoingTransferSize* attribute specifies the maximum size, in octets, of the application sub-layer data unit (ASDU) that can be transferred from this node in one single message transfer. The ASDU referred to is the ZCL frame, including header and payload, of any command sent by a protocol specific tunnel cluster on the same endpoint.

This value can not exceed the Maximum Outgoing Transfer Size field of the node descriptor on the device supporting this cluster.

9.2.2.2.3 *ProtocolAddress* Attribute

The *ProtocolAddress* attribute contains an octet string that is interpreted as a device address by the protocol being tunneled by an associated protocol specific tunnel cluster (if any). The overall maximum size of the string is 255 octets, but devices need only support the actual maximum size required by that protocol

9.2.2.3 Commands Received

The cluster specific commands received by the Generic Tunnel server cluster are listed in Table 9.3.

Table 9.3 Command IDs for the Generic Tunnel Cluster

Command Identifier Field Value	Description	Mandatory / Optional
0x00	Match Protocol Address	M
0x01 – 0xff	Reserved	-

9.2.2.3.1 Match Protocol Address Command

The Match Protocol Address command payload shall be formatted as illustrated in Figure 9.1.

Octets	Variable
Data Type	Octet string
Field Name	Protocol Address

Figure 9.1 Format of Match Protocol Address Command Payload

9.2.2.3.2 When Generated

This command is generated when an associated protocol specific tunnel cluster wishes to find the ZigBee address (node, endpoint) of the Generic Tunnel server cluster representing a protocol-specific device with a given protocol address. The command is typically multicast to a group of inter-communicating Generic Tunnel clusters.

9.2.2.3.3 Effect on Receipt

On receipt of this command, a device shall match the Protocol Address field of the received command to the ProtocolAddress attribute. If they are equal, it shall return the Match Protocol Address Response command (see A.3.3.2.4), otherwise it shall do nothing.

9.2.2.4 Commands Generated

The cluster specific commands generated by the Generic Tunnel server cluster are listed in Figure 9.4.

Table 9.4 Command IDs for the Generic Tunnel Cluster

Command Identifier Field Value	Description	Mandatory / Optional
0x00	Match Protocol Address Response	M
0x01	Advertise Protocol Address	O[a]
0x02 – 0xff	Reserved	-

a. CCB 1260

9.2.2.4.1 Match Protocol Address Response Command

The Match Protocol Address Response command payload shall be formatted as illustrated in Figure 9.2.

Octets	8	Variable
Data Type	IEEE address	Octet string
Field Name	Device IEEE Address	Protocol Address

Figure 9.2 Match Protocol Address Response Command Payload

The Device IEEE Address field shall be set equal to the IEEE address of the responding device. The Protocol Address field shall be set equal to the matched Protocol Address.

9.2.2.4.2 When Generated

This command is generated upon receipt of a Match Protocol Address command (see sub-clause 9.2.2.3.1), to indicate that the Protocol Address was successfully matched by the responding device.

9.2.2.4.3 Advertise Protocol Address Command

The Advertise Protocol Address command payload shall be formatted as illustrated in Figure 9.3.

Octets	Variable
Data Type	Octet string
Field Name	Protocol Address

Figure 9.3 Advertise Protocol Address Command Payload

The Protocol Address field shall be set to the value of the *ProtocolAddress* attribute.

9.2.2.4.4 When Generated

This command is typically sent upon startup, and whenever the *ProtocolAddress* attribute changes. It is typically multicast to a group of inter-communicating Generic Tunnel clusters.

9.2.3 Client

9.2.3.1 Dependencies

None.

9.2.3.2 Attributes

The client cluster has no attributes.

9.2.3.3 Commands Received

The client cluster receives the cluster specific commands detailed in sub-clause 9.2.2.4.

9.2.3.4 Commands Generated

The client cluster generates the cluster specific commands detailed in sub-clause 9.2.2.3.

9.3 BACnet Protocol Tunnel Cluster

9.3.1 Overview

The BACnet Protocol Tunnel cluster provides the commands and attributes required to tunnel the BACnet protocol (see [B8]). The server cluster receives BACnet NPDUs and the client cluster generates BACnet NPDUs, thus it is necessary to have both server and client on an endpoint to tunnel BACnet messages in both directions.

9.3.2 Server

9.3.2.1 Dependencies

Any endpoint that supports the BACnet Protocol Tunnel server cluster shall also support the Generic Tunnel server cluster.

The associated Generic Tunnel server cluster shall have its *ProtocolAddress* attribute equal to the device identifier of the BACnet device represented on that endpoint, expressed as an octet string (i.e. with identical format as a BACnet OID data type, but interpreted as an octet string). The special three octet value 0x3FFFFF of the *ProtocolAddress* attribute indicates that the associated BACnet device is not commissioned.

The associated Generic Tunnel server cluster shall also have its *MaximumIncomingTransferSize* attribute and *MaximumOutgoingTransferSize* attribute equal to or greater than 504 octets. Accordingly, this cluster requires fragmentation to be implemented, with maximum transfer sizes given by these attributes.

9.3.2.2 Attributes

The BACnet Protocol Tunnel cluster does not contain any attributes.

9.3.2.3 Commands Received

The cluster specific commands received by the BACnet Protocol Tunnel server cluster are listed in Table 9.5.

Table 9.5 Command IDs for the BACnet Protocol Tunnel Cluster

Command Identifier Field Value	Description	Mandatory / Optional
0x00	Transfer NPDU	M
0x01 – 0xff	Reserved	-

9.3.2.3.1 Transfer NPDU Command

9.3.2.3.1.1 Payload Format

The Transfer NPDU command payload shall be formatted as illustrated in Figure 9.4.

Octets	Variable
Data Type	Sequence of 8-bit data
Field Name	NPDU

Figure 9.4 Format of the Transfer NPDU Command Payload

9.3.2.3.1.2 NPDU Field

The NPDU field is of variable length and is a BACnet NPDU as defined in the BACnet standard [B8]. Its format is a sequence of 8-bit data (see General Data section of Table 2.16) of arbitrary length.

9.3.2.3.1.3 When Generated

This command is generated when a BACnet network layer wishes to transfer a BACnet NPDU across a ZigBee tunnel to another BACnet network layer.

9.3.2.3.1.4 Effect on Receipt

On receipt of this command, a device shall process the BACnet NPDU as specified in the BACnet standard [B8].

9.3.2.4 Commands Generated

No cluster specific commands are generated by the server cluster.

9.3.3 Client

9.3.3.1 Dependencies

None.

9.3.3.2 Attributes

The client cluster has no attributes.

9.3.3.3 Commands Received

The client does not receive any cluster specific commands.

9.3.3.4 Commands Generated

The cluster specific commands generated by the client cluster are listed in 9.3.2.3.

9.4 BACnet Input, Output and Value Clusters

This section specifies a number of clusters which are based on the Input, Output and Value objects specified by BACnet (see [B8]).

Each of these three objects is specified by BACnet in three different forms - Analog, Binary and Multistate. ZigBee clusters are specified here based on all nine such BACnet objects.

Each such BACnet object is represented in the ZCL by three related clusters - a Basic cluster (see 3.14), a BACnet Regular cluster and a BACnet Extended cluster. The properties of each BACnet object are implemented as ZigBee attributes, and are divided into three sets, which are allocated to the clusters as follows.

Basic clusters (3.14) implement attributes and functionality that can be readily employed either via interworking with a BACnet system, or by any general purpose ZigBee system. Accordingly, these clusters are included in the ZCL General functional domain.

BACnet Regular and BACnet Extended clusters implement attributes and functionality that are specifically intended for interworking with a BACnet system (through a BACnet gateway). Accordingly, these clusters are included in the ZCL Protocol Interface functional domain.

A BACnet Regular cluster may only be implemented on an endpoint that also implements its associated Basic cluster. Similarly, a BACnet Extended cluster may only be implemented on an endpoint that also implements both its associated BACnet Regular cluster and its associated Basic cluster.

The clusters specified herein are for use typically in ZigBee Commercial Building applications, but may be used in any application domain.

For these clusters, the Access field for each attribute specification may contain one of the following symbols:

R Readable, but not writeable

R/W Readable and writable

R*W Readable and optionally writable. The ability to write to this attribute is not mandatory but is determined by the vendor supplying the product. If not writable, a READ_ONLY error is returned for any write attempt.

9.4.1 Analog Input (BACnet Regular) cluster

The Analog Input (BACnet Regular) cluster provides an interface for accessing a number of commonly used BACnet based attributes of an analog measurement. It is used principally for interworking with BACnet systems.

9.4.1.1 Server

9.4.1.1.1 Dependencies

Any endpoint that supports this cluster must support the Analog Input (Basic) cluster.

9.4.1.1.2 Attributes

The attributes of this cluster are detailed in Table 9.6.

Table 9.6 Attributes of the Analog Input (BACnet Regular) Server

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0016	*COVIncrement*	Single precision	-	R*W	0	O
0x001F	*DeviceType*	Character string	-	R	Null string	O
0x004B	*ObjectIdentifier*	BACnet OID	0x00000000-0xffffffff	R	-	M
0x004D	*ObjectName*	Character string	-	R	Null string	M
0x004F	*ObjectType*	16-bit enumeration	-	R	-	M
0x0076	*UpdateInterval*	Unsigned 8-bit integer	-	R*W	0	O
0x00A8	*ProfileName*	Character string	-	R*W	Null string	O
All others < 0x0400	Reserved					
0x0400 - 0xFFFF	Reserved for vendor specific attributes					

For an explanation of the attributes, see section 9.4.19.

9.4.1.1.3 Commands

No cluster specific commands are received or generated.

9.4.1.1.4 Attribute Reporting

No attribute reporting is mandated for this cluster.

9.4.1.2 Client

The client has no dependencies, no attributes, and receives or generates no cluster specific commands.

9.4.2 Analog Input (BACnet Extended) Cluster

The Analog Input (BACnet Extended) cluster provides an interface for accessing a number of BACnet based attributes of an analog measurement. It is used principally for interworking with BACnet systems.

9.4.2.1 Server

9.4.2.1.1 Dependencies

Any endpoint that supports this cluster must support the Analog Input (Basic) cluster and the Analog Input (BACnet Regular) cluster.

9.4.2.1.2 Attributes

The attributes of this cluster are detailed in Table 9.7.

Table 9.7 Attributes of the Analog Input (BACnet Extended) Server

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0000	*AckedTransitions*	8-bit bitmap	-	R*W	0	M
0x0011	*NotificationClass*	Unsigned 16-bit integer	0x0000 - 0xffff	R*W	0	M
0x0019	*Deadband*	Single precision	-	R*W	0	M
0x0023	*EventEnable*	8-bit bitmap	-	R*W	0	M
0x0024	*EventState*	8-bit enumeration	-	R	0	O
0x002D	*HighLimit*	Single precision	-	R*W	0	M
0x0034	*LimitEnable*	8-bit bitmap	0x00 - 0x11	R*W	0x00	M
0x003B	*LowLimit*	Single precision	-	R*W	0	M
0x0048	*NotifyType*	8-bit enumeration	-	R*W	0	M
0x0071	*TimeDelay*	Unsigned 8-bit integer	-	R*W	0	M

Table 9.7 Attributes of the Analog Input (BACnet Extended) Server (Continued)

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0082	*EventTimeStamps*	Array[3] of (16-bit unsigned integer, time of day, or structure of (date, time of day))	-	R	-	M
All others < 0x0400	Reserved					
0x0400 - 0xFFFF	Reserved for vendor specific attributes					

For an explanation of the attributes, see section 9.4.20.

9.4.2.1.3 Commands

No cluster specific commands are received or generated.

9.4.2.1.4 Attribute Reporting

No attribute reporting is mandated for this cluster.

9.4.2.2 Client

The client has no dependencies, no attributes, and receives or generates no cluster specific commands.

9.4.3 Analog Output (BACnet Regular) Cluster

The Analog Output (BACnet Regular) cluster provides an interface for accessing a number of commonly used BACnet based attributes of an analog output. It is used principally for interworking with BACnet systems.

9.4.3.1 Server

9.4.3.1.1 Dependencies

Any endpoint that supports this cluster shall also support the Analog Output (Basic) cluster, and this cluster shall support the *PriorityArray* and *RelinquishDefault* attributes.

9.4.3.1.2 Attributes

The attributes of this cluster are detailed in Table 9.8.

Table 9.8 Attributes of the Analog Output (BACnet Regular) Server

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0016	*COVIncrement*	Single precision	-	R*W	0	O
0x001F	*DeviceType*	Character string	-	R	Null string	O
0x004B	*ObjectIdentifier*	BACnet OID	0x00000000-0xffffffff	R	-	M
0x004D	*ObjectName*	Character string	-	R	Null string	M
0x004F	*ObjectType*	16-bit enumeration	-	R	-	M
0x00A8	*ProfileName*	Character string	-	R*W	Null string	O
All others < 0x0400	Reserved					
0x0400 - 0xFFFF	Reserved for vendor specific attributes					

For an explanation of the attributes, see section 9.4.19.

9.4.3.1.3 Commands

No cluster specific commands are received or generated.

9.4.3.1.4 Attribute Reporting

No attribute reporting is mandated for this cluster.

9.4.3.2 Client

The client has no dependencies, no attributes, and receives or generates no cluster specific commands.

9.4.4 Analog Output (BACnet Extended) cluster

The Analog Output (BACnet Extended) cluster provides an interface for accessing a number of BACnet based attributes of an analog output. It is used principally for interworking with BACnet systems.

9.4.4.1 Server

9.4.4.1.1 Dependencies

Any endpoint that supports this cluster must support the Analog Output (Basic) cluster and the Analog Output (BACnet Regular) cluster.

9.4.4.1.2 Attributes

The attributes of this cluster are detailed in Table 9.9.

Table 9.9 Attributes of the Analog Output (BACnet Extended) Server

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0000	*AckedTransitions*	8-bit bitmap	-	R*W	0	M
0x0011	*NotificationClass*	Unsigned 16-bit integer	0x0000 - 0xffff	R*W	0	M
0x0019	*Deadband*	Single precision	-	R*W	0	M
0x0023	*EventEnable*	8-bit bitmap	-	R*W	0	M
0x0024	*EventState*	8-bit enumeration	-	R	0	O
0x002D	*HighLimit*	Single precision	-	R*W	0	M
0x0034	*LimitEnable*	8-bit bitmap	0x00 - 0x11	R*W	0x00	M
0x003B	*LowLimit*	Single precision	-	R*W	0	M
0x0048	*NotifyType*	8-bit enumeration	-	R*W	0	M
0x0071	*TimeDelay*	Unsigned 8-bit integer	-	R*W	0	M

Table 9.9 Attributes of the Analog Output (BACnet Extended) Server (Continued)

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0082	*EventTimeStamps*	Array[3] of (16-bit unsigned integer, time of day, or structure of (date, time of day))	-	R	-	M
All others < 0x0400	Reserved					
0x0400 - 0xFFFF	Reserved for vendor specific attributes					

For an explanation of the attributes, see section 9.4.20.

9.4.4.1.3 Commands

No cluster specific commands are received or generated.

9.4.4.1.4 Attribute Reporting

No attribute reporting is mandated for this cluster.

9.4.4.2 Client

The client has no dependencies, no attributes, and receives or generates no cluster specific commands.

9.4.5 Analog Value (BACnet Regular) Cluster

The Analog Value (BACnet Regular) cluster provides an interface for accessing commonly used BACnet based characteristics of an analog value, typically used as a control system parameter. It is principally used for interworking with BACnet systems.

9.4.5.1 Server

9.4.5.1.1 Dependencies

Any endpoint that supports this cluster must support the Analog Value (Basic) cluster.

9.4.5.1.2 Attributes

The attributes of this cluster are detailed in Table 9.10.

Table 9.10 Attributes of the Analog Value (BACnet Regular) Server

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0016	*COVIncrement*	Single precision	-	R*W	0	O
0x004B	*ObjectIdentifier*	BACnet OID	0x00000000-0xffffffff	R	-	M
0x004D	*ObjectName*	Character string	-	R	Null string	M
0x004F	*ObjectType*	16-bit enumeration	-	R	-	M
0x00A8	*ProfileName*	Character string	-	R*W	Null string	O
All others < 0x0400	Reserved					
0x0400 - 0xFFFF	Reserved for vendor specific attributes					

For an explanation of the attributes, see section 9.4.19.

9.4.5.1.3 Commands

No cluster specific commands are received or generated.

9.4.5.1.4 Attribute Reporting

No attribute reporting is mandated for this cluster.

9.4.5.2 Client

The client has no dependencies, no attributes, and receives or generates no cluster specific commands.

9.4.6 Analog Value (BACnet Extended) Cluster

The Analog Value (BACnet Extended) cluster provides an interface for accessing BACnet based characteristics of an analog value, typically used as a control system parameter. It is principally used for interworking with BACnet systems.

9.4.6.1 Server

9.4.6.1.1 Dependencies

Any endpoint that supports this cluster must support the Analog Value (Basic) cluster and the Analog Value (BACnet Regular) cluster.

9.4.6.1.2 Attributes

The attributes of this cluster are detailed in Table 9.11.

Table 9.11 Attributes of the Analog Value (BACnet Extended) Server

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0000	*AckedTransitions*	8-bit bitmap	-	R*W	0	M
0x0011	*NotificationClass*	Unsigned 16-bit integer	0x0000 - 0xffff	R*W	0	M
0x0019	*Deadband*	Single precision	-	R*W	0	M
0x0023	*EventEnable*	8-bit bitmap	-	R*W	0	M
0x0024	*EventState*	8-bit enumeration	-	R	0	O
0x002D	*HighLimit*	Single precision	-	R*W	0	M
0x0034	*LimitEnable*	8-bit bitmap	0x00 - 0x11	R*W	0x00	M
0x003B	*LowLimit*	Single precision	-	R*W	0	M
0x0048	*NotifyType*	8-bit enumeration	-	R*W	0	M
0x0071	*TimeDelay*	Unsigned 8-bit integer	-	R*W	0	M

Table 9.11 Attributes of the Analog Value (BACnet Extended) Server (Continued)

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0082	*EventTimeStamps*	Array[3] of (16-bit unsigned integer, time of day, or structure of (date, time of day))	-	R	-	M
All others < 0x0400	Reserved					
0x0400 - 0xFFFF	Reserved for vendor specific attributes					

For an explanation of the attributes, see section 9.4.20.

9.4.6.1.3 Commands

No cluster specific commands are received or generated.

9.4.6.1.4 Attribute Reporting

No attribute reporting is mandated for this cluster.

9.4.6.2 Client

The client has no dependencies, no attributes, and receives or generates no cluster specific commands.

9.4.7 Binary Input (BACnet Regular) Cluster

The Binary Input (BACnet Regular) cluster provides an interface for accessing a number of commonly used BACnet based attributes of a binary measurement. It is used principally for interworking with BACnet systems.

9.4.7.1 A.4.12.1 Server

9.4.7.1.1 Dependencies

Any endpoint that supports this cluster must support the Binary Input (Basic) cluster.

9.4.7.1.2 Attributes

The attributes of this cluster are detailed in Table 9.12.

Table 9.12 Attributes of the Binary Input (BACnet Regular) Server

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x000F	*ChangeOfState Count*	Unsigned 32-bit integer	-	R*W	0xffffffff	O
0x0010	*ChangeOfState Time*	Structure (Date, Time of Day)	-	R	0xffffffff 0xffffffff	O
0x001F	*DeviceType*	Character string	-	R	Null string	O
0x0021	*ElapsedActive Time*	Unsigned 32-bit integer	-	R*W	0xffffffff	O
0x004B	*ObjectIdentifier*	BACnet OID	0x00000000- 0xffffffff	R	-	M
0x004D	*ObjectName*	Character string	-	R	Null string	M
0x004F	*ObjectType*	16-bit enumeration	-	R	-	M
0x0072	*TimeOfATReset*	Structure (Date, Time of Day)	-	R	0xffffffff 0xffffffff	O
0x0073	*TimeOfSCReset*	Structure (Date, Time of Day)	-	R	0xffffffff 0xffffffff	O
0x00A8	*ProfileName*	Character string	-	R*W	Null string	O
All others < 0x0400	Reserved					
0x0400 - 0xFFFF	Reserved for vendor specific attributes					

For an explanation of the attributes, see section 9.4.19.

9.4.7.1.3 Commands

No cluster specific commands are received or generated.

9.4.7.1.4 Attribute Reporting

No attribute reporting is mandated for this cluster.

9.4.7.2 Client

The client has no dependencies, no attributes, and receives or generates no cluster specific commands.

9.4.8 Binary Input (BACnet Extended) Cluster

The Binary Input (BACnet Extended) cluster provides an interface for accessing a number of BACnet based attributes of a binary measurement. It is used principally for interworking with BACnet systems.

9.4.8.1 Server

9.4.8.1.1 Dependencies

Any endpoint that supports this cluster must support the Binary Input (Basic) cluster and the Binary Input (BACnet Regular) cluster.

9.4.8.1.2 Attributes

The attributes of this cluster are detailed in Table 9.13.

Table 9.13 Attributes of the Binary Input (BACnet Extended) Server

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0000	*AckedTransitions*	8-bit bitmap	-	R*W	0	M
0x0006	*AlarmValue*	Boolean	0 - 1	R*W	-	M
0x0011	*NotificationClass*	Unsigned 16-bit integer	0x0000 - 0xffff	R*W	0	M
0x0023	*EventEnable*	8-bit bitmap	-	R*W	0	M
0x0024	*EventState*	8-bit enumeration	-	R	0	O
0x0048	*NotifyType*	8-bit enumeration	-	R*W	0	M
0x0071	*TimeDelay*	Unsigned 8-bit integer	-	R*W	0	M

Table 9.13 Attributes of the Binary Input (BACnet Extended) Server (Continued)

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0082	*EventTimeStamps*	Array[3] of (16-bit unsigned integer, time of day, or structure of (date, time of day))	-	R	-	M
All others < 0x0400	*Reserved*					
0x0400 - 0xFFFF	*Reserved for vendor specific attributes*					

For an explanation of the attributes, see section 9.4.20.

9.4.8.1.3 Commands

No cluster specific commands are received or generated.

9.4.8.1.4 Attribute Reporting

No attribute reporting is mandated for this cluster.

9.4.8.2 Client

The client has no dependencies, no attributes, and receives or generates no cluster specific commands.

9.4.9 Binary Output (BACnet Regular) Cluster

The Analog Output (BACnet Regular) cluster provides an interface for accessing a number of commonly used BACnet based attributes of a binary output. It is used principally for interworking with BACnet systems.

9.4.9.1 Server

9.4.9.1.1 Dependencies

Any endpoint that supports this cluster shall also support the Binary Output (Basic) cluster, and this cluster shall support the PriorityArray and RelinquishDefault attributes.

9.4.9.1.2 Attributes

The attributes of this cluster are detailed in Table 9.14.

Table 9.14 Attributes of the Binary Output (BACnet Regular) Server

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x000F	*ChangeOfState Count*	Unsigned 32-bit integer	-	R*W	0xffffffff	O
0x0010	*ChangeOfState Time*	Structure (Date, Time of Day)	-	R	0xffffffff 0xffffffff	O
0x001F	*DeviceType*	Character string	-	R	Null string	O
0x0021	*ElapsedActive Time*	Unsigned 32-bit integer	-	R*W	0xffffffff	O
0x0028	*FeedBackValue*	8-bit enumeration	0 - 1	R*W	0	O
0x004B	*ObjectIdentifier*	BACnet OID	0x00000000- 0xffffffff	R	-	M
0x004D	*ObjectName*	Character string	-	R	Null string	M
0x004F	*ObjectType*	16-bit enumeration	-	R	-	M
0x0072	*TimeOfATReset*	Structure (Date, Time of Day)	-	R	0xffffffff 0xffffffff	O
0x0073	*TimeOfSCReset*	Structure (Date, Time of Day)	-	R	0xffffffff 0xffffffff	O
0x00A8	*ProfileName*	Character string	-	R*W	Null string	O
All others < 0x0400	Reserved					
0x0400 - 0xFFFF	Reserved for vendor specific attributes					

For an explanation of the attributes, see section 9.4.19.

9.4.9.1.3 Commands

No cluster specific commands are received or generated.

9.4.9.1.4 Attribute Reporting

No attribute reporting is mandated for this cluster.

9.4.9.2 Client

The client has no dependencies, no attributes, and receives or generates no cluster specific commands.

9.4.10 Binary Output (BACnet Extended) Cluster

The Binary Output (BACnet Extended) cluster provides an interface for accessing a number of BACnet based attributes of a binary output. It is used principally for interworking with BACnet systems.

9.4.10.1 Server

9.4.10.1.1 Dependencies

Any endpoint that supports this cluster must support the Binary Output (Basic) cluster and the Binary Output (BACnet Regular) cluster.

9.4.10.1.2 Attributes

The attributes of this cluster are detailed in Table 9.15.

Table 9.15 Attributes of the Binary Output (BACnet Extended) Server

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0000	*AckedTransitions*	8-bit bitmap	-	R*W	0	M
0x0011	*NotificationClass*	Unsigned 16-bit integer	0x0000 - 0xffff	R*W	0	M
0x0023	*EventEnable*	8-bit bitmap	-	R*W	0	M
0x0024	*EventState*	8-bit enumeration	-	R	0	O
0x0048	*NotifyType*	8-bit enumeration	-	R*W	0	M
0x0071	*TimeDelay*	Unsigned 8-bit integer	-	R*W	0	M

Table 9.15 Attributes of the Binary Output (BACnet Extended) Server (Continued)

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0082	*EventTimeStamps*	Array[3] of (16-bit unsigned integer, time of day, or structure of (date, time of day))	-	R	-	M
All others < 0x0400	Reserved					
0x0400 - 0xFFFF	Reserved for vendor specific attributes					

For an explanation of the attributes, see section 9.4.20.

9.4.10.1.3 Commands

No cluster specific commands are received or generated.

9.4.10.1.4 Attribute Reporting

No attribute reporting is mandated for this cluster.

9.4.10.2 Client

The client has no dependencies, no attributes, and receives or generates no cluster specific commands.

9.4.11 Binary Value (BACnet Regular) Cluster

The Binary Value (BACnet Regular) cluster provides an interface for accessing commonly used BACnet based characteristics of a binary value, typically used as a control system parameter. It is principally used for interworking with BACnet systems.

9.4.11.1 Server

9.4.11.1.1 Dependencies

Any endpoint that supports this cluster must support the Binary Value (Basic) cluster.

9.4.11.1.2 Attributes

The attributes of this cluster are detailed in Table 9.16.

Table 9.16 Attributes of the Binary Value (BACnet Regular) Server

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x000F	*ChangeOfState Count*	Unsigned 32-bit integer	-	R*W	0xffffffff	O
0x0010	*ChangeOfState Time*	Structure (Date, Time of Day)	-	R	0xffffffff 0xffffffff	O
0x0021	*ElapsedActive Time*	Unsigned 32-bit integer	-	R*W	0xffffffff	O
0x004B	*ObjectIdentifier*	BACnet OID	0-0xffffffff	R	-	M
0x004D	*ObjectName*	Character string	-	R	Null string	M
0x004F	*ObjectType*	16-bit enumeration	-	R	-	M
0x0072	*TimeOfATReset*	Structure (Date, Time of Day)	-	R	0xffffffff 0xffffffff	O
0x0073	*TimeOfSCReset*	Structure (Date, Time of Day)	-	R	0xffffffff 0xffffffff	O
0x00A8	*ProfileName*	Character string	-	R*W	Null string	O
All others < 0x0400	Reserved					
0x0400 - 0xFFFF	Reserved for vendor specific attributes					

For an explanation of the attributes, see section 9.4.19.

9.4.11.1.3 Commands

No cluster specific commands are received or generated.

9.4.11.1.4 Attribute Reporting

No attribute reporting is mandated for this cluster.

9.4.11.2 Client

The client has no dependencies, no attributes, and receives or generates no cluster specific commands.

9.4.12 Binary Value (BACnet Extended) Cluster

The Binary Value (BACnet Extended) cluster provides an interface for accessing BACnet based characteristics of a binary value, typically used as a control system parameter. It is principally used for interworking with BACnet systems.

9.4.12.1 Server

9.4.12.1.1 Dependencies

Any endpoint that supports this cluster must support the Binary Value (Basic) cluster and the Binary Value (BACnet Regular) cluster.

9.4.12.1.2 Attributes

The attributes of this cluster are detailed in Table 9.17.

Table 9.17 Attributes of the Binary Value (BACnet Extended) Server

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0000	*AckedTransitions*	8-bit bitmap	-	R*W	0	M
0x0006	*AlarmValue*	Boolean	0 - 1	R*W	-	M
0x0011	*NotificationClass*	Unsigned 16-bit integer	0x0000 - 0xffff	R*W	0	M
0x0023	*EventEnable*	8-bit bitmap	-	R*W	0	M
0x0024	*EventState*	8-bit enumeration	-	R	0	O
0x0048	*NotifyType*	8-bit enumeration	-	R*W	0	M
0x0071	*TimeDelay*	Unsigned 8-bit integer	-	R*W	0	M

Table 9.17 Attributes of the Binary Value (BACnet Extended) Server (Continued)

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0082	*EventTimeStamps*	Array[3] of (16-bit unsigned integer, time of day, or structure of (date, time of day))	-	R	-	M
All others < 0x0400	Reserved					
0x0400 - 0xFFFF	Reserved for vendor specific attributes					

For an explanation of the attributes, see section 9.4.20.

9.4.12.1.3 Commands

No cluster specific commands are received or generated.

9.4.12.1.4 Attribute Reporting

No attribute reporting is mandated for this cluster.

9.4.12.2 Client

The client has no dependencies, no attributes, and receives or generates no cluster specific commands.

9.4.13 Multistate Input (BACnet Regular) Cluster

The Multistate Input (BACnet Regular) cluster provides an interface for accessing a number of commonly used BACnet based attributes of a multistate measurement. It is used principally for interworking with BACnet systems.

9.4.13.1 Server

9.4.13.1.1 Dependencies

Any endpoint that supports this cluster must support the Multistate Input (Basic) cluster.

9.4.13.1.2 Attributes

The attributes of this cluster are detailed in Table 9.18.

Table 9.18 Attributes of the Multistate Input (BACnet Regular) Server

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x001F	*DeviceType*	Character string	-	R	Null string	O
0x004B	*ObjectIdentifier*	BACnet OID	0-0xffffffff	R	-	M
0x004D	*ObjectName*	Character string	-	R	Null string	M
0x004F	*ObjectType*	16-bit enumeration	-	R	-	M
0x00A8	*ProfileName*	Character string	-	R*W	Null string	O
All others < 0x0400	Reserved					
0x0400 - 0xFFFF	Reserved for vendor specific attributes					

For an explanation of the attributes, see section 9.4.19.

9.4.13.1.3 Commands

No cluster specific commands are received or generated.

9.4.13.1.4 Attribute Reporting

No attribute reporting is mandated for this cluster.

9.4.13.2 Client

The client has no dependencies, no attributes, and receives or generates no cluster specific commands.

9.4.14 Multistate Input (BACnet Extended) Cluster

The Multistate Input (BACnet Extended) cluster provides an interface for accessing a number of BACnet based attributes of a multistate measurement. It is used principally for interworking with BACnet systems.

9.4.14.1 Server

9.4.14.1.1 Dependencies

Any endpoint that supports this cluster must support the Multistate Input (Basic) cluster and the Multistate Input (BACnet Regular) cluster.

9.4.14.1.2 Attributes

The attributes of this cluster are detailed in Table 9.19.

Table 9.19 Attributes of Multistate Input (BACnet Extended) Server

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0000	*AckedTransitions*	8-bit bitmap	-	R*W	0	M
0x0006	*AlarmValues*	Set of unsigned 16-bit integer	0 - 0xffff	R*W	-	M
0x0011	*NotificationClass*	Unsigned 16-bit integer	0x0000 - 0xffff	R*W	0	M
0x0023	*EventEnable*	8-bit bitmap	-	R*W	0	M
0x0024	*EventState*	8-bit enumeration	-	R	0	O
0x0025	*FaultValues*	Set of unsigned 16-bit integer	0 - 0xffff	R*W	0	M
0x0048	*NotifyType*	8-bit enumeration	-	R*W	0	M
0x0071	*TimeDelay*	Unsigned 8-bit integer	-	R*W	0	M
0x0082	*EventTimeStamps*	Array[3] of (16-bit unsigned integer, time of day, or structure of (date, time of day))	-	R	-	M
All others < 0x0400	Reserved					
0x0400 - 0xFFFF	Reserved for vendor specific attributes					

For an explanation of the attributes, see section 9.4.20.

9.4.14.1.3 Commands

No cluster specific commands are received or generated.

9.4.14.1.4 Attribute Reporting

No attribute reporting is mandated for this cluster.

9.4.14.2 Client

The client has no dependencies, no attributes, and receives or generates no cluster specific commands.

9.4.15 Multistate Output (BACnet Regular) Cluster

The Multistate Output (BACnet Regular) cluster provides an interface for accessing a number of commonly used BACnet based attributes of a multistate output. It is used principally for interworking with BACnet systems.

9.4.15.1 Server

9.4.15.1.1 Dependencies

Any endpoint that supports this cluster shall also support the Multistate Output (Basic) cluster, and this cluster shall support the PriorityArray and RelinquishDefault attributes.

9.4.15.1.2 Attributes

The attributes of this cluster are detailed in Table 9.20.

Table 9.20 Attributes of Multistate Output (BACnet Regular) Server

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x001F	*DeviceType*	Character string	-	R	Null string	O
0x0028	*FeedBackValue*	8-bit enumeration	0 - 1	R*W	0	O
0x004B	*ObjectIdentifier*	BACnet OID	0x00000000-0xffffffff	R	-	M
0x004D	*ObjectName*	Character string	-	R	Null string	M
0x004F	*ObjectType*	16-bit enumeration	-	R	-	M

Table 9.20 Attributes of Multistate Output (BACnet Regular) Server (Continued)

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x00A8	*ProfileName*	Character string	-	R*W	Null string	O
All others < 0x0400	Reserved					
0x0400 - 0xFFFF	Reserved for vendor specific attributes					

For an explanation of the attributes, see section 9.4.19.

9.4.15.1.3 Commands

No cluster specific commands are received or generated.

9.4.15.1.4 Attribute Reporting

No attribute reporting is mandated for this cluster.

9.4.15.2 Client

The client has no dependencies, no attributes, and receives or generates no cluster specific commands.

9.4.16 Multistate Output (BACnet Extended) Cluster

The Multistate Output (BACnet Extended) cluster provides an interface for accessing a number of BACnet based attributes of a multistate output. It is used principally for interworking with BACnet systems.

9.4.16.1 Server

9.4.16.1.1 Dependencies

Any endpoint that supports this cluster must support the Multistate Output (Basic) cluster and the Multistate Output (BACnet Regular) cluster.

9.4.16.1.2 Attributes

The attributes of this cluster are detailed in Table 9.21.

Table 9.21 Attributes of Multistate Output (BACnet Extended) Server

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0000	*AckedTransitions*	8-bit bitmap	-	R*W	0	M
0x0011	*NotificationClass*	Unsigned 16-bit integer	0x0000 - 0xffff	R*W	0	M
0x0023	*EventEnable*	8-bit bitmap	-	R*W	0	M
0x0024	*EventState*	8-bit enumeration	-	R	0	O
0x0048	*NotifyType*	8-bit enumeration	-	R*W	0	M
0x0071	*TimeDelay*	Unsigned 8-bit integer	-	R*W	0	M
0x0082	*EventTimeStamps*	Array[3] of (16-bit unsigned integer, time of day, or structure of (date, time of day))	-	R	-	M
All others < 0x0400	Reserved					
0x0400 - 0xFFFF	Reserved for vendor specific attributes					

For an explanation of the attributes, see section 9.4.20.

9.4.16.1.3 Commands

No cluster specific commands are received or generated.

9.4.16.1.4 Attribute Reporting

No attribute reporting is mandated for this cluster.

9.4.16.2 Client

The client has no dependencies, no attributes, and receives or generates no cluster specific commands.

9.4.17 Multistate Value (BACnet Regular) Cluster

The Multistate Value (BACnet Regular) cluster provides an interface for accessing commonly used BACnet based characteristics of a multistate value, typically used as a control system parameter. It is principally used for interworking with BACnet systems.

9.4.17.1 Server

9.4.17.1.1 Dependencies

Any endpoint that supports this cluster must support the Multistate Value (Basic) cluster.

9.4.17.1.2 Attributes

The attributes of this cluster are detailed in Table 9.22.

Table 9.22 Attributes of Multistate Value (BACnet Regular) Server

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x004B	*ObjectIdentifier*	BACnet OID	0 - 0xffffffff	R	-	M
0x004D	*ObjectName*	Character string	-	R	Null string	M
0x004F	*ObjectType*	16-bit enumeration	-	R	-	M
0x00A8	*ProfileName*	Character string	-	R*W	Null string	O
All others < 0x0400	Reserved					
0x0400 - 0xFFFF	Reserved for vendor specific attributes					

For an explanation of the attributes, see section 9.4.19.

9.4.17.1.3 Commands

No cluster specific commands are received or generated.

9.4.17.1.4 Attribute Reporting

No attribute reporting is mandated for this cluster.

9.4.17.2 Client

The client has no dependencies, no attributes, and receives or generates no cluster specific commands.

9.4.18 Multistate Value (BACnet Extended) Cluster

The Multistate Value (BACnet Extended) cluster provides an interface for accessing BACnet based characteristics of a multistate value, typically used as a control system parameter. It is principally used for interworking with BACnet systems.

9.4.18.1 Server

9.4.18.1.1 Dependencies

Any endpoint that supports this cluster must support the Multistate Value (Basic) cluster and the Multistate Value (BACnet Regular) cluster.

9.4.18.1.2 Attributes

The attributes of this cluster are detailed in Table 9.23.

Table 9.23 Attributes of Multistate Value (BACnet Extended) Server

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0000	*AckedTransitions*	8-bit bitmap	-	R*W	0	M
0x0006	*AlarmValues*	Set of unsigned 16-bit integer	0 - 0xffff	R*W	-	M
0x0011	*NotificationClass*	Unsigned 16-bit integer	0x0000 - 0xffff	R*W	0	M
0x0023	*EventEnable*	8-bit bitmap	-	R*W	0	M
0x0024	*EventState*	8-bit enumeration	-	R	0	O
0x0025	*FaultValues*	Set of unsigned 16-bit integer	0 - 0xffff	R*W	0	M
0x0048	*NotifyType*	8-bit enumeration	-	R*W	0	M
0x0071	*TimeDelay*	Unsigned 8-bit integer	-	R*W	0	M

Table 9.23 Attributes of Multistate Value (BACnet Extended) Server (Continued)

Identifier	Name	Type	Range	Access	Default	Mandatory / Optional
0x0082	*EventTimeStamps*	Array[3] of (16-bit unsigned integer, time of day, or structure of (date, time of day))	-	R	-	M
All others < 0x0400	Reserved					
0x0400 - 0xFFFF	Reserved for vendor specific attributes					

For an explanation of the attributes, see section 9.4.20.

9.4.18.1.3 Commands

No cluster specific commands are received or generated.

9.4.18.1.4 Attribute Reporting

No attribute reporting is mandated for this cluster.

9.4.18.2 Client

The client has no dependencies, no attributes, and receives or generates no cluster specific commands.

9.4.19 Attributes of BACnet Regular Clusters

The attributes of BACnet Regular and BACnet Extended clusters are specifically intended for interworking with BACnet systems (via a BACnet gateway). They are based on BACnet properties with the same names. See the BACnet Reference Manual [B8] for detailed descriptions of these properties.

References to reports in this section refer to BACnet intrinsic reporting. Note that ZigBee attribute reporting may be used to send corresponding reports within a ZigBee system.

9.4.19.1 *ObjectIdentifier* Attribute

This attribute, of type BACnet OID, is a numeric code that is used to identify the object. It shall be unique within the BACnet Device that maintains it.

9.4.19.2 *ObjectName* Attribute

This attribute, of type Character String, shall represent a name for the object that is unique within the BACnet Device that maintains it. The minimum length of the string shall be one character. The set of characters used in the *ObjectName* shall be restricted to printable characters.

9.4.19.3 *ObjectType* Attribute

This attribute, of type enumeration, is set to the ID of the corresponding BACnet object type from which the cluster was derived.

9.4.19.4 *COVIncrement* Attribute

This attribute, of type single precision, specifies the minimum change in *PresentValue* that will cause a value change report to be initiated to bound report recipient clients. This value is the same as the Reportable Change value for the *PresentValue* attribute.

9.4.19.5 *DeviceType* Attribute

This attribute, of type Character String, is a text description of the physical device connected to the input, output or value.

9.4.19.6 *UpdateInterval* Attribute

This attribute indicates the maximum period of time between updates to the *PresentValue* of an Analog Input cluster, in hundredths of a second, when the input is not overridden and not out-of-service.

9.4.19.7 *ChangeOfStateCount* Attribute

This attribute, of type Unsigned 32-bit integer, represents the number of times that the *PresentValue* attribute of a Binary Input, Output or Value cluster has changed state (from 0 to 1, or from 1 to 0) since the *ChangeOfStateCount* attribute was most recently set to a zero value. The *ChangeOfStateCount* attribute shall have a range of 0-65535 or greater.

When *OutOfService* is FALSE, a change to the *Polarity* attribute shall alter *PresentValue* and thus be considered a change of state. When *OutOfService* is TRUE, changes to *Polarity* shall not cause changes of state. If one of the optional attributes *ChangeOfStateTime*, *ChangeOfStateCount*, or *TimeOfStateCountReset* is present, then all of these attributes shall be present.

9.4.19.8 *ChangeOfStateTime* Attribute

This attribute, of type Structure (Date, Time of Day), represents the most recent date and time at which the *PresentValue* attribute of a Binary Input, Output or Value cluster changed state (from 0 to 1, or from 1 to 0)

When *OutOfService* is FALSE, a change to the *Polarity* attribute shall alter *PresentValue* and thus be considered a change of state. When *OutOfService* is TRUE, changes to *Polarity* shall not cause changes of state. If one of the optional attributes *ChangeOfStateTime*, *ChangeOfStateCount*, or *TimeOfSCReset* is present, then all of these attributes shall be present.

9.4.19.9 *ElapsedActiveTime* Attribute

This attribute, of type Unsigned 32-bit integer, represents the accumulated number of seconds that the *PresentValue* attribute of a Binary Input, Output or Value cluster has had the value ACTIVE (1) since the *ElapsedActiveTime* attribute was most recently set to a zero value. If one of the optional properties *ElapsedActiveTime* or *TimeOfATReset* are present, then both of these attributes shall be present.

9.4.19.10 *TimeOfATReset* Attribute

This attribute, of type Structure (Date, Time of Day), represents the date and time at which the *ElapsedActiveTime* attribute of a Binary Input, Output or Value cluster was most recently set to a zero value. If one of the optional properties *ElapsedActiveTime* or *TimeOfATReset* are present, then both of these attributes shall be present.

9.4.19.11 *TimeOfSCReset* Attribute

This attribute, of type Structure (Date, Time of Day), represents the date and time at which the *ChangeOfStateCount* attribute of a Binary Input, Output or Value cluster was most recently set to a zero value. If one of the optional properties *ChangeOfStateTime*, *ChangeOfStateCount*, or *TimeOfSCReset* is present, then all of these attributes shall be present.

9.4.19.12 *FeedbackValue* Attribute

This property, of type enumeration, indicates a feedback value from which *PresentValue* must differ before an OFFNORMAL event is generated, and to which *PresentValue* must return before a TONORMAL event is generated. The manner by which the *FeedbackValue* is determined shall be a local matter.

9.4.19.13 *ProfileName* Attribute

This attribute, of type Character string, is the name of a BACnet object profile to which its associated cluster conforms. A profile defines a set of additional attributes, behavior, and/or requirements for the cluster beyond those specified here.

To ensure uniqueness, a profile name must begin with a vendor identifier code (see Clause 23 of [B8]) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile document named by the remainder of the profile name. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

9.4.20 Attributes of BACnet Extended Clusters

The attributes of BACnet Extended clusters are specifically intended for interworking with BACnet systems (via a BACnet gateway). They are based on BACnet properties with the same names. See the BACnet Reference Manual [B8] for detailed descriptions of these properties.

References to events and alarms in this section refer to BACnet intrinsic reporting. Note that ZigBee attribute reporting may be used to send corresponding reports within a ZigBee system.

9.4.20.1 *AckedTransitions* Attribute

This attribute, of type bitmap, holds three one-bit flags (b0, b1, b2) that respectively indicate the receipt of acknowledgments for TO-OFFNORMAL, TO-FAULT, and TO-NORMAL events

9.4.20.2 *AlarmValue* Attribute

This attribute, of type Boolean, specifies the value that the *PresentValue* attribute must have before a TO-OFFNORMAL event is generated.

9.4.20.3 *AlarmValues* Attribute

This attribute, of type Set of Unsigned 16-bit integer, specifies any values that the *PresentValue* attribute must equal before a TO-OFFNORMAL event is generated.

9.4.20.4 *FaultValues* Attribute

This attribute, of type Set of Unsigned 16-bit integer, specifies any values that the *PresentValue* attribute must equal before a TO-FAULT event is generated.

9.4.20.5 *NotificationClass* Attribute

This attribute, of type Unsigned 16-bit integer, specifies the notification class to be used when handling and generating event notifications for this object (over a BACnet gateway).

9.4.20.6 *Deadband* Attribute

This attribute, of type single precision, specifies a range (from *LowLimit* + *Deadband* to *HighLimit* - *Deadband*) which the *PresentValue* must return within for a TO-NORMAL event to be generated.

9.4.20.7 *EventEnable* Attribute

This attribute, of type bitmap, holds three one-bit flags (b0, b1, b2) that respectively enable (1) and disable (0) reporting of TO-OFFNORMAL, TO-FAULT, and TO-NORMAL events.

9.4.20.8 *EventState* Attribute

The *EventState* attribute, of type 8-bit enumeration, is included in order to provide a way to determine if this object has an active event state associated with it. The allowed values are

- NORMAL (0)
- FAULT (1)
- OFFNORMAL (2)
- HIGH-LIMIT (3)
- LOW-LIMIT (4)

9.4.20.9 *HighLimit* Attribute

This attribute, of type single precision, specifies a limit that *PresentValue* must exceed before an OFF-NORMAL (HIGH-LIMIT) event is generated.

9.4.20.10 *LimitEnable* Attribute

This attribute, of type 8-bit bitmap, holds two one-bit flags. The flag in bit position 0 enables reporting of low limit off-normal and return-to-normal events if it has the value 1, and disables reporting of these events if it has the value 0. The flag in bit position 1 enables reporting of high limit off-normal and return-to-normal events if it has the value 1, and disables reporting of these events if it has the value 0.

9.4.20.11 *LowLimit* Attribute

This attribute, of type single precision, shall specify a limit that *PresentValue* must fall below before an OFF-NORMAL (LOW-LIMIT) event is generated.

9.4.20.12 *NotifyType* Attribute

This attribute, of type enumeration, indicates whether the notifications generated by the cluster should be Events (0) or Alarms (1).

9.4.20.13 *TimeDelay* Attribute

This attribute, of type Unsigned 8-bit integer, specifies the minimum period of time in seconds that *PresentValue* must remain outside the band defined by the *HighLimit* and *LowLimit* attributes before a TO-OFFNORMAL event is generated, or within the band (from *LowLimit* + *Deadband* to *HighLimit* - *Deadband*) before a TO-NORMAL event is generated.

9.4.20.14 *EventTimeStamps* Attribute

This optional read-only attribute is of type Array[3]. The three elements each have a type which is one of:

- 16-bit unsigned integer - a sequence number
- Time of day
- Structure of (date, time of day)

The elements of the array hold the times (or sequence numbers) of the last event notifications for TO-OFFNORMAL, TO-FAULT, and TO-NORMAL events, respectively. The type of the elements are discovered by reading the attribute.

This page left intentionally blank