Matter Application Cluster Specification

Version 1.3.0.1

Document:	23-27350_Matter-1.3.0.1-Application-Cluster-Specification.pdf
	April 17, 2024
Sponsored by:	Connectivity Standards Alliance
Accepted by:	This document has been accepted for release by the Connectivity Standards Alliance Board of Directors on **April 17, 2024**
Abstract:	The Matter Application Clusters specified in this document are generic interfaces that are sufficiently general to be of use across a wide range of application domains.
Keywords:	Referenced in Chapter 1.

Matter Application Clusters

Version 1.3.0.1, 2024-05-29 21:38:18 -0700: Approved

Copyright Notice, License and Disclaimer

Connectivity Standards Alliance
508 Second Street, Suite 206
Davis, CA 95616, USA

Revision History

Revision	Date	Details	Editor
01	September 23, 2022	Version 1.0	Robert Szewczyk
02	May 17, 2023	Version 1.1	Robert Szewczyk
03	October 18, 2023	Version 1.2	Robert Szewczyk
04	April 17, 2024	Version 1.3	Robert Szewczyk
05	May 29, 2024	Version 1.3.0.1	Robert Szewczyk

Table of Contents

Introduction

The Matter Application Cluster specification defines generic interfaces that are sufficiently general to be of use across a wide range of application domains.

Scope and Purpose

This document specifies the Matter Application Cluster Library (MACL). The MACL is a repository for cluster functionality that is developed by the Connectivity Standards Alliance, and is a working library with regular updates as new functionality is added. A developer constructing a new application should use the MACL to find relevant cluster functionality that can be incorporated into the new application. Correspondingly, new clusters that are defined for applications should be considered for inclusion in the MACL.

The MACL consists of 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.

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.

CSA Reference Documents

Reference	Reference Location/URL	Description
[CSA-PNP]	https://groups.csa-iot.org/wg/members/document/21624	Organizational Processes and Procedures, 13-0625, revision 8, November 2021
[MatterCore]	https://groups.csa-iot.org/wg/members-all/document/27349	Matter Core Specification
[MatterDevLib]	https://github.com/CHIP-Specifications/connected-homeip-spec/raw/build-sample/pdf/device_library.pdf	Matter Device Library Specification
[StandardNamespaces]	https://groups.csa-iot.org/wg/members-all/document/31936	Standard Namespaces

External Reference Documents

Reference	Reference Location/URL	Description
[DIALRegistry]	http://www.dial-multiscreen.org/dial-registry/namespace-database	DIAL Registry
[EU Code of Conduct for ESAs]	https://ses.jrc.ec.europa.eu/development-of-policy-proposals-for-energy-smart-appliances	EU Code of Conduct for ESAs
[HDMI]	https://hdmiforum.org/hdmi-forum-releases-version-2-1-hdmi-specification/	HDMI CEC specification
[IEEE2030.5]	https://standards.ieee.org/ieee/2030.5/11216/	IEEE 2030.5
[ISO 4217]	https://www.iso.org/iso-4217-currency-codes.html	Currency Codes
[ISO 8601]	https://www.iso.org/iso-8601-date-and-time-format.html	Date and time format
[OpenADR]	https://www.openadr.org/	OpenADR Alliance
[PAS1878]	https://www.bsigroup.com/globalassets/localfiles/en-th/about-bsi/energy-smart-appliances-programme/bsi-pas-1878-energy-smart-appliances-system-functionality-and-architecture-th.pdf	PAS1878
[RFC 4122]	https://www.rfc-editor.org/rfc/rfc4122	A Universally Unique IDentifier (UUID) URN Namespace
[RFC 5646]	https://tools.ietf.org/html/rfc5646	Tags for Identifying Languages
[SAE J2847/3_202311]	https://www.sae.org/standards/content/j2847/3_202311/	Communication for Plug-in Vehicles as a Distributed Energy Source
[SAREF4ENER]	https://saref.etsi.org/saref4ener/v1.1.2/	SAREF ontology for energy (ETSI TS 103 410-1)
[WakeOnLAN]	https://www.amd.com/system/files/TechDocs/20213.pdf	Wake on LAN Magic Packet specification

Provisional

Per [CSA-PNP], when a specification is completed there may be sections of specification text (or smaller pieces of a section) that are not certifiable at this stage. These sections (or smaller pieces of a section) are marked as provisional prior to publishing the specification. This specification uses well-defined notation to mark Provisional Conformance (see [MatterCore], Section 7.3) or notes a section of text with the term "provisional".

List of Provisional Items

The following is a list of provisional items:

- Support for Pulse Width Modulation cluster and for the Frequency feature of the Level control cluster is provisional.
- Support for Ballast Configuration Cluster is provisional.
- Support for InflowError, DrainError, TempTooLow, TempTooHigh, and WaterLevelError alarms in Dishwasher Alarm Cluster is provisional.
- Support for Energy Preference Cluster is provisional.
- Support for Device Energy Management Cluster is provisional.
- Support for Device Energy Management Mode Cluster is provisional.
- Support for Content Control Cluster is provisional.
- Support for PowerInWatts feature of the Microwave Oven Control cluster is provisional
- Support for GoHome command in the RVC Operational State cluster is provisional.
- Support for Scenes Management Cluster is provisional.

Chapter 1. General

The Cluster Library is made of individual chapters such as this one. See Document Control in the Cluster Library for a list of all chapters and documents. References between chapters are made using a *X.Y* notation where *X* is the chapter and *Y* is the sub-section within that chapter. References to external documents are contained in Chapter 1 and are made using [*Rn*] notation.

1.1. General Description

1.1.1. Introduction

The clusters specified in this document are generic interfaces that are sufficiently general to be of use across a wide range of application domains.

1.1.2. Cluster List

This section lists the general clusters as specified in this chapter.

Table 1. Overview of the General Clusters

ID	Cluster Name	Description
0x0003	Identify	Attributes and commands for putting a device into Identification mode (e.g., flashing a light)
0x0004	Groups	Cluster to manage the associated endpoint's membership into one or more groups to support groupcast interactions.
0x0062	Scenes Management	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.
0x0006	On/Off	Attributes and commands for switching devices between On and Off states.
0x0008	Level Control	Attributes and commands for controlling a characteristic of devices that can be set to a level between fully On and fully Off.
0x0045	Boolean State	Attribute and event for a boolean state variable

ID	Cluster Name	Description
0x0080	Boolean State Configuration	Attributes and commands for configuration related to boolean state
0x0050	Mode Select	Allows a user to choose one mode option from several pre-defined values
n/a	Mode Base	Allows a user to choose one mode option from several pre-defined values
0x0508	Low Power	This cluster provides an interface for managing low power mode on a device.
0x0503	Wake On LAN	This cluster provides an interface for managing low power mode on a device that supports the Wake On LAN protocol.
0x003B	Switch	Attributes and events for various types of switch devices.
0x0060	Operational State	Commands and attributes for defining a device's operational state
n/a	Alarm Base	Base alarm cluster from which all alarms are derived
0x0097	Messages	Commands and attributes for sending messages to devices

1.2. Identify Cluster

This cluster supports an endpoint identification state (e.g., flashing a light), that indicates to an observer (e.g., an installer) which of several nodes and/or endpoints it is. It also supports a multicast request that any endpoint that is identifying itself to respond to the initiator.

The state of this cluster MAY be shared on more than one endpoint on a node.

> For Example: Two endpoints on a single node, one a temperature sensor, and one a humidity sensor, may both share the same cluster instance and therefore identification state (e.g. single LED on the node).

1.2.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Revision	Description
1	Global mandatory ClusterRevision attribute added
2	CCB 2808
3	All Hubs changes
4	New data model format and notation; add IdentifyType

1.2.2. Classification

Hierarchy	Role	Scope	PICS Code
Base	Utility	Endpoint	I

1.2.3. Cluster ID

ID	Name
0x0003	Identify

1.2.4. Data Types

1.2.4.1. IdentifyTypeEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0x00	**None**	No presentation.	M
0x01	**LightOutput**	Light output of a lighting product.	M
0x02	**VisibleIndicator**	Typically a small LED.	M
0x03	**AudibleBeep**		M
0x04	**Display**	Presentation will be visible on display screen.	M
0x05	**Actuator**	Presentation will be conveyed by actuator functionality such as through a window blind operation or in-wall relay.	M

1.2.4.2. EffectIdentifierEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0x00	**Blink**	e.g., Light is turned on/off once.	M
0x01	**Breathe**	e.g., Light is turned on/off over 1 second and repeated 15 times.	M
0x02	**Okay**	e.g., Colored light turns green for 1 second; non-colored light flashes twice.	M
0x0B	**ChannelChange**	e.g., Colored light turns orange for 8 seconds; non-colored light switches to the maximum brightness for 0.5s and then minimum brightness for 7.5s.	M
0xFE	**FinishEffect**	Complete the current effect sequence before terminating. e.g., if in the middle of a breathe effect (as above), first complete the current 1s breathe effect and then terminate the effect.	M
0xFF	**StopEffect**	Terminate the effect as soon as possible.	M

1.2.4.3. EffectVariantEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0x00	**Default**	Indicates the default effect is used	M

1.2.5. Attributes

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0000	**IdentifyTime**	uint16	all		0	RW VO	M
0x0001	**IdentifyType**	IdentifyTypeEnum	desc		MS	R V	M*

* IdentifyType represents a mandatory attribute that was previously not present or optional. Implementers should be aware that older devices may not implement it.

1.2.5.1. IdentifyTime Attribute

This attribute specifies the remaining length of time, in seconds, that the endpoint will continue to identify itself.

If this attribute is set to a value other than 0 then the device SHALL enter its identification state, in order to indicate to an observer which of several nodes and/or endpoints it is. It is RECOMMENDED that this state consists of flashing a light with a period of 0.5 seconds. The IdentifyTime attribute SHALL be decremented every second while in this state.

If this attribute reaches or is set to the value 0 then the device SHALL terminate its identification state.

1.2.5.2. IdentifyType Attribute

This attribute specifies how the identification state is presented to the user.

This field SHALL contain one of the values defined in IdentifyTypeEnum. The value None SHALL NOT be used if the device is capable of presenting its identification state using one of the other methods defined in IdentifyTypeEnum.

1.2.6. Commands

ID	Name	Direction	Response	Access	Conformance
0x00	**Identify**	client ⇒ server	Y	M	M
0x40	**TriggerEffect**	client ⇒ server	Y	M	O

1.2.6.1. Identify Command

This command starts or stops the receiving device identifying itself.

This command SHALL have the following data fields:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**IdentifyTime**	uint16	all			M

1.2.6.1.1. Effect on Receipt

On receipt of this command, the device SHALL set the IdentifyTime attribute to the value of the IdentifyTime field. This then starts, continues, or stops the device's identification state as detailed in IdentifyTime Attribute.

1.2.6.2. TriggerEffect Command

This command allows the support of feedback to the user, such as a certain light effect. It is used to allow an implementation to provide visual feedback to the user under certain circumstances such as a color light turning green when it has successfully connected to a network. The use of this command and the effects themselves are entirely up to the implementer to use whenever a visual feedback is useful but it is not the same as and does not replace the identify mechanism used during commissioning.

This command SHALL have the following data fields:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**EffectIdentifier**	EffectIdentifierEnum	desc			M
1	**EffectVariant**	EffectVariantEnum	desc			M

1.2.6.2.1. EffectIdentifier Field

This field specifies the identify effect to use and SHALL contain one of the non-reserved values in EffectIdentifierEnum.

All values of the EffectIdentifierEnum SHALL be supported. Implementors MAY deviate from the example light effects in EffectIdentifierEnum, but they SHOULD indicate during testing how they handle each effect.

1.2.6.2.2. EffectVariant Field

This field is used to indicate which variant of the effect, indicated in the EffectIdentifier field, SHOULD be triggered. If a device does not support the given variant, it SHALL use the default variant. This field SHALL contain one of the values in EffectVariantEnum.

1.2.6.2.3. Effect on Receipt

On receipt of this command, the device SHALL execute the trigger effect indicated in the EffectIdentifier and EffectVariant fields. If the EffectVariant field specifies a variant that is not supported on the device, it SHALL execute the default variant.

1.3. Groups Cluster

The Groups cluster manages, per endpoint, the content of the node-wide Group Table that is part of the underlying interaction layer.

In a network supporting fabrics, group IDs referenced by attributes or other elements of this cluster are scoped to the accessing fabric.

The Groups cluster is scoped to the endpoint. Groups cluster commands support discovering the endpoint membership in a group, adding the endpoint to a group, removing the endpoint from a group, removing endpoint membership from all groups. All commands defined in this cluster SHALL only affect groups scoped to the accessing fabric.

When group names are supported, the server stores a name string, which is set by the client for each assigned group and indicated in response to a client request.

Note that configuration of group addresses for outgoing commands is achieved using the Message Layer mechanisms where the Group Table is not involved. Hence this cluster does not play a part in that.

1.3.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Revision	Description
1	Global mandatory ClusterRevision attribute added; CCB 1745 2100
2	CCB 2289
3	CCB 2310 2704
4	New data model format and notation

1.3.2. Classification

Hierarchy	Role	Scope	PICS Code
Base	Utility	Endpoint	G

1.3.3. Cluster ID

ID	Name
0x0004	Groups

1.3.4. Features

This cluster SHALL support the FeatureMap bitmap attribute as defined below.

Bit	Code	Feature	Summary
0	**GN**	GroupNames	The ability to store a name for a group.

1.3.4.1. GroupNames Feature

The Group Names feature indicates the ability to store a name for a group when a group is added.

1.3.5. Data Types

1.3.5.1. NameSupportBitmap Type

This data type is derived from map8.

Bit	Name	Summary
7	**GroupNames**	The ability to store a name for a group.

1.3.6. Attributes

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0000	**NameSupport**	NameSupportBitmap	desc	F	0	R V	M

1.3.6.1. NameSupport Attribute

This attribute provides legacy, read-only access to whether the Group Names feature is supported. The most significant bit, bit 7 (GroupNames), SHALL be equal to bit 0 of the FeatureMap attribute (GN Feature). All other bits SHALL be 0.

1.3.7. Commands

ID	Name	Direction	Response	Access	Conformance
0x00	**AddGroup**	client ⇒ server	AddGroupResponse	F M	M
0x01	**ViewGroup**	client ⇒ server	ViewGroupResponse	F O	M
0x02	**GetGroupMembership**	client ⇒ server	GetGroupMembershipResponse	F O	M
0x03	**RemoveGroup**	client ⇒ server	RemoveGroupResponse	F M	M
0x04	**RemoveAllGroups**	client ⇒ server	Y	F M	M
0x05	**AddGroupIfIdentifying**	client ⇒ server	Y	F M	M

ID	Name	Direction	Response	Access	Conformance
0x00	**AddGroupResponse**	client ⇐ server	N		M
0x01	**ViewGroupResponse**	client ⇐ server	N		M
0x02	**GetGroupMembershipResponse**	client ⇐ server	N		M
0x03	**RemoveGroupResponse**	client ⇐ server	N		M

1.3.7.1. AddGroup Command

The AddGroup command allows a client to add group membership in a particular group for the server endpoint.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**GroupID**	group-id	min 1			M
1	**GroupName**	string	max 16			M

1.3.7.1.1. GroupID Field

This field SHALL be used to identify the group and any associated key material to which the server endpoint is to be added.

1.3.7.1.2. GroupName Field

This field MAY be set to a human-readable name for the group. If the client has no name for the group, the GroupName field SHALL be set to the empty string.

Support of group names is optional and is indicated by the FeatureMap and NameSupport attribute.

1.3.7.1.3. Effect on Receipt

If the server does not support group names, the GroupName field SHALL be ignored.

On receipt of the AddGroup command, the server SHALL perform the following procedure:

1. If the command fields are not within constraints, the status SHALL be CONSTRAINT_ERROR, and the server continues from step 6.
2. If the receiving node requires security material to support the group ID and that material does not exist for this group ID, the status SHALL be UNSUPPORTED_ACCESS and the server continues from step 6.
3. If the server endpoint is a member of the group indicated by the GroupID, the group name

SHALL be updated (if supported) to GroupName, the status SHALL be SUCCESS, and the server continues from step 6.

4. If there are no available resources to add the membership for the server endpoint, the status SHALL be RESOURCE_EXHAUSTED, and the server continues from step 6.
5. The server SHALL add the server endpoint as a member of the group indicated by the GroupID, the group name SHALL be updated (if supported) to GroupName, and the status SHALL be SUCCESS.
    a. If the GroupID had already been added to the Group Table because of a previous AddGroup or AddGroupIfIdentifying command and a GroupName is provided and the server supports GroupName storage, then the GroupName associated with the GroupID in the Group Table SHALL be updated to reflect the new GroupName provided for the Group, such that subsequent ViewGroup commands yield the same name for all endpoints which have a group association to the given GroupID.
6. If the AddGroup command was received as a unicast, the server SHALL generate an AddGroupResponse command with the Status field set to the evaluated status. If the AddGroup command was received as a groupcast, the server SHALL NOT generate an AddGroupResponse command.

See AddGroupResponse Command for a description of the response command.

1.3.7.2. ViewGroup Command

The ViewGroup command allows a client to request that the server responds with a ViewGroupResponse command containing the name string for a particular group.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**GroupID**	group-id	min 1			M

1.3.7.2.1. Effect on Receipt

On receipt of the ViewGroup command, the server SHALL perform the following procedure:

1. If the command fields are not within constraints, the status SHALL be CONSTRAINT_ERROR and the server continues from step 4.
2. If the server endpoint is a member of the group indicated by the GroupID, the status SHALL be SUCCESS, and the server continues from step 4.
3. Else the status SHALL be NOT_FOUND.
4. If the ViewGroup command was received as a unicast, the server SHALL generate a ViewGroupResponse command for the group, and the Status field set to the evaluated status. If the ViewGroup command was received as a groupcast, the server SHALL NOT generate a ViewGroupResponse command.

See ViewGroupResponse Command for a description of the response command.

1.3.7.3. GetGroupMembership Command

The GetGroupMembership command allows a client to inquire about the group membership of the server endpoint, in a number of ways.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**GroupList**	list[group-id]	all[min 1]			M

1.3.7.3.1. Effect on Receipt

On receipt of the GetGroupMembership command, the server SHALL respond with group membership information using the GetGroupMembershipResponse command as follows:

If the GroupList field is empty, the server SHALL respond with all group IDs indicating the groups of which the server endpoint is a member.

If the GroupList field contains at least one group ID indicating a group of which the server endpoint is a member, the server SHALL respond with each group ID indicating a group of which the server endpoint is a member that matches a group in the GroupList field.

If the GroupList field contains one or more group IDs but does not contain any group ID indicating a group of which the server endpoint is a member, the server SHALL only respond if the command is unicast. The response SHALL return with an empty GroupList field.

1.3.7.4. RemoveGroup Command

The RemoveGroup command allows a client to request that the server removes the membership for the server endpoint, if any, in a particular group.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**GroupID**	group-id	min 1			M

1.3.7.4.1. Effect on Receipt

On receipt of the RemoveGroup command, the server SHALL perform the following procedure:

1. If the command fields are not within constraints, the status SHALL be CONSTRAINT_ERROR and the server continues from step 4.
2. If the server endpoint is a member of the group indicated by the GroupID, the server SHALL remove the server endpoint membership in the group, the status SHALL be SUCCESS, and the server continues from step 4.
3. Else the status SHALL be NOT_FOUND.
4. If the RemoveGroup command was received as a unicast, the server SHALL generate a RemoveGroupResponse command with the Status field set to the evaluated status. If the RemoveGroup command was received as a groupcast, the server SHALL NOT generate a RemoveGroupResponse command.

See RemoveGroupResponse Command for a description of the response command.

Additionally, if the Scenes Management cluster is supported on the same endpoint, scenes associated with the indicated group SHALL be removed on that endpoint.

1.3.7.5. RemoveAllGroups Command

The RemoveAllGroups command allows a client to direct the server to remove all group associations for the server endpoint.

1.3.7.5.1. Effect on Receipt

On receipt of this command, the server SHALL remove all group memberships for the server endpoint from the Group Table. If the RemoveAllGroups command was received as unicast and a response is not suppressed, the server SHALL generate a response with the Status field set to SUCCESS.

Additionally, if the Scenes Management cluster is supported on the same endpoint, all scenes, except for scenes associated with group ID 0, SHALL be removed on that endpoint.

1.3.7.6. AddGroupIfIdentifying Command

The AddGroupIfIdentifying command allows a client to add group membership in a particular group for the server endpoint, on condition that the endpoint is identifying itself. Identifying functionality is controlled using the Identify cluster, (see Identify Cluster).

For correct operation of the AddGroupIfIdentifying command, any endpoint that supports the Groups server cluster SHALL also support the Identify server cluster.

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

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**GroupID**	group-id	min 1			M
1	**GroupName**	string	max 16			M

1.3.7.6.1. GroupID Field

This field SHALL be used to identify the group and any associated key material to which the server endpoint is to be added.

1.3.7.6.2. GroupName Field

This field MAY be set to a human-readable name for the group. If the client has no name for the group, the GroupName field SHALL be set to the empty string.

Support of group names is optional and is indicated by the FeatureMap and NameSupport attribute.

1.3.7.6.3. Effect on Receipt

If the server does not support group names, the GroupName field SHALL be ignored.

On receipt of the AddGroupIfIdentifying command, the server SHALL perform the following procedure:

1. The server verifies that it is currently identifying itself. If the server it not currently identifying itself, the status SHALL be SUCCESS, and the server continues from step 7.
2. If the command fields are not within constraints, the status SHALL be CONSTRAINT_ERROR and the server continues from step 7.
3. If the receiving node requires security material to support the group ID, and that material does not exist for this group ID, the status SHALL be UNSUPPORTED_ACCESS and the server continues from step 7.
4. If the server endpoint is a member of the group indicated by the GroupID, the status SHALL be SUCCESS and the server continues from step 7.
5. If there are no available resources to add the membership for the server endpoint, the status SHALL be RESOURCE_EXHAUSTED and the server continues from step 7.
6. The server SHALL add the server endpoint as a member of the group indicated by the GroupID, the group name SHALL be updated (if supported) to GroupName, and the status SHALL be SUCCESS.
    a. If the GroupID had already been added to the Group Table because of a previous AddGroup or AddGroupIfIdentifying command and a GroupName is provided and the server supports GroupName storage, then the GroupName associated with the GroupID in the Group Table SHALL be updated to reflect the new GroupName provided for the Group, such that subsequent ViewGroup commands yield the same name for all endpoints which have a group association to the given GroupID.
7. If the AddGroupIfIdentifying command was received as unicast and the evaluated status is not SUCCESS, or if the AddGroupIfIdentifying command was received as unicast and the evaluated status is SUCCESS and a response is not suppressed, the server SHALL generate a response with the Status field set to the evaluated status.

1.3.7.7. AddGroupResponse Command

The AddGroupResponse is sent by the Groups cluster server in response to an AddGroup command.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**Status**	enum8	desc			M
1	**GroupID**	group-id	min 1			M

1.3.7.7.1. Status Field

This field is set according to the Effect on Receipt section of the AddGroup command.

1.3.7.7.2. GroupID Field

This field is set to the GroupID field of the received AddGroup command.

1.3.7.8. ViewGroupResponse Command

The ViewGroupResponse command is sent by the Groups cluster server in response to a ViewGroup command.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**Status**	enum8	desc			M
1	**GroupID**	group-id	min 1			M
2	**GroupName**	string	max 16			M

1.3.7.8.1. Status Field

This field is according to the Effect on Receipt section of the ViewGroup command.

1.3.7.8.2. GroupID Field

This field is set to the GroupID field of the received ViewGroup command.

1.3.7.8.3. GroupName Field

If the status is SUCCESS, and group names are supported, this field is set to the group name associated with that group in the Group Table; otherwise it is set to the empty string.

1.3.7.9. GetGroupMembershipResponse Command

The GetGroupMembershipResponse command is sent by the Groups cluster server in response to a GetGroupMembership command.

The GetGroupMembershipResponse command SHALL have the following data fields:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**Capacity**	uint8	all	X		M
1	**GroupList**	list[group-id]	all[min 1]			M

1.3.7.9.1. Capacity Field

This field SHALL contain the remaining capacity of the Group Table of the node. The following values apply:

- 0 - No further groups MAY be added.
- 0 < Capacity < 0xFE - Capacity holds the number of groups that MAY be added.
- 0xFE - At least 1 further group MAY be added (exact number is unknown).

- null - It is unknown if any further groups MAY be added.

1.3.7.9.2. GroupList Field

The GroupList field SHALL contain either the group IDs of all the groups in the Group Table for which the server endpoint is a member of the group (in the case where the GroupList field of the received GetGroupMembership command was empty), or the group IDs of all the groups in the Group Table for which the server endpoint is a member of the group *and* for which the group ID was included in the the GroupList field of the received GetGroupMembership command (in the case where the GroupList field of the received GetGroupMembership command was not empty).
Zigbee: If the total number of groups will cause the maximum payload length of a frame to be exceeded, then the GroupList field SHALL contain only as many groups as will fit.

1.3.7.10. RemoveGroupResponse Command

The RemoveGroupResponse command is generated by the server in response to the receipt of a RemoveGroup command.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**Status**	enum8	desc			M
1	**GroupID**	group-id	min 1			M

1.3.7.10.1. Status Field

This field is according to the Effect on Receipt section of the RemoveGroup command.

1.3.7.10.2. GroupID Field

This field is set to the GroupID field of the received RemoveGroup command.

1.4. Scenes Management Cluster

The Scenes Management 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 Management cluster.

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

NOTE	Support for Scenes Management cluster is provisional.

1.4.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Revision	Description
1	Based on the ZCL Scenes Cluster, Updated the Cluster ID to 0x0062, the name to Scenes Management, removed the provisional status; Removed attributes SceneCount, CurrentScene, CurrentGroup, SceneValid, and NameSupport; Removed Explicit, TableSize and FabricScenes features; Removed EnhancedAddScene, EnhancedAddSceneResponse, EnhancedViewScene, EnhancedViewSceneResponse; TransitionTime field changed milliseconds in AddScene, ViewSceneResponse and RecallScene

1.4.2. Classification

Hierarchy	Role	Scope	PICS Code
Base	Application	Endpoint	S

1.4.3. Cluster ID

ID	Name
0x0062	Scenes Management

1.4.4. Features

This cluster SHALL support the FeatureMap bitmap attribute as defined below.

Bit	Code	Feature	Conformance	Summary
0	**SN**	SceneNames	O	The ability to store a name for a scene.

The following sections describe each feature in some detail. Further details are found within the specification.

1.4.4.1. SceneNames Feature

This feature indicates the ability to store a name for a scene when a scene is added.

1.4.5. Dependencies

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

Note that the RemoveGroup command and the RemoveAllGroups command of the Groups cluster also remove scenes.

1.4.6. Handling of fabric-scoping

Attributes and commands for this cluster are scoped to the accessing fabric and SHALL only affect or reflect data related to the accessing fabric, with the exception of the SceneValid Field of the FabricSceneInfo attribute.

The following constraints apply in addition to any other stated requirements in individual data model elements:

- Any attribute read, attribute write or command invoked on the server when no accessing fabric is available SHALL fail with a status code of UNSUPPORTED_ACCESS returned to the client.
- When accessing scene information, implementations SHALL ensure that scenes with identical Group ID and Scene ID across fabrics will only access the data for the accessing fabric, so that the same identifier values used by different accessing fabrics do not cause mixing or overwriting of another fabric's scenes.
- Upon leaving a fabric with the RemoveFabric command of the Operational Credentials Cluster, all scenes data for the associated fabric SHALL be removed from the Scene Table.
- The Scene Table capacity for a given fabric SHALL be less than half (rounded down towards 0) of the Scene Table entries (as indicated in the SceneTableSize attribute), with a maximum of 253 entries (to allow expressing it in the GetSceneMembershipResponse command). If the Scene Table capacity is about to be exceeded by adding or storing a scene, then the resource exhaustion behavior of the associated command SHALL apply.

1.4.7. Data Types

1.4.7.1. CopyModeBitmap Type

This data type is derived from map8.

Bit	Name	Summary
0	CopyAllScenes	Copy all scenes in the scene table

1.4.7.2. SceneInfoStruct Type

Access Quality: Fabric Scoped							
ID	**Name**	**Type**	**Constraint**	**Quality**	**Default**	**Access**	**Conformance**
0	**SceneCount**	uint8	all		0		M
1	**CurrentScene**	uint8	desc		0xFF	S	M
2	**CurrentGroup**	group-id	all		0	S	M

Access Quality: Fabric Scoped							
3	**SceneValid**	bool	all		False	S	M
4	**RemainingCapacity**	uint8	max 253		MS		M

1.4.7.2.1. SceneCount Field

This field SHALL indicate the number of scenes currently used in the server's Scene Table on the endpoint where the Scenes Management cluster appears.

This only includes the count for the associated fabric.

1.4.7.2.2. CurrentScene Field

This field SHALL indicate the scene identifier of the scene last invoked on the associated fabric. If no scene has been invoked, the value of this field SHALL be 0xFF, the undefined scene identifier.

1.4.7.2.3. CurrentGroup Field

This field SHALL indicate the group identifier of the scene last invoked on the associated fabric, or 0 if the scene last invoked is not associated with a group.

1.4.7.2.4. SceneValid Field

This field SHALL indicate whether the state of the server corresponds to that associated with the CurrentScene and CurrentGroup fields of the SceneInfoStruct they belong to. TRUE indicates that these fields are valid, FALSE indicates that they are not valid.

This field SHALL be set to False for all other fabrics when an attribute with the Scenes ("S") designation in the Quality column of another cluster present on the same endpoint is modified or when the current scene is modified by a fabric through the RecallScene or StoreScene commands, regardless of the fabric-scoped access quality of the command.

In the event where the SceneValid field is set to False for a fabric, the CurrentScene and CurrentGroup fields SHALL be the last invoked scene and group for that fabric. In the event where no scene was previously invoked for that fabric, the CurrentScene and CurrentGroup fields SHALL be their default values.

1.4.7.2.5. RemainingCapacity Field

This field SHALL indicate the remaining capacity of the Scene Table on this endpoint for the accessing fabric. Note that this value may change between reads, even if no entries are added or deleted on the accessing fabric, due to other clients associated with other fabrics adding or deleting entries that impact the resource usage on the device.

1.4.7.3. AttributeValuePairStruct Type

This data type indicates a combination of an identifier and the value of an attribute.

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0	**AttributeID**	attribute-id	all				M
1	**ValueUnsigned8**	uint8					O.a
2	**ValueSigned8**	int8					O.a
3	**ValueUnsigned16**	uint16					O.a
4	**ValueSigned16**	int16					O.a
5	**ValueUnsigned32**	uint32					O.a
6	**ValueSigned32**	int32					O.a
7	**ValueUnsigned64**	uint64					O.a
8	**ValueSigned64**	int64					O.a

1.4.7.3.1. AttributeID Field

This field SHALL be present for all instances in a given ExtensionFieldSetStruct.

The data type of AttributeValue SHALL be the data type of the attribute indicated by AttributeID.

The AttributeID field SHALL NOT refer to an attribute without the Scenes ("S") designation in the Quality column of the cluster specification.

1.4.7.3.2. ValueUnsigned8, ValueSigned8, ValueUnsigned16, ValueSigned16, ValueUnsigned32, ValueSigned32, ValueUnsigned64, ValueSigned64 Fields

These fields SHALL indicate the attribute value as part of an extension field set, associated with a given AttributeID under an ExtensionFieldSetStruct's ClusterID. The proper field SHALL be present that maps to the data type of the attribute indicated.

- Data types bool, map8, and uint8 SHALL map to ValueUnsigned8.
- Data types int8 SHALL map to ValueSigned8.
- Data types map16 and uint16 SHALL map to ValueUnsigned16.
- Data types int16 SHALL map to ValueSigned16.
- Data types map32, uint24, and uint32 SHALL map to ValueUnsigned32.
- Data types int24 and int32 SHALL map to ValueSigned32.

- Data types map64, uint48, uint56 and uint64 SHALL map to ValueUnsigned64.
- Data types int48, int56 and int64 SHALL map to ValueSigned64.
- For nullable attributes, any value that is not a valid numeric value for the attribute's type after accounting for range reductions due to being nullable and constraints SHALL be considered to have the null value for the type.
- For non-nullable attributes, any value that is not a valid numeric value for the attribute's type after accounting for constraints SHALL be considered to have the maximum legal value in the attribute's constrained range.

Examples of processing are:

- ColorControl cluster CurrentX (AttributeID 0x0003) has a type of uint16 and is not nullable.
  - AttributeValue of 0xAB12 would be used as-is, as it is in range.
  - AttributeValue of 0xAA0011 is outside of the range of uint16, and would be saturated to the maximum of the attribute's constraint range: 0xFEFF.
- LevelControl cluster CurrentLevel (AttributeID 0x0000) has a type of uint8 and is nullable.
  - AttributeValue of 0xA1 would be used as-is, as it is in range.
  - AttributeValue of 0xBB12 is outside the range of nullable uint8, and would be considered as the null value.

1.4.7.4. ExtensionFieldSetStruct Type

This data type indicates for a given cluster a set of attributes and their values.

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0	**ClusterID**	cluster-id	all				M
1	**AttributeValueList**	list[AttributeValuePairStruct]	desc				M

1.4.7.4.1. ClusterID Field

This field SHALL indicate the cluster-id of the cluster whose attributes are in the AttributeValueList field.

1.4.7.4.2. AttributeValueList Field

This field SHALL indicate a set of attributes and their values which are stored as part of a scene.

Attributes which do not have the Scenes ("S") designation in the Quality column of their cluster specification SHALL NOT be used in the AttributeValueList field.

1.4.7.4.3. Form of ExtensionFieldSetStruct

The AttributeValuePairStructs in the AttributeValueList MAY be in any order.

The AttributeValueList SHOULD contain all the attributes with the Scenes ("S") quality as the specification of the cluster identified by ClusterID describes, but AttributeValuePairStruct MAY be omitted.

An example using the Color Control cluster:

- Attribute 0x0001, CurrentSaturation, S quality, optional, implemented
- Attribute 0x0003, CurrentX, S quality, optional based on feature, implemented
- Attribute 0x0004, CurrentY, S quality, optional based on feature, implemented
- Attribute 0x0007, ColorTemperatureMireds, S quality, optional based on feature, implemented
- Attribute 0x4000, EnhancedCurrentHue, S quality, optional based on feature, implemented
- Attribute 0x4001, EnhancedColorMode, S quality, mandatory, implemented
- Attribute 0x4002, ColorLoopActive, S quality, optional based on feature, NOT implemented
- Attribute 0x4003, ColorLoopDirection, S quality, optional based on feature, NOT implemented
- Attribute 0x4004, ColorLoopTime, S quality, optional based on feature, NOT implemented

1.4.7.5. Logical Scene Table

The Scene Table is used to store information for each scene capable of being invoked on the server. Each scene is defined for a particular group. The Scene Table is defined here as a conceptual illustration to assist in understanding the underlying data to be stored when scenes are defined. Though the Scene Table is defined here using the data model architecture rules and format, the design is not normative.

The Scene table is logically a list of fabric-scoped structs. The logical fields of each Scene Table entry struct are illustrated below. An ExtensionFieldSetStruct MAY be present for each Scenes-supporting cluster implemented on the same endpoint.

Quality: Fabric-Scoped							
ID	**Name**	**Type**	**Constraint**	**Quality**	**Default**	**Access**	**Conformance**
0	**SceneGroupID**	group-id	all				M
1	**SceneID**	uint8	max 254				M
2	**SceneName**	string	max 16				SN
3	**SceneTransitionTime**	uint32	max 60000000		0		M
4	**ExtensionFields**	list[ExtensionFieldSetStruct]			empty		M

1.4.7.5.1. SceneGroupID Field

This field is the group identifier for which this scene applies, or 0 if the scene is not associated with a group.

1.4.7.5.2. SceneID Field

This field is unique within this group, which is used to identify this scene.

1.4.7.5.3. SceneName Field

The field is the name of the scene.

If scene names are not supported, any commands that write a scene name SHALL simply discard the name, and any command that returns a scene name SHALL return an empty string.

1.4.7.5.4. SceneTransitionTime Field

This field is the amount of time, in milliseconds, it will take for a cluster to change from its current state to the requested state.

1.4.7.5.5. ExtensionFields Field

See the Scene Table Extensions subsections of individual clusters. A Scene Table Extension SHALL only use attributes with the Scene quality. Each ExtensionFieldSetStruct holds a set of values of these attributes for a cluster implemented on the same endpoint where the Scene ("S") designation appears in the quality column. A scene is the aggregate of all such fields across all clusters on the endpoint.

1.4.8. Attributes

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0000	**LastConfiguredBy**	node-id	all	X	null	R V	O
0x0001	**SceneTableSize**	uint16	desc	F	16	R V	M
0x0002	**FabricSceneInfo**	list[SceneInfoStruct]	desc			R V F	M

1.4.8.1. LastConfiguredBy Attribute

This attribute SHALL indicate the Node ID of the node that last configured the Scene Table.

The null value indicates that the server has not been configured, or that the identifier of the node that last configured the Scenes Management cluster is not known.

The Node ID is scoped to the accessing fabric.

1.4.8.2. SceneTableSize Attribute

This attribute SHALL indicate the number of entries in the Scene Table on this endpoint. This is the total across all fabrics; note that a single fabric cannot use all those entries (see Handling of fabric-scoping). The minimum size of this table, (i.e., the minimum number of scenes to support across all fabrics per endpoint) SHALL be 16, unless a device type in which this cluster is used, defines a larger value in the device type definition.

1.4.8.3. FabricSceneInfo Attribute

This attribute SHALL indicate a list of fabric scoped information about scenes on this endpoint.

The number of list entries for this attribute SHALL NOT exceed the number of supported fabrics by the device.

1.4.9. Commands

ID	Name	Direction	Response	Access	Conformance
0x00	**AddScene**	client ⇒ server	AddSceneResponse	F M	M
0x00	**AddSceneResponse**	client ⇐ server	N		M
0x01	**ViewScene**	client ⇒ server	ViewSceneResponse	F O	M
0x01	**ViewSceneResponse**	client ⇐ server	N		M
0x02	**RemoveScene**	client ⇒ server	RemoveSceneResponse	F M	M
0x02	**RemoveSceneResponse**	client ⇐ server	N		M
0x03	**RemoveAllScenes**	client ⇒ server	RemoveAllScenesResponse	F M	M
0x03	**RemoveAllScenesResponse**	client ⇐ server	N		M
0x04	**StoreScene**	client ⇒ server	StoreSceneResponse	F M	M
0x04	**StoreSceneResponse**	client ⇐ server	N		M
0x05	**RecallScene**	client ⇒ server	Y	F O	M
0x06	**GetSceneMembership**	client ⇒ server	GetSceneMembershipResponse	F O	M

ID	Name	Direction	Response	Access	Conformance
0x06	**GetSceneMembershipResponse**	client ⇐ server	N		M
0x40	**CopyScene**	client ⇒ server	CopySceneResponse	F M	O
0x40	**CopySceneResponse**	client ⇐ server	N		CopyScene

1.4.9.1. Generic Usage Notes

The scene identifier 0, when used with group identifier 0, is reserved for the global scene used by the On/Off cluster.

1.4.9.2. AddScene Command

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**GroupID**	group-id	all			M
1	**SceneID**	uint8	max 254			M
2	**TransitionTime**	uint32	max 60000000			M
3	**SceneName**	string	max 16			M
4	**ExtensionFieldSetStructs**	list[ExtensionFieldSetStruct]	desc			M

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

1.4.9.2.1. GroupID Field

This field SHALL indicate the group identifier in the Group Table.

1.4.9.2.2. SceneID Field

This field SHALL indicate the scene identifier in the Scene Table.

1.4.9.2.3. TransitionTime Field

This field SHALL indicate the transition time of the scene, measured in milliseconds.

1.4.9.2.4. SceneName Field

This field SHALL indicate the name of the scene.

1.4.9.2.5. ExtensionFieldSetStructs Field

This field SHALL contains the list of extension fields.

1.4.9.2.6. Effect on Receipt

On receipt of this command, the server SHALL perform the following procedure:

1. If the value of the GroupID field is non-zero, the server verifies that the endpoint has an entry for that GroupID in the Group Table. If there is no such entry in the Group Table, the status SHALL be INVALID_COMMAND and the server continues from step 5.
2. If the ExtensionFieldSetStructs list is formatted in a way deemed invalid according to the ExtensionFieldSetStruct structure definition (see ExtensionFieldSetStruct), then the status SHALL be INVALID_COMMAND in the AddSceneResponse, and the server continues from step 5.
3. If a scene already exists under the same group/scene identifier pair, in step 4 the already existing scene entry SHALL be replaced with the new scene being added. Otherwise, if a scene cannot be created due to the lack of Scene Table capacity for the given fabric, the status SHALL be RESOURCE_EXHAUSTED and the server continues from step 5.
4. The server adds the scene entry into its Scene Table with fields copied from the AddScene command data fields and the status SHALL be SUCCESS.
    a. Any ExtensionFieldSetStruct referencing a ClusterID that is not implemented on the endpoint MAY be omitted during processing.
    b. Any AttributeValuePairStruct referencing an AttributeID from the referenced cluster that is not implemented on the endpoint MAY be omitted during processing.
    c. If the ExtensionFieldSetStructs list has multiple entries listing the same ClusterID, the last one within the list SHALL be the one recorded.
    d. Within a single entry of the ExtensionFieldSetStructs list, if an ExtensionFieldSet contains the same AttributeID more than once, the last one within the ExtensionFieldSet SHALL be the one recorded.
5. If the AddScene command was received as a unicast, the server SHALL then generate an AddSceneResponse command with the Status field set to the evaluated status. If the AddScene command was received as a groupcast, the server SHALL NOT generate an AddSceneResponse command.

The SceneTransitionTime field of the Scene Table SHALL be updated with the value of the TransitionTime.

1.4.9.3. AddSceneResponse Command

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**Status**	status	desc			M
1	**GroupID**	group-id	all			M
2	**SceneID**	uint8	max 254			M

1.4.9.3.1. Status Field

This field SHALL be set according to the Effect on Receipt section for AddScene command.

1.4.9.3.2. GroupID Field

The GroupID field SHALL be set to the corresponding field of the received AddScene command.

1.4.9.3.3. SceneID Field

The SceneID field SHALL be set to the corresponding field of the received AddScene command.

1.4.9.3.4. When Generated

This command is generated in response to a received AddScene command.

1.4.9.4. ViewScene Command

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**GroupID**	group-id	all			M
1	**SceneID**	uint8	max 254			M

1.4.9.4.1. GroupID Field

This field SHALL indicate the group identifier in the Group Table.

1.4.9.4.2. SceneID Field

This field SHALL indicate the scene identifier in the Scene Table.

1.4.9.4.3. Effect on Receipt

On receipt of this command, the server SHALL perform the following procedure:

1. If the value of the GroupID field is non-zero, the server verifies that the endpoint has an entry for that GroupID in the Group Table. If there is no such entry in the Group Table, the status SHALL be INVALID_COMMAND and the server continues from step 4.
2. The server verifies that the scene entry corresponding to the GroupID and SceneID fields exists in its Scene Table. If there is no such entry in its Scene Table, the status SHALL be NOT_FOUND and the server continues from step 4.
3. The server retrieves the requested scene entry from its Scene Table and the status SHALL be SUCCESS.
4. If the ViewScene command was received as a unicast, the server SHALL then generate a ViewSceneResponse command with the retrieved scene entry and the Status field set to the evaluated status. If the ViewScene command was received as a groupcast, the server SHALL NOT generate a ViewSceneResponse command.

NOTE The order of attributes within the ExtensionFieldSetStruct MAY differ in implemen-

tation-defined ways from any potential order given in a prior AddScene.

1.4.9.5. ViewSceneResponse Command

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**Status**	status	desc			M
1	**GroupID**	group-id	all			M
2	**SceneID**	uint8	max 254			M
3	**TransitionTime**	uint32	max 60000000			desc
4	**SceneName**	string	max 16			desc
5	**ExtensionFieldSetStructs**	list[ExtensionFieldSetStruct]				desc

1.4.9.5.1. Status Field

This field SHALL be set according to the Effect on Receipt section for ViewScene command.

1.4.9.5.2. GroupID Field

The GroupID field SHALL be set to the corresponding field of the received ViewScene command.

1.4.9.5.3. SceneID Field

The SceneID field SHALL be set to the corresponding field of the received ViewScene command.

1.4.9.5.4. TransitionTime Field

If the status is SUCCESS, this field SHALL be copied from the corresponding field in the Scene Table entry, otherwise it SHALL be omitted.

1.4.9.5.5. SceneName Field

If the status is SUCCESS, this field SHALL be copied from the corresponding field in the Scene Table entry, otherwise it SHALL be omitted.

1.4.9.5.6. ExtensionFieldSetStructs Field

If the status is SUCCESS, this field SHALL be copied from the corresponding field in the Scene Table entry, otherwise it SHALL be omitted.

1.4.9.5.7. When Generated

This command is generated in response to a received ViewScene command.

1.4.9.6. RemoveScene Command

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**GroupID**	group-id	all			M
1	**SceneID**	uint8	max 254			M

1.4.9.6.1. GroupID Field

This field SHALL indicate the group identifier in the Group Table.

1.4.9.6.2. SceneID Field

This field SHALL indicate the scene identifier in the Scene Table.

1.4.9.6.3. Effect on Receipt

On receipt of this command, the server SHALL perform the following procedure:

1. If the value of the GroupID field is non-zero, the server verifies that the endpoint has an entry for that GroupID in the Group Table. If there is no such entry in the Group Table, the status SHALL be INVALID_COMMAND and the server continues from step 4.
2. The server verifies that the scene entry corresponding to the GroupID and SceneID fields exists in its Scene Table. If there is no such entry in its Scene Table, the status SHALL be NOT_FOUND and the server continues from step 4.
3. The server removes the requested scene entry from its Scene Table and the status SHALL be SUCCESS.
4. If the RemoveScene command was received as a unicast, the server SHALL then generate a RemoveSceneResponse command with the Status field set to the evaluated status. If the RemoveScene command was received as a groupcast, the server SHALL NOT generate a RemoveSceneResponse command.

1.4.9.7. RemoveSceneResponse Command

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**Status**	status	desc			M
1	**GroupID**	group-id	all			M
2	**SceneID**	uint8	max 254			M

1.4.9.7.1. Status Field

This field SHALL be set according to the Effect on Receipt section for RemoveScene command.

1.4.9.7.2. GroupID Field

The GroupID field SHALL be set to the corresponding field of the received RemoveScene command.

1.4.9.7.3. SceneID Field

The SceneID field SHALL be set to the corresponding field of the received RemoveScene command.

1.4.9.7.4. When Generated

This command is generated in response to a received RemoveScene command.

1.4.9.8. RemoveAllScenes Command

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**GroupID**	group-id	all			M

1.4.9.8.1. GroupID Field

This field SHALL indicate the group identifier in the Group Table.

1.4.9.8.2. Effect on Receipt

On receipt of this command, the server SHALL perform the following procedure:

1. If the value of the GroupID field is non-zero, the server verifies that the endpoint has an entry for that GroupID in the Group Table. If there is no such entry in the Group Table, the status SHALL be INVALID_COMMAND and the server continues from step 3.
2. The server SHALL remove all scenes, corresponding to the value of the GroupID field, from its Scene Table and the status SHALL be SUCCESS.
3. If the RemoveAllScenes command was received as a unicast, the server SHALL then generate a RemoveAllScenesResponse command with the Status field set to the evaluated status. If the RemoveAllScenes command was received as a groupcast, the server SHALL NOT generate a RemoveAllScenesResponse command.

1.4.9.9. RemoveAllScenesResponse Command

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**Status**	status	desc			M
1	**GroupID**	group-id	all			M

1.4.9.9.1. Status Field

This field SHALL be set according to the Effect on Receipt section for RemoveAllScenes command.

1.4.9.9.2. GroupID Field

The GroupID field SHALL be set to the corresponding field of the received RemoveAllScenes command.

1.4.9.9.3. When Generated

This command is generated in response to a received RemoveAllScenes command.

1.4.9.10. StoreScene Command

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**GroupID**	group-id	all			M
1	**SceneID**	uint8	max 254			M

1.4.9.10.1. GroupID Field

This field SHALL indicate the group identifier in the Group Table.

1.4.9.10.2. SceneID Field

This field SHALL indicate the scene identifier in the Scene Table.

1.4.9.10.3. Effect on Receipt

On receipt of this command, the server SHALL perform the following procedure:

1. If the value of the GroupID field is non-zero, the server verifies that the endpoint has an entry for that GroupID in the Group Table. If there is no such entry in the Group Table, the status SHALL be INVALID_COMMAND and the server continues from step 4.
2. If a scene already exists under the same group/scene identifier pair, in step 3 the already existing scene entry SHALL be used to store the current state.
   If a scene cannot be created due to the lack of Scene Table capacity for the given fabric, the status SHALL be RESOURCE_EXHAUSTED and the server continues from step 4.
3. If a scene already exists under the same group/scene identifier pair, the ExtensionFieldSets of the stored scene SHALL be replaced with the ExtensionFieldSets corresponding to the current state of other clusters on the same endpoint and the other fields of the scene table entry SHALL remain unchanged.
   Otherwise, a new entry SHALL be added to the scene table, using the provided GroupID and SceneID, with SceneTransitionTime set to 0, with SceneName set to the empty string, and with ExtensionFieldSets corresponding to the current state of other clusters on the same endpoint.
   The status SHALL be SUCCESS.
4. If the StoreScene command was received as a unicast, the server SHALL then generate a StoreSceneResponse command with the Status field set to the evaluated status.
   If the StoreScene command was received as a groupcast, the server SHALL NOT generate a StoreSceneResponse command.

Note that if a scene to be stored requires a TransitionTime field and/or a SceneName field, these must be set up by a prior AddScene command, e.g., with no scene extension field sets.

1.4.9.11. StoreSceneResponse Command

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**Status**	status	desc			M
1	**GroupID**	group-id	all			M
2	**SceneID**	uint8	max 254			M

1.4.9.11.1. Status Field

This field SHALL be set according to the Effect on Receipt section for StoreScene command.

1.4.9.11.2. GroupID Field

The GroupID field SHALL be set to the corresponding field of the received StoreScene command.

1.4.9.11.3. SceneID Field

The SceneID field SHALL be set to the corresponding field of the received StoreScene command.

1.4.9.11.4. When Generated

This command is generated in response to a received StoreScene command.

1.4.9.12. RecallScene Command

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**GroupID**	group-id	all			M
1	**SceneID**	uint8	max 254			M
2	**TransitionTime**	uint32	max 60000000	X		O

1.4.9.12.1. GroupID Field

This field SHALL indicate the group identifier in the Group Table.

1.4.9.12.2. SceneID Field

This field SHALL indicate the scene identifier in the Scene Table.

1.4.9.12.3. TransitionTime Field

This field SHALL indicate the transition time of the scene, measured in milliseconds.

1.4.9.12.4. Effect on Receipt

On receipt of this command, the server SHALL perform the following procedure:

1. If the value of the GroupID field is non-zero, the server verifies that the endpoint has an entry for that GroupID in the Group Table. If there is no such entry in the Group Table, the status SHALL be INVALID_COMMAND and the server continues from step 4.
2. The server verifies that the scene entry corresponding to the GroupID and SceneID fields exists in its Scene Table. If there is no such entry in its Scene Table, the status SHALL be NOT_FOUND and the server continues from step 4.
3. The server retrieves the requested scene entry from its Scene Table. For each other cluster implemented on the endpoint, it SHALL retrieve any corresponding extension field sets 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 attributes, the values of these attributes SHALL remain unchanged. If an extension field set would cause an unknown or missing attribute to be set for any reason, that attribute SHALL be skipped. The status SHALL be SUCCESS.
4. If the RecallScene command was received as a unicast, the server SHALL then generate a response with the Status field set to the evaluated status. If the RecallScene command was received as a groupcast, the server SHALL NOT generate a response.

If the TransitionTime data field is present in the command and its value is not equal to null, this field SHALL indicate the transition time in milliseconds. In all other cases (command data field not present or value equal to null), the SceneTransitionTime field of the Scene Table entry SHALL indicate the transition time. The transition time determines 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 exact transition algorithm is implementation-defined.

1.4.9.13. GetSceneMembership Command

This command can be used to get the used scene identifiers within a certain group, for the endpoint that implements this cluster.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**GroupID**	group-id	all			M

1.4.9.13.1. GroupID Field

This field SHALL indicate the group identifier in the Group Table.

1.4.9.13.2. Effect on Receipt

On receipt of this command, the server SHALL perform the following procedure:

1. If the value of the GroupID field is non-zero, the server verifies that the endpoint has an entry for that GroupID in the Group Table. If there is no such entry in the Group Table, the status SHALL be INVALID_COMMAND and the server continues from step 3.
2. The status SHALL be SUCCESS.
3. If the GetSceneMembership command was received as a unicast, the server SHALL then gener-

ate a GetSceneMembershipResponse command with the Status field set to the evaluated status. If the GetSceneMembership command was not received as a unicast, the server SHALL only generate a GetSceneMembershipResponse command with the Status field set to the evaluated status if an entry within the Scene Table corresponds to the GroupID.

1.4.9.14. GetSceneMembershipResponse Command

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**Status**	status	desc			M
1	**Capacity**	uint8	all	X		M
2	**GroupID**	group-id	all			M
3	**SceneList**	list[uint8]				Status==Success

1.4.9.14.1. Status Field

This field SHALL be set according to the Effect on Receipt section for GetSceneMembership command.

1.4.9.14.2. Capacity Field

This field SHALL contain the remaining capacity of the Scene Table of the server (for all groups for the accessing fabric). The following values apply:

- 0 - No further scenes MAY be added.
- 0 < Capacity < 0xFE - Capacity holds the number of scenes that MAY be added.
- 0xFE - At least 1 further scene MAY be added (exact number is unknown).
- null - It is unknown if any further scenes MAY be added.

1.4.9.14.3. GroupID Field

This field SHALL be set to the corresponding field of the received GetSceneMembership command.

1.4.9.14.4. SceneList Field

If the status is not SUCCESS then this field SHALL be omitted, else this field SHALL contain the identifiers of all the scenes in the Scene Table with the corresponding Group ID.

1.4.9.14.5. When Generated

This command is generated in response to a received GetSceneMembership command.

1.4.9.15. CopyScene Command

This command allows a client to efficiently copy scenes from one group/scene identifier pair to another group/scene identifier pair.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**Mode**	CopyModeBitmap	desc			M
1	**GroupIdentifierFrom**	group-id	all			M
2	**SceneIdentifierFrom**	max 254	all			M
3	**GroupIdentifierTo**	group-id	all			M
4	**SceneIdentifierTo**	max 254	all			M

1.4.9.15.1. Mode Field

This field SHALL contain the information of how the scene copy is to proceed.

The CopyAllScenes bit of the Mode indicates whether all scenes are to be copied. If this value is set to 1, all scenes are to be copied and the SceneIdentifierFrom and SceneIdentifierTo fields SHALL be ignored. Otherwise this bit is set to 0.

1.4.9.15.2. GroupIdentifierFrom Field

This field SHALL indicate the identifier of the group from which the scene is to be copied. Together with the SceneIdentifierFrom field, this field uniquely identifies the scene to copy from the Scene Table.

1.4.9.15.3. SceneIdentifierFrom Field

This field SHALL indicate the identifier of the scene from which the scene is to be copied. Together with the GroupIdentifierFrom field, this field uniquely identifies the scene to copy from the Scene Table.

1.4.9.15.4. GroupIdentifierTo Field

This field SHALL indicate the identifier of the group to which the scene is to be copied. Together with the SceneIdentifierTo field, this field uniquely identifies the scene to copy to the Scene Table.

1.4.9.15.5. SceneIdentifierTo Field

This field SHALL indicate the identifier of the scene to which the scene is to be copied. Together with the GroupIdentifierTo field, this field uniquely identifies the scene to copy to the Scene Table.

1.4.9.15.6. Effect on Receipt

On receipt of this command, the server SHALL perform the following procedure:

1. If the value of either the GroupIdentifierFrom field or the Group Identifier To field is non-zero, the server verifies that the endpoint has an entry for these non-zero group identifiers in the

Group Table. If there are no such entries in the Group Table, the status SHALL be INVALID_-COMMAND and the server continues from step 5.

2. If the CopyAllScenes sub-field of the Mode field is set to 0, the server verifies that the scene entry corresponding to the GroupIdentifierFrom and SceneIdentifierFrom fields exists in its Scene Table. If there is no such entry in its Scene Table, the status SHALL be NOT_FOUND and the server continues from step 5.
3. If the CopyAllScenes sub-field of the Mode field is set to 1, the server SHALL copy all its available scenes with group identifier equal to the GroupIdentifierFrom field under the group identifier specified in the GroupIdentifierTo field, leaving the scene identifiers the same. In this case, the SceneIdentifierFrom and SceneIdentifierTo fields SHALL be ignored.
   If the CopyAllScenes sub-field of the Mode field is set to 0, the server SHALL copy the Scene Table entry corresponding to the GroupIdentifierFrom and SceneIdentifierFrom fields to the Scene Table entry corresponding to the GroupIdentifierTo and SceneIdentifierTo fields.
   Regardless of the value of the CopyAllScenes subfield, if a scene already exists under the same group/scene identifier pair, it SHALL be replaced with the scene being copied.
   Regardless of the value of the CopyAllScenes subfield, if a scene cannot be copied due to the lack of Scene Table capacity for the given fabric, the status SHALL be RESOURCE_EXHAUSTED and the server continues from step 5. In this case, scenes already copied SHALL be kept.
4. The status SHALL be SUCCESS.
5. If the CopyScene command was received as a unicast, the server SHALL then generate a CopySceneResponse command with the Status field set to the evaluated status. If the CopyScene command was received as a groupcast, the server SHALL NOT generate a CopySceneResponse command.

1.4.9.16. CopySceneResponse Command

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**Status**	status	desc			M
1	**GroupIdentifierFrom**	group-id	all			M
2	**SceneIdentifierFrom**	uint8	max 254			M

1.4.9.16.1. Status Field

This field SHALL be set according to the Effect on Receipt section for the CopyScene command.

1.4.9.16.2. GroupIdentifierFrom Field

This field SHALL be set to the same values as in the corresponding fields of the received CopyScene command.

1.4.9.16.3. SceneIdentifierFrom Field

This field SHALL be set to the same values as in the corresponding fields of the received CopyScene

command.

1.4.9.16.4. When Generated

The CopySceneResponse command is generated in response to a received CopyScene command.

1.5. On/Off Cluster

Attributes and commands for turning devices on and off.

1.5.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Revision	Description
1	Global mandatory ClusterRevision attribute added; CCB 1555
2	ZLO 1.0: StartUpOnOff
3	FeatureMap global attribute support with Level Control and Lighting feature
4	New data model format and notation
5	Addition of Dead Front behavior and associated FeatureMap entry
6	Addition of OffOnly feature and associated FeatureMap entry

1.5.2. Classification

Hierarchy	Role	Scope	PICS Code
Base	Application	Endpoint	OO

1.5.3. Cluster ID

ID	Name
0x0006	On/Off

1.5.4. Features

This cluster SHALL support the FeatureMap bitmap attribute as defined below.

Bit	Code	Feature	Conformance	Summary
0	**LT**	Lighting	[!OFFONLY]	Behavior that supports lighting applications.
1	**DF**	DeadFrontBehavior	[!OFFONLY]	Device has Dead Front behavior
2	**OFFONLY**	OffOnly	[!(LT | DF)]	Device supports the OffOnly feature

1.5.4.1. Lighting Feature

This cluster is used for a lighting application.

On receipt of a Level Control cluster command that causes the OnOff attribute to be set to FALSE, the OnTime attribute SHALL be set to 0.

On receipt of a Level Control cluster command that causes the OnOff attribute to be set to TRUE, if the value of the OnTime attribute is equal to 0, the server SHALL set the OffWaitTime attribute to 0.

1.5.4.2. DeadFrontBehavior Feature

When this feature is supported, the device exposing this server cluster exhibits "dead front" behavior when the "OnOff" attribute is FALSE (Off). This "dead front" behavior includes:

- clusters other than this cluster that are also exposed MAY respond with failures to Invoke and Write interactions. Such failure responses when in a "dead front" SHALL be with an INVALID_IN_STATE status code.
- clusters other than this cluster MAY change the values of their attributes to best-effort values, due to the actual values not being defined or available in this state. Device type specifications that require support for the DF feature SHOULD define what these best-effort values are.
- Report Transactions SHALL continue to be generated. Such transactions MAY include best-effort values as noted above.
- Event generation logic for clusters other than this cluster is unchanged (noting possible use of best-effort attribute values as in the preceding bullets).

When this feature is supported and the OnOff attribute changes from TRUE to FALSE (e.g. when receiving an Off Command, or due to a manual interaction on the device), it SHALL start executing this "dead front" behavior.

When this feature is supported and the OnOff attribute changes from FALSE to TRUE (e.g. when receiving an On Command, or due to a manual interaction on the device), it SHALL stop executing this "dead front" behavior.

When this feature is supported, and any change of the "dead front" state leads to changes in attributes of other clusters due to the "dead front" feature, these attribute changes SHALL NOT be skipped or omitted from the usual processing associated with attribute changes. For example, if an

attribute changes from value 4 to null on "dead front" behavior due to an Off command being received, this change SHALL be processed for reporting and subscriptions.

1.5.4.3. OffOnly Feature

When this feature is supported, the Off command SHALL be supported and the On and Toggle commands SHALL NOT be supported.

This feature is useful for devices which can be turned off via the Off command received by an instance of this cluster but cannot be turned on via commands received by an instance of this cluster due to regulatory requirements.

1.5.5. Data Types

1.5.5.1. OnOffControlBitmap Type

This data type is derived from map8.

Bit	Name	Summary	Conformance
0	**AcceptOnlyWhenOn**	Indicates a command is only accepted when in On state.	M

1.5.5.2. StartUpOnOffEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**Off**	Set the OnOff attribute to FALSE	M
1	**On**	Set the OnOff attribute to TRUE	M
2	**Toggle**	If the previous value of the OnOff attribute is equal to FALSE, set the OnOff attribute to TRUE. If the previous value of the OnOff attribute is equal to TRUE, set the OnOff attribute to FALSE (toggle).	M

1.5.5.3. EffectIdentifierEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0x00	DelayedAllOff	Delayed All Off	M
0x01	DyingLight	Dying Light	M

1.5.5.4. DelayedAllOffEffectVariantEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0x00	DelayedOffFastFade	Fade to off in 0.8 seconds	M
0x01	NoFade	No fade	M
0x02	DelayedOffSlowFade	50% dim down in 0.8 seconds then fade to off in 12 seconds	M

1.5.5.5. DyingLightEffectVariantEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0x00	DyingLightFadeOff	20% dim up in 0.5s then fade to off in 1 second	M

1.5.6. Attributes

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0000	**OnOff**	bool	all	SN	FALSE	R V	M
0x4000	**GlobalSceneControl**	bool	all		TRUE	R V	LT
0x4001	**OnTime**	uint16	all		0	RW VO	LT
0x4002	**OffWaitTime**	uint16	all		0	RW VO	LT
0x4003	**StartUpOnOff**	StartUpOnOffEnum	desc	XN	MS	RW VM	LT

1.5.6.1. Scene Table Extensions

If the Scenes Management server cluster is implemented on the same endpoint, the following attribute SHALL be part of the ExtensionFieldSetStruct of the Scene Table.

- OnOff

1.5.6.2. OnOff Attribute

This attribute indicates whether the device type implemented on the endpoint is turned off or turned on, in these cases the value of the OnOff attribute equals FALSE, or TRUE respectively.

1.5.6.3. GlobalSceneControl Attribute

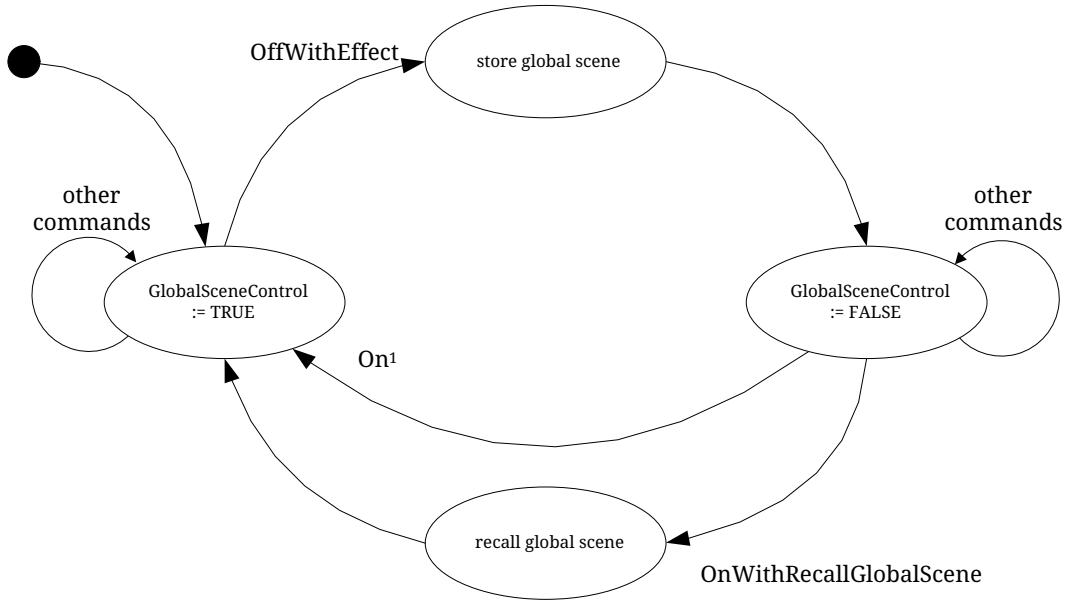
In order to support the use case where the user gets back the last setting of a set of devices (e.g. level settings for lights), a global scene is introduced which is stored when the devices are turned off and recalled when the devices are turned on. The global scene is defined as the scene that is stored with group identifier 0 and scene identifier 0.

This attribute is defined in order to prevent a second Off command storing the all-devices-off situation as a global scene, and to prevent a second On command destroying the current settings by going back to the global scene.

This attribute SHALL be set to TRUE after the reception of a command which causes the OnOff attribute to be set to TRUE, such as a standard On command, a MoveToLevel(WithOnOff) command, a RecallScene command or a OnWithRecallGlobalScene command.

This attribute is set to FALSE after reception of a OffWithEffect command.

These concepts are illustrated in Explanation of the Behavior of Store and Recall Global Scene functionality using a State Diagram.



Note 1: Any command which causes the OnOff attribute to be set to TRUE except OnWithRecallGlobalScene, e.g. On or Toggle.

Figure 1. Explanation of the Behavior of Store and Recall Global Scene functionality using a State Diagram

1.5.6.4. OnTime Attribute

This attribute specifies the length of time (in 1/10ths second) that the On state SHALL be maintained before automatically transitioning to the Off state when using the OnWithTimedOff command. This attribute can be written at any time, but writing a value only has effect when in the Timed On state. See OnWithTimedOff for more details.

1.5.6.5. OffWaitTime Attribute

This attribute specifies the length of time (in 1/10ths second) that the Off state SHALL be guarded to prevent another OnWithTimedOff command turning the server back to its On state (e.g., when leaving a room, the lights are turned off but an occupancy sensor detects the leaving person and attempts to turn the lights back on). This attribute can be written at any time, but writing a value only has an effect when in the Timed On state followed by a transition to the Delayed Off state, or in the Delayed Off state. See OnWithTimedOff for more details.

1.5.6.6. StartUpOnOff Attribute

This attribute SHALL define the desired startup behavior of a device when it is supplied with power and this state SHALL be reflected in the OnOff attribute. If the value is null, the OnOff attribute is set to its previous value. Otherwise, the behavior is defined in the table defining StartUpOnOffEnum.

This behavior does not apply to reboots associated with OTA. After an OTA restart, the OnOff attribute SHALL return to its value prior to the restart.

1.5.7. Commands

ID	Name	Direction	Response	Access	Conformance
0x00	**Off**	client ⇒ server	Y	O	M
0x01	**On**	client ⇒ server	Y	O	!OFFONLY
0x02	**Toggle**	client ⇒ server	Y	O	!OFFONLY
0x40	**OffWithEffect**	client ⇒ server	Y	O	LT
0x41	**OnWithRecallGlobalScene**	client ⇒ server	Y	O	LT
0x42	**OnWithTimedOff**	client ⇒ server	Y	O	LT

1.5.7.1. Off Command

1.5.7.1.1. Effect on Receipt

On receipt of the Off command, a server SHALL set the OnOff attribute to FALSE.

Additionally, when the OnTime attribute is supported, the server SHALL set the OnTime attribute to 0.

1.5.7.2. On Command

1.5.7.2.1. Effect on Receipt

If the OffOnly feature is supported, on receipt of the On command, an UNSUPPORTED_COMMAND failure status response SHALL be sent. Otherwise, on receipt of the On command, a server SHALL set the OnOff attribute to TRUE.

Additionally, when the OnTime and OffWaitTime attributes are both supported, if the value of the OnTime attribute is equal to 0, the server SHALL set the OffWaitTime attribute to 0.

1.5.7.3. Toggle Command

1.5.7.3.1. Effect on Receipt

If the OffOnly feature is supported, on receipt of the Toggle command, an UNSUPPORTED_COMMAND failure status response SHALL be sent. Otherwise, on receipt of the Toggle command, if the value of the OnOff attribute is equal to FALSE, the server SHALL set the OnOff attribute to TRUE, otherwise, the server SHALL set the OnOff attribute to FALSE.

Additionally, when the OnTime and OffWaitTime attributes are both supported, if the value of the OnOff attribute is equal to FALSE and if the value of the OnTime attribute is equal to 0, the server SHALL set the OffWaitTime attribute to 0. If the value of the OnOff attribute is equal to TRUE, the server SHALL set the OnTime attribute to 0.

1.5.7.4. OffWithEffect Command

The OffWithEffect command allows devices to be turned off using enhanced ways of fading.

The OffWithEffect command SHALL have the following data fields:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**EffectIdentifier**	EffectIdentifierEnum	desc			M
1	**EffectVariant**	enum8	desc		0	M

1.5.7.4.1. EffectIdentifier Field

This field specifies the fading effect to use when turning the device off. This field SHALL contain one of the non-reserved values listed in EffectIdentifierEnum.

1.5.7.4.2. EffectVariant Field

This field is used to indicate which variant of the effect, indicated in the EffectIdentifier field, SHOULD be triggered. If the server does not support the given variant, it SHALL use the default variant. This field is dependent on the value of the EffectIdentifier field and SHALL contain one of the non-reserved values listed in either DelayedAllOffEffectVariantEnum or DyingLightEffectVariantEnum.

1.5.7.4.3. Effect on Receipt

On receipt of the OffWithEffect command the server SHALL check the value of the GlobalSceneControl attribute.

If the GlobalSceneControl attribute is equal to TRUE, the server SHALL store its settings in its global scene then set the GlobalSceneControl attribute to FALSE, then set the OnOff attribute to FALSE and if the OnTime attribute is supported set the OnTime attribute to 0.

If the GlobalSceneControl attribute is equal to FALSE, the server SHALL only set the OnOff attribute to FALSE.

1.5.7.5. OnWithRecallGlobalScene Command

This command allows the recall of the settings when the device was turned off.

1.5.7.5.1. Effect on Receipt

On receipt of the OnWithRecallGlobalScene command, if the GlobalSceneControl attribute is equal to TRUE, the server SHALL discard the command.

If the GlobalSceneControl attribute is equal to FALSE, the Scene cluster server on the same endpoint SHALL recall its global scene, updating the OnOff attribute accordingly. The OnOff server SHALL then set the GlobalSceneControl attribute to TRUE.

Additionally, when the OnTime and OffWaitTime attributes are both supported, if the value of the OnTime attribute is equal to 0, the server SHALL set the OffWaitTime attribute to 0.

1.5.7.6. OnWithTimedOff Command

This command allows devices to be turned on for a specific duration with a guarded off duration so that SHOULD the device be subsequently turned off, further OnWithTimedOff commands, received during this time, are prevented from turning the devices back on. Further OnWithTimedOff commands received while the server is turned on, will update the period that the device is turned on.

The OnWithTimedOff command SHALL have the following data fields:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**OnOffControl**	OnOffControlBitmap	0 to 1			M
1	**OnTime**	uint16	max 0xFFFE			M
2	**OffWaitTime**	uint16	max 0xFFFE			M

1.5.7.6.1. OnOffControl Field

This field contains information on how the server is to be operated.

1.5.7.6.1.1. AcceptOnlyWhenOn Bit

This bit specifies whether the OnWithTimedOff command is to be processed unconditionally or only when the OnOff attribute is equal to TRUE. If this sub-field is set to 1, the OnWithTimedOff command SHALL only be accepted if the OnOff attribute is equal to TRUE. If this sub-field is set to 0, the OnWithTimedOff command SHALL be processed unconditionally.

1.5.7.6.2. OnTime Field

This field is used to adjust the value of the OnTime attribute.

1.5.7.6.3. OffWaitTime Field

This field is used to adjust the value of the OffWaitTime attribute.

1.5.7.6.4. Effect on Receipt

On receipt of this command, if the AcceptOnlyWhenOn sub-field of the OnOffControl field is set to 1, and the value of the OnOff attribute is equal to FALSE, the command SHALL be discarded.

If the value of the OffWaitTime attribute is greater than zero and the value of the OnOff attribute is equal to FALSE, then the server SHALL set the OffWaitTime attribute to the minimum of the OffWaitTime attribute and the value specified in the OffWaitTime field.

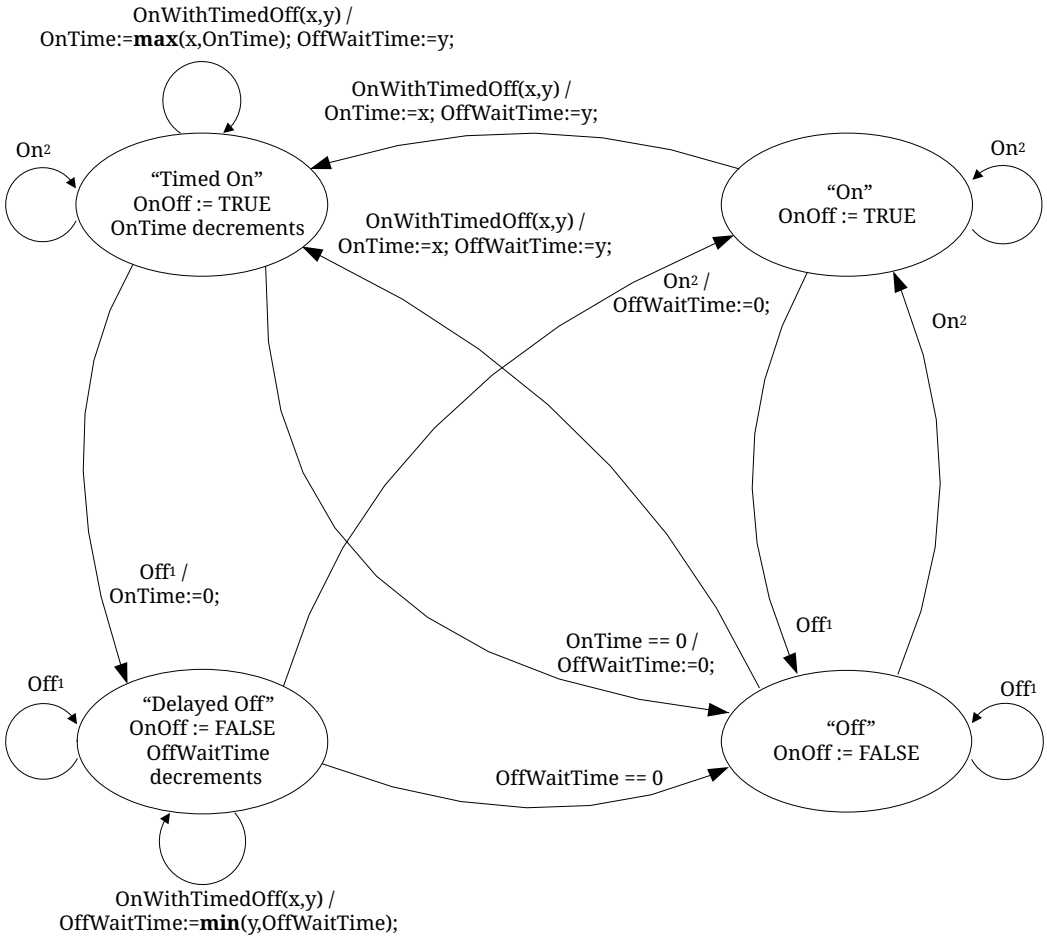
In all other cases, the server SHALL set the OnTime attribute to the maximum of the OnTime attribute and the value specified in the OnTime field, set the OffWaitTime attribute to the value specified in the OffWaitTime field and set the OnOff attribute to TRUE.

If the values of the OnTime and OffWaitTime attributes are both not equal to 0xFFFF, the server SHALL then update these attributes every 1/10th second until both the OnTime and OffWaitTime attributes are equal to 0, as follows:

- If the value of the OnOff attribute is equal to TRUE and the value of the OnTime attribute is greater than zero, the server SHALL decrement the value of the OnTime attribute. If the value of the OnTime attribute reaches 0, the server SHALL set the OffWaitTime and OnOff attributes to 0 and FALSE, respectively.
- If the value of the OnOff attribute is equal to FALSE and the value of the OffWaitTime attribute is greater than zero, the server SHALL decrement the value of the OffWaitTime attribute. If the value of the OffWaitTime attribute reaches 0, the server SHALL terminate the update.

1.5.8. State Description

The operation of the On/Off cluster with respect to the On, Off, and OnWithTimedOff commands is illustrated in On/Off Cluster Operation State Machine. In this diagram, the values X and Y correspond to the OnTime and OffWaitTime fields, respectively, of the OnWithTimedOff command. In the Timed On state, the OnTime attribute is decremented every 1/10th second, unless its value equals 0xFFFF. Similarly, in the Delayed Off state, the OffWaitTime attribute is decremented every 1/10th second, unless its value equals 0xFFFF.



Off[1] : Any command which causes the OnOff attribute to be set to FALSE, e.g. Off, Toggle or OffWithEffect.
On[2] : Any command which causes the OnOff attribute to be set to TRUE, e.g. On, Toggle or OnWithRecallGlobalScene.

Figure 2. On/Off Cluster Operation State Machine

1.6. Level Control Cluster

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.

1.6.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Revision	Description
1	Global mandatory ClusterRevision attribute added
2	Added Options attribute, state change table; ZLO 1.0; Base cluster (no change) CCB 2085 1775 2281 2147

Revision	Description
3	CCB 2574 2616 2659 2702 2814 2818 2819 2898
4	FeatureMap support with On/Off, Lighting and Frequency features
5	New data model format and notation

1.6.2. Classification

Hierarchy	Role	Scope	PICS Code
Base	Application	Endpoint	LVL

1.6.3. Cluster ID

Derived cluster specifications are defined elsewhere. This base cluster specification MAY be used for generic level control; however, it is recommended to derive another cluster to better define the application and domain requirements. If one of more derived cluster identifiers and the base identifier exists on a device endpoint, then they SHALL all represent a single instance of the device level control.

ID	Hierarchy	Name	Conformance
0x0008	Base	Level Control	
0x001C	Derived	Pulse Width Modulation	P

NOTE Pulse Width Modulation cluster is provisional.

1.6.4. Features

This cluster SHALL support the FeatureMap bitmap attribute as defined below.

Bit	Code	Feature	Default	Conformance	Summary
0	**OO**	OnOff	1	O	Dependency with the On/Off cluster
1	**LT**	Lighting	0	O	Behavior that supports lighting applications

Bit	Code	Feature	Default	Conformance	Summary
2	**FQ**	Frequency	0	P	Supports frequency attributes and behavior. The Pulse Width Modulation cluster was created for frequency control.

The following sections describe each feature in some detail. Further details are found within the specification.

1.6.4.1. On/Off Feature

For many applications, a close relationship between this cluster and the On/Off cluster is needed. This section describes the dependencies that are required when an endpoint that implements this server cluster and also implements the On/Off server cluster. Before the On/Off feature bit in the FeatureMap existed, there was a dependency between this cluster and the On/Off cluster.

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.

1.6.4.1.1. Effect of On/Off Commands on the CurrentLevel Attribute

The OnLevel attribute 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 equal to null) 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 the Effect of On/Off Commands when associated with Level Control table. In this table, and throughout this cluster specification, 'level' means the value of the CurrentLevel attribute.

Table 2. Effect of 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. Change CurrentLevel to OnLevel, or to the stored level if OnLevel is not defined, over the time period OnOffTransitionTime.

Command	Action On Receipt
Off	Temporarily store CurrentLevel. Change 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 FALSE, proceed as for the On command. Otherwise proceed as for the Off command.

Intention of the actions described in the table above is that CurrentLevel, which was in effect before any of the On, Off or Toggle commands were issued, SHALL be restored, after the transition is completed. If another of these commands is received, before the transition is completed, the originally stored CurrentLevel SHALL be preserved and restored.

1.6.4.1.2. 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 (MoveToLevel, Move and Step commands) SHALL NOT affect 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 separate 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 FALSE, and when the level is increased above its minimum the OnOff attribute is automatically turned to TRUE. As an example, this represents the behavior of a light dimmer with no independent on/off switch.

For the Stop command, the StopWithOnOff is included solely for symmetry, to allow easy choice of one or other set of commands – both Stop commands are identical, because the dependency on On/Off is determined by the original command that is being stopped.

1.6.4.1.3. Effect of Level Control Commands Depends on OnOff

Before the Options attribute was introduced, all commands except those postfixed with ‘with On/Off’, had no effect if the OnOff attribute of the On/Off cluster, on the same endpoint, was FALSE. Even if the On/Off (OO) feature set bit is set to zero, this is still true. To allow such commands to function, please see the Options attribute below.

1.6.4.1.4. GlobalSceneControl and Commands with On/Off

If a MoveToLevel(WithOnOff), Move(WithOnOff) or Step(WithOnOff) command is received that causes a change to the value of the OnOff attribute of the On/Off cluster, the value of the GlobalSceneControl attribute of the On/Off cluster SHALL be updated according to section GlobalSceneControl.

1.6.4.2. Lighting Feature

This feature supports an interface for controlling the level of a light source.

For the CurrentLevel attribute:

A value of 0x00 SHALL NOT be used.

A value of 0x01 SHALL indicate the minimum level that can be attained on a device.

A value of 0xFE SHALL indicate the maximum level that can be attained on a device.

A value of null SHALL represent an undefined value.

All other values are application specific gradations from the minimum to the maximum level.

1.6.4.3. Frequency Feature

NOTE	The Frequency feature is provisional.

1.6.5. Data Types

1.6.5.1. OptionsBitmap Type

This data type is derived from map8.

Bit	Name	Summary	Conformance
0	**ExecuteIfOff**	Dependency on On/Off cluster	LT | OO
1	**CoupleColorTemp-ToLevel**	Dependency on Color Control cluster	LT

1.6.5.1.1. ExecuteIfOff Bit

This bit indicates if this cluster has a dependency with the On/Off cluster.

1.6.5.1.2. CoupleColorTempToLevel Bit

This bit indicates if this cluster has a dependency with the Color Control cluster.

1.6.5.2. MoveModeEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**Up**	Increase the level	M
1	**Down**	Decrease the level	M

1.6.5.3. StepModeEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**Up**	Step upwards	M
1	**Down**	Step downwards	M

1.6.6. Attributes

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0000	**CurrentLevel**	uint8	MinLevel to MaxLevel	SNX	null	R V	M
0x0001	**RemainingTime**	uint16	all		0	R V	LT
0x0002	**MinLevel**	uint8	1 to MaxLevel		1	R V	[LT]
0x0002	**MinLevel**	uint8	0 to MaxLevel		0	R V	[!LT]
0x0003	**MaxLevel**	uint8	MinLevel to 254		254	R V	O
0x0004	**CurrentFrequency**	uint16	MinFrequency to MaxFrequency	PS	0	R V	FQ
0x0005	**MinFrequency**	uint16	0 to MaxFrequency		0	R V	FQ
0x0006	**MaxFrequency**	uint16	min MinFrequency		0	R V	FQ
0x0010	**OnOffTransitionTime**	uint16	all		0	RW VO	O
0x0011	**OnLevel**	uint8	MinLevel to MaxLevel	X	null	RW VO	M
0x0012	**OnTransitionTime**	uint16	all	X	null	RW VO	O
0x0013	**OffTransitionTime**	uint16	all	X	null	RW VO	O
0x0014	**DefaultMoveRate**	uint8	all	X	MS	RW VO	O

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x000F	**Options**	Options-Bitmap	desc		0	RW VO	M
0x4000	**StartUpCurrentLevel**	uint8	desc	XN	MS	RW VM	LT

1.6.6.1. Scene Table Extensions

If the Scenes Management server cluster is implemented on the same endpoint, the following attributes SHALL be part of the ExtensionFieldSetStruct of the Scene Table. If the implicit form of the ExtensionFieldSetStruct is used, the order of the attributes in the AttributeValueList is in the given order, i.e., the attribute listed as 1 is added first:

1. CurrentLevel
2. CurrentFrequency

Attributes in the scene table that are not supported by the device (according to the FeatureMap attribute) SHALL be present in the scene table but ignored. An extension field set value outside of the accepted values of the corresponding MoveTo<Target> command, e.g. a null value for the CurrentLevel attribute, corresponding to the MoveToLevel command, SHALL be ignored.

1.6.6.2. CurrentLevel Attribute

This attribute SHALL indicate the current level of this device. The meaning of 'level' is device dependent.

1.6.6.3. RemainingTime Attribute

This attribute SHALL indicate the time remaining until the current command is complete - it is specified in 1/10ths of a second.

1.6.6.4. MinLevel Attribute

This attribute SHALL indicate the minimum value of CurrentLevel that is capable of being assigned.

1.6.6.5. MaxLevel Attribute

This attribute SHALL indicate the maximum value of CurrentLevel that is capable of being assigned.

1.6.6.6. CurrentFrequency Attribute

This attribute SHALL indicate the frequency at which the device is at CurrentLevel. A CurrentFrequency of 0 is unknown.

1.6.6.7. MinFrequency Attribute

This attribute SHALL indicate the minimum value of CurrentFrequency that is capable of being assigned. MinFrequency SHALL be less than or equal to MaxFrequency. A value of 0 indicates undefined.

1.6.6.8. MaxFrequency Attribute

This attribute SHALL indicate the maximum value of CurrentFrequency that is capable of being assigned. MaxFrequency SHALL be greater than or equal to MinFrequency. A value of 0 indicates undefined.

1.6.6.9. Options Attribute

This attribute SHALL indicate the selected options of the device.

The Options attribute is a bitmap that determines the default behavior of some cluster commands. Each command that is dependent on the Options attribute SHALL first construct a temporary Options bitmap that is in effect during the command processing. The temporary Options bitmap has the same format and meaning as the Options attribute, but includes any bits that may be overridden by command fields.

This attribute is meant to be changed only during commissioning.

Command execution SHALL NOT continue beyond the Options processing if all of these criteria are true:

- The command is one of the ‘without On/Off’ commands: Move, Move to Level, Step, or Stop.
- The On/Off cluster exists on the same endpoint as this cluster.
- The OnOff attribute of the On/Off cluster, on this endpoint, is FALSE.
- The value of the ExecuteIfOff bit is 0.

1.6.6.9.1. ExecuteIfOff Bit

If this bit is set, commands in this cluster are executed and potentially change the CurrentLevel attribute when the OnOff attribute of the On/Off cluster is FALSE.

1.6.6.9.2. CoupleColorTempToLevel Bit

If this bit is set, changes to the CurrentLevel attribute SHALL be coupled with the color temperature set in the Color Control cluster.

When not supporting the Lighting feature, this bit SHALL be zero and ignored.

1.6.6.10. OnOffTransitionTime Attribute

This attribute SHALL indicate the time taken to move to or from the target level when On or 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. Please note that if the device is not able to move at a variable rate, the OnOffTransitionTime attribute SHOULD NOT be implemented.

1.6.6.11. OnLevel Attribute

This attribute SHALL indicate 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 TRUE, as a result of processing an On/Off cluster command. If the OnLevel attribute is not implemented, or is set to the null value, it has no effect. For more details see Effect of On/Off Commands on the CurrentLevel Attribute.

OnLevel represents a mandatory field that was previously not present or optional. Implementers should be aware that older devices may not implement it.

1.6.6.12. OnTransitionTime Attribute

This attribute SHALL indicate the time taken to move the current level from the minimum level to the maximum level when an On command is received by an On/Off cluster on the same endpoint. It is specified in 1/10ths of a second. If this attribute is not implemented, or contains a null value, the OnOffTransitionTime SHALL be used instead.

1.6.6.13. OffTransitionTime Attribute

This attribute SHALL indicate the time taken to move the current level from the maximum level to the minimum level when an Off command is received by an On/Off cluster on the same endpoint. It is specified in 1/10ths of a second. If this attribute is not implemented, or contains a null value, the OnOffTransitionTime SHALL be used instead.

1.6.6.14. DefaultMoveRate Attribute

This attribute SHALL indicate the movement rate, in units per second, when a Move command is received with a null value Rate parameter.

1.6.6.15. StartUpCurrentLevel Attribute

This attribute SHALL indicate the desired startup level for a device when it is supplied with power and this level SHALL be reflected in the CurrentLevel attribute. The values of the StartUpCurrentLevel attribute are listed below:

Value	Action on power up
0	Set the CurrentLevel attribute to the minimum value permitted on the device
null	Set the CurrentLevel attribute to its previous value
other values	Set the CurrentLevel attribute to this value

This behavior does not apply to reboots associated with OTA. After an OTA restart, the CurrentLevel attribute SHALL return to its value prior to the restart.

1.6.7. Commands

ID	Name	Direction	Response	Access	Conformance
0x00	**MoveToLevel**	client ⇒ server	Y	O	M
0x01	**Move**	client ⇒ server	Y	O	M
0x02	**Step**	client ⇒ server	Y	O	M
0x03	**Stop**	client ⇒ server	Y	O	M
0x04	**MoveToLevel-WithOnOff**	client ⇒ server	Y	O	M
0x05	**MoveWith-OnOff**	client ⇒ server	Y	O	M
0x06	**StepWith-OnOff**	client ⇒ server	Y	O	M
0x07	**StopWith-OnOff**	client ⇒ server	Y	O	M
0x08	**MoveToClos-estFrequency**	client ⇒ server	Y	O	FQ

1.6.7.1. MoveToLevel Command

ID	Name	Type	Constraint	Quality	Default	Confor-mance
0	**Level**	uint8	0 to 254			M
1	**Transition-Time**	uint16	all	X		M
2	**Options-Mask**	Options-Bitmap	desc		0	M
3	**OptionsOve rride**	Options-Bitmap	desc		0	M

1.6.7.1.1. Effect on Receipt

The OptionsMask and OptionsOverride fields SHALL both be present. Default values are provided to interpret missing fields from legacy devices. A temporary Options bitmap SHALL be created from the Options attribute, using the OptionsMask and OptionsOverride fields. Each bit of the temporary Options bitmap SHALL be determined as follows:

Each bit in the Options attribute SHALL determine the corresponding bit in the temporary Options bitmap, unless the OptionsMask field is present and has the corresponding bit set to 1, in which case the corresponding bit in the OptionsOverride field SHALL determine the corresponding bit in the temporary Options bitmap.

The resulting temporary Options bitmap SHALL then be processed as defined in the Options

attribute.

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 TransitionTime field, in tenths of a second, or as close to this as the device is able.

If the TransitionTime field takes the value null 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 TransitionTime field MAY be disregarded.

1.6.7.2. Move Command

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**MoveMode**	MoveModeEnum	desc			M
1	**Rate**	uint8	all	X		M
2	**OptionsMask**	OptionsBitmap	desc		0	M
3	**OptionsOverride**	OptionsBitmap	desc		0	M

1.6.7.2.1. MoveMode Field

This field SHALL be one of the non-reserved values in MoveModeEnum.

1.6.7.2.2. Rate Field

This field SHALL indicate 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 equal to null, then the value in DefaultMoveRate attribute SHALL be used. However, if the Rate field is equal to null and the DefaultMoveRate attribute is not supported, or if the Rate field is equal to null and the value of the DefaultMoveRate attribute is equal to null, then 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.

1.6.7.2.3. Effect on Receipt

On receipt of this command, a device SHALL first create and process a temporary Options bitmap as described in the MoveToLevel command.

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 below.

MoveMode	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.

1.6.7.3. Step Command

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**StepMode**	StepModeEnum	desc			M
1	**StepSize**	uint8	all			M
2	**TransitionTime**	uint16	all	X		M
3	**OptionsMask**	OptionsBitmap	desc		0	M
4	**OptionsOverride**	OptionsBitmap	desc		0	M

1.6.7.3.1. StepMode Field

This field SHALL be one of the non-reserved values in StepModeEnum.

1.6.7.3.2. StepSize Field

This field SHALL indicate the change to CurrentLevel.

1.6.7.3.3. TransitionTime Field

This field SHALL indicate the time that SHALL be taken to perform the step, in tenths of a second. A step is a change in the CurrentLevel of StepSize units. The actual time taken SHOULD be as close to this as the device is able. If the TransitionTime field is equal to null, the device SHOULD move as fast as it is able.

If the device is not able to move at a variable rate, the TransitionTime field MAY be disregarded.

1.6.7.3.4. Effect on Receipt

On receipt of this command, a device SHALL first create and process a temporary Options bitmap as described in the MoveToLevel command.

On receipt of this command, a device SHALL move from its current level in an up or down direction as detailed below.

StepMode	Action on Receipt
Up	Increase CurrentLevel by StepSize 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 StepSize 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.

1.6.7.4. Stop Command

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**OptionsMask**	Options-Bitmap	desc		0	M
1	**OptionsOverride**	Options-Bitmap	desc		0	M

1.6.7.4.1. Effect of Receipt

On receipt of this command, a device SHALL first create and process a temporary Options bitmap as described in the MoveToLevel command.

Upon receipt of this command, any MoveToLevel, 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 the Commands list, one for the MoveToLevel, Move and Step 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, because the dependency on On/Off is determined by the original command that is being stopped.

1.6.7.5. MoveToClosestFrequency Command

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**Frequency**	uint16	all		0	M

1.6.7.5.1. Effect of Receipt

Upon receipt of this command, the device SHALL change its current frequency to the requested frequency, or to the closest frequency that it can generate. If the device cannot approximate the frequency, then it SHALL return a default response with an error code of CONSTRAINT_ERROR. Determining if a requested frequency can be approximated by a supported frequency is a manufacturer-

specific decision.

1.6.7.6. 'With On/Off' Commands

The MoveToLevelWithOnOff, MoveWithOnOff and StepWithOnOff commands have identical data fields compared to the MoveToLevel, Move and Step commands respectively. They also have the same effects, except for the following additions.

Before commencing any command that has the effect of setting the CurrentLevel attribute above 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 TRUE (‘On’).

If any command that has the effect of setting the CurrentLevel attribute 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 FALSE (‘Off’).

The StopWithOnOff command has identical data fields compared to the Stop command. Both Stop commands are identical, because the dependency on On/Off is determined by the original command that is being stopped.

1.6.8. State Change Table for Lighting

Below is a table of examples of state changes when Level Control and On/Off clusters are on the same endpoint and the Lighting feature is set.

EiO - ExecuteIfOff field in the Options attribute

OnOff – Attribute value of On/Off cluster: FALSE=‘Off’, TRUE=‘On’

MIN - MinLevel

MAX - MaxLevel

MID – Midpoint between MinLevel and MaxLevel

1.6.8.1. Lighting Device State Change examples

CurrentLevel	**EiO**	**OnOff**	**Physical Device**	**Command Before After**	**CurrentLevel**	**OnOff**	**Physical Device**	**Device Output Result**
any	0	FALSE	Off	MoveToLevel(l=*MID*, t=2 sec)	*same*	FALSE	Off	Stays off
any	0	FALSE	Off	MoveToLevelWithOnOff(l=*MID*, t=2 sec	*MID*	TRUE	On (midpoint brightness)	Turns on and output level adjusts or stays at half

CurrentLevel	**EiO**	**OnOff**	**Physical Device**	**Command Before After**	**CurrentLevel**	**OnOff**	**Physical Device**	**Device Output Result**
any	1	FALSE	Off	MoveToLevel(l=*MID*, t=2 sec)	*MID*	FALSE	Off	Stays off
any	1	FALSE	Off	MoveToLevelWithOnOff(l=*MID*, t=2 sec)	*MID*	TRUE	On	Turns on and output level adjusts to or stays at half
any	1	FALSE	Off	Move(up, rate=64/s)	*MAX*	FALSE	Off	Stays off
any	1	FALSE	Off	MoveWithOnOff(up, rate=64/s)	*MAX*	TRUE	On	Turn on and output level adjusts to or stays at full
any	1	FALSE	Off	MoveWithOnOff(down, rate=64/s)	*MIN*	FALSE	Off	Stays off
any	*any*	TRUE	On (any brightness)	MoveToLevelWithOnOff(l=*MID*, t=2 sec)	*MID*	TRUE	On (midpoint brightness)	Output level adjusts to or stays at half
any	*any*	TRUE	On (any brightness)	MoveWithOnOff(up, rate=64/s)	*MAX*	TRUE	On (full brightness)	Output level adjusts to or stays at full
any	*any*	TRUE	On (any brightness)	Move(down, rate=64/s)	*MIN*	TRUE	On (at minimum brightness)	Output level adjusts to minimum

Cur-rentLevel	EiO	OnOff	Physical Device	Com-mand Before After	Cur-rentLevel	OnOff	Physical Device	Device Output Result
any	*any*	TRUE	On (any bright-ness)	Move-With-OnOff(down, rate=64/s)	*MIN*	FALSE	Off	Output level adjusts to off

1.7. Boolean State Cluster

This cluster provides an interface to a boolean state.

1.7.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Revision	Description
1	Initial release

1.7.2. Classification

Hierarchy	Role	Scope	PICS Code
Base	Application	Endpoint	BOOL

1.7.3. Cluster ID

ID	Name
0x0045	Boolean State

1.7.4. Attributes

ID	Name	Type	Constraint	Quality	Default	Access	Confor-mance
0x0000	**StateValue**	bool		P		R V	M

1.7.4.1. StateValue Attribute

This represents a boolean state.

The semantics of this boolean state are defined by the device type using this cluster.
For example, in a Contact Sensor device type, FALSE=open or no contact, TRUE=closed or contact.

1.7.5. Events

ID	Name	Priority	Access	Conformance
0x00	**StateChange**	INFO	V	O

1.7.5.1. StateChange Event

If this event is supported, it SHALL be generated when the StateValue attribute changes.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**StateValue**	bool				M

1.7.5.1.1. StateValue Field

This field SHALL indicate the new value of the StateValue attribute.

1.8. Boolean State Configuration Cluster

This cluster is used to configure a boolean sensor, including optional state change alarm features and configuration of the sensitivity level associated with the sensor.

1.8.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Revision	Description
1	Initial Release

1.8.2. Classification

Hierarchy	Role	Scope	PICS Code
Base	Application	Endpoint	BOOLCFG

1.8.3. Cluster ID

ID	Name
0x0080	Boolean State Configuration

1.8.4. Features

This cluster SHALL support the FeatureMap bitmap attribute as defined below.

Bit	Code	Feature	Conformance	Summary
0	**VIS**	Visual	O	Supports visual alarms
1	**AUD**	Audible	O	Supports audible alarms
2	**SPRS**	AlarmSuppress	[VIS | AUD]	Supports ability to suppress or acknowledge alarms
3	**SENSLVL**	SensitivityLevel	O	Supports ability to set sensor sensitivity

1.8.4.1. AlarmSuppress Feature

This feature SHALL indicate that the device is able to suppress the supported alarm modes, when the user acknowledges the alarm. This is intended to stop visual and/or audible alarms, when the user has become aware that the sensor is triggered, but it is no longer desired to have the alarm modes active on the device, e.g.:

- The triggering cause have been resolved by the user, but the sensor has not yet stopped detecting the triggering cause.
- The user is not able to address the triggering cause, but is aware of the alarm and suppress/acknowledge it be addressed at a later point.

Acknowledge of alarms will for the remainder of this cluster be referred to as suppress.

A suppressed alarm is still considered active and will remain so unless it is actively disabled or the triggering condition is not longer present. The action of suppressing an alarm mode is only applicable to and is intended to stop the physical alarming, e.g. emitting a sound or blinking a light; it does not impact alarm reporting in AlarmsActive.

1.8.5. Data Types

1.8.5.1. AlarmModeBitmap Type

This data type is derived from map8.

Bit	Name	Summary	Conformance
0	**Visual**	Visual alarming	VIS
1	**Audible**	Audible alarming	AUD

1.8.5.2. SensorFaultBitmap Type

This data type is derived from map16.

Bit	Name	Summary	Conformance
0	**GeneralFault**	Unspecified fault detected	M

1.8.6. Attributes

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0000	**CurrentSensitivityLevel**	uint8	max SupportedSensitivityLevels - 1	N	MS	RW VO	SENSLVL
0x0001	**SupportedSensitivityLevels**	uint8	2 to 10	F	MS	R V	SENSLVL
0x0002	**DefaultSensitivityLevel**	uint8	max SupportedSensitivityLevels - 1	F	MS	R V	[SENSLVL]
0x0003	**AlarmsActive**	AlarmModeBitmap	all		0	R V	VIS | AUD
0x0004	**AlarmsSuppressed**	AlarmModeBitmap	all		0	R V	SPRS
0x0005	**AlarmsEnabled**	AlarmModeBitmap	all	N	MS	R V	[VIS | AUD]
0x0006	**AlarmsSupported**	AlarmModeBitmap	all	F	0	R V	VIS | AUD
0x0007	**SensorFault**	SensorFaultBitmap	all		0	R V	O

1.8.6.1. CurrentSensitivityLevel Attribute

This attribute SHALL indicate the currently selected sensitivity level.

If a write interaction to this attribute contains an unsupported sensitivity value, a CONSTRAINT_ERROR status SHALL be returned.

1.8.6.2. SupportedSensitivityLevels Attribute

This attribute SHALL indicate the number of supported sensitivity levels by the device.

These supported sensitivity levels SHALL be ordered by sensitivity, where a value of 0 SHALL be considered the lowest sensitivity level (least sensitive) and the highest supported value SHALL be considered the highest sensitivity level.
The number of supported sensitivity levels SHOULD represent unique sensitivity levels supported by the device.

1.8.6.3. DefaultSensitivityLevel Attribute

This attribute SHALL indicate the default sensitivity level selected by the manufacturer.

1.8.6.4. AlarmsActive Attribute

This attribute SHALL indicate which specific alarm modes on the server are currently active. When the sensor is no longer triggered, this attribute SHALL be set to the inactive state, by setting the bit to 0, for all supported alarm modes.

If an alarm mode is not supported, the bit indicating this alarm mode SHALL always be 0.

A bit SHALL indicate whether the alarm mode inactive or not:

- 0 = Inactive
- 1 = Active

1.8.6.5. AlarmsSuppressed Attribute

This attribute SHALL indicate which specific alarm modes on the server are currently suppressed. When the sensor is no longer triggered, this attribute SHALL be set to the unsuppressed state, by setting the bit to 0, for all supported alarm modes.

If an alarm mode is not supported, the bit indicating this alarm mode SHALL always be 0.

A bit SHALL indicate whether the alarm mode is suppressed or not:

- 0 = Not suppressed
- 1 = Suppressed

1.8.6.6. AlarmsEnabled Attribute

This attribute SHALL indicate the alarm modes that will be emitted if the sensor is triggered.

If an alarm mode is not supported, the bit indicating this alarm mode SHALL always be 0.

A bit SHALL indicate whether the alarm mode is enabled or disabled:

- 0 = Disabled
- 1 = Enabled

1.8.6.7. AlarmsSupported Attribute

This attribute SHALL indicate the alarms supported by the sensor.

A bit SHALL indicate whether the alarm mode is supported:

- 0 = Not supported
- 1 = Supported

1.8.6.8. SensorFault Attribute

This attribute SHALL indicate any faults registered by the device.

1.8.7. Commands

ID	Name	Direction	Response	Access	Conformance
0x00	**SuppressAlarm**	client ⇒ server	Y	O	SPRS
0x01	**EnableDisableAlarm**	client ⇒ server	Y	O	VIS | AUD

1.8.7.1. SuppressAlarm Command

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**AlarmsToSuppress**	AlarmModeBitmap	all			M

1.8.7.1.1. AlarmsToSuppress Field

This field SHALL indicate the alarm modes to suppress.

1.8.7.1.2. Effect on Receipt

If any of the requested alarm modes are not supported this command SHALL be ignored and the server SHALL return an CONSTRAINT_ERROR status code.

If any of the requested alarm modes are not active, in case the sensor is not triggered or the alarm mode is disabled/not supported, this command SHALL be ignored and the server SHALL return an INVALID_IN_STATE status code.

In the case an alarm is already suppressed and the specific bit is set in the AlarmsSuppressed attribute, the bit SHALL be ignored and the remaining bits SHALL be evaluated.

For the valid bits in the AlarmsToSuppress field, the device SHALL suppress the specified alarm modes as requested in the AlarmsToSuppress field and set the AlarmsSuppressed attribute accordingly.

1.8.7.2. EnableDisableAlarm Command

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**AlarmsToEnableDisable**	AlarmModeBitmap	all			M

1.8.7.2.1. AlarmsToEnableDisable Field

This field SHALL indicate the alarm modes to either enable or disable depending on the bit status, as specified for the AlarmsEnabled attribute.

1.8.7.2.2. Effect on Receipt

If any of the requested alarm modes are not supported this command SHALL be ignored and the server SHALL return an CONSTRAINT_ERROR status code.

If all the bits are valid, the value of the AlarmsEnabled attribute SHALL be set to the value of the AlarmsToEnableDisable field.

If an alarm mode is being enabled and the trigger condition is met, the device SHALL immediately activate the alarm mode and set the associated bit in the AlarmsActive attribute.

If an alarm mode is being disabled, any alarm mode which is either in the active or suppressed state SHALL be cleared and the alarm mode SHALL be considered not active.

1.8.8. Events

ID	Name	Priority	Access	Conformance
0x00	**AlarmsStateChanged**	INFO	V	VIS | AUD
0x01	**SensorFault**	INFO	V	O

1.8.8.1. AlarmsStateChanged Event

This event SHALL be generated after any bits in the AlarmsActive and/or AlarmsSuppressed attributes change. This MAY occur in situations such as when internal processing by the server determines that an alarm mode becomes active or inactive, or when the SuppressAlarm or EnableDisableAlarm commands are processed in a way that some alarm modes becomes suppressed, active or inactive.

If several alarm modes change state at the same time, a single event combining multiple changes MAY be emitted instead of multiple events each representing a single change.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**AlarmsActive**	AlarmModeBitmap	all			M
1	**AlarmsSuppressed**	AlarmModeBitmap	all			SPRS

1.8.8.1.1. AlarmsActive Field

This field SHALL indicate the state of active alarm modes, as indicated by the AlarmsActive attribute, at the time the event was generated.

1.8.8.1.2. AlarmsSuppressed Field

This field SHALL indicate the state of suppressed alarm modes, as indicated by the AlarmsSuppressed attribute, at the time the event was generated.

1.8.8.2. SensorFault Event

This event SHALL be generated when the device registers or clears a fault.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**SensorFault**	SensorFaultBitmap	all			M

1.8.8.2.1. SensorFault Field

This field SHALL indicate the value of the SensorFault attribute, at the time this event is generated.

1.9. Mode Select Cluster

This cluster provides an interface for controlling a characteristic of a device that can be set to one of several predefined values. For example, the light pattern of a disco ball, the mode of a massage chair, or the wash cycle of a laundry machine.

The server allows the client to set a mode on the server. A mode is one of a list of options that may be presented by a client for a user choice, or understood by the client, via the semantic tags on the mode.

A semantic tag is either a standard tag within a standard category namespace, or a manufacturer specific tag, within the namespace of the vendor ID of the manufacturer. If there is no semantic tag, the mode is anonymous, and the selection is made by the user solely based on the Label string.

Each cluster ID that indicates this specification SHALL define a distinct purpose for the cluster instance. For example: A LightBlinking cluster ID supports blinking modes for a light (and is described that way).

An anonymous mode SHALL support the derived cluster purpose. A manufacturer specific semantic tag SHALL support the derived cluster purpose. An anonymous mode SHALL NOT replace the meaning of a standard semantic tag, when one exists, for the cluster purpose.

1.9.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Revision	Description
1	Initial version
2	The MfgCode field was marked non-nullable. Updated the related text. Reorder sections.

1.9.2. Classification

Hierarchy	Role	Scope	PICS Code
Base	Application	Endpoint	MOD

1.9.3. Cluster ID

ID	Name
0x0050	Mode Select

1.9.4. Features

This cluster SHALL support the FeatureMap global attribute:

ID	Code	Feature	Summary
0	**DEPONOFF**	OnOff	Dependency with the OnOff cluster

1.9.4.1. OnOff Feature

This feature creates a dependency between an OnOff cluster instance and this cluster instance on the same endpoint. See OnMode for more information.

1.9.5. Data Types

1.9.5.1. SemanticTagStruct Type

A Semantic Tag is meant to be interpreted by the client for the purpose the cluster serves.

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0	**MfgCode**	vendor-id	desc	F			M
1	**Value**	enum16	all	F			M

1.9.5.1.1. Value Field

This field SHALL indicate the semantic tag within a semantic tag namespace which is either manufacturer specific or standard. For semantic tags in a standard namespace, see Standard Namespace.

1.9.5.1.2. MfgCode Field

This field SHALL indicate a manufacturer code (Vendor ID), and the Value field SHALL indicate a semantic tag defined by the manufacturer. Each manufacturer code supports a single namespace of values. The same manufacturer code and semantic tag value in separate cluster instances are part of the same namespace and have the same meaning. For example: a manufacturer tag meaning "pinch", has the same meaning in a cluster whose purpose is to choose the amount of sugar, or amount of salt.

1.9.5.2. ModeOptionStruct Type

This is a struct representing a possible mode of the server.

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0	**Label**	string	max 64	F	MS		M
1	**Mode**	uint8		F	MS		M
2	**SemanticTags**	list[SemanticTagStruct]	max 64	F	MS		M

1.9.5.2.1. Label Field

This field is readable text that describes the mode option that can be used by a client to indicate to the user what this option means. This field is meant to be readable and understandable by the user.

1.9.5.2.2. Mode Field

The Mode field is used to identify the mode option. The value SHALL be unique for every item in the SupportedModes attribute.

1.9.5.2.3. SemanticTags Field

This field is a list of semantic tags that map to the mode option. This MAY be used by clients to determine the meaning of the mode option as defined in a standard or manufacturer specific namespace. Semantic tags can help clients look for options that meet certain criteria. A semantic tag SHALL be either a standard tag or manufacturer specific tag as defined in each SemanticTagStruct list entry.

A mode option MAY have more than one semantic tag. A mode option MAY be mapped to a mixture of standard and manufacturer specific semantic tags.

All standard semantic tags are from a single namespace indicated by the StandardNamespace attribute.

For example: A mode labeled "100%" can have both the `HIGH` (MS) and `MAX` (standard) semantic tag. Clients seeking the option for either `HIGH` or `MAX` will find the same option in this case.

1.9.6. Attributes

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0000	**Description**	**string**	max 64	F	MS	R V	M
0x0001	**StandardNamespace**	**enum16**	desc	FX	null	R V	M
0x0002	**SupportedModes**	**list[ModeOptionStruct]**	max 255	F	MS	R V	M
0x0003	**CurrentMode**	**uint8**	desc	SN	MS	R V	M
0x0004	**StartUpMode**	**uint8**	desc	NX	MS	RW VO	O
0x0005	**OnMode**	**uint8**	desc	NX	null	RW VO	DEPONOFF

1.9.6.1. Scene Table Extensions

If the Scenes Management server cluster is implemented on the same endpoint, the following attribute SHALL be part of the ExtensionFieldSetStruct of the Scene Table.

- CurrentMode

1.9.6.2. Description Attribute

This attribute describes the purpose of the server, in readable text.

For example, a coffee machine may have a Mode Select cluster for the amount of milk to add, and another Mode Select cluster for the amount of sugar to add. In this case, the first instance can have the description `Milk` and the second instance can have the description `Sugar`. This allows the user to tell the purpose of each of the instances.

1.9.6.3. StandardNamespace Attribute

This attribute, when not null, SHALL indicate a single standard namespace for any standard semantic tag value supported in this or any other cluster instance with the same value of this attribute. A

null value indicates no standard namespace, and therefore, no standard semantic tags are provided in this cluster instance. Each standard namespace and corresponding values and value meanings SHALL be defined in another document.

1.9.6.4. SupportedModes Attribute

This attribute is the list of supported modes that may be selected for the CurrentMode attribute. Each item in this list represents a unique mode as indicated by the Mode field of the ModeOptionStruct. Each entry in this list SHALL have a unique value for the Mode field.

1.9.6.5. CurrentMode Attribute

This attribute represents the current mode of the server.

The value of this field must match the Mode field of one of the entries in the `SupportedModes` attribute.

1.9.6.6. StartUpMode Attribute

The StartUpMode attribute value indicates the desired startup mode for the server when it is supplied with power.

If this attribute is not null, the CurrentMode attribute SHALL be set to the StartUpMode value, when the server is powered up, except in the case when the OnMode attribute overrides the StartUpMode attribute (see OnModeWithPowerUp).

This behavior does not apply to reboots associated with OTA. After an OTA restart, the CurrentMode attribute SHALL return to its value prior to the restart.

The value of this field SHALL match the Mode field of one of the entries in the `SupportedModes` attribute.

If this attribute is not implemented, or is set to the null value, it SHALL have no effect.

1.9.6.7. OnMode Attribute

This attribute SHALL indicate the value of CurrentMode that depends on the state of the On/Off cluster on the same endpoint. If this attribute is not present or is set to null, it SHALL NOT have an effect, otherwise the CurrentMode attribute SHALL depend on the OnOff attribute of the On/Off cluster as described in the table below:

OnOff Change	CurrentMode Change
ON → OFF	No change
OFF → ON	Change CurrentMode to OnMode
OFF → OFF	No change
ON → ON	No change

The value of this field SHALL match the Mode field of one of the entries in the SupportedModes attribute.

1.9.6.7.1. OnMode with Power Up

If the On/Off feature is supported and the On/Off cluster attribute StartUpOnOff is present, with a value of On (turn on at power up), then the CurrentMode attribute SHALL be set to the OnMode attribute value when the server is supplied with power, except if the OnMode attribute is null.

1.9.7. Commands

ID	Name	Direction	Response	Access	Conformance
0x00	**ChangeToMode**	client ⇒ server	Y	O	M

1.9.7.1. ChangeToMode Command

On receipt of this command, if the `NewMode` field indicates a valid mode transition within the supported list, the server SHALL set the CurrentMode attribute to the `NewMode` value, otherwise, the server SHALL respond with an `INVALID_COMMAND` status response.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**NewMode**	uint8	desc			M

1.10. Mode Base Cluster

This cluster provides an interface for controlling a characteristic of a device that can be set to one of several predefined values. For example, the light pattern of a disco ball, the mode of a massage chair, or the wash cycle of a laundry machine.

The server allows the client to set a mode on the server. A mode is one of a list of options that may be presented by a client for a user choice, or understood by the client, via the mode's tags.

A mode tag is either a standard tag within a standard category namespace, or a manufacturer specific tag, within the namespace of the vendor ID of the manufacturer.

Any derived cluster specification based on this cluster SHALL support the standard mode tag value definitions and command status definitions defined in this cluster and MAY define additional standard mode tag values and standard command status values that are supported in the respective derived cluster instances.

Each cluster ID that indicates this specification SHALL define a distinct purpose for the cluster instance. For example: A LightBlinking cluster ID supports blinking modes for a light (and is described that way).

An anonymous mode SHALL NOT replace the meaning of a standard mode tag, when one exists, for the cluster purpose.

1.10.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Revision	Description
1	Initial version
2	ChangeToModeResponse command: StatusText must be provided for InvalidInMode status. Require at least one standard mode tag. Define reserved ranges for base/derived clusters

1.10.2. Classification

Hierarchy	Role	Scope	PICS Code
Base	Application	Endpoint	MODB

1.10.3. Cluster ID

ID	Name
n/a	Mode Base

1.10.4. Features

This cluster SHALL support the FeatureMap bitmap attribute as defined below.

Bit	Code	Feature	Summary
0	**DEPONOFF**	OnOff	Dependency with the OnOff cluster

1.10.4.1. OnOff Feature

This feature creates a dependency between an OnOff cluster instance and this cluster instance on the same endpoint. See OnMode for more information.

1.10.5. Data Types

1.10.5.1. ModeTagStruct Type

A Mode Tag is meant to be interpreted by the client for the purpose the cluster serves.

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0	**MfgCode**	vendor-id	desc				O
1	**Value**	enum16	all				M

1.10.5.1.1. MfgCode Field

If the MfgCode field exists, the Value field SHALL be in the manufacturer-specific value range (see Section 1.10.8, “Mode Namespace”).

This field SHALL indicate the manufacturer’s VendorID and it SHALL determine the meaning of the Value field.

The same manufacturer code and mode tag value in separate cluster instances are part of the same namespace and have the same meaning. For example: a manufacturer tag meaning "pinch" can be used both in a cluster whose purpose is to choose the amount of sugar, or in a cluster whose purpose is to choose the amount of salt.

1.10.5.1.2. Value Field

This field SHALL indicate the mode tag within a mode tag namespace which is either manufacturer specific or standard.

1.10.5.2. ModeOptionStruct Type

This is a struct representing a possible mode of the server.

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0	**Label**	string	max 64	F	MS		M
1	**Mode**	uint8		F	MS		M
2	**ModeTags**	list[ModeTagStruct]	max 8	F	MS		M

1.10.5.2.1. Label Field

This field SHALL indicate readable text that describes the mode option, so that a client can provide it to the user to indicate what this option means. This field is meant to be readable and understandable by the user.

1.10.5.2.2. Mode Field

This field is used to identify the mode option.

1.10.5.2.3. ModeTags Field

This field SHALL contain a list of tags that are associated with the mode option. This MAY be used by clients to determine the full or the partial semantics of a certain mode, depending on which tags they understand, using standard definitions and/or manufacturer specific namespace definitions.

The standard mode tags are defined in this cluster specification. For the derived cluster instances, if the specification of the derived cluster defines a namespace, the set of standard mode tags also includes the mode tag values from that namespace.

Mode tags can help clients look for options that meet certain criteria, render the user interface, use

the mode in an automation, or to craft help text their voice-driven interfaces. A mode tag SHALL be either a standard tag or a manufacturer specific tag, as defined in each ModeTagStruct list entry.

A mode option MAY have more than one mode tag. A mode option MAY be associated with a mixture of standard and manufacturer specific mode tags. A mode option SHALL be associated with at least one standard mode tag.

A few examples are provided below.

- A mode named "100%" can have both the High (manufacturer specific) and Max (standard) mode tag. Clients seeking the mode for either High or Max will find the same mode in this case.
- A mode that includes a LowEnergy tag can be displayed by the client using a widget icon that shows a green leaf.
- A mode that includes a LowNoise tag may be used by the client when the user wishes for a lower level of audible sound, less likely to disturb the household's activities.
- A mode that includes a LowEnergy tag (standard, defined in this cluster specification) and also a Delicate tag (standard, defined in the namespace of a Laundry Mode derived cluster).
- A mode that includes both a generic Quick tag (defined here), and Vacuum and Mop tags, (defined in the RVC Clean cluster that is a derivation of this cluster).

1.10.6. Attributes

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0000	**SupportedModes**	list[ModeOptionStruct]	2 to 255	F	MS	R V	M
0x0001	**CurrentMode**	uint8	desc	SN	MS	R V	M
0x0002	**StartUpMode**	uint8	desc	NX	MS	RW VO	O
0x0003	**OnMode**	uint8	desc	NX	null	RW VO	DEPONOFF

1.10.6.1. Scene Table Extensions

If the Scenes Management server cluster is implemented on the same endpoint, the following attribute SHALL be part of the ExtensionFieldSetStruct of the Scene Table.

- CurrentMode

1.10.6.2. SupportedModes Attribute

This attribute SHALL contain the list of supported modes that may be selected for the CurrentMode attribute. Each item in this list represents a unique mode as indicated by the Mode field of the ModeOptionStruct.

Each entry in this list SHALL have a unique value for the Mode field.

Each entry in this list SHALL have a unique value for the Label field.

1.10.6.3. CurrentMode Attribute

This attribute SHALL indicate the current mode of the server.

The value of this field SHALL match the Mode field of one of the entries in the SupportedModes attribute.

The value of this attribute may change at any time via an out-of-band interaction outside of the server, such as interactions with a user interface, via internal mode changes due to autonomously progressing through a sequence of operations, on system time-outs or idle delays, or via interactions coming from a fabric other than the one which last executed a ChangeToMode.

1.10.6.4. StartUpMode Attribute

This attribute SHALL indicate the desired startup mode for the server when it is supplied with power.

If this attribute is not null, the CurrentMode attribute SHALL be set to the StartUpMode value, when the server is powered up, except in the case when the OnMode attribute overrides the StartUpMode attribute (see OnModeWithPowerUp).

This behavior does not apply to reboots associated with OTA. After an OTA restart, the CurrentMode attribute SHALL return to its value prior to the restart.

The value of this field SHALL match the Mode field of one of the entries in the SupportedModes attribute.

If this attribute is not implemented, or is set to the null value, it SHALL have no effect.

1.10.6.5. OnMode Attribute

This attribute SHALL indicate whether the value of CurrentMode depends on the state of the On/Off cluster on the same endpoint. If this attribute is not present or is set to null, there is no dependency, otherwise the CurrentMode attribute SHALL depend on the OnOff attribute in the On/Off cluster as described in the table below:

OnOff Change	CurrentMode Change
ON → OFF	No change
OFF → ON	Change CurrentMode to OnMode
OFF → OFF	No change
ON → ON	No change

The value of this field SHALL match the Mode field of one of the entries in the SupportedModes attribute.

1.10.6.5.1. OnMode with Power Up

If the On/Off feature is supported and the On/Off cluster attribute StartUpOnOff is present, with a value of On (turn on at power up), then the CurrentMode attribute SHALL be set to the OnMode attribute value when the server is supplied with power, except if the OnMode attribute is null.

1.10.7. Commands

ID	Name	Direction	Response	Access	Conformance
0x00	**ChangeToMode**	client ⇒ server	ChangeToModeResponse	O	M
0x01	**ChangeToModeResponse**	client ⇐ server	N	O	M

1.10.7.1. ChangeToMode Command

This command is used to change device modes.

On receipt of this command the device SHALL respond with a ChangeToModeResponse command.

This command SHALL have the following data fields:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**NewMode**	uint8	desc			M

1.10.7.1.1. NewMode Field

If the NewMode field doesn't match the Mode field of any entry of the SupportedModes list, the ChangeToModeResponse command's Status field SHALL indicate UnsupportedMode and the StatusText field SHALL be included and MAY be used to indicate the issue, with a human readable string, or include an empty string.

If the NewMode field matches the Mode field of one entry of the SupportedModes list, but the device is not able to transition as requested, the ChangeToModeResponse command SHALL:

- Have the Status set to a product-specific Status value representing the error, or GenericFailure if a more specific error cannot be provided. See Status Field for details.
- Provide a human readable string in the StatusText field.

If the NewMode field matches the Mode field of one entry of the SupportedModes list and the device is able to transition as requested, the server SHALL transition into the mode associated with NewMode, the ChangeToModeResponse command SHALL have the Status field set to Success, the StatusText field MAY be supplied with a human readable string or include an empty string and the CurrentMode field SHALL be set to the value of the NewMode field.

If the NewMode field is the same as the value of the CurrentMode attribute the ChangeToModeRe-

sponse command SHALL have the Status field set to Success and the StatusText field MAY be supplied with a human readable string or include an empty string.

1.10.7.2. ChangeToModeResponse Command

This command is sent by the device on receipt of the ChangeToMode command. This command SHALL have the following data fields:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**Status**	enum8	desc			M
1	**StatusText**	string	max 64			[Status == Success], M

1.10.7.2.1. Status Field

1.10.7.2.1.1. Mode Base Status Code Ranges

The following table defines the enumeration ranges for the ChangeToModeResponse command's Status field values.

Status Code Range	Range Name	Summary
0x00 to 0x3F	**CommonCodes**	Common standard values defined in the generic Mode Base cluster specification.
0x40 to 0x7F	**DerivedClusterCodes**	Derived cluster specific standard values defined in the derived Mode Base cluster specification.
0x80 to 0xBF	**MfgCodes**	Manufacturer specific values. For the derived Mode Base cluster instances, these are manufacturer specific under the derived cluster.

The set of Status field values defined in each of the generic or derived Mode Base cluster specifications is called StatusCode.

1.10.7.2.1.2. Mode Base Status CommonCodes Range

The following table defines the common StatusCode values.

Status Code	Name	Summary
0x00	**Success**	Switching to the mode indicated by the NewMode field is allowed and possible. The CurrentMode attribute is set to the value of the NewMode field.
0x01	**UnsupportedMode**	The value of the NewMode field doesn't match any entries in the SupportedMode attribute.
0x02	**GenericFailure**	Generic failure code, indicating that switching to the mode indicated by the NewMode field is not allowed or not possible.
0x03	**InvalidInMode**	The received request cannot be handled due to the current mode of the device

The derived cluster code definitions SHALL NOT duplicate the common code definitions. For example, a derived cluster specification SHALL NOT define a status code with the same semantic as the common code of 0x01 (UnsupportedMode).

If the Status field is set to Success, the StatusText field is optional.

If the Status field is set to UnsupportedMode, the StatusText field SHALL be an empty string.

If the Status field is not set to Success or UnsupportedMode, the StatusText field SHALL include a vendor-defined error description which can be used to explain the error to the user. For example, if the Status field is set to InvalidInMode, the StatusText field SHOULD indicate why the transition to the requested mode is not allowed, given the current mode of the device, which may involve other clusters.

1.10.8. Mode Namespace

This section provides the definitions of the mode tag ranges and of the common standard mode tag values available in the instances of this cluster.

The following table defines the enumeration ranges for the ModeTagStruct Value field values.

Mode Tag Value Range	Range Name	Summary
0x0000 to 0x3FFF	**CommonTags**	Common standard values defined in this cluster specification.
0x4000 to 0x7FFF	**DerivedClusterTags**	Derived cluster specific standard values defined in the derived cluster specification.

Mode Tag Value Range	Range Name	Summary
0x8000 to 0xBFFF	**MfgTags**	Manufacturer-specific values. For the derived cluster instances, these are manufacturer specific under the derived cluster.

The derived cluster specific standard value definitions SHALL not duplicate the common standard value definitions. For example, a derived cluster specification can't define a mode tag value with the same mode as the common standard tag value of 0x0001 (Quick).

The set of ModeTagStruct Value field values defined in each of the generic or derived Mode Base cluster specifications is called ModeTag.

The following table defines the common ModeTag values.

Mode Tag Value	Name	Summary
0x0000	**Auto**	The device decides which options, features and setting values to use.
0x0001	**Quick**	The mode of the device is optimizing for faster completion.
0x0002	**Quiet**	The device is silent or barely audible while in this mode.
0x0003	**LowNoise**	Either the mode is inherently low noise or the device optimizes for that.
0x0004	**LowEnergy**	The device is optimizing for lower energy usage in this mode. Sometimes called "Eco mode".
0x0005	**Vacation**	A mode suitable for use during vacations or other extended absences.
0x0006	**Min**	The mode uses the lowest available setting value.
0x0007	**Max**	The mode uses the highest available setting value.
0x0008	**Night**	The mode is recommended or suitable for use during night time.
0x0009	**Day**	The mode is recommended or suitable for use during day time.

1.11. Low Power Cluster

This cluster provides an interface for managing low power mode on a device.

This cluster would be supported on an endpoint that represents a physical device with a low power mode. This cluster provides a sleep() command to allow clients to manually put the device into low power mode. There is no command here to wake up a sleeping device because that operation often involves other protocols such as Wake On LAN. Most devices automatically enter low power mode based upon inactivity.

The cluster server for Low Power is implemented by a device that supports a low power mode, such as a TV, Set-top box, or Smart Speaker.

NOTE	We have considered a “DisableLowPowerMode” command but have not added it due to suspected issues with energy consumption regulations. This can be added in the future.

1.11.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Revision	Description
1	Initial Release

1.11.2. Classification

Hierarchy	Role	Scope	PICS Code
Base	Application	Endpoint	LOWPOWER

1.11.3. Cluster ID

ID	Name
0x0508	Low Power

1.11.4. Commands

ID	Name	Direction	Response	Access	Conformance
0x00	**Sleep**	client ⇒ server	Y	O	M

1.11.4.1. Sleep Command

This command SHALL put the device into low power mode.

1.12. Wake On LAN Cluster

This cluster provides an interface for managing low power mode on a device that supports the Wake On LAN or Wake On Wireless LAN (WLAN) protocol (see [Wake On LAN]).

This cluster would be supported on IP devices that have a low power mode AND support the ability to be woken up using the Wake on LAN or Wake on WLAN protocol. This cluster provides the device MAC address which is a required input to the Wake on LAN protocol. Besides the MAC address, this cluster provides an optional link-local IPv6 address which is useful to support "Wake on Direct Packet" used by some Ethernet and Wi-Fi devices.

Acting on the MAC address or link-local IPv6 address information does require the caller to be in the same broadcast domain as the destination. To wake the destination up, the caller sends a multicast-based magic UDP packet that contains destination's MAC address in the UDP payload to FF02::1, the IPv6 all-nodes link-local multicast group address. If the optional link-local address is provided by the destination through this cluster, the caller also sends the magic UDP packet in unicast to that link-local address. This unicast-based method is particularly useful for Wi-Fi devices, since due to lack of MAC layer retransmission mechanism, multicast over Wi-Fi is not as reliable as unicast. If a device provides the link-local address in this cluster, its Ethernet controller or Wi-Fi radio SHALL respond to the IPv6 neighbor solicitation message for the link-local address without the need to wake host CPU up. In order to receive the magic or neighbor solicitation packets in multicast, the Wi-Fi devices must support Group Temporal Key (GTK) rekey operation in low power mode.

Most devices automatically enter low power mode based upon inactivity.

The cluster server for Wake on LAN or Wake on WLAN is implemented by a device that supports the Wake on LAN/WLAN protocol, such as a TV, Set-top Box, or Smart Speaker.

1.12.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Revision	Description
1	Initial Release

1.12.2. Classification

Hierarchy	Role	Scope	PICS Code
Base	Application	Endpoint	WAKEONLAN

1.12.3. Cluster ID

ID	Name
0x0503	Wake on LAN

1.12.4. Attributes

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0000	**MACAddress**	string	max 12	F		R V	O
0x0001	**LinkLocalAddress**	ipv6adr	desc	F		R V	O

1.12.4.1. MACAddress Attribute

This attribute SHALL indicate the current MAC address of the device. Only 48-bit MAC Addresses SHALL be used for this attribute as required by the Wake on LAN protocol.

Format of this attribute SHALL be an upper-case hex-encoded string representing the hex address, like `12345678ABCD`.

1.12.4.2. LinkLocalAddress Attribute

This attribute SHALL indicate the current link-local address of the device. Only 128-bit IPv6 link-local addresses SHALL be used for this attribute.

NOTE	Some companies may consider MAC Address to be protected data subject to PII handling considerations and will therefore choose not to include it or read it. The MAC Address can often be determined using ARP in IPv4 or NDP in IPv6.

1.13. Switch Cluster

This cluster exposes interactions with a switch device, for the purpose of using those interactions by other devices.
Two types of switch devices are supported: latching switch (e.g. rocker switch) and momentary switch (e.g. push button), distinguished with their feature flags.
Interactions with the switch device are exposed as attributes (for the latching switch) and as events (for both types of switches).
An interested client MAY subscribe to these attributes/events and thus be informed of the interactions, and can perform actions based on this, for example by sending commands to perform an action such as controlling a light or a window shade.

1.13.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Revision	Description
1	Initial Release

1.13.2. Classification

Hierarchy	Role	Scope	PICS Code
Base	Application	Endpoint	SWTCH

1.13.3. Cluster ID

ID	Name
0x003B	Switch

1.13.4. Features

This cluster SHALL support the FeatureMap bitmap attribute as defined below.

These feature flags SHALL be used to indicate the physics as well as the detection/reporting capabilities of the device represented by the cluster.

Bit	Code	Feature	Conformance	Summary
0	**LS**	LatchingSwitch	O.a	Switch is latching
1	**MS**	MomentarySwitch	O.a	Switch is momentary
2	**MSR**	MomentarySwitchRelease	[MS]	Switch supports release
3	**MSL**	MomentarySwitchLongPress	[MS & MSR]	Switch supports long press
4	**MSM**	MomentarySwitchMultiPress	[MS & MSR]	Switch supports multi-press

1.13.4.1. LatchingSwitch Feature

This feature is for a switch that maintains its position after being pressed (or turned).

1.13.4.2. MomentarySwitch Feature

This feature is for a switch that does not maintain its position after being pressed (or turned). After releasing, it goes back to its idle position.

1.13.4.3. MomentarySwitchRelease Feature

This feature is for a momentary switch that can distinguish and report release events. When this feature flag MSR is present, MS SHALL be present as well.

1.13.4.4. MomentarySwitchLongPress Feature

This feature is for a momentary switch that can distinguish and report long presses from short presses. When this feature flag MSL is present, MS and MSR SHALL be present as well.

1.13.4.5. MomentarySwitchMultiPress Feature

This feature is for a momentary switch that can distinguish and report double press and potentially multiple presses with more events, such as triple press, etc. When this feature flag MSM is present, MS and MSR SHALL be present as well.

1.13.5. Attributes

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0000	**NumberOfPositions**	uint8	min 2	F	2	R V	M
0x0001	**CurrentPosition**	uint8	max NumberOfPositions-1	N	0	R V	M
0x0002	**MultiPressMax**	uint8	min 2	F	2	R V	MSM

1.13.5.1. NumberOfPositions Attribute

This attribute SHALL indicate the maximum number of positions the switch has. Any kind of switch has a minimum of 2 positions. Also see Multi Position Details for the case NumberOfPositions>2.

1.13.5.2. CurrentPosition Attribute

This attribute SHALL indicate the position of the switch. The valid range is zero to NumberOfPositions-1. CurrentPosition value 0 SHALL be assigned to the default position of the switch: for example the "open" state of a rocker switch, or the "idle" state of a push button switch.

1.13.5.3. MultiPressMax Attribute

This attribute SHALL indicate how many consecutive presses can be detected and reported by a momentary switch which supports multi-press (e.g. it will report the value 3 if it can detect single press, double press and triple press, but not quad press and beyond).

1.13.6. Events

ID	Name	Priority	Access	Conformance
0x00	**SwitchLatched**	INFO	V	LS
0x01	**InitialPress**	INFO	V	MS
0x02	**LongPress**	INFO	V	MSL

ID	Name	Priority	Access	Conformance
0x03	**ShortRelease**	INFO	V	MSR
0x04	**LongRelease**	INFO	V	MSL
0x05	**MultiPressOngoing**	INFO	V	MSM
0x06	**MultiPressComplete**	INFO	V	MSM

1.13.6.1. SwitchLatched Event

This event SHALL be generated, when the latching switch is moved to a new position.
It MAY have been delayed by debouncing within the switch.

The data of this event SHALL contain the following information:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**NewPosition**	uint8	0 to NumberOfPositions-1			M

1.13.6.1.1. NewPosition Field

This field SHALL indicate the new value of the CurrentPosition attribute, i.e. after the move.

1.13.6.2. InitialPress Event

This event SHALL be generated, when the momentary switch starts to be pressed (after debouncing).

The data of this event SHALL contain the following information:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**NewPosition**	uint8	0 to NumberOfPositions-1			M

1.13.6.2.1. NewPosition Field

This field SHALL indicate the new value of the CurrentPosition attribute, i.e. while pressed.

1.13.6.3. LongPress Event

This event SHALL be generated, when the momentary switch has been pressed for a "long" time (this time interval is manufacturer determined (e.g. since it depends on the switch physics)).

The data of this event SHALL contain the following information:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**NewPosition**	uint8	0 to NumberOfPositions-1			M

1.13.6.3.1. NewPosition Field

This field SHALL indicate the new value of the CurrentPosition attribute, i.e. while pressed.

1.13.6.4. ShortRelease Event

This event SHALL be generated, when the momentary switch has been released (after debouncing).

- If the server supports the Momentary Switch LongPress (MSL) feature, this event SHALL be generated when the switch is released if ***no*** LongPress event had been generated since the previous InitialPress event.
- If the server does not support the Momentary Switch LongPress (MSL) feature, this event SHALL be generated when the switch is released - even when the switch was pressed for a long time.
- Also see Section 1.13.7, “Sequence of generated events”.

The data of this event SHALL contain the following information:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**PreviousPosition**	uint8	0 to NumberOfPositions-1			M

1.13.6.4.1. PreviousPosition Field

This field SHALL indicate the previous value of the CurrentPosition attribute, i.e. just prior to release.

1.13.6.5. LongRelease Event

This event SHALL be generated, when the momentary switch has been released (after debouncing) and after having been pressed for a long time, i.e. this event SHALL be generated when the switch is released if a LongPress event has been generated since the previous InitialPress event. Also see Section 1.13.7, “Sequence of generated events”.

The data of this event SHALL contain the following information:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**PreviousPosition**	uint8	0 to NumberOfPositions-1			M

1.13.6.5.1. PreviousPosition Field

This field SHALL indicate the previous value of the CurrentPosition attribute, i.e. just prior to release.

1.13.6.6. MultiPressOngoing Event

This event SHALL be generated to indicate how many times the momentary switch has been pressed in a multi-press sequence, *during* that sequence. See Multi Press Details below.

The data of this event SHALL contain the following information:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**NewPosition**	uint8	0 to NumberOfPositions-1			M
1	**CurrentNumberOfPressesCounted**	uint8	2 to MultiPressMax			M

1.13.6.6.1. NewPosition Field

This field SHALL indicate the new value of the CurrentPosition attribute, i.e. while pressed.

1.13.6.6.2. CurrentNumberOfPressesCounted Field

This field SHALL contain:

- a value of 2 when the second press of a multi-press sequence has been detected,
- a value of 3 when the third press of a multi-press sequence has been detected,
- a value of N when the N^{th} press of a multi-press sequence has been detected.

1.13.6.7. MultiPressComplete Event

This event SHALL be generated to indicate how many times the momentary switch has been pressed in a multi-press sequence, after it has been detected that the sequence has *ended*. See Multi Press Details.

The data of this event SHALL contain the following information:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**PreviousPosition**	uint8	0 to NumberOfPositions-1			M
1	**TotalNumberOfPressesCounted**	uint8	1 to MultiPressMax			M

The PreviousPosition field SHALL indicate the previous value of the CurrentPosition attribute, i.e. just prior to release.

The TotalNumberOfPressesCounted field SHALL contain:

- a value of 1 when there was one press in a multi-press sequence (and the sequence has ended), i.e. there was no double press (or more),
- a value of 2 when there were exactly two presses in a multi-press sequence (and the sequence has ended),
- a value of 3 when there were exactly three presses in a multi-press sequence (and the sequence has ended),
- a value of N when there were exactly N presses in a multi-press sequence (and the sequence has ended).

1.13.7. Sequence of generated events

This section describes the sequence of events that will be generated by three types of momentary switches (distinguished by their feature flags). For each switch type, we will define the sequence of generated events for these three interactions:

1. Sequence for a switch which is pressed briefly.
2. Sequence for a switch pressed for a long time.
3. Sequence for a switch pressed for a very long time.

In the three interactions described in the subsections below, if the NumberOfPositions attribute is equal to 2, the InitialPress and LongPress events have the NewPosition field set to 1 and the ShortRelease and LongRelease events have the PreviousPosition field set to 1. For larger values of the NumberOfPositions attribute, see Multi Position Details.

1.13.7.1. Supports InitialPress + LongPress + ShortRelease + LongRelease

This switch (with feature flags MS & MSR & MSL) SHALL generate either a sequence of two or three (depending on how long the switch is pressed) events for one interaction cycle, in the order given below and illustrated in the figure below.

- short press: InitialPress, then ShortRelease
- long press (or very long press): InitialPress, then LongPress, and finally LongRelease. Please

note that the LongPress event SHALL be generated exactly once for this case and SHALL not be repeated, irrespective how long the switch is pressed.

The image shows a time representation of the state of the switch (low=not pressed, high=pressed) with the colored dots indicating the various events generated at that moment in time.

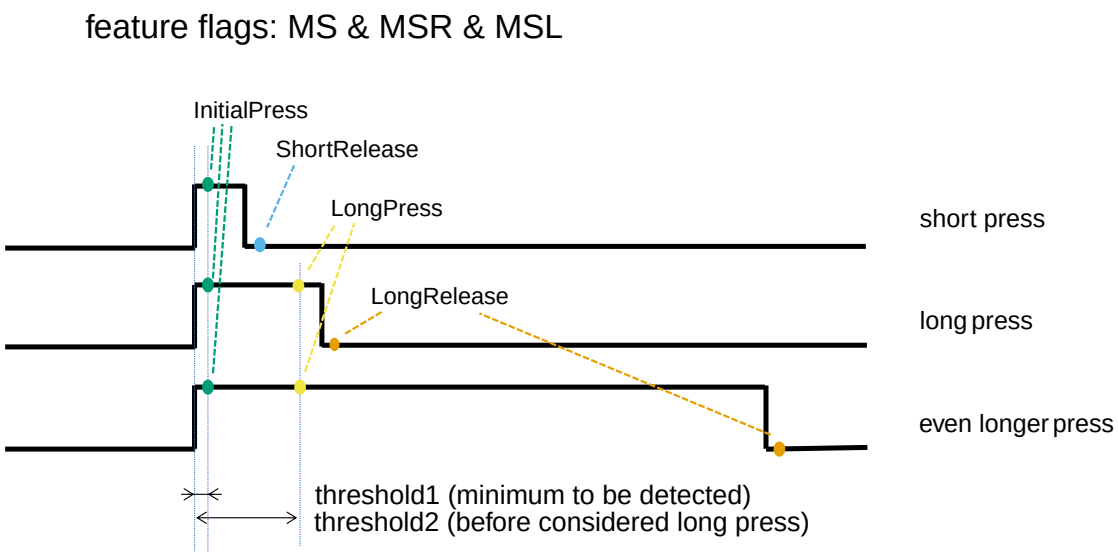


*Figure 3. Switch device delivering 'InitialPress', 'LongPress' and both '*Release' events*

1.13.7.2. Supports InitialPress + ShortRelease (but not LongPress, LongRelease)

This switch (with feature flags MS & MSR & !MSL) does not generate events for the "longpress" case and therefore it SHALL generate a sequence of two events for one interaction cycle, irrespective of how long the switch is pressed, in the order given and illustrated below.

- any press length: InitialPress, then ShortRelease

Please note that even after a "long" period being pressed, the release event is ShortRelease. A device with this set of feature flags SHALL NOT generate the LongPress and LongRelease events.

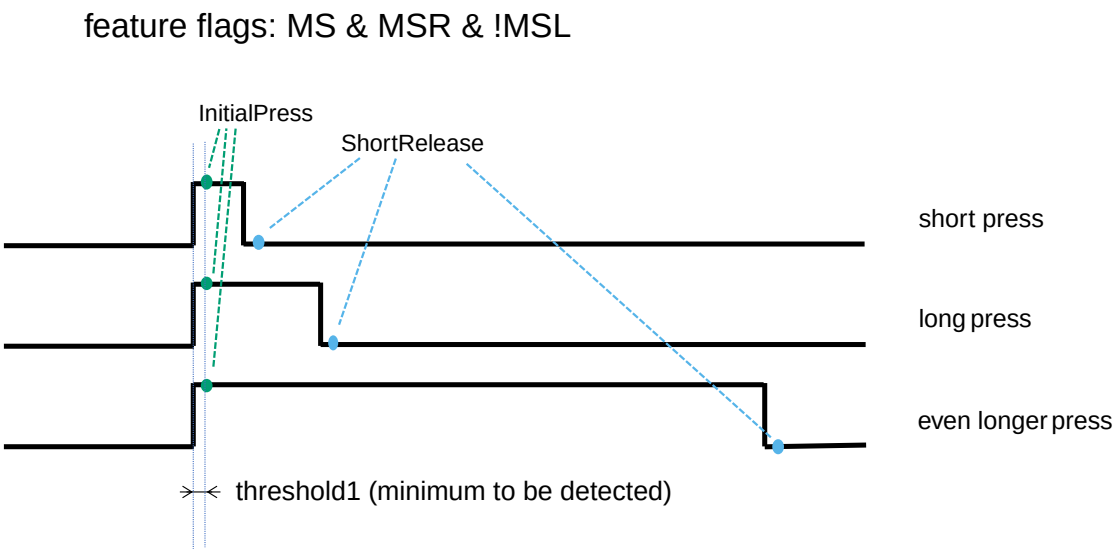


Figure 4. Switch device delivering only 'InitialPress' and 'ShortRelease' events

1.13.7.3. Supports InitialPress (but not LongPress, ShortRelease and LongRelease)

This switch (with feature flags MS & !MSR & !MSL) SHALL generate a single InitialPress event for one interaction cycle, irrespective of how long the switch is pressed, as illustrated in the figure below.

A device with this set of feature flags SHALL NOT generate any of the ShortRelease, LongPress and LongRelease events.

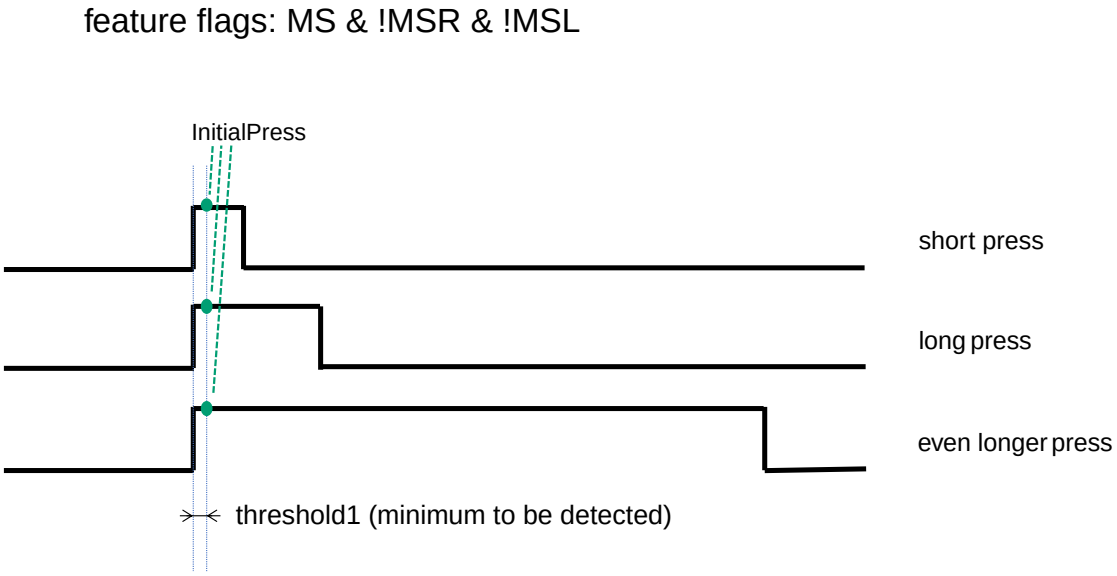


Figure 5. Switch device delivering only 'InitialPress' events

1.13.8. Sequence of events for MultiPress

Multi-press detection is a feature of momentary switches (indicated with feature flag MSM) that they can count and report sequences of press-release cycles within a certain time frame, for example to indicate that the user has pressed the switch once, twice or three (or even more) times in suc-

cession. The definition of the time window for this detection is manufacturer-specific since it depends on the switch physics. The maximum number of presses which can be detected and reported SHALL be indicated in attribute MultiPressMax. The minimum value and default value for MultiPressMax are both 2.
A switch supporting MultiPress SHALL set feature flags MS & MSR & MSM, and optionally also feature flag MSL. It SHALL generate the MultiPressOngoing and MultiPressFinished events in *addition* to the other events defined for momentary switches (i.e. InitialPress, ShortRelease, and in case of MSL, LongPress and LongRelease).

The MultiPressOngoing event is provided for parties interested in keeping track of the actual presses during the multi-press sequence. The MultiPressComplete event is provided for parties interested in the outcome of the whole sequence: after the multi-press sequence has ended, they will receive the MultiPressComplete event indicating how many times the switch was pressed.

In the figure below, three sequences of user interaction are indicated:

- single press sequence: after the press and release moments, the InitialPress and ShortRelease events SHALL be generated. After some further time, when the switch has detected that there is no second press, it SHALL generate MultiPressComplete(1) since it has detected that the sequence consisted of one press. No MultiPressOngoing event SHALL be generated for this case.
- double press sequence: after each of the press and release moments, the InitialPress and ShortRelease events SHALL be generated. Additionally, when the switch is pressed for the second time, the MultiPressOngoing(2) event SHALL be generated, as the switch has detected the second press. Note that this event coincides with the second InitialPress event; both SHALL be generated. After some further time, when the switch has detected that there is no third press, it SHALL generate MultiPressComplete(2) since it has detected that the sequence consisted of two presses.
- third press sequence: after each press and release moments, the InitialPress and ShortRelease events SHALL be generated. Additionally, when the switch is pressed for the second time, the MultiPressOngoing(2) event SHALL be generated, as the switch has detected the second press. Note that this event coincides with the second InitialPress event; both SHALL be generated. Additionally, when the switch is pressed for the third time, the MultiPressOngoing(3) event SHALL be generated, as the switch has detected the third press. Note that this event coincides with the third InitialPress event; both SHALL be generated. After some further time, when the switch has detected that there is no fourth press, it SHALL generate MultiPressComplete(3) since it has detected that the sequence consisted of three presses.

For the above cases where multiple events need to be generated at the same time, the MultiPressOngoing event SHALL be generated directly after the InitialPress event.

NOTE	The numbers in parentheses in the bulleted text above and in the figure below indicate the value of the CurrentNumberOfPressesCounted resp. TotalNumberOfPressesCounted field in the event data.

NOTE	As with the other figures, sufficient debounce time needs to be take into account for the detection of press and release events. This is included in the figure, and has been left out of the description above for readability, but SHALL be applied.

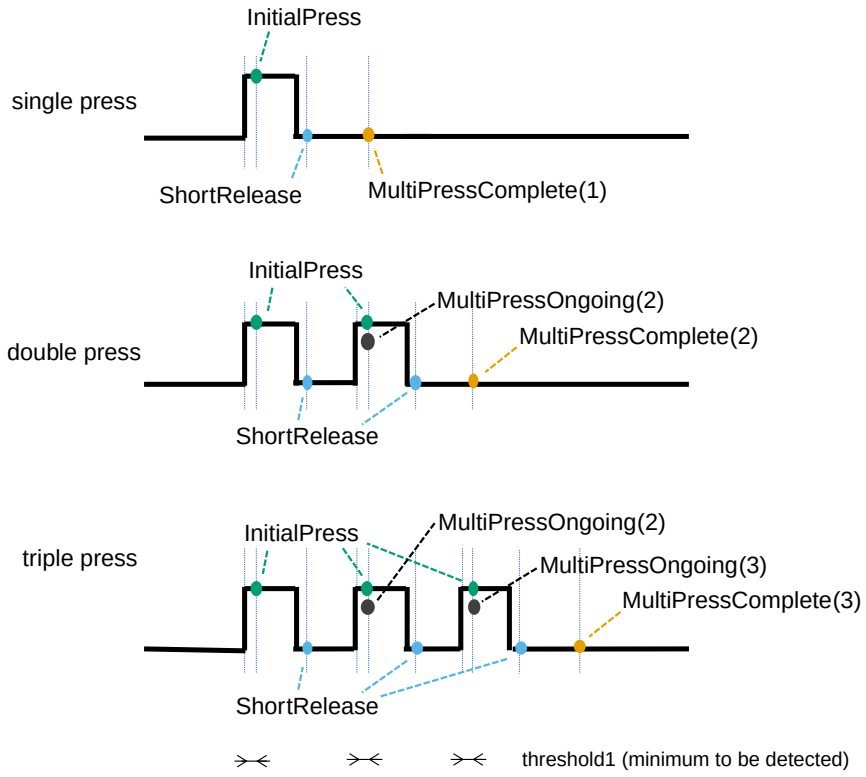


Figure 6. Switch device with multi-press events

1.13.9. Multi Position Details

This section will discuss some archetypes of switch devices with more than 2 positions and how they SHALL set attribute values and generate events matching their characteristics.

For the SwitchLatched, InitialPress, LongPress and MultiPressOngoing events, the field NewPosition SHALL be set to the value corresponding to the new position to which the switch was moved. For the ShortRelease, LongRelease and MultiPressComplete events, the field PreviousPosition SHALL be set to the value corresponding to the position of the switch just preceding the (latest) release.

1.13.9.1. Latching Switch with N stable positions (N>2) with fixed sequence

With such a device, the user can move the switch from a position M to positions M-1 and M+1 (either with a wraparound between the end positions, or fixed stop at the end positions).
On each interaction with the switch device, it SHALL generate a SwitchLatched event with the New-Position field set to the value associated with the new position.
Due to the physical constraints, such an event will have a NewPosition field which is equal to the previous NewPosition field plus or minus 1 (modulo NumPositions if the switch does not have end stops).

In a first example, a switch has 3 positions, associated with values 0, 1 and 2. In this case, wrap-around is not possible: from position 0 it can only be moved to position 1.

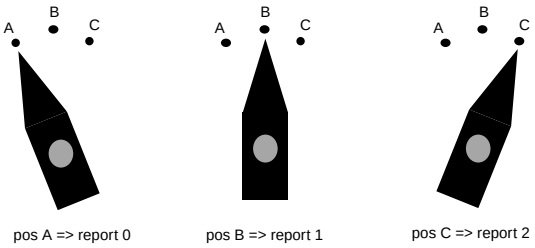


Figure 7. Rotary switch device with 3 positions

In another example, a switch has 8 positions, associated with values 0 through 7. In this case, the physics of the switch allow wraparound: from position 0 it can be moved to position 1 or to position 7.

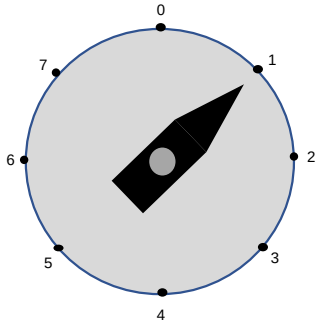


Figure 8. Rotary switch device with 8 positions and wraparound

1.13.9.2. Latching Switch with N stable positions (N>2) with random sequence (example: radio buttons)

With such a device, the user can press any of the available buttons, so this switch does not show the incrementing or decrementing behavior of NewPosition which we discussed for the latching switch with fixed sequence. In the example in the figure below, the 5 buttons are labeled "A" through "E" for the user and are associated with values 0 through 4. The user first presses the "A" button, and the switch device generates a SwitchLatched event with NewPosition set to 0. Then the user first presses the "D" button, and the switch device generates a SwitchLatched event with NewPosition set to 4.

Figure 9. Switch device with radio buttons

1.13.9.3. Momentary Switch with 2 or more non-stable positions

For a Momentary Switch with more than 1 stable position, it depends on the physics of the switch device which sequence of events will be generated.

NOTE	In this section we will mention only the InitialPress and ShortRelease events. The switch device could also generate the other events defined above for a momentary switch, depending on the capabilities of the switch device and the interaction with the switch device.

The first variant (figure below, example: up/down control for window blinds) shows a switch in neutral position (left figure) which corresponds to CurrentPosition=0. The user can press the top side (position value 1) or the bottom side (position value 2). It is not possible to go directly from position 1 to position 2 or vice versa - the switch will always need to go through the neutral position 0.

So when the user presses the top side of the switch, the InitialPress (NewPosition=1) event will be generated. When they release the top side, the ShortRelease (PreviousPosition=1) event will be generated. The user continues to press the bottom side, and the event InitialPress (NewPosition=2) is generated. Upon release of the bottom side, the event ShortRelease (PreviousPosition=2) is generated.

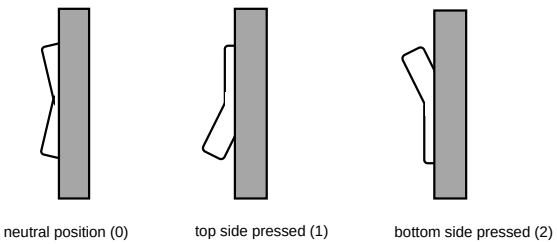


Figure 10. Up/down control switch device

Another variant (figure below, example: joystick) has a control handle with a neutral position in the

middle (left figure) which corresponds to CurrentPosition=0. The handle can be moved along the dotted lines.

In the middle figure, the user moves the handle to the East position and then releases it (which makes it return to the neutral middle position). This generates this sequence of events:

```
InitialPress (NewPosition=3)       // move to East
ShortRelease (PreviousPosition=3)  // back to middle (from East)
```

In the righthand figure, the user moves the handle to the SouthWest position, then to the South position and then releases it (which makes it return to the neutral middle position). This generates this sequence of events:

```
InitialPress (NewPosition=6)       // move to SouthWest
InitialPress (NewPosition=5)       // move to South
ShortRelease (PreviousPosition=5)  // back to middle (from South)
```

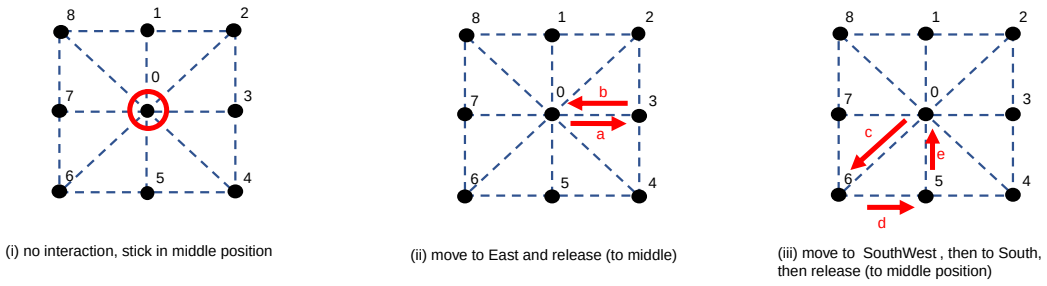


Figure 11. Switch device with joystick

Therefore, in the "joystick" variation, there could be a succession of InitialPress events without an intermediate ShortRelease events - unlike the "window blinds control" variation which will always have such an intermediate ShortRelease event.

1.14. Operational State Cluster

This cluster supports remotely monitoring and, where supported, changing the operational state of any device where a state machine is a part of the operation.

This cluster defines common states, scoped to this cluster (e.g. Stopped, Running, Paused, Error). A derived cluster specification may define more states scoped to the derivation. Manufacturer specific states are supported in this cluster and any derived clusters thereof. When defined in a derived instance, such states are scoped to the derivation.

Actual state transitions are dependent on both the implementation, and the requirements that may additionally be imposed by a derived cluster.

An implementation that supports remotely starting its operation can make use of this cluster's Start command to do so. A device that supports remote pause or stop of its currently selected operation can similarly make use of this cluster's Pause and Stop commands to do so. The ability to remotely pause or stop is independent of how the operation was started (for example, an operation started by using a manual button press can be stopped by using a Stop command if the device supports remotely stopping the operation).

Additionally, this cluster provides events for monitoring the operational state of the device.

1.14.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Rev	Description
1	Initial release
2	The Pause and Resume commands are usable in all compatible states. Define reserved ranges for base/derived clusters.

1.14.2. Classification

Hierarchy	Role	Scope	PICS Code
Base	Application	Endpoint	OPSTATE

1.14.3. Cluster ID

ID	Name
0x0060	Operational State

1.14.4. Data Types

1.14.4.1. OperationalStateEnum Type

This type defines the set of known operational state values, and is derived from enum8. The following table defines the applicable ranges for values that are defined within this type. All values that are undefined SHALL be treated as reserved. As shown by the table, states that may be specific to a certain Device Type or other modality SHALL be defined in a derived cluster of this cluster.

Value	Name	Summary
0x00 to 0x3F	**GeneralStates**	Generally applicable values for state, defined herein
0x40 to 0x7F	**DerivedClusterStates**	Derived Cluster defined states
0x80 to 0xBF	**ManufacturerStates**	Vendor specific states

The derived cluster-specific state definitions SHALL NOT duplicate any general state definitions. That is, a derived cluster specification of this cluster cannot define states with the same semantics as the general states defined below.

A manufacturer-specific state definition SHALL NOT duplicate the general state definitions or derived cluster state definitions. That is, a manufacturer-defined state defined for this cluster or a derived cluster thereof cannot define a state with the same semantics as the general states defined below or states defined in a derived cluster. Such manufacturer-specific state definitions SHALL be scoped in the context of the Vendor ID present in the Basic Information cluster.

The following table defines the generally applicable states.

Value	Name	Summary	Conformance
0x00	**Stopped**	The device is stopped	M
0x01	**Running**	The device is operating	M
0x02	**Paused**	The device is paused during an operation	M
0x03	**Error**	The device is in an error state	M

1.14.4.2. OperationalStateStruct Type

The OperationalStateStruct is used to indicate a possible state of the device.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**OperationalStateID**	OperationalStateEnum	all		0	M
1	**OperationalStateLabel**	string	max 64			desc

1.14.4.2.1. OperationalStateID Field

This SHALL be populated with a value from the OperationalStateEnum.

1.14.4.2.2. OperationalStateLabel Field

This field SHALL be present if the OperationalStateID is from the set reserved for Manufacturer Specific States, otherwise it SHALL NOT be present. If present, this SHALL contain a human-readable description of the operational state.

1.14.4.3. ErrorStateEnum Type

This type defines the set of known operational error values, and is derived from enum8. The following table defines the applicable ranges for values that are defined within this type. All values that

are undefined SHALL be treated as reserved. As shown by the table, errors that may be specific to a certain Device Type or other modality SHALL be defined in a derived cluster of this cluster.

Value	Name	Summary
0x00 to 0x3F	**GeneralErrors**	Generally applicable values for error, defined herein
0x40 to 0x7F	**DerivedClusterErrors**	Derived Cluster defined errors
0x80 to 0xBF	**ManufacturerError**	Vendor specific errors

The derived cluster-specific error definitions SHALL NOT duplicate the general error definitions. That is, a derived cluster specification of this cluster cannot define errors with the same semantics as the general errors defined below.

The manufacturer-specific error definitions SHALL NOT duplicate the general error definitions or derived cluster-specific error definitions. That is, a manufacturer-defined error defined for this cluster or a derived cluster thereof cannot define errors with the same semantics as the general errors defined below or errors defined in a derived cluster. Such manufacturer-specific error definitions SHALL be scoped in the context of the Vendor ID present in the Basic Information cluster.

The set of ErrorStateID field values defined in each of the generic or derived Operational State cluster specifications is called ErrorState.

1.14.4.3.1. ErrorStateEnum GeneralErrors Range

The following table defines the generally applicable ErrorState values.

Value	Name	Summary	Conformance
0x00	**NoError**	The device is not in an error state	M
0x01	**UnableToStartOrResume**	The device is unable to start or resume operation	M
0x02	**UnableToCompleteOperation**	The device was unable to complete the current operation	M
0x03	**CommandInvalidInState**	The device cannot process the command in its current state	M

1.14.4.4. ErrorStateStruct Type

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**ErrorStateID**	ErrorStateEnum	all		0	M

ID	Name	Type	Constraint	Quality	Default	Conformance
1	**ErrorStateLabel**	string	max 64		empty	desc
2	**ErrorStateDetails**	string	max 64		empty	O

1.14.4.4.1. ErrorStateID Field

This SHALL be populated with a value from the ErrorStateEnum.

1.14.4.4.2. ErrorStateLabel Field

This field SHALL be present if the ErrorStateID is from the set reserved for Manufacturer Specific Errors, otherwise it SHALL NOT be present. If present, this SHALL contain a human-readable description of the ErrorStateID; e.g. for a manufacturer specific ErrorStateID of "0x80" the ErrorStateLabel MAY contain "My special error".

1.14.4.4.3. ErrorStateDetails Field

This SHALL be a human-readable string that provides details about the error condition. As an example, if the ErrorStateID indicates that the device is a Robotic Vacuum that is stuck, the ErrorStateDetails contains "left wheel blocked".

1.14.5. Attributes

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0000	**PhaseList**	list[string]	max 32[max 64]	X	MS	R V	M
0x0001	**CurrentPhase**	uint8	desc	X	MS	R V	M
0x0002	**CountdownTime**	elapsed-s	max 259200	X C	null	R V	O
0x0003	**OperationalStateList**	list[OperationalStateStruct]	desc		MS	R V	M
0x0004	**OperationalState**	OperationalStateEnum	all			R V	M
0x0005	**OperationalError**	ErrorStateStruct	desc			R V	M

1.14.5.1. PhaseList Attribute

This attribute SHALL indicate a list of names of different phases that the device can go through for the selected function or mode. The list may not be in sequence order. For example in a washing machine this could include items such as "pre-soak", "rinse", and "spin". These phases are manufacturer specific and may change when a different function or mode is selected.

A null value indicates that the device does not present phases during its operation. When this attribute's value is null, the CurrentPhase attribute SHALL also be set to null.

1.14.5.2. CurrentPhase Attribute

This attribute represents the current phase of operation being performed by the server. This SHALL be the positional index representing the value from the set provided in the PhaseList Attribute, where the first item in that list is an index of 0. Thus, this attribute SHALL have a maximum value that is "length(PhaseList) - 1".

This attribute SHALL be null if the PhaseList attribute is null or if the PhaseList attribute is an empty list.

1.14.5.3. CountdownTime Attribute

This attribute SHALL represent the estimated time left before the operation is completed, in seconds. Changes to this value SHALL NOT be reported in a subscription (note the C Quality). A Client implementation MAY periodically poll this value to ensure alignment of any local rendering of the CountdownTime with the device provided value.

A value of 0 means that the operation has completed.

When this attribute is null, that represents that there is no time currently defined until operation completion. This MAY happen, for example, because no operation is in progress or because the completion time is unknown.

1.14.5.4. OperationalStateList Attribute

This attribute describes the set of possible operational states that the device exposes. An operational state is a fundamental device state such as Running or Error. Details of the phase of a device when, for example, in a state of Running are provided by the CurrentPhase attribute.

All devices SHALL, at a minimum, expose the set of states matching the commands that are also supported by the cluster instance, in addition to Error. The set of possible device states are defined in the OperationalStateEnum. A device type requiring implementation of this cluster SHALL define the set of states that are applicable to that specific device type.

1.14.5.5. OperationalState Attribute

This attribute specifies the current operational state of a device. This SHALL be populated with a valid OperationalStateID from the set of values in the OperationalStateList Attribute.

1.14.5.6. OperationalError Attribute

This attribute SHALL specify the details of any current error condition being experienced on the device when the OperationalState attribute is populated with Error. Please see ErrorStateStruct for general requirements on the population of this attribute.

When there is no error detected, this SHALL have an ErrorStateID of NoError.

1.14.6. Commands

ID	Name	Direction	Response	Access	Conformance
0x00	**Pause**	client ⇒ server	Operational-Comman-dResponse	O	Resume, O
0x01	**Stop**	client ⇒ server	Operational-Comman-dResponse	O	Start, O
0x02	**Start**	client ⇒ server	Operational-Comman-dResponse	O	O
0x03	**Resume**	client ⇒ server	Operational-Comman-dResponse	O	Pause, O
0x04	**Operational-Comman-dResponse**	client ⇐ server	N	O	Pause | Stop | Start | Resume

Note that it is entirely possible due to regulatory or other reasons for an instance of this cluster to expose no possible commands. When that occurs, this cluster does not provide any ability to actuate the device, instead it provides readable (and by extension, can be subscribed to) information as to the state of the device only. The commands that are supported SHALL be exposed by the device in the AcceptedCommandList global attribute.

1.14.6.1. Pause Command

This command SHALL be supported if the device supports remotely pausing the operation. If this command is supported, the Resume command SHALL also be supported.

On receipt of this command, the device SHALL pause its operation if it is possible based on the current function of the server. For example, if it is at a point where it is safe to do so and/or permitted, but can be restarted from the point at which pause occurred.

If this command is received when already in the Paused state the device SHALL respond with an OperationalCommandResponse command with an ErrorStateID of NoError but take no further action.

A device that receives this command in any state which is not Pause-compatible SHALL respond

with an OperationalCommandResponse command with an ErrorStateID of CommandInvalidInState and SHALL take no further action.

States are defined as Pause-compatible as follows:

- For states defined in this cluster specification, in Table 3, “Pause Compatibility”.
- For states defined by derived cluster specifications, in the corresponding specifications.
- For manufacturer-specific states, by the manufacturer.

A device that is unable to honor the Pause command for whatever reason SHALL respond with an OperationalCommandResponse command with an ErrorStateID of CommandInvalidInState but take no further action.

Otherwise, on success:

- The OperationalState attribute SHALL be set to Paused.
- The device SHALL respond with an OperationalCommandResponse command with an ErrorStateID of NoError.

The following table defines the compatibility of this cluster’s states with the Pause command.

Table 3. Pause Compatibility

State Value	State Name	Pause-Compatible
0x00	**Stopped**	N
0x01	**Running**	Y
0x02	**Paused**	Y
0x03	**Error**	N

1.14.6.2. Stop Command

This command SHALL be supported if the device supports remotely stopping the operation.

On receipt of this command, the device SHALL stop its operation if it is at a position where it is safe to do so and/or permitted. Restart of the device following the receipt of the Stop command SHALL require attended operation unless remote start is allowed by the device type and any jurisdiction governing remote operation of the device.

If this command is received when already in the Stopped state the device SHALL respond with an OperationalCommandResponse command with an ErrorStateID of NoError but take no further action.

A device that is unable to honor the Stop command for whatever reason SHALL respond with an OperationalCommandResponse command with an ErrorStateID of CommandInvalidInState but take no further action.

Otherwise, on success:

- The OperationalState attribute SHALL be set to Stopped.

- The device SHALL respond with an OperationalCommandResponse command with an ErrorStateID of NoError.

1.14.6.3. Start Command

This command SHALL be supported if the device supports remotely starting the operation. If this command is supported, the 'Stop command SHALL also be supported.

On receipt of this command, the device SHALL start its operation if it is safe to do so and the device is in an operational state from which it can be started. There may be either regulatory or manufacturer-imposed safety and security requirements that first necessitate some specific action at the device before a Start command can be honored. In such instances, a device SHALL respond with a status code of CommandInvalidInState if a Start command is received prior to the required on-device action.

If this command is received when already in the Running state the device SHALL respond with an OperationalCommandResponse command with an ErrorStateID of NoError but take no further action.

A device that is unable to honor the Start command for whatever reason SHALL respond with an OperationalCommandResponse command with an ErrorStateID of UnableToStartOrResume but take no further action.

Otherwise, on success:

- The OperationalState attribute SHALL be set to Running.
- The device SHALL respond with an OperationalCommandResponse command with an ErrorStateID of NoError.

1.14.6.4. Resume Command

This command SHALL be supported if the device supports remotely resuming the operation. If this command is supported, the Pause command SHALL also be supported.

On receipt of this command, the device SHALL resume its operation from the point it was at when it received the Pause command, or from the point when it was paused by means outside of this cluster (for example by manual button press).

If this command is received when already in the Running state the device SHALL respond with an OperationalCommandResponse command with an ErrorStateID of NoError but take no further action.

A device that receives this command in any state which is not Resume-compatible SHALL respond with an OperationalCommandResponse command with an ErrorStateID of CommandInvalidInState and SHALL take no further action.

States are defined as Resume-compatible as follows:

- For states defined in this cluster specification, in Table 4, “Resume Compatibility”.
- For states defined by derived cluster specifications, in the corresponding specifications.

- For manufacturer-specific states, by the manufacturer.

The following table defines the compatibility of this cluster's states with the Resume command.

Table 4. Resume Compatibility

State Value	State Name	Resume-Compatible
0x00	**Stopped**	N
0x01	**Running**	Y
0x02	**Paused**	Y
0x03	**Error**	N

A device that is unable to honor the Resume command for any other reason SHALL respond with an OperationalCommandResponse command with an ErrorStateID of UnableToStartOrResume but take no further action.

Otherwise, on success:

- The OperationalState attribute SHALL be set to the most recent non-Error operational state prior to entering the Paused state.
- The device SHALL respond with an OperationalCommandResponse command with an ErrorStateID of NoError.

1.14.6.5. OperationalCommandResponse Command

This command SHALL be supported by an implementation if any of the other commands defined by this cluster are supported (i.e. listed in the AcceptedCommandList global attribute). This command SHALL also be supported by an implementation of a derived cluster as a response to any commands that MAY be additionally defined therein.

This command SHALL be generated in response to any of the Start, Stop, Pause, or Resume commands. The data for this command SHALL be as follows:

ID	Name	Type	Constraint	Quality	Default	Conformance
0x00	**CommandResponseState**	ErrorStateStruct	all			M

1.14.6.5.1. CommandResponseState Field

This SHALL indicate the success or otherwise of the attempted command invocation. On a successful invocation of the attempted command, the ErrorStateID SHALL be populated with NoError. Please see the individual command sections for additional specific requirements on population.

1.14.7. Events

ID	Name	Priority	Access	Conformance
0x00	**OperationalError**	CRITICAL	V	M
0x01	**OperationCompletion**	INFO	V	P, O

1.14.7.1. OperationalError Event

This event is generated when a reportable error condition is detected. A device that generates this event SHALL also set the OperationalState attribute to Error, indicating an error condition.

This event SHALL contain the following fields:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	ErrorState	ErrorStateStruct	all			M

1.14.7.2. OperationCompletion Event

This event is generated when the overall operation ends, successfully or otherwise. For example, the completion of a cleaning operation in a Robot Vacuum Cleaner, or the completion of a wash cycle in a Washing Machine.

This event SHALL contain the following fields:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	CompletionErrorCode	enum8	all			M
1	TotalOperationalTime	elapsed-s	all	X		O
2	PausedTime	elapsed-s	all	X		O

1.14.7.2.1. CompletionErrorCode Field

This field provides an indication of the state at the end of the operation. This field SHALL have a value from the ErrorStateEnum set. A value of NoError indicates success, that is, no error has been detected.

1.14.7.2.2. TotalOperationalTime Field

The total operational time, in seconds, from when the operation was started via an initial Start command or manual action, until the operation completed. This includes any time spent while paused. There may be cases whereby the total operational time exceeds the maximum value that can be conveyed by this attribute, in such instances, this attribute SHALL be populated with null.

1.14.7.2.3. PausedTime Field

The total time spent in the paused state, in seconds. There may be cases whereby the total paused time exceeds the maximum value that can be conveyed by this attribute, in such instances, this attribute SHALL be populated with null.

1.15. Alarm Base Cluster

This cluster is a base cluster from which clusters for particular alarms for a device type can be derived. Each derivation SHALL define the values for the AlarmBitmap data type used in this cluster. Each derivation SHALL define which alarms are latched.

1.15.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Revision	Description
1	Initial revision

1.15.2. Classification

Hierarchy	Role	Scope	PICS Code
Base	Application	Endpoint	ALARM

1.15.3. Cluster ID

ID	Name
n/a	Alarm Base

1.15.4. Features

This cluster SHALL support the FeatureMap bitmap attribute as defined below.

Bit	Code	Feature	Summary
0	**RESET**	Reset	Supports the ability to reset alarms

1.15.4.1. Reset Feature

This feature indicates that alarms can be reset via the Reset command.

1.15.5. Data Types

1.15.5.1. AlarmBitmap Type

This data type SHALL be a map32 with values defined by the derived cluster. The meaning of each bit position SHALL be consistent for all attributes in a derived cluster. That is, if bit 0 is defined for an alarm, the Latch, State, and Supported information for that alarm are also bit 0.

1.15.6. Attributes

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0000	**Mask**	Alarm-Bitmap	all		0	R V	M
0x0001	**Latch**	Alarm-Bitmap	all	F	0	R V	RESET
0x0002	**State**	Alarm-Bitmap	all		0	R V	M
0x0003	**Supported**	Alarm-Bitmap	all	F	0	R V	M

1.15.6.1. Mask Attribute

This attribute SHALL indicate a bitmap where each bit set in the Mask attribute corresponds to an alarm that SHALL be enabled.

1.15.6.2. Latch Attribute

This attribute SHALL indicate a bitmap where each bit set in the Latch attribute SHALL indicate that the corresponding alarm will be latched when set, and will not reset to inactive when the underlying condition which caused the alarm is no longer present, and so requires an explicit reset using the Reset command.

1.15.6.3. State Attribute

This attribute SHALL indicate a bitmap where each bit SHALL represent the state of an alarm. The value of true means the alarm is active, otherwise the alarm is inactive.

1.15.6.4. Supported Attribute

This attribute SHALL indicate a bitmap where each bit SHALL represent whether or not an alarm is supported. The value of true means the alarm is supported, otherwise the alarm is not supported.

If an alarm is not supported, the corresponding bit in Mask, Latch, and State SHALL be false.

1.15.7. Commands

ID	Name	Direction	Response	Access	Conformance
0x00	**Reset**	client ⇒ server	Y	O	RESET

ID	Name	Direction	Response	Access	Conformance
0x01	**ModifyEnabledAlarms**	client ⇒ server	Y	O	O

1.15.7.1. Reset Command

This command resets active and latched alarms (if possible). Any generated Notify event SHALL contain fields that represent the state of the server after the command has been processed.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**Alarms**	AlarmBitmap	all		0	M

1.15.7.1.1. Alarms Field

This field SHALL indicate a bitmap where each bit set in this field corresponds to an alarm that SHALL be reset to inactive in the State attribute unless the alarm definition requires manual intervention. If the alarms indicated are successfully reset, the response status code SHALL be SUCCESS, otherwise, the response status code SHALL be FAILURE.

1.15.7.2. ModifyEnabledAlarms Command

This command allows a client to request that an alarm be enabled or suppressed at the server.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**Mask**	AlarmBitmap	all		0	M

1.15.7.2.1. Mask Field

This field SHALL indicate a bitmap where each bit set in the this field corresponds to an alarm that SHOULD be enabled or suppressed. A value of 1 SHALL indicate that the alarm SHOULD be enabled while a value of 0 SHALL indicate that the alarm SHOULD be suppressed.

A server that receives this command with a Mask that includes bits that are set for unknown alarms SHALL respond with a status code of INVALID_COMMAND.

A server that receives this command with a Mask that includes bits that are set for alarms which are not supported, as indicated in the Supported attribute, SHALL respond with a status code of INVALID_COMMAND.

A server that is unable to enable a currently suppressed alarm, or is unable to suppress a currently enabled alarm SHALL respond with a status code of FAILURE; otherwise the server SHALL respond with a status code of SUCCESS.

On a SUCCESS case, the server SHALL also change the value of the Mask attribute to the value of the

Mask field from this command. After that the server SHALL also update the value of its State attribute to reflect the status of the new alarm set as indicated by the new value of the Mask attribute.

1.15.8. Events

ID	Name	Priority	Access	Conformance
0x00	**Notify**	INFO	V	M

1.15.8.1. Notify Event

This event SHALL be generated when one or more alarms change state, and SHALL have these fields:

ID	Name	Type	Constraint	Quality	Default	Conformance
1	**Active**	Alarm-Bitmap	all		0	M
2	**Inactive**	Alarm-Bitmap	all		0	M
3	**State**	Alarm-Bitmap	all		0	M
4	**Mask**	Alarm-Bitmap	all		0	M

1.15.8.1.1. Active Field

This field SHALL indicate those alarms that have become active.

1.15.8.1.2. Inactive Field

This field SHALL indicate those alarms that have become inactive.

1.15.8.1.3. Mask Field

This field SHALL be a copy of the Mask attribute when this event was generated.

1.15.8.1.4. State Field

This field SHALL be a copy of the new State attribute value that resulted in the event being generated. That is, this field SHALL have all the bits in Active set and SHALL NOT have any of the bits in Inactive set.

1.16. Messages Cluster

This cluster provides an interface for passing messages to be presented by a device.

1.16.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Revision	Description
1	mandatory global *ClusterRevision* attribute added
2	Updated from SE1.4 version; CCB 1819
3	Initial Matter release; renamed from EnergyMessaging to Messages

1.16.2. Classification

Hierarchy	Role	Scope	PICS Code
Base	Application	Endpoint	MESS

1.16.3. Cluster ID

ID	Name
0x0097	Messages

1.16.4. Features

This cluster SHALL support the FeatureMap bitmap attribute as defined below.

Bit	Code	Feature	Conformance	Summary
0	**CONF**	ReceivedConfirmation	O	
1	**RESP**	ConfirmationResponse	[CONF]	
2	**RPLY**	ConfirmationReply	[CONF]	
3	**PROT**	ProtectedMessages	O	

1.16.4.1. ReceivedConfirmation

This feature SHALL indicate that the device can get confirmation from a user that the message was received.

1.16.4.2. ConfirmationResponse

This feature SHALL indicate that the device is capable of presenting a list of responses to the user and recording the user's choice of response.

1.16.4.3. ConfirmationReply

This feature SHALL indicate that the device is capable of collecting a free-form text response to a message.

1.16.4.4. ProtectedMessages

This feature SHALL indicate that the device is capable of requiring the user to authenticate before viewing a message; e.g. entering a PIN or password before viewing a message with billing information.

1.16.5. Data Types

1.16.5.1. MessageID Type

This data type is an octstr of fixed length 16, containing the binary encoding of a UUID as specified in RFC 4122.

1.16.5.2. MessageControlBitmap Type

This data type is derived from map16, and indicates control information related to a message.

Bit	Name	Summary	Conformance
0	**ConfirmationRequired**	Message requires confirmation from user	CONF
1	**ResponseRequired**	Message requires response from user	RESP
2	**ReplyMessage**	Message supports reply message from user	RPLY
3	**MessageConfirmed**	Message has already been confirmed	CONF
4	**MessageProtected**	Message required PIN/password protection	PROT

1.16.5.2.1. ConfirmationRequired Bit

This bit SHALL indicate that the message originator requests a confirmation of receipt by the user. If confirmation is required, the device SHOULD present the message until it is either confirmed by the user selecting a confirmation option, or the message expires.

1.16.5.2.2. ResponseRequired Bit

This bit SHALL indicate that a MessagePresented event SHOULD be generated based on the response of the user to the message.

1.16.5.2.3. ReplyMessage Bit

This bit SHALL indicate that a free-form user reply is to be included in the confirmation of receipt.

1.16.5.2.4. MessageConfirmed Bit

This bit SHALL indicate the current confirmation state of a message, which is useful in the event that there are multiple Messages cluster client devices on a network.

1.16.5.2.5. MessageProtected Bit

This bit SHALL indicate that user authentication (e.g. by password or PIN) is required before viewing a message.

1.16.5.3. FutureMessagePreferenceEnum Type

This data type is derived from enum8.

A display device MAY include this preference in the MessageComplete event as a hint to clients about how to handle future similar messages.

Value	Name	Summary	Conformance
0	**Allowed**	Similar messages are allowed	M
1	**Increased**	Similar messages should be sent more often	M
2	**Reduced**	Similar messages should be sent less often	M
3	**Disallowed**	Similar messages should not be sent	M
4	**Banned**	No further messages should be sent	M

1.16.5.4. MessagePriorityEnum Type

This data type is derived from enum8.

Priority SHOULD be used to decide which messages to show when the number of eligible messages is larger than the device's capacity to present them.

Value	Name	Summary	Conformance
0	**Low**	Message to be transferred with a low level of importance	M

Value	Name	Summary	Conformance
1	**Medium**	Message to be transferred with a medium level of importance	M
2	**High**	Message to be transferred with a high level of importance	M
3	**Critical**	Message to be transferred with a critical level of importance	M

1.16.5.5. MessageStruct Type

This represents a single message.

Access Quality: Fabric Scoped							
ID	**Name**	**Type**	**Constraint**	**Quality**	**Default**	**Access**	**Conformance**
0	**MessageID**	MessageID	all			S	M
1	**Priority**	MessagePriorityEnum			0	S	M
2	**MessageControl**	MessageControlBitmap			0	S	M
3	**StartTime**	epoch-s		X	0	S	M
4	**Duration**	uint64	all	X	0	S	M
5	**MessageText**	string	max 256			S	M
6	**Responses**	list[MessageResponseOptionStruct]	max 4		empty	S	RESP

1.16.5.5.1. MessageID Field

This field SHALL indicate a globally unique ID for this message.

1.16.5.5.2. Priority Field

This field SHALL indicate the priority level for this message.

1.16.5.5.3. MessageControl Field

This field SHALL indicate control information related to the message.

1.16.5.5.4. StartTime Field

This field SHALL indicate the time in UTC at which the message becomes available to be presented. A null value SHALL indicate "now."

1.16.5.5.5. Duration Field

This field SHALL indicate the amount of time, in milliseconds, after the StartTime during which the message is available to be presented. A null value SHALL indicate "until changed".

1.16.5.5.6. MessageText Field

This field SHALL indicate a string containing the message to be presented.

1.16.5.5.7. Responses Field

This field SHALL indicate a list of potential responses to the message. The entries in this list SHALL have unique values of MessageResponseID.

If the ResponseRequired bit is set on the message but this list is empty, the device SHALL provide a generic acknowledgement button, e.g. "OK".

If the ResponseRequired bit is not set on the message, this list SHALL be ignored.

1.16.5.6. MessageResponseOptionStruct Type

This represents a possible response to a message.

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0	**MessageResponseID**	uint32	min 1				M
1	**Label**	string	max 32				M

1.16.5.6.1. MessageResponseID Field

This field SHALL indicate a unique unsigned 32-bit number identifier for this message response option.

1.16.5.6.2. Label Field

This field SHALL indicate the text for this option; e.g. "Yes", "No", etc.

1.16.6. Attributes

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0000	**Messages**	list[MessageStruct]	max 8		empty	R V F	M
0x0001	**ActiveMessageIDs**	list[MessageID]	max 8		empty	R V	M

1.16.6.1. Messages Attribute

This attribute SHALL indicate a list of queued messages.

In addition to filtering based upon fabric, to preserve user privacy, the server MAY further limit the set of messages returned in a read request. At minimum, the server SHALL return to a client those messages that the client itself created/submitted.

1.16.6.2. ActiveMessageIDs Attribute

This attribute SHALL indicate a list of the MessageIDs of the Messages currently being presented. If this list is empty, no messages are currently being presented.

This list SHALL NOT be fabric-scoped; it SHALL contain MessageIDs for all Messages being presented, no matter what fabric the client that queued them is on.

1.16.7. Commands

ID	Name	Direction	Response	Access	Conformance
0x00	**PresentMessagesRequest**	client ⇒ server	Y	O F	M
0x01	**CancelMessagesRequest**	client ⇒ server	Y	O F	M

1.16.7.1. PresentMessagesRequest Command

Upon receipt, this SHALL cause the message in the passed fields to be appended to the Messages attribute.

If appending the message would cause the number of messages to be greater than the capacity of the list, the device SHALL NOT append any message to Messages, and SHALL return a status code of RESOURCE_EXHAUSTED.

When displaying a message in response to this command, an indication (ex. visual) of the origin node of the command SHALL be provided. This could be in the form of a friendly name label which uniquely identifies the node to the user. This friendly name label is typically assigned by the Matter Admin at the time of commissioning and, when it's a device, is often editable by the user. It might be a combination of a company name and friendly name, for example, "Acme" or "Acme Streaming Service on Alice's Phone".

NOTE	It is currently not specified where the friendly name label can be found on the node, meaning that clients SHOULD NOT rely on a certain method they happen to observe in a particular server instance, since other instances could employ a different method.

The device SHOULD make it possible for the user to view which nodes have access to this cluster and to individually remove privileges for each node.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**MessageID**	MessageID	all			M
1	**Priority**	MessagePriorityEnum			0	M
2	**MessageControl**	MessageControlBitmap			0	M
3	**StartTime**	epoch-s		X	0	M
4	**Duration**	uint64	all	X	0	M
5	**MessageText**	string	max 256			M
6	**Responses**	list[MessageResponseOptionStruct]	max 4		empty	RESP

1.16.7.1.1. MessageID Field

This field SHALL indicate a globally unique ID for this message. See MessageID.

1.16.7.1.2. Priority Field

This field SHALL indicate the priority level for this message. See Priority.

1.16.7.1.3. MessageControl Field

This field SHALL indicate control information related to the message. See MessageControl.

1.16.7.1.4. StartTime Field

This field SHALL indicate the time in UTC at which the message becomes available to be presented. A null value SHALL indicate "now." See StartTime.

1.16.7.1.5. Duration Field

This field SHALL indicate the amount of time, in milliseconds, after the StartTime during which the message is available to be presented. A null value SHALL indicate "until changed". See Duration.

1.16.7.1.6. MessageText Field

This field SHALL indicate a string containing the message to be presented. See MessageText.

1.16.7.1.7. Responses Field

This field SHALL indicate a list of potential responses to the message. The entries in this list SHALL have unique values of MessageResponseID.

If the ResponseRequired bit is set on the message but this list is empty, the device SHALL provide a generic acknowledgement button, e.g. "OK".

If the ResponseRequired bit is not set on the message, this list SHALL be ignored.

See Responses.

1.16.7.2. CancelMessagesRequest Command

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**MessageIDs**	list[MessageID]	max 8			M

1.16.7.2.1. MessageIDs Field

This field SHALL indicate the MessageIDs for the messages being cancelled.

Cancelling a message SHALL cause it to be removed from Messages, cause its MessageID to be removed from ActiveMessageIDs and cause any active presentation of the message to cease.

Message IDs in this command that indicate messages that do not exist in Messages, or that are not scoped to the fabric of the sender, SHALL be ignored.

1.16.8. Events

ID	Name	Priority	Access	Conformance
0x00	**MessageQueued**	INFO	V	M
0x01	**MessagePresented**	INFO	V	M
0x02	**MessageComplete**	INFO	V	M

1.16.8.1. MessageQueued Event

This event SHALL be generated when a message is added to the messages attribute.

Access Quality: Fabric-Sensitive						
ID	**Name**	**Type**	**Constraint**	**Quality**	**Default**	**Conformance**
0	**MessageID**	MessageID				M

1.16.8.1.1. MessageID Field

This field SHALL indicate the MessageID for newly added message.

1.16.8.2. MessagePresented Event

This event SHALL be generated when the message is presented to the user.

Access Quality: Fabric-Sensitive						
ID	**Name**	**Type**	**Constraint**	**Quality**	**Default**	**Conformance**
0	**MessageID**	MessageID				M

1.16.8.2.1. MessageID Field

This field SHALL indicate the MessageID for the message being presented.

1.16.8.3. MessageComplete Event

This event SHALL be generated when the message is confirmed by the user, or when the Duration of the message has elapsed without confirmation.

Access Quality: Fabric-Sensitive						
ID	**Name**	**Type**	**Constraint**	**Quality**	**Default**	**Conformance**
0	**MessageID**	MessageID				M
1	**ResponseID**	uint32	all	X	null	RESP
2	**Reply**	string	max 256	X	null	RPLY
3	**FutureMessagesPreference**	FutureMessagePreferenceEnum	all	X	null	M

1.16.8.3.1. MessageID Field

This field SHALL indicate the MessageID for the message being confirmed.

1.16.8.3.2. ResponseID Field

This field SHALL indicate the MessageResponseID selected by the user. If there was no response before the Duration of the message has elapsed, this field SHALL be null.

1.16.8.3.3. Reply Field

This field SHALL indicate a user-provided reply to the message. If there was no reply, or the message did not have the ReplyRequired bit set, this field SHALL be null.

1.16.8.3.4. FutureMessagesPref Field

This field SHALL indicate a user-provided preference for the delivery of similar messages in the future. A null value SHALL indicate no change in preference.

Chapter 2. Measurement and Sensing

The Cluster Library is made of individual chapters such as this one. See Document Control in the Cluster Library for a list of all chapters and documents. References between chapters are made using a *X.Y* notation where *X* is the chapter and *Y* is the sub-section within that chapter. References to external documents are contained in Chapter 1 and are made using [*Rn*] notation.

2.1. General Description

2.1.1. 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.

2.1.2. Cluster List

This section lists the measurement and sensing clusters as specified in this chapter.

Table 5. Overview of the Measurement and Sensing Clusters

Cluster ID	Cluster Name	Description
0x0400	Illuminance Measurement	Attributes and commands for configuring the measurement of illuminance, and reporting illuminance measurements
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	Supports configuring the measurement of relative humidity, and reporting relative humidity measurements of water in the air
0x0406	Occupancy Sensing	Attributes and commands for configuring occupancy sensing, and reporting occupancy status

Cluster ID	Cluster Name	Description
various	Resource Monitoring	Attributes and commands for reporting conditions of various resources
0x005B	Air Quality Measurement	Attributes for reporting air quality classification
various	Concentration Measurement	Attributes and aliases for concentration measurements
0x005C	Smoke and CO Alarm	An interface to smoke and CO alarms
0x0091	Electrical Energy Measurement	Attributes and commands for measuring electrical energy
0x0090	Electrical Power Measurement	Attributes and commands for measuring electrical power

2.1.3. Measured Value

This section provides requirements on the attributes MeasuredValue, MinMeasuredValue, MaxMeasuredValue. Accuracy of MeasuredValue is discussed in the following section.

2.1.3.1. Constraint

Where MinMeasuredValue or MaxMeasuredValue attributes are mandatory the null value MAY be used to indicate that a limit is unknown.

For any measurement cluster with MeasuredValue, MinMeasuredValue and MaxMeasuredValue attributes, the following SHALL be always be true:

- If MinMeasuredValue and MaxMeasuredValue are both known, then MaxMeasuredValue SHALL be greater than MinMeasuredValue.
- If MaxMeasuredValue is known, then MeasuredValue SHALL be less than or equal to MaxMeasuredValue.
- If MinMeasuredValue is known, then MeasuredValue SHALL be greater than or equal to MinMeasuredValue.

2.1.4. Measurement Accuracy

Measurement clusters MAY express the accuracy of their measurements with a Tolerance attribute expressing a simple magnitude of error, or with a MeasurementAccuracyStruct expressing magnitude or percentage ranges of error for different ranges of measured values.

2.1.4.1. Tolerance Attribute

For any measurement cluster with a MeasuredValue and Tolerance attribute, when Tolerance is implemented the following SHALL always be true:

- The Tolerance attribute SHALL indicate the magnitude of the possible error that is associated with MeasuredValue attribute, using the same units and resolution. The true value SHALL be in the range (MeasuredValue – Tolerance) to (MeasuredValue + Tolerance).
- If known, the true value SHALL never be outside the possible physical range. Some examples:
  - a temperature SHALL NOT be below absolute zero
  - a concentration SHALL NOT be negative

2.1.4.2. MeasurementTypeEnum Type

This data type is derived from enum16.

Value	Name	Summary	Conformance
0	Unspecified		M
1	Voltage	Voltage in millivolts (mV)	M
2	ActiveCurrent	Active current in milliamps (mA)	M
3	ReactiveCurrent	Reactive current in milliamps (mA)	M
4	ApparentCurrent	Apparent current in milliamps (mA)	M
5	ActivePower	Active power in milliwatts (mW)	M
6	ReactivePower	Reactive power in millivolt-amps reactive (mVAR)	M
7	ApparentPower	Apparent power in millivolt-amps (mVA)	M
8	RMSVoltage	Root mean squared voltage in millivolts (mV)	M
9	RMSCurrent	Root mean squared current in milliamps (mA)	M
10	RMSPower	Root mean squared power in milliwatts (mW)	M
11	Frequency	AC frequency in millihertz (mHz)	M
12	PowerFactor	Power Factor ratio in +/- 1/100ths of a percent.	M

Value	Name	Summary	Conformance
13	NeutralCurrent	AC neutral current in milliamps (mA)	M
14	ElectricalEnergy	Electrical energy in milliwatt-hours (mWh)	M

2.1.4.3. MeasurementAccuracyRangeStruct Type

This struct represents the accuracy of a measurement for a range of measurement values. Accuracy SHALL be expressed as a maximum +/- percentage of the true value, a maximum +/- fixed value of the true value, or both.

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0	**RangeMin**	int64	-2^{62} to 2^{62}	F		R	M
1	**RangeMax**	int64	-2^{62} to 2^{62}	F		R	M
2	**PercentMax**	percent100ths	all	F		R	O.a+
3	**PercentMin**	percent100ths	max PercentTypical	F		R	[PercentMax]
4	**PercentTypical**	percent100ths	PercentMin to PercentMax	F		R	[PercentMin]
5	**FixedMax**	uint64	max $2^{62} - 1$	F		R	O.a+
6	**FixedMin**	uint64	max FixedMax	F		R	[FixedMax]
7	**FixedTypical**	uint64	FixedMin to FixedMax	F		R	[FixedMin]

- If both PercentMax and FixedMax are indicated, then for a given true value in the range between RangeMin and RangeMax,
  - the reported value SHALL be less than or equal to the sum of the true value, FixedMax and PercentMax percent of the true value.
  - the reported value SHALL be greater than or equal to the true value minus the sum of FixedMax and PercentMax percent of the true value.
- If only PercentMax is indicated, then for a given true value in the range between RangeMin and RangeMax,
  - the reported value SHALL be less than or equal to the sum of the true value and PercentMax percent of the true value.
  - the reported value SHALL be greater than or equal to the true value minus PercentMax per-

cent of the true value.

- If only FixedMax is indicated, then for a given true value in the range between RangeMin and RangeMax,
  - the reported value SHALL be less than or equal to the sum of the true value and FixedMax.
  - the reported value SHALL be greater than or equal to the true value minus FixedMax.

2.1.4.3.1. RangeMin Field

This field SHALL indicate the minimum measurement value for the specified level of accuracy.

The value of this field SHALL be greater than or equal to the value of the MinMeasuredValue field on the encompassing MeasurementAccuracyStruct.

The value of this field SHALL be less than or equal to the value of the MaxMeasuredValue field on the encompassing MeasurementAccuracyStruct.

2.1.4.3.2. RangeMax Field

This field SHALL indicate the maximum measurement value for the specified level of accuracy.

The value of this field SHALL be greater than the value of the RangeMin field.

The value of this field SHALL be greater than or equal to the value of the MinMeasuredValue field on the encompassing MeasurementAccuracyStruct.

The value of this field SHALL be less than or equal to the value of the MaxMeasuredValue field on the encompassing MeasurementAccuracyStruct.

2.1.4.3.3. PercentMax Field

This field SHALL indicate the maximum +/- percentage accuracy for the associated measurement.

2.1.4.3.4. PercentMin Field

This field SHALL indicate the minimum +/- percentage accuracy for the associated measurement.

2.1.4.3.5. PercentTypical Field

This field SHALL indicate the typical +/- percentage accuracy for the associated measurement.

2.1.4.3.6. FixedMax Field

This field SHALL indicate the maximum +/- fixed accuracy for the associated measurement, in the unit indicated by MeasurementType.

2.1.4.3.7. FixedMin Field

This field SHALL indicate the minimum +/- fixed accuracy for the associated measurement, in the unit indicated by MeasurementType.

2.1.4.3.8. FixedTypical Field

This field SHALL indicate the typical +/- fixed accuracy for the associated measurement, in the unit indicated by MeasurementType.

2.1.4.4. MeasurementAccuracyStruct Type

This struct represents the set of accuracy ranges for a given measurement, the maximum and minimum values for the measurement, and whether the measurement is directly measured or just estimated from other information.

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0	**MeasurementType**	MeasurementTypeEnum		F		R	M
1	**Measured**	bool		F	false	R	M
2	**MinMeasuredValue**	int64	-2^{62} to 2^{62}	F		R	M
3	**MaxMeasuredValue**	int64	-2^{62} to 2^{62}	F		R	M
4	**AccuracyRanges**	list[MeasurementAccuracyRangeStruct]	min 1	F		R	M

2.1.4.4.1. MeasurementType Field

This field SHALL indicate the type of measurement for the accuracy provided.

2.1.4.4.2. Measured Field

This field SHALL indicate whether the associated measurement was directly measured. If this field is not set to true, then the associated measurement was estimated.

2.1.4.4.3. MinMeasuredValue Attribute

This field SHALL indicate the minimum value that can be measured.

2.1.4.4.4. MaxMeasuredValue Attribute

This field SHALL indicate the maximum value that can be measured.

2.1.4.4.5. AccuracyRanges Field

This field SHALL indicate a list of measurement ranges and their associated accuracies.

The value of the RangeMin field on the first MeasurementAccuracyRangeStruct in this list SHALL be equal to the value of the MinMeasuredValue field.

The value of the RangeMax field on the last MeasurementAccuracyRangeStruct in this list SHALL be less than or equal to the value of the MaxMeasuredValue field.

The value of the RangeMin field on each MeasurementAccuracyRangeStruct in this list other than the first SHALL be one more the value of the RangeMax field on the previous MeasurementAccuracyRangeStruct in this list (i.e. there SHALL be no gaps in the accuracy ranges, and the ranges SHALL be in increasing order).

2.2. Illuminance Measurement Cluster

The Illuminance Measurement cluster provides an interface to illuminance measurement functionality, including configuration and provision of notifications of illuminance measurements.

2.2.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Revision	Description
1	Mandatory global ClusterRevision attribute added; CCB 2048 2049 2050
2	CCB 2167
3	New data model format and notation

2.2.2. Classification

Hierarchy	Role	Scope	PICS Code
Base	Application	Endpoint	ILL

2.2.3. Cluster ID

ID	Name
0x0400	Illuminance Measurement

2.2.4. Data Types

2.2.4.1. LightSensorTypeEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**Photodiode**	Indicates photodiode sensor type	M
1	**CMOS**	Indicates CMOS sensor type	M
64 to 254	**MS**	Reserved for manufacturer specific light sensor types	O

2.2.5. Attributes

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0000	**MeasuredValue**	uint16	0, MinMeasuredValue to MaxMeasuredValue	PX	0	R V	M
0x0001	**MinMeasuredValue**	uint16	1 to MaxMeasuredValue-1	X		R V	M
0x0002	**MaxMeasuredValue**	uint16	MinMeasuredValue+1 to 65534	X		R V	M
0x0003	**Tolerance**	uint16	0 to 2048			R V	O
0x0004	**LightSensorType**	LightSensorType Enum	all	X	null	R V	O

2.2.5.1. MeasuredValue Attribute

The MeasuredValue attribute represents the illuminance in Lux (symbol lx) as follows:

- MeasuredValue = 10,000 x $\log_{10}$(illuminance) + 1,

where 1 lx <= illuminance <= 3.576 Mlx, corresponding to a MeasuredValue in the range 1 to 0xFFFE.

The MeasuredValue attribute can take the following values:

- 0 indicates a value of illuminance that is too low to be measured,
- MinMeasuredValue <= MeasuredValue <= MaxMeasuredValue under normal circumstances,
- null indicates that the illuminance measurement is invalid.

The MeasuredValue attribute is updated continuously as new measurements are made.

2.2.5.2. MinMeasuredValue Attribute

The MinMeasuredValue attribute indicates the minimum value of MeasuredValue that can be measured. A value of null indicates that this attribute is not defined. See Measured Value for more details.

2.2.5.3. MaxMeasuredValue Attribute

The MaxMeasuredValue attribute indicates the maximum value of MeasuredValue that can be measured. A value of null indicates that this attribute is not defined. See Measured Value for more details.

2.2.5.4. Tolerance Attribute

See Measured Value.

2.2.5.5. 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 LightSensorTypeEnum or null in case the sensor type is unknown.

2.3. Temperature Measurement Cluster

This cluster provides an interface to temperature measurement functionality, including configuration and provision of notifications of temperature measurements.

2.3.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Revision	Description
1	Mandatory global ClusterRevision attribute added
2	CCB 2241 2370
3	CCB 2823
4	New data model format and notation

2.3.2. Classification

Hierarchy	Role	Scope	PICS Code
Base	Application	Endpoint	TMP

2.3.3. Cluster ID

ID	Name
0x0402	Temperature Measurement

2.3.4. Attributes

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0000	**MeasuredValue**	temperature	MinMeasuredValue to MaxMeasuredValue	XP		R V	M
0x0001	**MinMeasuredValue**	temperature	-27315 to MaxMeasuredValue-1	X		R V	M
0x0002	**MaxMeasuredValue**	temperature	MinMeasuredValue+1 to 32767	X		R V	M
0x0003	**Tolerance**	uint16	0 to 2048		0	R V	O

2.3.4.1. MeasuredValue Attribute

This attribute SHALL indicate the measured temperature.

The null value indicates that the temperature is unknown.

2.3.4.2. MinMeasuredValue Attribute

This attribute SHALL indicate the minimum value of MeasuredValue that is capable of being measured. See Measured Value for more details.

The null value indicates that the value is not available.

2.3.4.3. MaxMeasuredValue Attribute

This attribute indicates the maximum value of MeasuredValue that is capable of being measured. See Measured Value for more details.

The null value indicates that the value is not available.

2.3.4.4. Tolerance Attribute

See Measured Value.

2.4. Pressure Measurement Cluster

This cluster provides an interface to pressure measurement functionality, including configuration and provision of notifications of pressure measurements.

2.4.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Revision	Description
1	Mandatory global ClusterRevision attribute added
2	CCB 2241 2370
3	New data model format and notation

2.4.2. Classification

Hierarchy	Role	Scope	PICS Code
Base	Application	Endpoint	PRS

2.4.3. Cluster ID

ID	Name
0x0403	Pressure Measurement

2.4.4. Features

This cluster SHALL support the FeatureMap bitmap attribute as defined below.

Bit	Code	Feature	Conformance	Summary
0	**EXT**	Extended	O	Extended range and resolution

2.4.5. Attributes

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0000	**MeasuredValue**	int16	MinMeasuredValue to MaxMeasuredValue	XP		R V	M

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0001	**MinMeasuredValue**	int16	-32767 to MaxMeasuredValue-1	X		R V	M
0x0002	**MaxMeasuredValue**	int16	MinMeasuredValue+1 to 32767	X		R V	M
0x0003	**Tolerance**	uint16	0 to 2048		0	R V	O
0x0010	**ScaledValue**	int16	MinScaledValue to MaxScaledValue	X	0	R V	EXT
0x0011	**MinScaledValue**	int16	-32767 to MaxScaledValue-1	X	0	R V	EXT
0x0012	**MaxScaledValue**	int16	MinScaledValue+1 to 32767	X	0	R V	EXT
0x0013	**ScaledTolerance**	uint16	0 to 2048		0	R V	[EXT]
0x0014	**Scale**	int8	-127 to 127		0	R V	EXT

2.4.5.1. MeasuredValue Attribute

This attribute SHALL represent the pressure in kPa as follows:

MeasuredValue = 10 x Pressure [kPa]

The null value indicates that the value is not available.

2.4.5.2. MinMeasuredValue Attribute

This attribute SHALL indicate the minimum value of MeasuredValue that can be measured. See Measured Value for more details.

The null value indicates that the value is not available.

2.4.5.3. MaxMeasuredValue Attribute

This attribute SHALL indicate the maximum value of MeasuredValue that can be measured. See Measured Value for more details.

The null value indicates that the value is not available.

2.4.5.4. Tolerance Attribute

See Measured Value.

2.4.5.5. ScaledValue Attribute

This attribute SHALL represent the pressure in Pascals as follows:

ScaledValue = 10^{Scale} x Pressure [Pa]

The null value indicates that the value is not available.

2.4.5.6. MinScaledValue Attribute

This attribute SHALL indicate the minimum value of ScaledValue that can be measured.

The null value indicates that the value is not available.

2.4.5.7. MaxScaledValue Attribute

This attribute SHALL indicate the maximum value of ScaledValue that can be measured.

The null value indicates that the value is not available.

2.4.5.8. ScaledTolerance Attribute

This attribute SHALL indicate the magnitude of the possible error that is associated with Scaled-Value. The true value is located in the range

(ScaledValue – ScaledTolerance) to (ScaledValue + ScaledTolerance).

2.4.5.9. Scale Attribute

This attribute SHALL indicate the base 10 exponent used to obtain ScaledValue (see ScaledValue).

2.5. Flow Measurement Cluster

This cluster provides an interface to flow measurement functionality, including configuration and provision of notifications of flow measurements.

2.5.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Revision	Description
1	Mandatory global ClusterRevision attribute added
2	CCB 2241 2370

Revision	Description
3	New data model format and notation

2.5.2. Classification

Hierarchy	Role	Scope	PICS Code
Base	Application	Endpoint	FLW

2.5.3. Cluster ID

ID	Name
0x0404	Flow Measurement

2.5.4. Attributes

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0000	**MeasuredValue**	uint16	MinMeasuredValue to MaxMeasuredValue	XP	null	R V	M
0x0001	**MinMeasuredValue**	uint16	0 to MaxMeasuredValue-1	X		R V	M
0x0002	**MaxMeasuredValue**	uint16	MinMeasuredValue+1 to 65534	X		R V	M
0x0003	**Tolerance**	uint16	0 to 2048		0	R V	O

2.5.4.1. MeasuredValue Attribute

MeasuredValue represents the flow in m^3/h as follows:

MeasuredValue = 10 x Flow

The null value indicates that the flow measurement is unknown, otherwise the range SHALL be as described in Measured Value.

2.5.4.2. MinMeasuredValue Attribute

The MinMeasuredValue attribute indicates the minimum value of MeasuredValue that can be measured. See Measured Value for more details.

The null value indicates that the value is not available.

2.5.4.3. MaxMeasuredValue Attribute

The MaxMeasuredValue attribute indicates the maximum value of MeasuredValue that can be measured. See Measured Value for more details.

The null value indicates that the value is not available.

2.5.4.4. Tolerance Attribute

See Measured Value.

2.6. Water Content Measurement Clusters

This is a base cluster. The server cluster provides an interface to water content measurement functionality. The measurement is reportable and may be configured for reporting. Water content measurements currently is, but are not limited to relative humidity.

2.6.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Revision	Description
1	Mandatory global ClusterRevision attribute added
2	CCB 2241
3	New data model format and notation

2.6.2. Classification

Hierarchy	Role	Scope	PICS Code
Base	Application	Endpoint	RH

2.6.3. Cluster ID

ID	Name	Measurement Type
0x0405	Relative Humidity Measurement	Percentage of water in the air

2.6.4. Attributes

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0000	**MeasuredValue**	uint16	MinMeasuredValue to MaxMeasuredValue	XP		R V	M
0x0001	**MinMeasuredValue**	uint16	0 to MaxMeasuredValue-1	X		R V	M
0x0002	**MaxMeasuredValue**	uint16	MinMeasuredValue+1 to 10000	X		R V	M
0x0003	**Tolerance**	uint16	0 to 2048			R V	O

2.6.4.1. MeasuredValue Attribute

MeasuredValue represents the water content in % as follows:

MeasuredValue = 100 x water content

Where 0% < = water content < = 100%, corresponding to a MeasuredValue in the range 0 to 10000.

The maximum resolution this format allows is 0.01%.

MinMeasuredValue and MaxMeasuredValue define the range of the sensor.

The null value indicates that the measurement is unknown, otherwise the range SHALL be as described in Measured Value.

MeasuredValue is updated continuously as new measurements are made.

2.6.4.2. MinMeasuredValue Attribute

The MinMeasuredValue attribute indicates the minimum value of MeasuredValue that can be measured. The null value means this attribute is not defined. See Measured Value for more details.

2.6.4.3. MaxMeasuredValue Attribute

The MaxMeasuredValue attribute indicates the maximum value of MeasuredValue that can be measured. The null value means this attribute is not defined. See Measured Value for more details.

2.6.4.4. Tolerance Attribute

See Measured Value.

2.7. Occupancy Sensing Cluster

The server cluster provides an interface to occupancy sensing functionality, including configuration and provision of notifications of occupancy status.

2.7.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Revision	Description
1	Mandatory global ClusterRevision attribute added
2	Physical Contact Occupancy feature with mandatory OccupancySensorTypeBitmap
3	New data model format and notation
4	Remove nullable from PhysicalContact delay attributes and ability to not report transitions

2.7.2. Classification

Hierarchy	Role	Scope	PICS Code
Base	Application	Endpoint	OCC

2.7.3. Cluster ID

ID	Name
0x0406	Occupancy Sensing

2.7.4. Data Types

2.7.4.1. OccupancyBitmap Type

This data type is derived from map8.

Bit	Name	Summary	Conformance
0	**Occupied**	Indicates the sensed occupancy state	M

2.7.4.1.1. Occupied Bit

If this bit is set, it SHALL indicate the occupied state else if the bit if not set, it SHALL indicate the unoccupied state.

2.7.4.2. OccupancySensorTypeBitmap Type

This data type is derived from map8.

Bit	Name	Summary	Conformance
0	**PIR**	Indicates a passive infrared sensor.	M
1	**Ultrasonic**	Indicates a ultrasonic sensor.	M
2	**PhysicalContact**	Indicates a physical contact sensor.	M

2.7.4.3. OccupancySensorTypeEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**PIR**	Indicates a passive infrared sensor.	M
1	**Ultrasonic**	Indicates a ultrasonic sensor.	M
2	**PIRAndUltrasonic**	Indicates a passive infrared and ultrasonic sensor.	M
3	**PhysicalContact**	Indicates a physical contact sensor.	M

2.7.5. Attributes

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0000	**Occupancy**	OccupancyBitmap	0 to 1	P		R V	M
0x0001	**OccupancySensorType**	OccupancySensorTypeEnum	desc		MS	R V	M
0x0002	**OccupancySensorTypeBitmap**	OccupancySensorTypeBitmap	0 to 7			R V	M

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0010	**PIROccupiedToUnoccupiedDelay**	uint16	all		0	RW VM	O
0x0011	**PIRUnoccupiedToOccupiedDelay**	uint16	all		0	RW VM	PIRUnoccupiedToOccupiedThreshold, O
0x0012	**PIRUnoccupiedToOccupiedThreshold**	uint8	1 to 254		1	RW VM	PIRUnoccupiedToOccupiedDelay, O
0x0020	**UltrasonicOccupiedToUnoccupiedDelay**	uint16	all		0	RW VM	O
0x0021	**UltrasonicUnoccupiedToOccupiedDelay**	uint16	all		0	RW VM	UltrasonicUnoccupiedToOccupiedThreshold, O
0x0022	**UltrasonicUnoccupiedToOccupiedThreshold**	uint8	1 to 254		1	RW VM	UltrasonicUnoccupiedToOccupiedDelay, O
0x0030	**PhysicalContactOccupiedToUnoccupiedDelay**	uint16	all		0	RW VM	O
0x0031	**PhysicalContactUnoccupiedToOccupiedDelay**	uint16	all		0	RW VM	O

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0032	**PhysicalContactUnoccupiedToOccupiedThreshold**	uint8	1 to 254		1	RW VM	PhysicalContactUnoccupiedToOccupiedDelay, O

2.7.5.1. Occupancy Attribute

This attribute indicates the sensed (processed) status of occupancy.

2.7.5.2. OccupancySensorType Attribute

This attribute specifies the type of the occupancy sensor.

2.7.5.3. OccupancySensorTypeBitmap Attribute

This attribute specifies the types of the occupancy sensor. Each bit position, if set, indicates the corresponding sensing capability is implemented.

The value of the OccupancySensorType SHALL be aligned to the value of the OccupancySensorTypeBitmap attribute as defined below.

OccupancySensorTypeBitmap	OccupancySensorType
PIR	PIR
Ultrasonic	Ultrasonic
PIR + Ultrasonic	PIRAndUltrasonic
PhysicalContact	PhysicalContact
PhysicalContact + PIR	PIR
PhysicalContact + Ultrasonic	Ultrasonic
PhysicalContact + PIR + Ultrasonic	PIRAndUltrasonic

2.7.5.4. PIROccupiedToUnoccupiedDelay Attribute

This attribute 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.

2.7.5.5. PIRUnoccupiedToOccupiedDelay Attribute

This attribute specifies the time delay, in seconds, before the PIR sensor changes to its occupied state after the detection of movement in the sensed area.

2.7.5.6. PIRUnoccupiedToOccupiedThreshold Attribute

This attribute specifies the number of movement detection events that must occur in the period PIRUnoccupiedToOccupiedDelay, before the PIR sensor changes to its occupied state.

2.7.5.7. UltrasonicOccupiedToUnoccupiedDelay Attribute

This attribute 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.

2.7.5.8. UltrasonicUnoccupiedToOccupiedDelay Attribute

This attribute specifies the time delay, in seconds, before the Ultrasonic sensor changes to its occupied state after the detection of movement in the sensed area.

2.7.5.9. UltrasonicUnoccupiedToOccupiedThreshold Attribute

This attribute specifies the number of movement detection events that must occur in the period UltrasonicUnoccupiedToOccupiedDelay, before the Ultrasonic sensor changes to its occupied state.

2.7.5.10. PhysicalContactOccupiedToUnoccupiedDelay Attribute

This attribute specifies the time delay, in seconds, before the physical contact occupancy sensor changes to its unoccupied state after detecting the unoccupied event.

2.7.5.11. PhysicalContactUnoccupiedToOccupiedDelay Attribute

This attribute specifies the time delay, in seconds, before the physical contact sensor changes to its occupied state after the detection of the occupied event.

2.7.5.12. PhysicalContactUnoccupiedToOccupiedThreshold Attribute

This attribute specifies the number of movement detection events that must occur in the period PhysicalContactUnoccupiedToOccupiedDelay, before the PhysicalContact sensor changes to its occupied state.

2.8. Resource Monitoring Clusters

This generic cluster provides an interface to the current condition of a resource. A resource is a component of a device that is designed to be replaced, refilled, or emptied when exhausted or full. Examples of resources include filters, cartridges, and water tanks. While batteries fit this definition they are not intended to be used with this cluster. Use the power source cluster for batteries instead.

NOTE	This cluster is not meant to be used for monitoring of the system resources, such as processing, memory utilization, networking properties, etc.

This cluster SHALL be used via an alias to a specific resource type (see Cluster IDs).

2.8.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Revision	Description
1	Initial version of the Resource Monitoring cluster

2.8.2. Classification

Hierarchy	Role	Scope	PICS Code
Base	Application	Endpoint	REPM

2.8.3. Cluster IDs

The table below is a list of aliased Cluster IDs which represent different resource types and conform to this cluster definition.

ID	Name	Resource Type	PICS Code
0x0071	HEPA Filter Monitoring	HEPA Filter	HEPAFREMON
0x0072	Activated Carbon Filter Monitoring	Activated Carbon Filter	ACFREMON

2.8.4. Features

This cluster SHALL support the FeatureMap bitmap attribute as defined below.

Bit	Code	Feature	Conformance	Summary
0	**CON**	Condition	O	Supports monitoring the condition of the resource in percentage
1	**WRN**	Warning	O	Supports warning indication
2	**REP**	Replacement Product List	O	Supports specifying the list of replacement products

2.8.5. Data Types

2.8.5.1. DegradationDirectionEnum Type

This data type is derived from enum8.

Indicates the direction in which the condition of the resource changes over time.

Value	Name	Summary	Conformance
0	**Up**	The degradation of the resource is indicated by an upwards moving/increasing value	M
1	**Down**	The degradation of the resource is indicated by a downwards moving/decreasing value	M

2.8.5.2. ChangeIndicationEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**OK**	Resource is in good condition, no intervention required	M
1	**Warning**	Resource will be exhausted soon, intervention will shortly be required	WRN
2	**Critical**	Resource is exhausted, immediate intervention is required	M

2.8.5.3. ProductIdentifierTypeEnum Type

This data type is derived from enum8.

Indicate the type of identifier used to describe the product. Devices SHOULD use globally-recognized IDs over OEM specific ones.

Value	Name	Summary	Conformance
0	**UPC**	12-digit Universal Product Code	M
1	**GTIN-8**	8-digit Global Trade Item Number	M
2	**EAN**	13-digit European Article Number	M
3	**GTIN-14**	14-digit Global Trade Item Number	M

Value	Name	Summary	Conformance
4	**OEM**	Original Equipment Manufacturer part number	M

2.8.5.4. ReplacementProductStruct Type

Indicates the product identifier that can be used as a replacement for the resource.

ID	Name	Type	Constraint	Quality	Access	Default	Conformance
0	**ProductIdentifierType**	ProductIdentifierTypeEnum	desc				M
1	**ProductIdentifierValue**	string	max 20				M

2.8.6. Attributes

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0000	**Condition**	percent				R V	CON
0x0001	**DegradationDirection**	DegradationDirectionEnum	desc	F		R V	CON
0x0002	**ChangeIndication**	ChangeIndicationEnum			0	R V	M
0x0003	**InPlaceIndicator**	bool				R V	O
0x0004	**LastChangedTime**	epoch-s	all	X N	null	RW VO	O
0x0005	**ReplacementProductList**	list[ReplacementProductStruct]	max 5	F		R V	REP

2.8.6.1. Condition Attribute

This attribute SHALL indicate the current condition of the resource in percent.

2.8.6.2. DegradationDirection Attribute

This attribute SHALL indicate the direction of change for the condition of the resource over time, which helps to determine whether a higher or lower condition value is considered optimal.

2.8.6.3. ChangeIndication Attribute

This attribute SHALL be populated with a value from ChangeIndicationEnum that is indicative of the current requirement to change the resource.

2.8.6.4. InPlaceIndicator Attribute

This attribute SHALL indicate whether a resource is currently installed. A value of true SHALL indicate that a resource is installed. A value of false SHALL indicate that a resource is not installed.

2.8.6.5. LastChangedTime Attribute

This attribute MAY indicates the time at which the resource has been changed, if supported by the server. The attribute SHALL be null if it was never set or is unknown.

2.8.6.6. ReplacementProductList Attribute

This attribute SHALL indicate the list of supported products that may be used as replacements for the current resource. Each item in this list represents a unique ReplacementProductStruct.

2.8.7. Commands

ID	Name	Direction	Response	Access	Conformance
0x00	**ResetCondition**	client ⇒ server	Y	O	O

2.8.7.1. ResetCondition Command

Upon receipt, the device SHALL reset the Condition and ChangeIndicator attributes, indicating full resource availability and readiness for use, as initially configured. Invocation of this command MAY cause the LastChangedTime to be updated automatically based on the clock of the server, if the server supports setting the attribute.

2.9. Air Quality Cluster

This cluster provides an interface to air quality classification using distinct levels with human-readable labels.

2.9.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Revision	Description
1	Initial version of the Air Quality cluster

2.9.2. Classification

Hierarchy	Role	Scope	PICS Code
Base	Application	Endpoint	AIRQUAL

2.9.3. Cluster ID

ID	Name
0x005B	Air Quality

2.9.4. Features

This cluster SHALL support the FeatureMap bitmap attribute as defined below.

Bit	Code	Feature	Conformance	Summary
0	**FAIR**	Fair	O	Cluster supports the Fair air quality level
1	**MOD**	Moderate	O	Cluster supports the Moderate air quality level
2	**VPOOR**	VeryPoor	O	Cluster supports the Very poor air quality level
3	**XPOOR**	ExtremelyPoor	O	Cluster supports the Extremely poor air quality level

2.9.5. Data Types

2.9.5.1. AirQualityEnum Type

This data type is derived from enum8.

The AirQualityEnum provides a representation of the quality of the analyzed air. It is up to the device manufacturer to determine the mapping between the measured values and their corresponding enumeration values.

Value	Name	Summary	Conformance
0	**Unknown**	The air quality is unknown.	M
1	**Good**	The air quality is good.	M
2	**Fair**	The air quality is fair.	FAIR
3	**Moderate**	The air quality is moderate.	MOD
4	**Poor**	The air quality is poor.	M
5	**VeryPoor**	The air quality is very poor.	VPOOR
6	**ExtremelyPoor**	The air quality is extremely poor.	XPOOR

2.9.6. Attributes

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0000	**AirQuality**	AirQualityEnum	desc		0	R V	M

2.9.6.1. AirQuality Attribute

This attribute SHALL indicate a value from AirQualityEnum that is indicative of the currently measured air quality.

2.10. Concentration Measurement Clusters

The server cluster provides an interface to concentration measurement functionality.

This cluster SHALL to be used via an alias to a specific substance (see Cluster IDs).

2.10.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Revision	Description
1	Mandatory global ClusterRevision attribute added
2	CCB 2882
3	Cluster redesigned to add support for Level Indication, Peak/Average Measurement, Medium/Unit of Measurement and Uncertainty.

2.10.2. Classification

Hierarchy	Role	Scope	PICS Code
Base	Application	Endpoint	CONC

2.10.3. Cluster IDs

The table below is a list of Cluster IDs that conform to this specification.

ID	Name	Substance Measured	PICS Code
0x040C	Carbon Monoxide Concentration Measurement	Carbon Monoxide (CO)	CMOCONC
0x040D	Carbon Dioxide Concentration Measurement	Carbon Dioxide (CO_2)	CDOCONC
0x0413	Nitrogen Dioxide Concentration Measurement	Nitrogen Dioxide (NO_2)	NDOCONC
0x0415	Ozone Concentration Measurement	Ozone (O_3)	OZCONC
0x042A	PM2.5 Concentration Measurement	PM2.5	PMICONC
0x042B	Formaldehyde Concentration Measurement	Formaldehyde (CH2O)	FLDCONC
0x042C	PM1 Concentration Measurement	PM1	PMHCONC
0x042D	PM10 Concentration Measurement	PM10	PMKCONC
0x042E	Total Volatile Organic Compounds Concentration Measurement	Total Volatile Organic Compounds (TVOC)	TVOCCONC
0x042F	Radon Concentration Measurement	Radon (Rn)	RNCONC

2.10.4. Features

This cluster SHALL support the FeatureMap bitmap attribute as defined below.

Bit	Code	Feature	Conformance	Summary
0	**MEA**	NumericMeasurement	O.a+	Cluster supports numeric measurement of substance

Bit	Code	Feature	Conformance	Summary
1	**LEV**	LevelIndication	O.a+	Cluster supports basic level indication for substance using the ConcentrationLevel enum
2	**MED**	MediumLevel	[LEV]	Cluster supports the Medium Concentration Level
3	**CRI**	CriticalLevel	[LEV]	Cluster supports the Critical Concentration Level
4	**PEA**	PeakMeasurement	[MEA]	Cluster supports peak numeric measurement of substance
5	**AVG**	AverageMeasurement	[MEA]	Cluster supports average numeric measurement of substance

2.10.5. Data Types

2.10.5.1. MeasurementUnitEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**PPM**	Parts per Million (10^6)	MEA
1	**PPB**	Parts per Billion (10^9)	MEA
2	**PPT**	Parts per Trillion (10^{12})	MEA
3	**MGM3**	Milligram per m^3	MEA
4	**UGM3**	Microgram per m^3	MEA
5	**NGM3**	Nanogram per m^3	MEA
6	**PM3**	Particles per m^3	MEA
7	**BQM3**	Becquerel per m^3	MEA

Where mentioned, Billion refers to 10^9, Trillion refers to 10^{12} (short scale).

2.10.5.2. MeasurementMediumEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**Air**	The measurement is being made in Air	M
1	**Water**	The measurement is being made in Water	M
2	**Soil**	The measurement is being made in Soil	M

2.10.5.3. LevelValueEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**Unknown**	The level is Unknown	M
1	**Low**	The level is considered Low	M
2	**Medium**	The level is considered Medium	MED
3	**High**	The level is considered High	M
4	**Critical**	The level is considered Critical	CRI

2.10.6. Attributes

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0000	**MeasuredValue**	single	MinMeasuredValue to MaxMeasuredValue	X P	null	R V	MEA
0x0001	**MinMeasuredValue**	single	max MaxMeasuredValue	X	null	R V	MEA
0x0002	**MaxMeasuredValue**	single	min MinMeasuredValue	X	null	R V	MEA
0x0003	**PeakMeasuredValue**	single	MinMeasuredValue to MaxMeasuredValue	X P	null	R V	PEA

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0004	**PeakMeasuredValueWindow**	elapsed-s	max 604800	P	1	R V	PEA
0x0005	**AverageMeasuredValue**	single	MinMeasuredValue to MaxMeasuredValue	X P	null	R V	AVG
0x0006	**AverageMeasuredValueWindow**	elapsed-s	max 604800	P	1	R V	AVG
0x0007	**Uncertainty**	single	MS		MS	R V	[MEA]
0x0008	**MeasurementUnit**	MeasurementUnitEnum		F	MS	R V	MEA
0x0009	**MeasurementMedium**	MeasurementMediumEnum		F	MS	R V	M
0x000A	**LevelValue**	LevelValueEnum			0	R V	LEV

2.10.6.1. MeasuredValue Attribute

This attribute SHALL represent the most recent measurement as a single-precision floating-point number. MeasuredValue's unit is represented by MeasurementUnit.

A value of null indicates that the measurement is unknown or outside the valid range.

MinMeasuredValue and MaxMeasuredValue define the valid range for MeasuredValue.

2.10.6.2. MinMeasuredValue Attribute

This attribute SHALL represent the minimum value of MeasuredValue that is capable of being measured. A MinMeasuredValue of null indicates that the MinMeasuredValue is not defined.

2.10.6.3. MaxMeasuredValue Attribute

This attribute SHALL represent the maximum value of MeasuredValue that is capable of being measured. A MaxMeasuredValue of null indicates that the MaxMeasuredValue is not defined.

2.10.6.4. PeakMeasuredValue Attribute

This attribute SHALL represent the maximum value of MeasuredValue that has been measured during the PeakMeasuredValueWindow. If this attribute is provided, the PeakMeasuredValueWindow attribute SHALL also be provided.

2.10.6.5. PeakMeasuredValueWindow Attribute

This attribute SHALL represent the window of time used for determining the PeakMeasuredValue. The value is in seconds.

2.10.6.6. AverageMeasuredValue Attribute

This attribute SHALL represent the average value of MeasuredValue that has been measured during the AverageMeasuredValueWindow. If this attribute is provided, the AverageMeasuredValueWindow attribute SHALL also be provided.

2.10.6.7. AverageMeasuredValueWindow Attribute

This attribute SHALL represent the window of time used for determining the AverageMeasuredValue. The value is in seconds.

2.10.6.8. Uncertainty Attribute

This attribute SHALL represent the range of error or deviation that can be found in MeasuredValue and PeakMeasuredValue. This is considered a +/- value and should be considered to be in MeasurementUnit.

2.10.6.9. MeasurementUnit Attribute

This attribute SHALL represent the unit of MeasuredValue. See MeasurementUnitEnum.

2.10.6.10. MeasurementMedium Attribute

This attribute SHALL represent the medium in which MeasuredValue is being measured. See MeasurementMediumEnum.

2.10.6.11. LevelValue Attribute

This attribute SHALL represent the level of the substance detected. See LevelValueEnum.

2.11. Smoke CO Alarm Cluster

This cluster provides an interface for observing and managing the state of smoke and CO alarms.

2.11.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Revision	Description
1	Initial Release

2.11.2. Classification

Hierarchy	Role	Scope	PICS Code
Base	Application	Endpoint	SMOKECO

2.11.3. Cluster ID

ID	Name
0x005C	Smoke CO Alarm

2.11.4. Features

This cluster SHALL support the FeatureMap bitmap attribute as defined below.

Bit	Code	Feature	Conformance	Summary
0	**SMOKE**	SmokeAlarm	O.a+	Supports Smoke alarm
1	**CO**	COAlarm	O.a+	Supports CO alarm

2.11.5. Data Types

2.11.5.1. AlarmStateEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**Normal**	Nominal state, the device is not alarming	M
1	**Warning**	Warning state	O
2	**Critical**	Critical state	M

2.11.5.1.1. Normal Value

This value SHALL indicate that this alarm is not alarming.

2.11.5.1.2. Warning Value

This value SHALL indicate that this alarm is in a warning state. Alarms in this state SHOULD be subject to being muted via physical interaction.

2.11.5.1.3. Critical Value

This value SHALL indicate that this alarm is in a critical state. Alarms in this state SHALL NOT be subject to being muted via physical interaction.

2.11.5.2. SensitivityEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**High**	High sensitivity	O
1	**Standard**	Standard Sensitivity	M
2	**Low**	Low sensitivity	O

2.11.5.3. ExpressedStateEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**Normal**	Nominal state, the device is not alarming	M
1	**SmokeAlarm**	Smoke Alarm state	SMOKE
2	**COAlarm**	CO Alarm state	CO
3	**BatteryAlert**	Battery Alert State	M
4	**Testing**	Test in Progress	M
5	**HardwareFault**	Hardware Fault Alert State	M
6	**EndOfService**	End of Service Alert State	M
7	**InterconnectSmoke**	Interconnected Smoke Alarm State	O
8	**InterconnectCO**	Interconnected CO Alarm State	O

2.11.5.3.1. Normal Value

This value SHALL indicate that this alarm is not alarming.

2.11.5.3.2. SmokeAlarm Value

This value SHALL indicate that this alarm is currently expressing visual indication of Smoke Alarm. This value SHALL indicate that the alarm is currently expressing audible indication of Smoke Alarm unless the DeviceMuted attribute is supported and set to Muted.

2.11.5.3.3. COAlarm Value

This value SHALL indicate that this alarm is currently expressing visual indication of CO Alarm. This value SHALL indicate that the alarm is currently expressing audible indication of CO Alarm unless the DeviceMuted attribute is supported and set to Muted.

2.11.5.3.4. BatteryAlert Value

This value SHALL indicate that this alarm is currently expressing visual indication of Critical Low Battery. This value SHALL indicate that the alarm is currently expressing audible indication of Critical Low Battery unless the DeviceMuted attribute is supported and set to Muted.

2.11.5.3.5. Testing Value

This value SHALL indicate that this alarm is currently expressing visual and audible indication of SelfTest.

2.11.5.3.6. HardwareFault Value

This value SHALL indicate that this alarm is currently expressing visual indication of Hardware Fault. This value SHALL indicate that the alarm is currently expressing audible indication of Hardware Fault unless the DeviceMuted attribute is supported and set to Muted.

2.11.5.3.7. EndOfService Value

This value SHALL indicate that this alarm is currently expressing visual indication of End Of Service. This value SHALL indicate that the alarm is currently expressing audible indication of End of Service unless the DeviceMuted attribute is supported and set to Muted.

2.11.5.3.8. InterconnectSmoke Value

This value SHALL indicate that this alarm is currently expressing visual indication of Smoke Alarm caused by Interconnect. This value SHALL indicate that the alarm is currently expressing audible indication of Smoke Alarm caused by Interconnect unless the DeviceMuted attribute is supported and set to Muted.

2.11.5.3.9. InterconnectCO Value

This value SHALL indicate that this alarm is currently expressing visual indication of CO Alarm caused by Interconnect. This value SHALL indicate that the alarm is currently expressing audible indication of CO Alarm caused by Interconnect unless the DeviceMuted attribute is supported and set to Muted.

2.11.5.4. MuteStateEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**NotMuted**	Not Muted	M
1	**Muted**	Muted	M

2.11.5.4.1. NotMuted Value

This value SHALL indicate that the device is not muted.

2.11.5.4.2. Muted Value

This value SHALL indicate that the device is muted.

2.11.5.5. EndOfServiceEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**Normal**	Device has not expired	M
1	**Expired**	Device has reached its end of service	M

2.11.5.5.1. Expired Value

This value SHALL indicate that the device has reached its end of service, and needs to be replaced.

2.11.5.5.2. Normal Value

This value SHALL indicate that the device has not yet reached its end of service, and does not need to be imminently replaced.

2.11.5.6. ContaminationStateEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**Normal**	Nominal state, the sensor is not contaminated	M
1	**Low**	Low contamination	O
2	**Warning**	Warning state	O
3	**Critical**	Critical state, will cause nuisance alarms	M

2.11.5.6.1. Normal Value

This value SHALL indicate that the smoke sensor has nominal contamination levels, no customer action is required.

2.11.5.6.2. Low Value

This value SHALL indicate that the smoke sensor has detectable contamination levels, but the contamination is too low to cause a visible or audible alarm.

2.11.5.6.3. Warning Value

This value SHALL indicate that the smoke sensor has contamination levels in a warning state. At this level, the contamination may cause a visible or audible alarm. User intervention is suggested.

2.11.5.6.4. Critical Value

This value SHALL indicate that the smoke sensor has contamination levels in a critical state. At this level, the contamination should cause a visible or audible alarm. User intervention is required. Critical contamination of the sensor SHALL also be reflected as a HardwareFault.

2.11.6. Attributes

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0000	**ExpressedState**	ExpressedStateEnum	all	N		R V	M
0x0001	**SmokeState**	AlarmStateEnum	all	N		R V	SMOKE
0x0002	**COState**	AlarmStateEnum	all	N		R V	CO
0x0003	**BatteryAlert**	AlarmStateEnum	all	N		R V	M
0x0004	**DeviceMuted**	MuteStateEnum	all	N		R V	O
0x0005	**TestInProgress**	bool	all			R V	M
0x0006	**HardwareFaultAlert**	bool	all	N		R V	M
0x0007	**EndOfServiceAlert**	EndOfServiceEnum	all	N		R V	M
0x0008	**InterconnectSmokeAlarm**	AlarmStateEnum	all			R V	O
0x0009	**InterconnectCOAlarm**	AlarmStateEnum	all			R V	O
0x000A	**ContaminationState**	ContaminationStateEnum	all			R V	[SMOKE]
0x000B	**SmokeSensitivityLevel**	SensitivityEnum	all			RW VM	[SMOKE]

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x000C	**ExpiryDate**	epoch-s	all	F		R V	O

2.11.6.1. ExpressedState Attribute

This attribute SHALL indicate the visibly- and audibly-expressed state of the alarm. When multiple alarm conditions are being reflected in the server, this attribute SHALL indicate the condition with the highest priority. Priority order of conditions is determined by the manufacturer and SHALL be supplied as a part of certification procedure. If the value of ExpressedState is not Normal, the attribute corresponding to the value SHALL NOT be Normal. For example, if the ExpressedState is set to SmokeAlarm, the value of the SmokeState will indicate the severity of the alarm (Warning or Critical). Clients SHOULD also read the other attributes to be aware of further alarm conditions beyond the one indicated in ExpressedState.

Visible expression is typically a LED light pattern. Audible expression is a horn or speaker pattern. Audible expression SHALL BE suppressed if the DeviceMuted attribute is supported and set to Muted.

2.11.6.2. SmokeState Attribute

This attribute SHALL indicate whether the device’s smoke sensor is currently triggering a smoke alarm.

2.11.6.3. COState Attribute

This attribute SHALL indicate whether the device’s CO sensor is currently triggering a CO alarm.

2.11.6.4. BatteryAlert Attribute

This attribute SHALL indicate whether the power resource fault detection mechanism is currently triggered at the device. If the detection mechanism is triggered, this attribute SHALL be set to Warning or Critical, otherwise it SHALL be set to Normal. The battery state SHALL also be reflected in the Power Source cluster representing the device’s battery using the appropriate supported attributes and events.

2.11.6.5. DeviceMuted Attribute

This attribute SHALL indicate the whether the audible expression of the device is currently muted. Audible expression is typically a horn or speaker pattern.

2.11.6.6. TestInProgress Attribute

This attribute SHALL indicate whether the device self-test is currently activated. If the device self-test is activated, this attribute SHALL be set to True, otherwise it SHALL be set to False.

2.11.6.7. HardwareFaultAlert Attribute

This attribute SHALL indicate whether the hardware fault detection mechanism is currently triggered. If the detection mechanism is triggered, this attribute SHALL be set to True, otherwise it SHALL be set to False.

2.11.6.8. EndOfServiceAlert Attribute

This attribute SHALL indicate whether the end-of-service has been triggered at the device. This attribute SHALL be set to Expired when the device reaches the end-of-service.

2.11.6.9. InterconnectSmokeAlarm Attribute

This attribute SHALL indicate whether the interconnected smoke alarm is currently triggering by branching devices. When the interconnected smoke alarm is being triggered, this attribute SHALL be set to Warning or Critical, otherwise it SHALL be set to Normal.

2.11.6.10. InterconnectCOAlarm Attribute

This attribute SHALL indicate whether the interconnected CO alarm is currently triggering by branching devices. When the interconnected CO alarm is being triggered, this attribute SHALL be set to Warning or Critical, otherwise it SHALL be set to Normal.

2.11.6.11. ContaminationState Attribute

This attribute SHALL indicate the contamination level of the smoke sensor.

2.11.6.12. SmokeSensitivityLevel Attribute

This attribute SHALL indicate the sensitivity level of the smoke sensor configured on the device.

2.11.6.13. ExpiryDate Attribute

This attribute SHALL indicate the date when the device reaches its stated expiry date. After the ExpiryDate has been reached, the EndOfServiceAlert SHALL start to be triggered. To account for better customer experience across time zones, the EndOfServiceAlert MAY be delayed by up to 24 hours after the ExpiryDate. Similarly, clients MAY delay any actions based on the ExpiryDate by up to 24 hours to best align with the local time zone.

2.11.7. Commands

ID	Name	Direction	Response	Access	Conformance
0x00	**SelfTestRequest**	client ⇒ server	Y	O	O

2.11.7.1. SelfTestRequest Command

This command SHALL initiate a device self-test. The return status SHALL indicate whether the test was successfully initiated. Only one SelfTestRequest may be processed at a time. When the value of the ExpressedState attribute is any of SmokeAlarm, COAlarm, Testing, InterconnectSmoke, Inter-

connectCO, the device SHALL NOT execute the self-test, and SHALL return status code BUSY.

Upon successful acceptance of SelfTestRequest, the TestInProgress attribute SHALL be set to True and ExpressedState attribute SHALL be set to Testing. Any faults identified during the test SHALL be reflected in the appropriate attributes and events. Upon completion of the self test procedure, the SelfTestComplete event SHALL be generated, the TestInProgress attribute SHALL be set to False and ExpressedState attribute SHALL be updated to reflect the current state of the server.

2.11.8. Events

ID	Name	Priority	Access	Conformance
0x00	**SmokeAlarm**	CRITICAL	V	SMOKE
0x01	**COAlarm**	CRITICAL	V	CO
0x02	**LowBattery**	INFO	V	M
0x03	**HardwareFault**	INFO	V	M
0x04	**EndOfService**	INFO	V	M
0x05	**SelfTestComplete**	INFO	V	M
0x06	**AlarmMuted**	INFO	V	O
0x07	**MuteEnded**	INFO	V	O
0x08	**InterconnectSmokeAlarm**	CRITICAL	V	[SMOKE]
0x09	**InterconnectCOAlarm**	CRITICAL	V	[CO]
0x0A	**AllClear**	INFO	V	M

2.11.8.1. SmokeAlarm Event

This event SHALL be generated when SmokeState attribute changes to either Warning or Critical state.

The data of this event SHALL contain the following information:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**AlarmSeverityLevel**	AlarmStateEnum	all			M

2.11.8.1.1. AlarmSeverityLevel Field

This field SHALL indicate the current value of the SmokeState attribute.

2.11.8.2. COAlarm Event

This event SHALL be generated when COState attribute changes to either Warning or Critical state.

The data of this event SHALL contain the following information:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**AlarmSeverityLevel**	AlarmStateEnum	all			M

2.11.8.2.1. AlarmSeverityLevel Field

This field SHALL indicate the current value of the COState attribute.

2.11.8.3. LowBattery Event

This event SHALL be generated when BatteryAlert attribute changes to either Warning or Critical state.

The data of this event SHALL contain the following information:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**AlarmSeverityLevel**	AlarmStateEnum	all			M

2.11.8.3.1. AlarmSeverityLevel Field

This field SHALL indicate the current value of the BatteryAlert attribute.

2.11.8.4. HardwareFault Event

This event SHALL be generated when the device detects a hardware fault that leads to setting HardwareFaultAlert to True.

2.11.8.5. EndOfService Event

This event SHALL be generated when the EndOfServiceAlert is set to Expired.

2.11.8.6. SelfTestComplete Event

This event SHALL be generated when the SelfTest completes, and the attribute TestInProgress changes to False.

2.11.8.7. AlarmMuted Event

This event SHALL be generated when the DeviceMuted attribute changes to Muted.

2.11.8.8. MuteEnded Event

This event SHALL be generated when DeviceMuted attribute changes to NotMuted.

2.11.8.9. InterconnectSmokeAlarm Event

This event SHALL be generated when the device hosting the server receives a smoke alarm from an interconnected sensor.

The data of this event SHALL contain the following information:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**AlarmSeverityLevel**	AlarmStateEnum	all			M

2.11.8.9.1. AlarmSeverityLevel Field

This field SHALL indicate the current value of the InterconnectSmokeAlarm attribute.

2.11.8.10. InterconnectCOAlarm Event

This event SHALL be generated when the device hosting the server receives a CO alarm from an interconnected sensor.

The data of this event SHALL contain the following information:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**AlarmSeverityLevel**	AlarmStateEnum	all			M

2.11.8.10.1. AlarmSeverityLevel Field

This field SHALL indicate the current value of the InterconnectCOAlarm attribute.

2.11.8.11. AllClear Event

This event SHALL be generated when ExpressedState attribute returns to Normal state.

2.12. Electrical Energy Measurement Cluster

This cluster provides a mechanism for querying data about the electrical energy imported or provided by the server.

2.12.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Revision	Description
1	Initial Release

2.12.2. Classification

Hierarchy	Role	Scope	PICS Code
Base	Application	Endpoint	EEM

2.12.3. Cluster ID

ID	Name
0x0091	Electrical Energy Measurement

2.12.4. Features

This cluster SHALL support the FeatureMap bitmap attribute as defined below.

Bit	Code	Feature	Conformance	Summary
0	IMPE	ImportedEnergy	O.a+	Measurement of energy imported by the server
1	EXPE	ExportedEnergy	O.a+	Measurement of energy provided by the server
2	CUME	CumulativeEnergy	O.b+	Measurements are cumulative
3	PERE	PeriodicEnergy	O.b+	Measurements are periodic

2.12.4.1. ImportedEnergy Feature

The feature indicates the server is capable of measuring how much energy is imported by the server.

2.12.4.2. ExportedEnergy Feature

The feature indicates the server is capable of measuring how much energy is exported by the server.

2.12.4.3. CumulativeEnergy Feature

The feature indicates the server is capable of measuring how much energy has been imported or exported by the server over the device's lifetime. This measurement MAY start from when a device's firmware is updated to include this feature, when a device's firmware is updated to correct measurement errors, or when a device is factory reset.

2.12.4.4. PeriodicEnergy Feature

The feature indicates the server is capable of measuring how much energy has been imported or exported by the server during a certain period of time. The start and end times for measurement periods SHALL be determined by the server, and MAY represent overlapping periods.

2.12.5. Data Types

2.12.5.1. EnergyMeasurementStruct Type

This struct SHALL indicate the amount of energy measured during a given measurement period.

A server which does not have the ability to determine the time in UTC, or has not yet done so, SHALL use the system time fields to specify the measurement period and observation times.

A server which has determined the time in UTC SHALL use the timestamp fields to specify the measurement period. Such a server MAY also include the systime fields to indicate how many seconds had passed since boot for a given timestamp; this allows for client-side resolution of UTC time for previous reports that only included systime.

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0	**Energy**	energy-mWh	0 to 2^{62}			R V	M
1	**StartTimestamp**	epoch-s				R V	desc
2	**EndTimestamp**	epoch-s	min (StartTimestamp+1)			R V	desc
3	**StartSystime**	systime-ms				R V	desc
4	**EndSystime**	systime-ms	min (StartSystime+1)			R V	desc

2.12.5.1.1. Energy Field

This field SHALL be the reported energy.

If the EnergyMeasurementStruct represents cumulative energy, then this SHALL represent the cumulative energy recorded at either the value of the EndTimestamp field or the value of the EndSystime field, or both.

If the EnergyMeasurementStruct represents periodic energy, then this SHALL represent the energy recorded during the period specified by either the StartTimestamp and EndTimestamp fields, the period specified by the StartSystime and EndSystime fields, or both.

2.12.5.1.2. StartTimestamp Field

This field SHALL indicate the timestamp in UTC of the beginning of the period during which the value of the Energy field was measured.

If this EnergyMeasurementStruct represents cumulative energy, this field SHALL be omitted.

Otherwise, if the server had determined the time in UTC at or before the beginning of the measurement period, this field SHALL be indicated.

Otherwise, if the server had not yet determined the time in UTC at or before the beginning of the measurement period, or does not have the capability of determining the time in UTC, this field SHALL be omitted.

2.12.5.1.3. EndTimestamp Field

This field SHALL indicate the timestamp in UTC of the end of the period during which the value of the Energy field was measured.

If the server had determined the time in UTC by the end of the measurement period, this field SHALL be indicated.

Otherwise, if the server had not yet determined the time in UTC by the end of the measurement period, or does not have the capability of determining the time in UTC, this field SHALL be omitted.

2.12.5.1.4. StartSystime Field

This field SHALL indicate the seconds since boot at the beginning of the period during which the value of the Energy field was measured.

If this EnergyMeasurementStruct represents cumulative energy, this field SHALL be omitted.

Otherwise, if the server had not yet determined the time in UTC at the start of the measurement period, or does not have the capability of determining the time in UTC, this field SHALL be indicated.

Otherwise, if the server had determined the time in UTC at or before the beginning of the measurement period, this field MAY be omitted; if it is indicated, its value SHALL be the seconds since boot at the UTC time indicated in StartTimestamp.

2.12.5.1.5. EndSystime Field

This field SHALL indicate the seconds since boot at the end of the period during which the value of the Energy field was measured.

If the server had not yet determined the time in UTC by the end of the measurement period, or does not have the capability of determining the time in UTC, this field SHALL be indicated.

Otherwise, if the server had determined the time in UTC by the end of the measurement period, this field MAY be omitted; if it is indicated, its value SHALL be the seconds since boot at the UTC time indicated in EndTimestamp.

2.12.5.2. CumulativeEnergyResetStruct Type

This struct SHALL represent the times at which cumulative measurements were last zero, either due to initialization of the device, or an internal reset of the cumulative value.

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0	**ImportedResetTimestamp**	epoch-s		X	null	R V	[IMPE]
1	**ExportedResetTimestamp**	epoch-s		X	null	R V	[EXPE]
2	**ImportedResetSystime**	systime-ms		X	null	R V	[IMPE]
3	**ExportedResetSystime**	systime-ms		X	null	R V	[EXPE]

2.12.5.2.1. ImportedResetTimestamp Field

This field SHALL indicate the timestamp in UTC when the value of the Energy field on the CumulativeEnergyImported attribute was most recently zero.

If the server had determined the time in UTC when the value of the Energy field on the CumulativeEnergyImported attribute was most recently zero, this field SHALL be indicated.

Otherwise, if the server had not yet determined the time in UTC when the value of the Energy field on the CumulativeEnergyImported attribute was most recently zero, or does not have the capability of determining the time in UTC, this field SHALL be omitted.

If the timestamp in UTC when the value of the Energy field on the CumulativeEnergyImported attribute was most recently zero cannot currently be determined, a value of null SHALL be returned.

2.12.5.2.2. ExportedResetTimestamp Field

This field SHALL indicate the timestamp in UTC when the value of the Energy field on the CumulativeEnergyExported attribute was most recently zero.

If the server had determined the time in UTC when the value of the Energy field on the Cumula-

tiveEnergyExported attribute was most recently zero, this field SHALL be indicated.

Otherwise, if the server had not yet determined the time in UTC when the value of the Energy field on the CumulativeEnergyExported attribute was most recently zero, or does not have the capability of determining the time in UTC, this field SHALL be omitted.

If the timestamp in UTC when the value of the Energy field on the CumulativeEnergyExported attribute was most recently zero cannot currently be determined, a value of null SHALL be returned.

2.12.5.2.3. ImportedResetSystime Field

This field SHALL indicate the seconds since boot when the value of the Energy field on the CumulativeEnergyImported attribute was most recently zero.

If the server had not yet determined the time in UTC when the value of the Energy field on the CumulativeEnergyImported attribute was most recently zero, or does not have the capability of determining the time in UTC, this field SHALL be indicated.

Otherwise, if the server had determined the time in UTC when the value of the Energy field on the CumulativeEnergyImported attribute was most recently zero, this field MAY be omitted; if it is indicated, its value SHALL be the seconds since boot at the UTC time indicated in ImportedResetTimestamp.

2.12.5.2.4. ExportedResetSystime Field

This field SHALL indicate the seconds since boot when the value of the Energy field on the CumulativeEnergyExported attribute was most recently zero.

If the server had not yet determined the time in UTC when the value of the Energy field on the CumulativeEnergyExported attribute was most recently zero, or does not have the capability of determining the time in UTC, this field SHALL be indicated.

Otherwise, if the server had determined the time in UTC when the value of the Energy field on the CumulativeEnergyExported attribute was most recently zero, this field MAY be omitted; if it is indicated, its value SHALL be the seconds since boot at the UTC time indicated in ImportedResetTimestamp.

2.12.6. Attributes

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0000	**Accuracy**	MeasurementAccuracyStruct		F		R V	M
0x0001	**CumulativeEnergyImported**	EnergyMeasurementStruct		XQ	null	R V	IMPE & CUME

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0002	**CumulativeEnergyExported**	EnergyMeasurementStruct		XQ	null	R V	EXPE & CUME
0x0003	**PeriodicEnergyImported**	EnergyMeasurementStruct		XQ	null	R V	IMPE & PERE
0x0004	**PeriodicEnergyExported**	EnergyMeasurementStruct		XQ	null	R V	EXPE & PERE
0x0005	**CumulativeEnergyReset**	CumulativeEnergyResetStruct		X	null	R V	[CUME]

2.12.6.1. Accuracy Attribute

This attribute SHALL indicate the accuracy of energy measurement by this server. The value of the MeasurementType field on this MeasurementAccuracyStruct SHALL be ElectricalEnergy.

2.12.6.2. CumulativeEnergyImported Attribute

This attribute SHALL indicate the most recent measurement of cumulative energy imported by the server over the lifetime of the device, and the timestamp of when the measurement was recorded.

The reporting interval of this attribute SHALL be manufacturer dependent. The server MAY choose to omit publication of deltas considered not meaningful.

The server SHALL NOT mark this attribute ready for report if the last time this was done was more recently than 1 second ago.

The server MAY delay marking this attribute ready for report for longer periods if needed, however the server SHALL NOT delay marking this attribute as ready for report for longer than 60 seconds.

If the cumulative energy imported cannot currently be determined, a value of null SHALL be returned.

2.12.6.3. CumulativeEnergyExported Attribute

This attribute SHALL indicate the most recent measurement of cumulative energy exported by the server over the lifetime of the device, and the timestamp of when the measurement was recorded.

The reporting interval of this attribute SHALL be manufacturer dependent. The server MAY choose to omit publication of deltas considered not meaningful.

The server SHALL NOT mark this attribute ready for report if the last time this was done was more recently than 1 second ago.

The server MAY delay marking this attribute ready for report for longer periods if needed, however the server SHALL NOT delay marking this attribute as ready for report for longer than 60 seconds.

If the cumulative energy exported cannot currently be determined, a value of null SHALL be returned.

2.12.6.4. PeriodicEnergyImported Attribute

This attribute SHALL indicate the most recent measurement of energy imported by the server and the period during which it was measured.

The reporting interval of this attribute SHALL be manufacturer dependent. The server MAY choose to omit publication of deltas considered not meaningful.

The server SHALL NOT mark this attribute ready for report if the last time this was done was more recently than 1 second ago.

The server MAY delay marking this attribute ready for report for longer periods if needed, however the server SHALL NOT delay marking this attribute as ready for report for longer than 60 seconds.

If the periodic energy imported cannot currently be determined, a value of null SHALL be returned.

2.12.6.5. PeriodicEnergyExported Attribute

This attribute SHALL indicate the most recent measurement of energy exported by the server and the period during which it was measured.

The reporting interval of this attribute SHALL be manufacturer dependent. The server MAY choose to omit publication of deltas considered not meaningful.

The server SHALL NOT mark this attribute ready for report if the last time this was done was more recently than 1 second ago.

The server MAY delay marking this attribute ready for report for longer periods if needed, however the server SHALL NOT delay marking this attribute as ready for report for longer than 60 seconds.

If the periodic energy exported cannot currently be determined, a value of null SHALL be returned.

2.12.6.6. CumulativeEnergyReset Attribute

This attribute SHALL indicate when cumulative measurements were most recently zero.

2.12.7. Events

This cluster SHALL support these events:

ID	Name	Priority	Access	Conformance
0	CumulativeEnergyMeasured	INFO	R V	CUME
1	PeriodicEnergyMeasured	INFO	R V	PERE

2.12.7.1. CumulativeEnergyMeasured Event

This event SHALL be generated when the server takes a snapshot of the cumulative energy imported by the server, exported from the server, or both, but not more frequently than the rate mentioned in the description above of the related attribute.

The data of this event SHALL contain the following information:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**EnergyImported**	EnergyMeasurementStruct				CUME & IMPE
1	**EnergyExported**	EnergyMeasurementStruct				CUME & EXPE

2.12.7.1.1. EnergyImported Field

This field SHALL be the value of CumulativeEnergyImported attribute at the timestamp indicated in its EndTimestamp field, EndSystime field, or both.

2.12.7.1.2. EnergyExported Field

This field SHALL be the value of CumulativeEnergyExported attribute at the timestamp indicated in its EndTimestamp field, EndSystime field, or both.

2.12.7.2. PeriodicEnergyMeasured Event

This event SHALL be generated when the server reaches the end of a reporting period for imported energy, exported energy, or both.

The data of this event SHALL contain the following information:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**EnergyImported**	EnergyMeasurementStruct				PERE & IMPE

ID	Name	Type	Constraint	Quality	Default	Conformance
1	**EnergyExported**	EnergyMeasurementStruct				PERE & EXPE

2.12.7.2.1. EnergyImported Field

This field SHALL be the value of PeriodicEnergyImported attribute at the timestamp indicated in its EndTimestamp field, EndSystime field, or both.

2.12.7.2.2. EnergyExported Field

This field SHALL be the value of PeriodicEnergyExported attribute at the timestamp indicated in its EndTimestamp field, EndSystime field, or both.

2.13. Electrical Power Measurement Cluster

This cluster provides a mechanism for querying data about electrical power as measured by the server.

2.13.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Revision	Description
1	Initial Release

2.13.2. Classification

Hierarchy	Role	Scope	PICS Code
Base	Application	Endpoint	EPM

2.13.3. Cluster ID

ID	Name
0x0090	Electrical Power Measurement

2.13.4. Features

This cluster SHALL support the FeatureMap bitmap attribute as defined below.

Bit	Code	Feature	Conformance	Summary
0	DIRC	DirectCurrent	O.a+	Supports measurement of direct current
1	ALTC	AlternatingCurrent	O.a+	Supports measurement of alternating current
2	POLY	PolyphasePower	[ALTC]	Supports polyphase measurements
3	HARM	Harmonics	[ALTC]	Supports measurement of AC harmonics
4	PWRQ	PowerQuality	[ALTC]	Supports measurement of AC harmonic phases

2.13.4.1. DirectCurrent Feature

This feature indicates the cluster can measure a direct current.

2.13.4.2. AlternatingCurrent Feature

This feature indicates the cluster can measure an alternating current.

2.13.4.3. PolyphasePower Feature

This feature indicates the cluster represents the collective measurements for a Polyphase power supply.

2.13.4.4. Harmonics Feature

This feature indicates the cluster can measure the harmonics of an alternating current.

2.13.4.5. PowerQuality Feature

This feature indicates the cluster can measure the harmonic phases of an alternating current.

2.13.5. Data Types

2.13.5.1. PowerModeEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	Unknown		M

Value	Name	Summary	Conformance
1	DC	Direct current	M
2	AC	Alternating current, either single-phase or polyphase	M

2.13.5.2. MeasurementRangeStruct Type

This struct SHALL indicate the maximum and minimum values of a given measurement type during a measurement period, along with the observation times of these values.

A server which does not have the ability to determine the time in UTC, or has not yet done so, SHALL use the system time fields to specify the measurement period and observation times.

A server which has determined the time in UTC SHALL use the timestamp fields to specify the measurement period and observation times. Such a server MAY also include the systime fields to indicate how many seconds had passed since boot for a given timestamp; this allows for client-side resolution of UTC time for previous reports that only included systime.

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0	**MeasurementType**	MeasurementTypeEnum					M
1	**Min**	int64	-2^{62} to 2^{62}				M
2	**Max**	int64	-2^{62} to 2^{62}				M
3	**StartTimestamp**	epoch-s					EndTimestamp
4	**EndTimestamp**	epoch-s	min (StartTimestamp+1)				desc
5	**MinTimestamp**	epoch-s					EndTimestamp
6	**MaxTimestamp**	epoch-s	min (MinTimestamp+1)				EndTimestamp
7	**StartSystime**	systime-ms					EndSystime
8	**EndSystime**	systime-ms	min (StartSystime+1)				desc
9	**MinSystime**	systime-ms					EndSystime

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
10	**MaxSystime**	systime-ms	min (MinSystime+1)				EndSystime

2.13.5.2.1. MeasurementType Field

This field SHALL be the type of measurement for the range provided.

2.13.5.2.2. Min Field

This field SHALL be the smallest measured value for the associated measurement over either the period between StartTimestamp and EndTimestamp, or the period between StartSystime and EndSystime, or both.

2.13.5.2.3. Max Field

This field SHALL be the largest measured value for the associated measurement over the period between either StartTimestamp and EndTimestamp or the period between StartSystime and EndSystime, or both.

2.13.5.2.4. StartTimestamp Field

This field SHALL be the timestamp in UTC of the beginning of the measurement period.

If the server had not yet determined the time in UTC at or before the beginning of the measurement period, or does not have the capability of determining the time in UTC, this field SHALL be omitted.

2.13.5.2.5. EndTimestamp Field

This field SHALL be the timestamp in UTC of the end of the measurement period.

If the server had not yet determined the time in UTC at or before the beginning of the measurement period, or does not have the capability of determining the time in UTC, this field SHALL be omitted.

2.13.5.2.6. MinTimestamp Field

This field SHALL be the most recent timestamp in UTC that the value in the Min field was measured.

This field SHALL be greater than or equal to the value of the StartTimestamp field.

This field SHALL be less than or equal to the value of the EndTimestamp field.

2.13.5.2.7. MaxTimestamp Field

This field SHALL be the most recent timestamp in UTC of the value in the Max field.

This field SHALL be greater than or equal to the value of the StartTimestamp field.

This field SHALL be less than or equal to the value of the EndTimestamp field.

2.13.5.2.8. StartSystime Field

This field SHALL be the time since boot of the beginning of the measurement period.

If the server had determined the time in UTC at or before the start of the measurement period, this field MAY be omitted along with the EndSystime, MinSystime, and MaxSystime fields.

2.13.5.2.9. EndSystime Field

This field SHALL be the time since boot of the end of the measurement period.

If the server had determined the time in UTC at the end of the measurement period, this field MAY be omitted along with the StartSystime field, MinSystime, and MaxSystime fields.

2.13.5.2.10. MinSystime Field

This field SHALL be the measurement time since boot of the value in the Min field was measured.

This field SHALL be greater than or equal to the value of the StartSystime field.

This field SHALL be less than or equal to the value of the EndSystime field.

2.13.5.2.11. MaxSystime Field

This field SHALL be the measurement time since boot of the value in the Max field.

This field SHALL be greater than or equal to the value of the StartSystime field.

This field SHALL be less than or equal to the value of the EndSystime field.

2.13.5.3. HarmonicMeasurementStruct Type

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0	**Order**	uint8	min 1		1		M
1	**Measurement**	int64	-2^{62} to 2^{62}	X	null		M

2.13.5.3.1. Order Field

This field SHALL be the order of the harmonic being measured. Typically this is an odd number, but servers may choose to report even harmonics.

2.13.5.3.2. Measurement Field

This field SHALL be the measured value for the given harmonic order.

For the Harmonic Currents attribute, this value is the most recently measured harmonic current reading in milliamps (mA). A positive value indicates that the measured harmonic current is positive, and a negative value indicates that the measured harmonic current is negative.

For the Harmonic Phases attribute, this value is the most recent phase of the given harmonic order

in millidegrees (mDeg). A positive value indicates that the measured phase is leading, and a negative value indicates that the measured phase is lagging.

If this measurement is not currently available, a value of null SHALL be returned.

2.13.6. Attributes

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0000	**PowerMode**	PowerModeEnum				R V	M
0x0001	**NumberOfMeasurementTypes**	uint8	min 1	F		R V	M
0x0002	**Accuracy**	list[MeasurementAccuracyStruct]	1 to NumberOfMeasurementTypes	F		R V	M
0x0003	**Ranges**	list[MeasurementRangeStruct]	0 to NumberOfMeasurementTypes	Q	empty	R V	O
0x0004	**Voltage**	voltage-mV	-2^{62} to 2^{62}	XQ	null	R V	O
0x0005	**ActiveCurrent**	amperage-mA	-2^{62} to 2^{62}	XQ	null	R V	O
0x0006	**ReactiveCurrent**	amperage-mA	-2^{62} to 2^{62}	XQ	null	R V	[ALTC]
0x0007	**ApparentCurrent**	amperage-mA	0 to 2^{62}	XQ	null	R V	[ALTC]
0x0008	**ActivePower**	power-mW	-2^{62} to 2^{62}	XQ	null	R V	M
0x0009	**ReactivePower**	power-mW	-2^{62} to 2^{62}	XQ	null	R V	[ALTC]
0x000A	**ApparentPower**	power-mW	-2^{62} to 2^{62}	XQ	null	R V	[ALTC]
0x000B	**RMSVoltage**	voltage-mV	-2^{62} to 2^{62}	XQ	null	R V	[ALTC]
0x000C	**RMSCurrent**	amperage-mA	-2^{62} to 2^{62}	XQ	null	R V	[ALTC]

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x000D	**RMSPower**	power-mW	-2^{62} to 2^{62}	XQ	null	R V	[ALTC]
0x000E	**Frequency**	int64	0 to 1000000	XQ	null	R V	[ALTC]
0x000F	**HarmonicCurrents**	list[HarmonicMeasurementStruct]	desc	XQ	null	R V	HARM
0x0010	**HarmonicPhases**	list[HarmonicMeasurementStruct]	desc	XQ	null	R V	PWRQ
0x0011	**PowerFactor**	int64	-10000 to 10000	XQ	null	R V	[ALTC]
0x0012	**NeutralCurrent**	amperage-mA	-2^{62} to 2^{62}	XQ	null	R V	[POLY]

2.13.6.1. PowerMode Attribute

This SHALL indicate the current mode of the server. For some servers, such as an EV, this may change depending on the mode of charging or discharging.

2.13.6.2. NumberOfMeasurementTypes Attribute

This SHALL indicate the maximum number of measurement types the server is capable of reporting.

2.13.6.3. Accuracy Attribute

This SHALL indicate a list of accuracy specifications for the measurement types supported by the server. There SHALL be an entry for ActivePower, as well as any other measurement types implemented by this server.

2.13.6.4. Ranges Attribute

This SHALL indicate a list of measured ranges for different measurement types. Each measurement type SHALL have at most one entry in this list, representing the range of measurements in the most recent measurement period.

The reporting interval of this attribute SHALL be manufacturer dependent. The server MAY choose to omit publication of deltas considered not meaningful.

The server SHALL NOT mark this attribute ready for report if the last time this was done was more recently than 1 second ago.

The server MAY delay marking this attribute ready for report for longer periods if needed, however the server SHALL NOT delay marking this attribute as ready for report for longer than 60 seconds.

2.13.6.5. Voltage Attribute

This SHALL indicate the most recent Voltage reading in millivolts (mV).

The reporting interval of this attribute SHALL be manufacturer dependent. The server MAY choose to omit publication of deltas considered not meaningful.

The server SHALL NOT mark this attribute ready for report if the last time this was done was more recently than 1 second ago.

The server MAY delay marking this attribute ready for report for longer periods if needed, however the server SHALL NOT delay marking this attribute as ready for report for longer than 60 seconds.

If the voltage cannot be measured, a value of null SHALL be returned.

2.13.6.6. ActiveCurrent Attribute

This SHALL indicate the most recent ActiveCurrent reading in milliamps (mA).

A positive value represents current flowing into the server, while a negative value represents current flowing out of the server.

The reporting interval of this attribute SHALL be manufacturer dependent. The server MAY choose to omit publication of deltas considered not meaningful.

The server SHALL NOT mark this attribute ready for report if the last time this was done was more recently than 1 second ago.

The server MAY delay marking this attribute ready for report for longer periods if needed, however the server SHALL NOT delay marking this attribute as ready for report for longer than 60 seconds.

If the current cannot be measured, a value of null SHALL be returned.

2.13.6.7. ReactiveCurrent Attribute

This SHALL indicate the most recent ReactiveCurrent reading in milliamps (mA).

A positive value represents current flowing into the server, while a negative value represents current flowing out of the server.

The reporting interval of this attribute SHALL be manufacturer dependent. The server MAY choose to omit publication of deltas considered not meaningful.

The server SHALL NOT mark this attribute ready for report if the last time this was done was more recently than 1 second ago.

The server MAY delay marking this attribute ready for report for longer periods if needed, however the server SHALL NOT delay marking this attribute as ready for report for longer than 60 seconds.

If the current cannot be measured, a value of null SHALL be returned.

2.13.6.8. ApparentCurrent Attribute

This SHALL indicate the most recent ApparentCurrent (square root sum of the squares of active and reactive currents) reading in milliamps (mA).

The reporting interval of this attribute SHALL be manufacturer dependent. The server MAY choose to omit publication of deltas considered not meaningful.

The server SHALL NOT mark this attribute ready for report if the last time this was done was more recently than 1 second ago.

The server MAY delay marking this attribute ready for report for longer periods if needed, however the server SHALL NOT delay marking this attribute as ready for report for longer than 60 seconds.

If the active or reactive currents cannot be measured, a value of null SHALL be returned.

2.13.6.9. ActivePower Attribute

This SHALL indicate the most recent ActivePower reading in milliwatts (mW). If the power cannot be measured, a value of null SHALL be returned.

A positive value represents power imported, while a negative value represents power exported.

The reporting interval of this attribute SHALL be manufacturer dependent. The server MAY choose to omit publication of deltas considered not meaningful.

The server SHALL NOT mark this attribute ready for report if the last time this was done was more recently than 1 second ago.

The server MAY delay marking this attribute ready for report for longer periods if needed, however the server SHALL NOT delay marking this attribute as ready for report for longer than 60 seconds.

If the Polyphase Power feature is set, this value represents the combined active power imported or exported.

2.13.6.10. ReactivePower Attribute

This SHALL indicate the most recent ReactivePower reading in millivolt-amps reactive (mVAR).

A positive value represents power imported, while a negative value represents power exported.

The reporting interval of this attribute SHALL be manufacturer dependent. The server MAY choose to omit publication of deltas considered not meaningful.

The server SHALL NOT mark this attribute ready for report if the last time this was done was more recently than 1 second ago.

The server MAY delay marking this attribute ready for report for longer periods if needed, however the server SHALL NOT delay marking this attribute as ready for report for longer than 60 seconds.

If the reactive power cannot be measured, a value of null SHALL be returned.

If the Polyphase Power feature is supported, this value represents the combined reactive power imported or exported.

2.13.6.11. ApparentPower Attribute

This SHALL indicate the most recent ApparentPower reading in millivolt-amps (mVA).

A positive value represents power imported, while a negative value represents power exported.

The reporting interval of this attribute SHALL be manufacturer dependent. The server MAY choose to omit publication of deltas considered not meaningful.

The server SHALL NOT mark this attribute ready for report if the last time this was done was more recently than 1 second ago.

The server MAY delay marking this attribute ready for report for longer periods if needed, however the server SHALL NOT delay marking this attribute as ready for report for longer than 60 seconds.

If the apparent power cannot be measured, a value of null SHALL be returned.

2.13.6.12. RMSVoltage Attribute

This SHALL indicate the most recent RMSVoltage reading in millivolts (mV).

The reporting interval of this attribute SHALL be manufacturer dependent. The server MAY choose to omit publication of deltas considered not meaningful.

The server SHALL NOT mark this attribute ready for report if the last time this was done was more recently than 1 second ago.

The server MAY delay marking this attribute ready for report for longer periods if needed, however the server SHALL NOT delay marking this attribute as ready for report for longer than 60 seconds.

If the RMS voltage cannot be measured, a value of null SHALL be returned.

2.13.6.13. RMSCurrent Attribute

This SHALL indicate the most recent RMSCurrent reading in milliamps (mA).

A positive value represents current flowing into the server, while a negative value represents current flowing out of the server.

The reporting interval of this attribute SHALL be manufacturer dependent. The server MAY choose to omit publication of deltas considered not meaningful.

The server SHALL NOT mark this attribute ready for report if the last time this was done was more recently than 1 second ago.

The server MAY delay marking this attribute ready for report for longer periods if needed, however the server SHALL NOT delay marking this attribute as ready for report for longer than 60 seconds.

If the RMS current cannot be measured, a value of null SHALL be returned.

2.13.6.14. RMSPower Attribute

This SHALL indicate the most recent RMSPower reading in milliwatts (mW).

A positive value represents power imported, while a negative value represents power exported.

The reporting interval of this attribute SHALL be manufacturer dependent. The server MAY choose to omit publication of deltas considered not meaningful.

The server SHALL NOT mark this attribute ready for report if the last time this was done was more recently than 1 second ago.

The server MAY delay marking this attribute ready for report for longer periods if needed, however the server SHALL NOT delay marking this attribute as ready for report for longer than 60 seconds.

If the RMS power cannot be measured, a value of null SHALL be returned.

2.13.6.15. Frequency Attribute

This SHALL indicate the most recent Frequency reading in millihertz (mHz).

The reporting interval of this attribute SHALL be manufacturer dependent. The server MAY choose to omit publication of deltas considered not meaningful.

The server SHALL NOT mark this attribute ready for report if the last time this was done was more recently than 1 second ago.

The server MAY delay marking this attribute ready for report for longer periods if needed, however the server SHALL NOT delay marking this attribute as ready for report for longer than 60 seconds.

If the frequency cannot be measured, a value of null SHALL be returned.

2.13.6.16. HarmonicCurrents Attribute

This SHALL indicate a list of HarmonicMeasurementStruct values, with each HarmonicMeasurementStruct representing the harmonic current reading for the harmonic order specified by Order.

The reporting interval of this attribute SHALL be manufacturer dependent. The server MAY choose to omit publication of deltas considered not meaningful.

The server SHALL NOT mark this attribute ready for report if the last time this was done was more recently than 1 second ago.

The server MAY delay marking this attribute ready for report for longer periods if needed, however the server SHALL NOT delay marking this attribute as ready for report for longer than 60 seconds.

2.13.6.17. HarmonicPhases Attribute

This SHALL indicate a list of HarmonicMeasurementStruct values, with each HarmonicMeasurementStruct representing the most recent phase of the harmonic current reading for the harmonic

order specified by Order.

The reporting interval of this attribute SHALL be manufacturer dependent. The server MAY choose to omit publication of deltas considered not meaningful.

The server SHALL NOT mark this attribute ready for report if the last time this was done was more recently than 1 second ago.

The server MAY delay marking this attribute ready for report for longer periods if needed, however the server SHALL NOT delay marking this attribute as ready for report for longer than 60 seconds.

2.13.6.18. PowerFactor Attribute

This SHALL indicate the Power Factor ratio in +/- 1/100ths of a percent.

The reporting interval of this attribute SHALL be manufacturer dependent. The server MAY choose to omit publication of deltas considered not meaningful.

The server SHALL NOT mark this attribute ready for report if the last time this was done was more recently than 1 second ago.

The server MAY delay marking this attribute ready for report for longer periods if needed, however the server SHALL NOT delay marking this attribute as ready for report for longer than 60 seconds.

2.13.6.19. NeutralCurrent Attribute

This SHALL indicate the most recent NeutralCurrent reading in milliamps (mA). Typically this is a derived value, taking the magnitude of the vector sum of phase currents.

If the neutral current cannot be measured or derived, a value of null SHALL be returned.

A positive value represents an imbalance between the phase currents when power is imported.

A negative value represents an imbalance between the phase currents when power is exported.

The reporting interval of this attribute SHALL be manufacturer dependent. The server MAY choose to omit publication of deltas considered not meaningful.

The server SHALL NOT mark this attribute ready for report if the last time this was done was more recently than 1 second ago.

The server MAY delay marking this attribute ready for report for longer periods if needed, however the server SHALL NOT delay marking this attribute as ready for report for longer than 60 seconds.

2.13.7. Events

This cluster SHALL support the following event:

ID	Name	Priority	Access	Conformance
0	MeasurementPeri-odRanges	INFO	R V	Ranges

2.13.7.1. MeasurementPeriodRanges Event

If supported, this event SHALL be generated at the end of a measurement period. The start and end times for measurement periods SHALL be determined by the server, and MAY represent overlapping periods.

The data of this event SHALL contain the following information:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**Ranges**	list[MeasurementRangeStruct]			R V	M

2.13.7.1.1. Ranges Field

This SHALL indicate the value of the Ranges attribute at the time of event generation.

Chapter 3. Lighting

The Cluster Library is made of individual chapters such as this one. See Document Control in the Cluster Library for a list of all chapters and documents. References between chapters are made using a *X.Y* notation where *X* is the chapter and *Y* is the sub-section within that chapter. References to external documents are contained in Chapter 1 and are made using [*Rn*] notation.

3.1. General Description

3.1.1. Introduction

The clusters specified in this document are for use typically in lighting applications, but MAY be used in any application domain.

3.1.2. Terms

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).

3.1.3. Cluster List

This section lists the lighting specific clusters as specified in this chapter.

Table 6. Overview of the Lighting Clusters

Cluster ID	Cluster Name	Description
0x0300	Color Control	Attributes and commands for controlling the color of a color-capable light.
0x0301	Ballast Configuration	Attributes and commands for configuring a lighting ballast

3.2. Color Control Cluster

3.2.1. Introduction

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, [I1]. 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 (a.k.a. HSB) color model.

Control over luminance is not included, as this is provided by means of the Level Control for Lighting cluster. 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.

3.2.2. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Rev	Description
1	mandatory global ClusterRevision attribute added; CCB 2028
2	added Options attribute, CCB 2085 2104 2124 2230; ZLO 1.0
3	CCB 2501 2814 2839 2840 2843 2861
4	All Hubs changes
5	new data model format and notation, FeatureMap support
6	Added clarifications to Scenes support for Matter

3.2.3. Classification

Hierarchy	Role	PICS Code	Primary Transaction
Base	Application	CC	Type 1 (client to server)

3.2.4. Cluster Identifiers

Identifier	Name
0x0300	Color Control

3.2.5. Features

This cluster SHALL support the FeatureMap bitmap attribute as defined below.

Bit	Code	Feature	Description
0	**HS**	Hue/Saturation	Supports color specification via hue/saturation.
1	**EHUE**	Enhanced Hue	Enhanced hue is supported.
2	**CL**	Color loop	Color loop is supported.

Bit	Code	Feature	Description
3	**XY**	XY	Supports color specification via XY.
4	**CT**	Color temperature	Supports specification of color temperature.

Support for EHUE SHALL require support for HS.

Support for CL SHALL require support for EHUE.

3.2.6. Dependencies

3.2.6.1. Coupling color temperature to Level Control

If the Level Control for Lighting cluster identifier 0x0008 is supported on the same endpoint as the Color Control cluster and color temperature is supported, it is possible to couple changes in the current level to the color temperature.

The CoupleColorTempToLevel bit of the Options attribute of the Level Control cluster indicates whether the color temperature is to be linked with the CurrentLevel attribute in the Level Control cluster.

If the CoupleColorTempToLevel bit of the Options attribute of the Level Control cluster is equal to 1 and the ColorMode or EnhancedColorMode attribute is set to 2 (color temperature) then a change in the CurrentLevel attribute SHALL affect the ColorTemperatureMireds attribute. This relationship is manufacturer specific, with the qualification that the maximum value of the CurrentLevel attribute SHALL correspond to a ColorTemperatureMired attribute value equal to the CoupleColorTempToLevelMinMireds attribute. This relationship is one-way so a change to the ColorTemperatureMireds attribute SHALL NOT have any effect on the CurrentLevel attribute.

In order to simulate the behavior of an incandescent bulb, a low value of the CurrentLevel attribute SHALL be associated with a high value of the ColorTemperatureMireds attribute (i.e., a low value of color temperature in kelvins).

If the CoupleColorTempToLevel bit of the Options attribute of the Level Control cluster is equal to 0, there SHALL be no link between color temperature and current level.

3.2.6.2. Independent transition in hue and saturation

Various commands in this cluster can be used to start transitions in hue and/or saturation.

- When a transition in hue is in progress, and a command to change saturation (MoveSaturation (with MoveMode!=Stop), StepSaturation, MoveToSaturation) is received by the server, this latter command SHALL NOT interrupt the ongoing transition in hue.
- When a transition in saturation is in progress, and a command to change hue (MoveHue (with MoveMode!=Stop), EnhancedMoveHue (with MoveMode!=Stop) StepHue, EnhancedStepHue, MoveToHue, EnhancedMoveToHue) is received by the server, this latter command SHALL NOT interrupt the ongoing transition in saturation.

3.2.7. Color Information Attribute Set

The attributes defined in this cluster specification are arranged into four sets of related attributes.

The Color Information attribute set contains the attributes summarized below.

Table 7. Attributes of the Color Information Attribute Set

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0000	CurrentHue	uint8	0 to 254	PN	0	R V	HS
0x0001	CurrentSaturation	uint8	0 to 254	PSN	0	R V	HS
0x0002	RemainingTime	uint16	0 to 65534		0	R V	O
0x0003	CurrentX	uint16	0 to 0xFEFF	PSN	0x616B (0.381)	R V	XY
0x0004	CurrentY	uint16	0 to 0xFEFF	PSN	0x607D (0.377)	R V	XY
0x0005	DriftCompensation	enum8	0 to 4		-	R V	O
0x0006	CompensationText	string	max 254		-	R V	O
0x0007	ColorTemperatureMireds	uint16	0 to 0xFEFF	PSN	0x00FA (4000K)	R V	CT
0x0008	ColorMode	enum8	0 to 2	N	1	R V	M
0x000F	Options	map8	desc		0	RW VO	M
0x4000	EnhancedCurrentHue	uint16	all	SN	0	R V	EHUE
0x4001	EnhancedColorMode	enum8	0 to 3	SN	1	R V	M
0x4002	ColorLoopActive	uint8	all	SN	0	R V	CL
0x4003	ColorLoopDirection	uint8	all	SN	0	R V	CL
0x4004	ColorLoopTime	uint16	all	SN	0x0019	R V	CL
0x4005	ColorLoopStartEnhancedHue	uint16	all		0x2300	R V	CL
0x4006	ColorLoopStoredEnhancedHue	uint16	all		0	R V	CL
0x400A	ColorCapabilities	map16	0 to 0x001F		0	R V	M
0x400B	ColorTempPhysicalMinMireds	uint16	0 to 0xFEFF		0	R V	CT
0x400C	ColorTempPhysicalMaxMireds	uint16	0 to 0xFEFF		0xFEFF	R V	CT

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x400D	CoupleColorTempToLevelMinMireds	uint16	ColorTempPhysicalMinMireds to ColorTemperatureMireds		MS	R V	CT | ColorTemperatureMireds
0x4010	StartUpColorTemperatureMireds	uint16	0 to 0xFEFF	NX	MS	RW VM	CT | ColorTemperatureMireds

Regarding attributes whose values persistent across an OTA restart (designated by 'N' in the Access column in the table above), only those attributes that are supported (due to the color capabilities of the device) need to be stored.

3.2.7.1. Scene Table Extensions

If the Scenes Management server cluster is implemented on the same endpoint, the following attributes SHALL be part of the ExtensionFieldSetStruct of the Scene Table. If an attribute is not implemented, the value that will be used for it in the AttributeValuePairStruct is given in parentheses. If the implicit form of the ExtensionFieldSetStruct is used, the order of the attributes in the AttributeValueList is in the given order, i.e., the attribute listed as 1 is added first:

1. CurrentX (0)
2. CurrentY (0)
3. EnhancedCurrentHue (null)
4. CurrentSaturation (null)
5. ColorLoopActive (0)
6. ColorLoopDirection (0)
7. ColorLoopTime (0)
8. ColorTemperatureMireds (null)
9. EnhancedColorMode

Since there is a direct relation between ColorTemperatureMireds and XY, color temperature, if supported, is stored as XY in the scenes table.

Attributes in the scene table that are not supported by the device (according to the FeatureMap attribute) SHALL be present in the scene table but ignored.

If the Scenes Management cluster server is implemented on the same endpoint, and the cluster

revision is 6 or above, then all extension fields in the list above, including up to EnhancedColorMode (item 9), SHALL be present in the extension fields set for a Scene to be considered valid. This disambiguates the color mode used within the Scene.

3.2.7.2. 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 to 254 inclusive)

If this attribute is implemented then the CurrentSaturation and ColorMode attributes SHALL also be implemented.

3.2.7.3. 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 to 254 inclusive)

If this attribute is implemented then the CurrentHue and ColorMode attributes SHALL also be implemented.

3.2.7.4. RemainingTime Attribute

The RemainingTime attribute holds the time remaining, in 1/10ths of a second, until the currently active command will be complete.

3.2.7.5. 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)

3.2.7.6. 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 / 65535 (CurrentY in the range 0 to 65279 inclusive)

3.2.7.7. 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 Values of the DriftCompensation Attribute.

Table 8. Values of the DriftCompensation Attribute

Value	Description
0	None
1	Other / Unknown
2	Temperature monitoring
3	Optical luminance monitoring and feedback
4	Optical color monitoring and feedback

3.2.7.8. 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.

3.2.7.9. ColorTemperatureMireds Attribute

The ColorTemperatureMireds attribute contains **a scaled inverse** of the current value of the color temperature. The unit of ColorTemperatureMireds is the mired (micro reciprocal degree), a.k.a. mirek (micro reciprocal kelvin). It is updated as fast as practical during commands that change the color.

The color temperature value in kelvins SHALL be related to the ColorTemperatureMireds attribute in mireds by the relationship

Color temperature in kelvins = 1,000,000 / ColorTemperatureMireds, where ColorTemperatureMireds is in the range 1 to 65279 mireds inclusive, giving a color temperature range from 1,000,000 kelvins to 15.32 kelvins.

If this attribute is implemented then the ColorMode attribute SHALL also be implemented.

3.2.7.10. ColorMode Attribute

The ColorMode attribute indicates which attributes are currently determining the color of the device.

The value of the ColorMode attribute cannot be written directly - it is set upon reception of any command in section Commands to the appropriate mode for that command.

Table 9. Values of the ColorMode Attribute

Value	Attributes that Determine the Color
0	CurrentHue and CurrentSaturation
1	CurrentX and CurrentY

Value	Attributes that Determine the Color
2	ColorTemperatureMireds

3.2.7.11. Options Attribute

The Options attribute is meant to be changed only during commissioning. The Options attribute is a bitmap that determines the default behavior of some cluster commands. Each command that is dependent on the Options attribute SHALL first construct a temporary Options bitmap that is in effect during the command processing. The temporary Options bitmap has the same format and meaning as the Options attribute, but includes any bits that may be overridden by command fields.

Below is the format and description of the Options attribute and temporary Options bitmap and the effect on dependent commands.

Table 10. Options Attribute

Bit	Name	Values & Summary
0	ExecuteIfOff	0 – Do not execute command if the On/Off cluster, OnOff attribute is FALSE. 1 – Execute command if the On/Off cluster, OnOff attribute is FALSE.

ExecuteIfOff Options bit: Command execution SHALL NOT continue beyond the Options processing if all of these criteria are true:

- The On/Off cluster exists on the same endpoint as this cluster.
- The OnOff attribute of the On/Off cluster, on this endpoint, is FALSE.
- The value of the ExecuteIfOff bit is 0.

3.2.7.12. EnhancedCurrentHue Attribute

The EnhancedCurrentHue attribute represents non-equidistant steps along the CIE 1931 color triangle, and it provides 16-bits precision.

The upper 8 bits of this attribute SHALL be used as an index in the implementation specific XY lookup table to provide the non-equidistance steps. The lower 8 bits SHALL be used to interpolate between these steps in a linear way in order to provide color zoom for the user.

To provide compatibility with standard ZCL, the CurrentHue attribute SHALL contain a hue value in the range 0 to 254, calculated from the EnhancedCurrentHue attribute.

3.2.7.13. EnhancedColorMode Attribute

The EnhancedColorMode attribute specifies which attributes are currently determining the color of the device, as detailed in Values of the EnhancedColorMode Attribute.

Table 11. Values of the EnhancedColorMode Attribute

Value	Attributes That Determine the Color
0	CurrentHue and CurrentSaturation
1	CurrentX and CurrentY
2	ColorTemperatureMireds
3	EnhancedCurrentHue and CurrentSaturation

To provide compatibility with standard ZCL, the original ColorMode attribute SHALL indicate 'CurrentHue and CurrentSaturation' when the light uses the EnhancedCurrentHue attribute. If the ColorMode attribute is changed, e.g., due to one of the standard Color Control cluster commands defined in the ZCL, its new value SHALL be copied to the EnhancedColorMode attribute.

3.2.7.14. ColorLoopActive Attribute

The ColorLoopActive attribute specifies the current active status of the color loop. If this attribute has the value 0, the color loop SHALL not be active. If this attribute has the value 1, the color loop SHALL be active. All other values (2 to 254) are reserved.

3.2.7.15. ColorLoopDirection Attribute

The ColorLoopDirection attribute specifies the current direction of the color loop. If this attribute has the value 0, the EnhancedCurrentHue attribute SHALL be decremented. If this attribute has the value 1, the EnhancedCurrentHue attribute SHALL be incremented. All other values (2 to 254) are reserved.

3.2.7.16. ColorLoopTime Attribute

The ColorLoopTime attribute specifies the number of seconds it SHALL take to perform a full color loop, i.e., to cycle all values of the EnhancedCurrentHue attribute (between 0 and 0xFFFE).

3.2.7.17. ColorLoopStartEnhancedHue Attribute

The ColorLoopStartEnhancedHue attribute specifies the value of the EnhancedCurrentHue attribute from which the color loop SHALL be started.

3.2.7.18. ColorLoopStoredEnhancedHue Attribute

The ColorLoopStoredEnhancedHue attribute specifies the value of the EnhancedCurrentHue attribute before the color loop was started. Once the color loop is complete, the EnhancedCurrentHue attribute SHALL be restored to this value.

3.2.7.19. ColorCapabilities Attribute

Bits 0-4 of the ColorCapabilities attribute SHALL have the same values as the corresponding bits of the FeatureMap attribute. All other bits in ColorCapabilities SHALL be 0.

3.2.7.20. ColorTempPhysicalMinMireds Attribute

The ColorTempPhysicalMinMireds attribute indicates the minimum mired value supported by the hardware. ColorTempPhysicalMinMireds corresponds to the maximum color temperature in kelvins supported by the hardware. ColorTempPhysicalMinMireds <= ColorTemperatureMireds.

3.2.7.21. ColorTempPhysicalMaxMireds Attribute

The ColorTempPhysicalMaxMireds attribute indicates the maximum mired value supported by the hardware. ColorTempPhysicalMaxMireds corresponds to the minimum color temperature in kelvins supported by the hardware. ColorTemperatureMireds <= ColorTempPhysicalMaxMireds.

3.2.7.22. CoupleColorTempToLevelMinMireds Attribute

The CoupleColorTempToLevelMinMireds attribute specifies a lower bound on the value of the ColorTemperatureMireds attribute for the purposes of coupling the ColorTemperatureMireds attribute to the CurrentLevel attribute when the CoupleColorTempToLevel bit of the Options attribute of the Level Control cluster is equal to 1. When coupling the ColorTemperatureMireds attribute to the CurrentLevel attribute, this value SHALL correspond to a CurrentLevel value of 0xFE (100%).

This attribute SHALL be set such that the following relationship exists:
ColorTempPhysicalMinMireds ≤ CoupleColorTempToLevelMinMireds ≤ ColorTemperatureMireds

Note that since this attribute is stored as a micro reciprocal degree (mired) value (i.e. color temperature in kelvins = 1,000,000 / CoupleColorTempToLevelMinMireds), the CoupleColorTempToLevelMinMireds attribute corresponds to an upper bound on the value of the color temperature in kelvins supported by the device.

3.2.7.23. StartUpColorTemperatureMireds Attribute

The StartUpColorTemperatureMireds attribute SHALL define the desired startup color temperature value a lamp SHALL use when it is supplied with power and this value SHALL be reflected in the ColorTemperatureMireds attribute. In addition, the ColorMode and EnhancedColorMode attributes SHALL be set to 0x02 (color temperature). The values of the StartUpColorTemperatureMireds attribute are listed in the table below,

Table 12. Values of the StartUpColorTemperatureMireds attribute

Value	Action on power up
0 to 0xFEFF	Set the ColorTemperatureMireds attribute to this value.
null	Set the ColorTemperatureMireds attribute to its previous value.

3.2.8. Defined Primaries Information Attribute Set

The Defined Primaries Information attribute set contains the attributes summarized in Defined Primaries Information Attribute Set.

Table 13. Defined Primaries Information Attribute Set

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0010	NumberOfPrimaries	uint8	0 to 6	FX	-	R V	M
0x0011	Primary1X	uint16	0 to 0xFEFF	F	-	R V	M^0
0x0012	Primary1Y	uint16	0 to 0xFEFF	F	-	R V	M^0
0x0013	Primary1Intensity	uint8	all	FX	-	R V	M^0
0x0015	Primary2X	uint16	0 to 0xFEFF	F	-	R V	M^1
0x0016	Primary2Y	uint16	0 to 0xFEFF	F	-	R V	M^1
0x0017	Primary2Intensity	uint8	all	FX	-	R V	M^1
0x0019	Primary3X	uint16	0 to 0xFEFF	F	-	R V	M^2
0x001A	Primary3Y	uint16	0 to 0xFEFF	F	-	R V	M^2
0x001B	Primary3Intensity	uint8	all	FX	-	R V	M^2

M^i = Mandatory if the value of the NumberOfPrimaries attribute is greater than *i*, otherwise optional.

3.2.8.1. NumberOfPrimaries Attribute

The NumberOfPrimaries attribute contains the number of color primaries implemented on this device. A value of null 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.

3.2.8.2. 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)

3.2.8.3. 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)

3.2.8.4. 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 Ballast Configuration Cluster), normalized such that the primary with the highest maximum intensity contains the value 0xFE.

A value of null SHALL indicate that this primary is not available.

3.2.8.5. 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.

3.2.9. Additional Defined Primaries Information Attribute Set

The Additional Defined Primaries Information attribute set contains the attributes summarized in Additional Defined Primaries Information Attribute Set.

Table 14. Additional Defined Primaries Information Attribute Set

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0020	Primary4X	uint16	0 to 0xFEFF	F	-	R V	M^3
0x0021	Primary4Y	uint16	0 to 0xFEFF	F	-	R V	M^3
0x0022	Primary4Intensity	uint8	all	FX	-	R V	M^3
0x0024	Primary5X	uint16	0 to 0xFEFF	F	-	R V	M^4
0x0025	Primary5Y	uint16	0 to 0xFEFF	F	-	R V	M^4
0x0026	Primary5Intensity	uint8	all	FX	-	R V	M^4
0x0028	Primary6X	uint16	0 to 0xFEFF	F	-	R V	M^5
0x0029	Primary6Y	uint16	0 to 0xFEFF	F	-	R V	M^5
0x002A	Primary6Intensity	uint8	all	FX	-	R V	M^5

M^i = Mandatory if the value of the NumberOfPrimaries attribute is greater than *i*, otherwise optional.

3.2.9.1. Remaining 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.

3.2.10. Defined Color Points Settings Attribute Set

The Defined Color Points Settings attribute set contains the attributes summarized in Defined Color Points Settings Attribute Set.

Table 15. Defined Color Points Settings Attribute Set

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0030	WhitePointX	uint16	0 to 0xFEFF		-	RW VM	O
0x0031	WhitePointY	uint16	0 to 0xFEFF		-	RW VM	O
0x0032	ColorPointRX	uint16	0 to 0xFEFF		-	RW VM	O
0x0033	ColorPointRY	uint16	0 to 0xFEFF		-	RW VM	O
0x0034	ColorPointRIntensity	uint8	all	X	-	RW VM	O
0x0036	ColorPointGX	uint16	0 to 0xFEFF		-	RW VM	O
0x0037	ColorPointGY	uint16	0 to 0xFEFF		-	RW VM	O
0x0038	ColorPointGIntensity	uint8	all	X	-	RW VM	O
0x003A	ColorPointBX	uint16	0 to 0xFEFF		-	RW VM	O
0x003B	ColorPointBY	uint16	0 to 0xFEFF		-	RW VM	O
0x003C	ColorPointBIntensity	uint8	all	X	-	RW VM	O

3.2.10.1. 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)

3.2.10.2. 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)

3.2.10.3. 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)

3.2.10.4. 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)

3.2.10.5. 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 Ballast Configuration Cluster), normalized such that the color point with the highest relative intensity contains the value 0xFE.

A value of null SHALL indicate an invalid value.

3.2.10.6. 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.

3.2.11. Commands

The command IDs for the Color Control cluster are listed in Command IDs for the Color Control Cluster.

Table 16. Command IDs for the Color Control Cluster

ID	Name	Direction	Response	Access	Conformance
0x00	MoveToHue	client ⇒ server	Y	O	HS
0x01	MoveHue	client ⇒ server	Y	O	HS
0x02	StepHue	client ⇒ server	Y	O	HS
0x03	MoveToSaturation	client ⇒ server	Y	O	HS
0x04	MoveSaturation	client ⇒ server	Y	O	HS
0x05	StepSaturation	client ⇒ server	Y	O	HS
0x06	MoveToHueAndSaturation	client ⇒ server	Y	O	HS
0x07	MoveToColor	client ⇒ server	Y	O	XY
0x08	MoveColor	client ⇒ server	Y	O	XY
0x09	StepColor	client ⇒ server	Y	O	XY
0x0A	MoveToColorTemperature	client ⇒ server	Y	O	CT
0x40	EnhancedMoveToHue	client ⇒ server	Y	O	EHUE
0x41	EnhancedMoveHue	client ⇒ server	Y	O	EHUE
0x42	EnhancedStepHue	client ⇒ server	Y	O	EHUE
0x43	EnhancedMoveToHueAndSaturation	client ⇒ server	Y	O	EHUE
0x44	ColorLoopSet	client ⇒ server	Y	O	CL
0x47	StopMoveStep	client ⇒ server	Y	O	HS | XY | CT
0x4B	MoveColorTemperature	client ⇒ server	Y	O	CT
0x4C	StepColorTemperature	client ⇒ server	Y	O	CT

3.2.11.1. Generic Usage Notes

When asked to change color via one of these commands, the implementation SHALL select a color, within the limits of the hardware of the device, which is as close as possible to that requested. The determination as to the true representations of color is out of the scope of this specification. However, as long as the color data fields of the received command are within the permitted range of this specification and no error condition applies, the resulting status code SHALL be SUCCESS.

For example the MoveToColorTemperature command: if the target color temperature is not achievable by the hardware then the color temperature SHALL be clipped at the physical minimum or maximum achievable (depending on the direction of the color temperature transition) when the device reaches that color temperature (which MAY be before the requested transition time).

If a color loop is active (i.e., the ColorLoopActive attribute is equal to 1), it SHALL only be stopped

by sending a specific ColorLoopSet command frame with a request to deactivate the color loop (i.e., the color loop SHALL not be stopped on receipt of another command which has a 'stop' semantic such as the EnhancedMoveToHue command with MoveMode==Stop, or the StopMoveStep command). In addition, while a color loop is active, a manufacturer MAY choose to ignore incoming color commands which affect a change in hue.

3.2.11.2. 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 ColorTemperatureMireds) 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.

3.2.11.3. Use of the OptionsMask and OptionsOverride fields

The OptionsMask and OptionsOverride fields SHALL both be present. Default values are provided to interpret missing fields from legacy devices. A temporary Options bitmap SHALL be created from the Options attribute, using the OptionsMask and OptionsOverride fields. Each bit of the temporary Options bitmap SHALL be determined as follows:

Each bit in the Options attribute SHALL determine the corresponding bit in the temporary Options bitmap, unless the OptionsMask field is present and has the corresponding bit set to 1, in which case the corresponding bit in the OptionsOverride field SHALL determine the corresponding bit in the temporary Options bitmap.

The resulting temporary Options bitmap SHALL then be processed as defined in section Options Attribute.

3.2.11.4. MoveToHue Command

The MoveToHue command SHALL have the following data fields:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	Hue	uint8	0 to 254			M
1	Direction	enum8	desc			M
2	TransitionTime	uint16	0 to 65534			M
3	OptionsMask	map8	desc		0	M
4	OptionsOverride	map8	desc		0	M

3.2.11.4.1. Hue Field

The Hue field specifies the hue to be moved to.

3.2.11.4.2. Direction Field

The Direction field SHALL be one of the non-reserved values in Values of the Direction Field.

Table 17. Values of the Direction Field

Value	Description
0	Shortest distance
1	Longest distance
2	Up
3	Down

3.2.11.4.3. TransitionTime Field

The TransitionTime field specifies, in 1/10ths of a second, the time that SHALL be taken to move to the new hue.

3.2.11.4.4. OptionsMask and OptionsOverride fields

The OptionsMask and OptionsOverride fields SHALL be processed according to section Use of the OptionsMask and OptionsOverride fields.

3.2.11.4.5. Effect on Receipt

On receipt of this command, a device SHALL also set the ColorMode attribute to the value 0 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 TransitionTime 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.

3.2.11.5. MoveHue Command

The MoveHue command SHALL have the following data fields:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	MoveMode	enum8	desc			M
1	Rate	uint8	all			M

ID	Name	Type	Constraint	Quality	Default	Conformance
2	OptionsMask	map8	desc		0	M
3	OptionsOverride	map8	desc		0	M

3.2.11.5.1. MoveMode Field

The MoveMode field SHALL be one of the non-reserved values in Values of the MoveMode Field. If the MoveMode field is equal to 0 (Stop), the Rate field SHALL be ignored.

Table 18. Values of the MoveMode Field

Value	Description
0	Stop
1	Up
2	Reserved
3	Down

3.2.11.5.2. 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 MoveMode field is set to 1 (up) or 3 (down) and the Rate field has a value of zero, the command has no effect and a response command SHALL be sent in response, with the status code set to INVALID_COMMAND. If the MoveMode field is set to 0 (stop) the Rate field SHALL be ignored.

3.2.11.5.3. OptionsMask and OptionsOverride field

The OptionsMask and OptionsOverride fields SHALL be processed according to section Use of the OptionsMask and OptionsOverride fields.

3.2.11.5.4. Effect on Receipt

On receipt of this command, a device SHALL set the ColorMode attribute to the value 0 and SHALL then move from its current hue in an up or down direction in a continuous fashion, as detailed in Actions on Receipt for MoveHue Command.

Table 19. Actions on Receipt for MoveHue Command

MoveMode	Action on Receipt
Stop	If moving, stop, else ignore the command (i.e., the command is accepted but has no effect). This SHALL stop any ongoing hue and/or saturation transition(s).
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 wraparound and proceed from its minimum allowed value.

MoveMode	Action on Receipt
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 wraparound and proceed from its maximum allowed value.

3.2.11.6. StepHue Command

The StepHue command SHALL have the following data fields:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	StepMode	enum8	desc			M
1	StepSize	uint8	all			M
2	TransitionTime	uint8	all			M
3	OptionsMask	map8	desc		0	M
4	OptionsOverride	map8	desc		0	M

3.2.11.6.1. StepMode Field

The StepMode field SHALL be one of the non-reserved values in Values of the StepMode Field.

Table 20. Values of the StepMode Field

Value	Description
0	Reserved
1	Up
2	Reserved
3	Down

3.2.11.6.2. StepSize Field

The change to be added to (or subtracted from) the current value of the device's hue.

3.2.11.6.3. TransitionTime Field

The TransitionTime 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.

NOTE Here the TransitionTime data field is of data type uint8, where uint16 is more common for TransitionTime data fields in other clusters / commands.

3.2.11.6.4. OptionsMask and OptionsOverride fields

The OptionsMask and OptionsOverride fields SHALL be processed according to section Use of the OptionsMask and OptionsOverride fields.

3.2.11.6.5. Effect on Receipt

On receipt of this command, a device SHALL set the ColorMode attribute to the value 0 and SHALL then move from its current hue in an up or down direction by one step, as detailed in Actions on Receipt for StepHue Command.

Table 21. Actions on Receipt for StepHue Command

StepMode	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 wraparound and proceed from 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 wraparound and proceed from the maximum allowed value.

3.2.11.7. MoveToSaturation Command

The MoveToSaturation command SHALL have the following data fields:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	Saturation	uint8	0 to 254			M
1	TransitionTime	uint16	0 to 65534			M
2	OptionsMask	map8	desc		0	M
3	OptionsOverride	map8	desc		0	M

3.2.11.7.1. OptionsMask and OptionsOverride fields

The OptionsMask and OptionsOverride fields SHALL be processed according to section Use of the OptionsMask and OptionsOverride fields.

3.2.11.7.2. Effect on Receipt

On receipt of this command, a device SHALL set the ColorMode attribute to the value 0 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 TransitionTime field, in 1/10ths of a second.

3.2.11.8. MoveSaturation Command

The MoveSaturation command SHALL have the following data fields:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	MoveMode	enum8	desc			M

ID	Name	Type	Constraint	Quality	Default	Conformance
1	Rate	uint8	all			M
2	OptionsMask	map8	desc		0	M
3	OptionsOverride	map8	desc		0	M

3.2.11.8.1. MoveMode Field

The MoveMode field SHALL be one of the non-reserved values in Values of the MoveMode Field. If the MoveMode field is equal to 0 (Stop), the Rate field SHALL be ignored.

Table 22. Values of the MoveMode Field

Value	Description
0	Stop
1	Up
2	Reserved
3	Down

3.2.11.8.2. 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 MoveMode field is set to 1 (up) or 3 (down) and the Rate field has a value of zero, the command has no effect and a response command SHALL be sent in response, with the status code set to INVALID_COMMAND. If the MoveMode field is set to 0 (stop) the Rate field SHALL be ignored.

3.2.11.8.3. OptionsMask and OptionsOverride fields

The OptionsMask and OptionsOverride fields SHALL be processed according to section Use of the OptionsMask and OptionsOverride fields.

3.2.11.8.4. Effect on Receipt

On receipt of this command, a device SHALL set the ColorMode attribute to the value 0 and SHALL then move from its current saturation in an up or down direction in a continuous fashion, as detailed in Actions on Receipt for MoveSaturation Command.

Table 23. Actions on Receipt for MoveSaturation Command

MoveMode	Action on Receipt
Stop	If moving, stop, else ignore the command (i.e., the command is accepted but has no effect). This SHALL stop any ongoing hue and/or saturation transition(s).
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.

3.2.11.9. StepSaturation Command

The StepSaturation command SHALL have the following data fields:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	StepMode	enum8	desc			M
1	StepSize	uint8	all			M
2	TransitionTime	uint8	all			M
3	OptionsMask	map8	desc		0	M
4	OptionsOverride	map8	desc		0	M

3.2.11.9.1. StepMode Field

The StepMode field SHALL be one of the non-reserved values in Values of the StepMode Field.

Table 24. Values of the StepMode Field

Value	Description
0	Reserved
1	Up
2	Reserved
3	Down

3.2.11.9.2. StepSize Field

The change to be added to (or subtracted from) the current value of the device's saturation.

3.2.11.9.3. TransitionTime Field

The TransitionTime 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.

NOTE Here the TransitionTime data field is of data type uint8, where uint16 is more common for TransitionTime data fields in other clusters / commands.

3.2.11.9.4. OptionsMask and OptionsOverride fields

The OptionsMask and OptionsOverride fields SHALL be processed according to section Use of the OptionsMask and OptionsOverride fields.

3.2.11.9.5. Effect on Receipt

On receipt of this command, a device SHALL set the ColorMode attribute to the value 0 and SHALL then move from its current saturation in an up or down direction by one step, as detailed in Actions on Receipt for StepSaturation Command.

Table 25. Actions on Receipt for StepSaturation Command

StepMode	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.

3.2.11.10. MoveToHueAndSaturation Command

The MoveToHueAndSaturation command SHALL have the following data fields:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	Hue	uint8	0 to 254			M
1	Saturation	uint8	0 to 254			M
2	TransitionTime	uint16	0 to 65534			M
3	OptionsMask	map8	desc		0	M
4	OptionsOverride	map8	desc		0	M

3.2.11.10.1. OptionsMask and OptionsOverride fields

The OptionsMask and OptionsOverride fields SHALL be processed according to section Use of the OptionsMask and OptionsOverride fields.

3.2.11.10.2. Effect on Receipt

On receipt of this command, a device SHALL set the ColorMode attribute to the value 0 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 TransitionTime 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 through color space, i.e., movement is 'in a straight line' across the CIE xyY Color Space.

3.2.11.11. MoveToColor Command

The MoveToColor command SHALL have the following data fields:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	ColorX	uint16	0 to 0xFEFF			M

ID	Name	Type	Constraint	Quality	Default	Conformance
1	ColorY	uint16	0 to 0xFEFF			M
2	TransitionTime	uint16	0 to 65534			M
3	OptionsMask	map8	desc		0	M
4	OptionsOverride	map8	desc		0	M

3.2.11.11.1. OptionsMask and OptionsOverride fields

The OptionsMask and OptionsOverride fields SHALL be processed according to section Use of the OptionsMask and OptionsOverride fields.

3.2.11.11.2. Effect on Receipt

On receipt of this command, a device SHALL set the value of the ColorMode attribute, where implemented, to 1, 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 TransitionTime 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 through color space, i.e., movement is 'in a straight line' across the CIE xyY Color Space.

3.2.11.12. MoveColor Command

The MoveColor command SHALL have the following data fields:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	RateX	int16	all			M
1	RateY	int16	all			M
2	OptionsMask	map8	desc		0	M
3	OptionsOverride	map8	desc		0	M

3.2.11.12.1. 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.

3.2.11.12.2. 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.

3.2.11.12.3. OptionsMask and OptionsOverride fields

The OptionsMask and OptionsOverride fields SHALL be processed according to section Use of the OptionsMask and OptionsOverride fields.

3.2.11.12.4. Effect on Receipt

On receipt of this command, a device SHALL set the value of the ColorMode attribute, where implemented, to 1, 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.

3.2.11.13. StepColor Command

The StepColor command SHALL have the following data fields:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	StepX	int16	all			M
1	StepY	int16	all			M
2	TransitionTime	uint16	0 to 65534			M
3	OptionsMask	map8	desc		0	M
4	OptionsOverride	map8	desc		0	M

3.2.11.13.1. 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.

3.2.11.13.2. TransitionTime Field

The TransitionTime field specifies, in 1/10ths of a second, the time that SHALL be taken to perform the color change.

3.2.11.13.3. OptionsMask and OptionsOverride fields

The OptionsMask and OptionsOverride fields SHALL be processed according to section Use of the OptionsMask and OptionsOverride fields.

3.2.11.13.4. Effect on Receipt

On receipt of this command, a device SHALL set the value of the ColorMode attribute, where implemented, to 1, 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 TransitionTime 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 through color space, i.e., movement is 'in a straight line' across the CIE xyY Color Space.

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.

3.2.11.14. MoveToColorTemperature Command

The MoveToColorTemperature command SHALL have the following data fields:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	ColorTemperatureMireds	uint16	0 to 0xFEFF			M
1	TransitionTime	uint16	0 to 65534			M
2	OptionsMask	map8	desc		0	M
3	OptionsOverride	map8	desc		0	M

3.2.11.14.1. OptionsMask and OptionsOverride fields

The OptionsMask and OptionsOverride fields SHALL be processed according to section Use of the OptionsMask and OptionsOverride fields.

3.2.11.14.2. Effect on Receipt

On receipt of this command, a device SHALL set the value of the ColorMode attribute, where implemented, to 2, and SHALL then move from its current color to the color given by the ColorTemperatureMireds 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 TransitionTime 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'.

3.2.11.15. EnhancedMoveToHue Command

The EnhancedMoveToHue command allows lamps to be moved in a smooth continuous transition from their current hue to a target hue.

The EnhancedMoveToHue command SHALL have the following data fields:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	EnhancedHue	uint16	all			M
1	Direction	enum8	desc			M
2	TransitionTime	uint16	0 to 65534			M
3	OptionsMask	map8	desc		0	M
4	OptionsOverride	map8	desc		0	M

3.2.11.15.1. EnhancedHue Field

The EnhancedHue field specifies the target extended hue for the lamp.

3.2.11.15.2. Direction Field

This field is identical to the Direction field of the MoveToHue command of the Color Control cluster (see sub-clause Use of the OptionsMask and OptionsOverride fields).

3.2.11.15.3. TransitionTime Field

This field is identical to the TransitionTime field of the MoveToHue command of the Color Control cluster (see sub-clause Use of the OptionsMask and OptionsOverride fields).

3.2.11.15.4. OptionsMask and OptionsOverride fields

The OptionsMask and OptionsOverride fields SHALL be processed according to section Use of the OptionsMask and OptionsOverride fields.

3.2.11.15.5. Effect on Receipt

On receipt of this command, a device SHALL set the ColorMode attribute to 0 and set the EnhancedColorMode attribute to the value 3. The device SHALL then move from its current enhanced hue to the value given in the EnhancedHue field.

The movement SHALL be continuous, i.e., not a step function, and the time taken to move to the new enhanced hue SHALL be equal to the TransitionTime field.

3.2.11.16. EnhancedMoveHue Command

The EnhancedMoveHue command allows lamps to be moved in a continuous stepped transition from their current hue to a target hue.

The EnhancedMoveHue command SHALL have the following data fields:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	MoveMode	enum8	desc			M
1	Rate	uint16	all			M

ID	Name	Type	Constraint	Quality	Default	Conformance
2	OptionsMask	map8	desc		0	M
3	OptionsOverride	map8	desc		0	M

3.2.11.16.1. MoveMode Field

This field is identical to the MoveMode field of the MoveHue command of the Color Control cluster (see sub-clause MoveHue Command). If the MoveMode field is equal to 0 (Stop), the Rate field SHALL be ignored.

3.2.11.16.2. Rate field

The Rate field specifies the rate of movement in steps per second. A step is a change in the extended hue of a device by one unit. If the MoveMode field is set to 1 (up) or 3 (down) and the Rate field has a value of zero, the command has no effect and a response command SHALL be sent in response, with the status code set to INVALID_COMMAND. If the MoveMode field is set to 0 (stop) the Rate field SHALL be ignored.

3.2.11.16.3. OptionsMask and OptionsOverride fields

The OptionsMask and OptionsOverride fields SHALL be processed according to section Use of the OptionsMask and OptionsOverride fields.

3.2.11.16.4. Effect on receipt

On receipt of this command, a device SHALL set the ColorMode attribute to 0 and set the EnhancedColorMode attribute to the value 3. The device SHALL then move from its current enhanced hue in an up or down direction in a continuous fashion, as detailed in Actions on Receipt of the Enhanced-MoveHueCommand.

Table 26. Actions on Receipt of the EnhancedMoveHueCommand

MoveMode	Action on Receipt
Stop	If moving, stop, else ignore the command (i.e., the command is accepted but has no effect). This SHALL stop any ongoing hue and/or saturation transition(s).
Up	Increase the device's enhanced hue at the rate given in the Rate field. If the enhanced hue reaches the maximum allowed for the device, wraparound and proceed from its minimum allowed value.
Down	Decrease the device's enhanced hue at the rate given in the Rate field. If the hue reaches the minimum allowed for the device, wraparound and proceed from its maximum allowed value.

3.2.11.17. EnhancedStepHue Command

The EnhancedStepHue command allows lamps to be moved in a stepped transition from their current hue to a target hue, resulting in a linear transition through XY space.

The EnhancedStepHue command SHALL have the following data fields:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	StepMode	enum8	desc			M
1	StepSize	uint16	all			M
2	TransitionTime	uint16	0 to 65534			M
3	OptionsMask	map8	desc		0	M
4	OptionsOverride	map8	desc		0	M

3.2.11.17.1. StepMode Field

This field is identical to the StepMode field of the StepHue command of the Color Control cluster (see sub-clause StepHue Command).

3.2.11.17.2. StepSize Field

The StepSize field specifies the change to be added to (or subtracted from) the current value of the device's enhanced hue.

3.2.11.17.3. TransitionTime Field

The TransitionTime field specifies, in units of 1/10ths of a second, the time that SHALL be taken to perform the step. A step is a change to the device's enhanced hue of a magnitude corresponding to the StepSize field.

NOTE	Here TransitionTime data field is of data type uint16, while the TransitionTime data field of the StepHue command is of data type uint8.

3.2.11.17.4. OptionsMask and OptionsOverride fields

The OptionsMask and OptionsOverride fields SHALL be processed according to section Use of the OptionsMask and OptionsOverride fields.

3.2.11.17.5. Effect on Receipt

On receipt of this command, a device SHALL set the ColorMode attribute to 0 and the EnhancedColorMode attribute to the value 3. The device SHALL then move from its current enhanced hue in an up or down direction by one step, as detailed in Actions on Receipt for the EnhancedStepHue Command.

Table 27. Actions on Receipt for the EnhancedStepHue Command

StepMode	Action on Receipt
Up	Increase the device's enhanced hue by one step. If the enhanced hue reaches the maximum allowed for the device, wraparound and proceed from its minimum allowed value.

StepMode	Action on Receipt
Down	Decrease the device's enhanced hue by one step. If the hue reaches the minimum allowed for the device, wraparound and proceed from its maximum allowed value.

3.2.11.18. EnhancedMoveToHueAndSaturation Command

The EnhancedMoveToHueAndSaturation command allows lamps to be moved in a smooth continuous transition from their current hue to a target hue and from their current saturation to a target saturation.

The EnhancedMoveToHueAndSaturation command SHALL have the following data fields:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	EnhancedHue	uint16	all			M
1	Saturation	uint8	0 to 254			M
2	TransitionTime	uint16	0 to 65534			M
3	OptionsMask	map8	desc		0	M
4	OptionsOverride	map8	desc		0	M

3.2.11.18.1. EnhancedHue Field

The EnhancedHue field specifies the target extended hue for the lamp.

3.2.11.18.2. Saturation Field

This field is identical to the Saturation field of the MoveToHueAndSaturation command of the Color Control cluster (see sub-clause MoveToHueAndSaturation Command).

3.2.11.18.3. TransitionTime Field

This field is identical to the TransitionTime field of the MoveToHue command of the Color Control cluster (see sub-clause MoveToHueAndSaturation Command).

3.2.11.18.4. OptionsMask and OptionsOverride fields

The OptionsMask and OptionsOverride fields SHALL be processed according to section Use of the OptionsMask and OptionsOverride fields.

3.2.11.18.5. Effect on Receipt

On receipt of this command, a device SHALL set the ColorMode attribute to the value 0 and set the EnhancedColorMode attribute to the value 3. The device SHALL then move from its current enhanced hue and saturation to the values given in the EnhancedHue 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 TransitionTime 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 through color space, i.e., movement is 'in a straight line' across the CIE xyY Color Space.

3.2.11.19. ColorLoopSet Command

The Color Loop Set command allows a color loop to be activated such that the color lamp cycles through its range of hues.

The ColorLoopSet command SHALL have the following data fields:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	UpdateFlags	map8	desc			M
1	Action	enum8	desc			M
2	Direction	enum8	desc			M
3	Time	uint16	all			M
4	StartHue	uint16	all			M
5	OptionsMask	map8	desc		0	M
6	OptionsOverride	map8	desc		0	M

3.2.11.19.1. UpdateFlags Field

The UpdateFlags field specifies which color loop attributes to update before the color loop is started. This field SHALL be formatted as illustrated in Format of the UpdateFlags Field of the ColorLoopSet Command.

Table 28. Format of the UpdateFlags Field of the ColorLoopSet Command

Bit	Name
0	UpdateAction
1	UpdateDirection
2	UpdateTime
3	UpdateStartHue
4-7	(Reserved)

The UpdateAction sub-field is 1 bit in length and specifies whether the device SHALL adhere to the action field in order to process the command. If this sub-field is set to 1, the device SHALL adhere to the action field. If this sub-field is set to 0, the device SHALL ignore the Action field.

The UpdateDirection sub-field is 1 bit in length and specifies whether the device SHALL update the ColorLoopDirection attribute with the Direction field. If this sub-field is set to 1, the device SHALL update the value of the ColorLoopDirection attribute with the value of the Direction field. If this sub-field is set to 0, the device SHALL ignore the Direction field.

The UpdateTime sub-field is 1 bit in length and specifies whether the device SHALL update the ColorLoopTime attribute with the Time field. If this sub-field is set to 1, the device SHALL update the value of the ColorLoopTime attribute with the value of the Time field. If this sub-field is set to 0, the device SHALL ignore the Time field.

The UpdateStartHue sub-field is 1 bit in length and specifies whether the device SHALL update the ColorLoopStartEnhancedHue attribute with the StartHue field. If this sub-field is set to 1, the device SHALL update the value of the ColorLoopStartEnhancedHue attribute with the value of the StartHue field. If this sub-field is set to 0, the device SHALL ignore the StartHue field.

3.2.11.19.2. Action Field

The Action field specifies the action to take for the color loop if the UpdateAction sub-field of the UpdateFlags field is set to 1. This field SHALL be set to one of the non-reserved values listed in Values of the Action Field of the ColorLoopSet Command.

Table 29. Values of the Action Field of the ColorLoopSet Command

Value	Description
0	De-activate the color loop.
1	Activate the color loop from the value in the ColorLoopStartEnhancedHue field.
2	Activate the color loop from the value of the EnhancedCurrentHue attribute.

3.2.11.19.3. Direction Field

The Direction field specifies the direction for the color loop if the Update Direction field of the UpdateFlags field is set to 1. This field SHALL be set to one of the non-reserved values listed in Values of the Direction Field of the ColorLoopSet Command.

Table 30. Values of the Direction Field of the ColorLoopSet Command

Value	Description
0	Decrement the hue in the color loop.
1	Increment the hue in the color loop.

3.2.11.19.4. Time Field

The Time field specifies the number of seconds over which to perform a full color loop if the UpdateTime sub-field of the UpdateFlags field is set to 1.

3.2.11.19.5. Start Hue Field

The StartHue field specifies the starting hue to use for the color loop if the Update StartHue field of the Update Flags field is set to 1.

3.2.11.19.6. OptionsMask and OptionsOverride fields

The OptionsMask and OptionsOverride fields SHALL be processed according to section Use of the OptionsMask and OptionsOverride fields.

3.2.11.19.7. Effect on Receipt

On receipt of this command, the device SHALL first update its color loop attributes according to the value of the UpdateFlags field, as follows. If the UpdateDirection sub-field is set to 1, the device SHALL set the ColorLoopDirection attribute to the value of the Direction field. If the UpdateTime sub-field is set to 1, the device SHALL set the ColorLoopTime attribute to the value of the Time field. If the UpdateStartHue sub-field is set to 1, the device SHALL set the ColorLoopStartEnhancedHue attribute to the value of the StartHue field. If the color loop is active (and stays active), the device SHALL immediately react on updates of the ColorLoopDirection and ColorLoopTime attributes.

If the UpdateAction sub-field of the UpdateFlags field is set to 1, the device SHALL adhere to the action specified in the Action field, as follows. If the value of the Action field is set to 0 and the color loop is active, i.e. the ColorLoopActive attribute is set to 1, the device SHALL de-active the color loop, set the ColorLoopActive attribute to 0 and set the EnhancedCurrentHue attribute to the value of the ColorLoopStoredEnhancedHue attribute. If the value of the Action field is set to 0 and the color loop is inactive, i.e. the ColorLoopActive attribute is set to 0, the device SHALL ignore the action update component of the command. If the value of the action field is set to 1, the device SHALL set the ColorLoopStoredEnhancedHue attribute to the value of the EnhancedCurrentHue attribute, set the ColorLoopActive attribute to 1 and activate the color loop, starting from the value of the ColorLoopStartEnhancedHue attribute. If the value of the Action field is set to 2, the device SHALL set the ColorLoopStoredEnhancedHue attribute to the value of the EnhancedCurrentHue attribute, set the ColorLoopActive attribute to 1 and activate the color loop, starting from the value of the EnhancedCurrentHue attribute.

If the color loop is active, the device SHALL cycle over the complete range of values of the EnhancedCurrentHue attribute in the direction of the ColorLoopDirection attribute over the time specified in the ColorLoopTime attribute. The level of increments/decrements is application specific.

3.2.11.20. StopMoveStep Command

The StopMoveStep command is provided to allow MoveTo and Step commands to be stopped. (Note this automatically provides symmetry to the Level Control cluster.)

NOTE	the StopMoveStep command has no effect on an active color loop.

The StopMoveStep command SHALL have the following data fields:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	OptionsMask	map8	desc		0	M
1	OptionsOverride	map8	desc		0	M

3.2.11.20.1. OptionsMask and OptionsOverride fields

The OptionsMask and OptionsOverride fields SHALL be processed according to section Use of the OptionsMask and OptionsOverride fields.

3.2.11.20.2. Effect on Receipt

Upon receipt of this command, any MoveTo, Move or Step command currently in process SHALL be terminated. The values of the CurrentHue, EnhancedCurrentHue and CurrentSaturation attributes SHALL be left at their present value upon receipt of the StopMoveStep command, and the RemainingTime attribute SHALL be set to zero.

3.2.11.21. MoveColorTemperature Command

The MoveColorTemperature command allows the color temperature of a lamp to be moved at a specified rate.

The MoveColorTemperature command SHALL have the following data fields:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	MoveMode	map8	desc			M
1	Rate	uint16	all			M
2	ColorTemperatureMinimumMireds	uint16	0 to 0xFEFF			M
3	ColorTemperatureMaximumMireds	uint16	0 to 0xFEFF			M
4	OptionsMask	map8	desc		0	M
5	OptionsOverride	map8	desc		0	M

3.2.11.21.1. MoveMode Field

This field is identical to the MoveMode field of the MoveHue command of the Color Control cluster (see sub-clause MoveHue Command). If the MoveMode field is equal to 0 (Stop), the Rate field SHALL be ignored.

3.2.11.21.2. Rate Field

The Rate field specifies the rate of movement in steps per second. A step is a change in the color temperature of a device by one unit. If the MoveMode field is set to 1 (up) or 3 (down) and the Rate field has a value of zero, the command has no effect and a response command SHALL be sent in response, with the status code set to INVALID_COMMAND. If the MoveMode field is set to 0 (stop) the Rate field SHALL be ignored.

3.2.11.21.3. ColorTemperatureMinimumMireds Field

The ColorTemperatureMinimumMireds field specifies a lower bound on the ColorTemperatureMireds attribute (≡ an upper bound on the color temperature in kelvins) for the current move operation such that:

ColorTempPhysicalMinMireds <= ColorTemperatureMinimumMireds field <= ColorTemperatureMireds

As such if the move operation takes the ColorTemperatureMireds attribute towards the value of the ColorTemperatureMinimumMireds field it SHALL be clipped so that the above invariant is satisfied. If the ColorTemperatureMinimumMireds field is set to 0, ColorTempPhysicalMinMireds SHALL be used as the lower bound for the ColorTemperatureMireds attribute.

3.2.11.21.4. ColorTemperatureMaximumMireds Field

The ColorTemperatureMaximumMireds field specifies an upper bound on the ColorTemperatureMireds attribute (≡ a lower bound on the color temperature in kelvins) for the current move operation such that:

ColorTemperatureMireds <= ColorTemperatureMaximumMireds field <= ColorTempPhysicalMaxMireds

As such if the move operation takes the ColorTemperatureMireds attribute towards the value of the ColorTemperatureMaximumMireds field it SHALL be clipped so that the above invariant is satisfied. If the ColorTemperatureMaximumMireds field is set to 0, ColorTempPhysicalMaxMireds SHALL be used as the upper bound for the ColorTemperatureMireds attribute.

3.2.11.21.5. OptionsMask and OptionsOverride fields

The OptionsMask and OptionsOverride fields SHALL be processed according to section Use of the OptionsMask and OptionsOverride fields.

3.2.11.21.6. Effect on Receipt

On receipt of this command, a device SHALL set both the ColorMode and EnhancedColorMode attributes to 2. The device SHALL then move from its current color temperature in an up or down direction in a continuous fashion, as detailed in Actions on Receipt of the MoveColorTemperature Command.

Table 31. Actions on Receipt of the MoveColorTemperature Command

MoveMode	Action on Receipt
Stop	If moving, stop the operation, else ignore the command (i.e., the command is accepted but has no effect).
Up	Increase the ColorTemperatureMireds attribute (≡ decrease the color temperature in kelvins) at the rate given in the Rate field. If the ColorTemperatureMireds attribute reaches the maximum allowed for the device (via either the ColorTemperatureMaximumMireds field or the ColorTempPhysicalMaxMireds attribute), the move operation SHALL be stopped.
Down	Decrease the ColorTemperatureMireds attribute (≡ increase the color temperature in kelvins) at the rate given in the Rate field. If the ColorTemperatureMireds attribute reaches the minimum allowed for the device (via either the ColorTemperatureMinimumMireds field or the ColorTempPhysicalMinMireds attribute), the move operation SHALL be stopped.

3.2.11.22. StepColorTemperature Command

The StepColorTemperature command allows the color temperature of a lamp to be stepped with a specified step size.

The StepColorTemperature command SHALL have the following data fields:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	StepMode	map8	desc			M
1	StepSize	uint16	all			M
2	TransitionTime	uint16	0 to 65534			M
3	ColorTemperatureMinimumMireds	uint16	0 to 0xFEFF			M
4	ColorTemperatureMaximumMireds	uint16	0 to 0xFEFF			M
5	OptionsMask	map8	desc		0	M
6	OptionsOverride	map8	desc		0	M

3.2.11.22.1. StepMode Field

This field is identical to the StepMode field of the StepHue command of the Color Control cluster (see sub-clause StepHue Command).

3.2.11.22.2. StepSize Field

The StepSize field specifies the change to be added to (or subtracted from) the current value of the device's color temperature.

3.2.11.22.3. TransitionTime Field

The TransitionTime field specifies, in units of 1/10ths of a second, the time that SHALL be taken to perform the step. A step is a change to the device's color temperature of a magnitude corresponding to the StepSize field.

3.2.11.22.4. ColorTemperatureMinimumMireds Field

The ColorTemperatureMinimumMireds field specifies a lower bound on the ColorTemperatureMireds attribute (≡ an upper bound on the color temperature in kelvins) for the current step operation such that:

ColorTempPhysicalMinMireds <= ColorTemperatureMinimumMireds field <= ColorTemperatureMireds

As such if the step operation takes the ColorTemperatureMireds attribute towards the value of the Color Temperature Minimum Mireds field it SHALL be clipped so that the above invariant is satisfied. If the ColorTemperatureMinimumMireds field is set to 0, ColorTempPhysicalMinMireds SHALL be used as the lower bound for the ColorTemperatureMireds attribute.

3.2.11.22.5. ColorTemperatureMaximumMireds Field

The ColorTemperatureMaximumMireds field specifies an upper bound on the ColorTemperatureMireds attribute (≡ a lower bound on the color temperature in kelvins) for the current step operation such that:

ColorTemperatureMireds ≤ ColorTemperatureMaximumMireds field ≤ ColorTempPhysicalMaxMireds

As such if the step operation takes the ColorTemperatureMireds attribute towards the value of the ColorTemperatureMaximumMireds field it SHALL be clipped so that the above invariant is satisfied. If the ColorTemperatureMaximum Mireds field is set to 0, ColorTempPhysicalMaxMireds SHALL be used as the upper bound for the ColorTemperatureMireds attribute.

3.2.11.22.6. OptionsMask and OptionsOverride fields

The OptionsMask and OptionsOverride fields SHALL be processed according to section Use of the OptionsMask and OptionsOverride fields.

3.2.11.22.7. Effect on Receipt

On receipt of this command, a device SHALL set both the ColorMode and EnhancedColorMode attributes to 2. The device SHALL then move from its current color temperature in an up or down direction by one step, as detailed in Actions on Receipt of the StepColorTemperature Command.

Table 32. Actions on Receipt of the StepColorTemperature Command

StepMode	Action on Receipt
Up	Increase the ColorTemperatureMireds attribute (≡ decrease the color temperature in kelvins) by one step. If the ColorTemperatureMireds attribute reaches the maximum allowed for the device (via either the ColorTemperatureMaximumMireds field or the ColorTempPhysicalMaxMireds attribute), the step operation SHALL be stopped.
Down	Decrease the ColorTemperatureMireds attribute (≡ increase the color temperature in kelvins) by one step. If the ColorTemperatureMireds attribute reaches the minimum allowed for the device (via either the ColorTemperatureMinimumMireds field or the ColorTempPhysicalMinMireds attribute), the step operation SHALL be stopped.

3.3. Ballast Configuration Cluster

This cluster is used for configuring a lighting ballast.

NOTE Support for Ballast Configuration cluster is provisional.

3.3.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Revision	Description
1	Mandatory global ClusterRevision attribute added
2	CCB 2104 2193 2230 2393 Deprecated some attributes
3	CCB 2881
4	New data model format and notation

3.3.2. Classification

Hierarchy	Role	Scope	PICS Code
Base	Application	Endpoint	BC

3.3.3. Cluster ID

ID	Name	Conformance
0x0301	Ballast Configuration	P

3.3.4. Dependencies

If the Alarms server cluster is supported on the same endpoint then the Alarms functionality is enabled and the LampAlarmMode attribute SHALL be supported.

3.3.5. Data Types

3.3.5.1. BallastStatusBitmap Type

This data type is derived from map8.

Bit	Name	Summary
0	**BallastNonOperational**	Operational state of the ballast.
1	**LampFailure**	Operational state of the lamps.

3.3.5.1.1. BallastNonOperational Bit

This bit SHALL indicate whether the ballast is operational.

- 0 = The ballast is fully operational
- 1 = The ballast is not fully operational

3.3.5.1.2. LampFailure Bit

This bit SHALL indicate whether all lamps is operational.

- 0 = All lamps are operational

- 1 = One or more lamp is not in its socket or is faulty

3.3.5.2. LampAlarmModeBitmap Type

This data type is derived from map8.

Bit	Name	Summary
0	**LampBurnHours**	State of LampBurnHours alarm generation

3.3.5.2.1. LampBurnHours Bit

This bit SHALL indicate that the LampBurnHours attribute MAY generate an alarm.

3.3.6. Attributes

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0000	**PhysicalMinLevel**	uint8	1 to 254		1	R V	M
0x0001	**PhysicalMaxLevel**	uint8	1 to 254		254	R V	M
0x0002	**BallastStatus**	BallastStatusBitmap	all		0	R V	O
0x0010	**MinLevel**	uint8	PhysicalMinLevel to MaxLevel		PhysicalMinLevel	RW VM	M
0x0011	**MaxLevel**	uint8	MinLevel to PhysicalMaxLevel		PhysicalMaxLevel	RW VM	M
0x0012	**PowerOnLevel**						D
0x0013	**PowerOnFadeTime**						D
0x0014	**IntrinsicBallastFactor**	uint8	all	X		RW VM	O
0x0015	**BallastFactorAdjustment**	uint8	100 to MS	X	null	RW VM	O
0x0020	**LampQuantity**	uint8	all			R V	M

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0030	**LampType**	string	max 16		""	RW VM	O
0x0031	**LampManufacturer**	string	max 16		""	RW VM	O
0x0032	**LampRatedHours**	uint24	all	X	null	RW VM	O
0x0033	**LampBurnHours**	uint24	all	X	0	RW VM	O
0x0034	**LampAlarmMode**	LampAlarmModeBitmap	all		0	RW VM	O
0x0035	**LampBurnHoursTripPoint**	uint24	all	X	null	RW VM	O

3.3.6.1. PhysicalMinLevel Attribute

This attribute SHALL specify the minimum light output the ballast can achieve according to the dimming light curve (see Dimming Curve).

3.3.6.2. PhysicalMaxLevel Attribute

This attribute SHALL specify the maximum light output the ballast can achieve according to the dimming light curve (see Dimming Curve).

3.3.6.3. BallastStatus Attribute

This attribute SHALL specify the status of various aspects of the ballast or the connected lights, see BallastStatusBitmap.

3.3.6.4. MinLevel Attribute

This attribute SHALL specify the light output of the ballast according to the dimming light curve (see Dimming Curve) when the Level Control Cluster's CurrentLevel attribute equals to 1 (and the On/Off Cluster's OnOff attribute equals to TRUE).

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 response SHALL be returned with status code set to CONSTRAINT_ERROR, and the level SHALL NOT be set.

3.3.6.5. MaxLevel Attribute

This attribute SHALL specify the light output of the ballast according to the dimming light curve

(see Dimming Curve) when the Level Control Cluster's CurrentLevel attribute equals to 254 (and the On/Off Cluster's OnOff attribute equals to TRUE).

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 response SHALL be returned with status code set to CONSTRAINT_ERROR, and the level SHALL NOT be set.

3.3.6.6. IntrinsicBallastFactor Attribute

This attribute SHALL specify the ballast factor, as a percentage, of the ballast/lamp combination, prior to any adjustment.

A value of null indicates in invalid value.

3.3.6.7. BallastFactorAdjustment Attribute

This attribute SHALL specify the multiplication factor, as a percentage, to be applied to the configured light output of the lamps. A typical use for this attribute 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 response SHALL be returned with status code set to CONSTRAINT_ERROR, and the level SHALL NOT be changed. The value of null indicates that ballast factor scaling is not in use.

3.3.6.8. LampQuantity Attribute

This attribute SHALL specify 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).

3.3.6.9. LampType Attribute

This attribute SHALL specify the type of lamps (including their wattage) connected to the ballast.

3.3.6.10. LampManufacturer Attribute

This attribute SHALL specify the name of the manufacturer of the currently connected lamps.

3.3.6.11. LampRatedHours Attribute

This attribute SHALL specify the number of hours of use the lamps are rated for by the manufacturer.

A value of null indicates an invalid or unknown time.

3.3.6.12. LampBurnHours Attribute

This attribute SHALL specify 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 lamps are changed. If partially used lamps are connected, LampBurnHours SHOULD be updated to reflect the burn hours of the lamps.

A value of null indicates an invalid or unknown time.

3.3.6.13. LampAlarmMode Attribute

This attribute SHALL specify which attributes MAY cause an alarm notification to be generated. A 1 in each bit position means that its associated attribute is able to generate an alarm.

NOTE	All alarms are also logged in the alarm table – see Alarms cluster.

3.3.6.14. LampBurnHoursTripPoint Attribute

This attribute SHALL specify 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 Dependencies).

The Alarm Code field included in the generated alarm SHALL be 0x01.

If this attribute has the value of null, then this alarm SHALL NOT be generated.

3.3.7. 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 and

Level is an 8-bit integer between 1 (0.1% light output) and 254 (100% output) that is adjusted for MinLevel and MaxLevel using the following equation:

Level = (MaxLevel – MinLevel) * CurrentLevel / 253 + (254 * MinLevel – MaxLevel) / 253.

Note 1: Value 255 is not used.

Note 2: The light output is determined by this curve together with the IntrinsicBallastFactor and BallastFactorAdjustment attributes.

The table below gives a couple of examples of the dimming light curve for different values of MinLevel, MaxLevel and CurrentLevel.

Table 33. Examples of The Dimming Light Curve

MinLevel	MaxLevel	CurrentLevel	Level	%Light
1	254	1	1	0.10%
1	254	10	10	0.13%
1	254	100	100	1.49%
1	254	254	254	100%
170	254	1	170	10.1%
170	254	10	173	11.0%
170	254	100	203	24.8%
170	254	254	254	100%
170	230	1	170	10.1%
170	230	10	172	10.7%
170	230	100	193	19.2%
170	230	254	230	51.9%

Chapter 4. HVAC

The Cluster Library is made of individual chapters such as this one. References between chapters are made using a *X.Y* notation where *X* is the chapter and *Y* is the sub-section within that chapter.

4.1. General Description

4.1.1. Introduction

The clusters specified in this document are for use typically in HVAC applications, but MAY be used in any application domain.

4.1.2. Terms

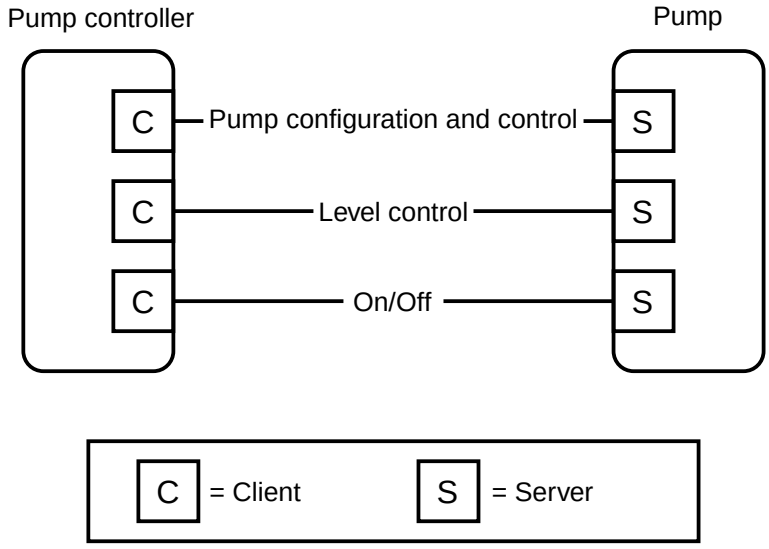
Precooling: Cooling a building in the early (cooler) part of the day, so that the thermal mass of the building decreases cooling needs in the later (hotter) part of the day.

4.1.3. Cluster List

This section lists the HVAC specific clusters as specified in this chapter.

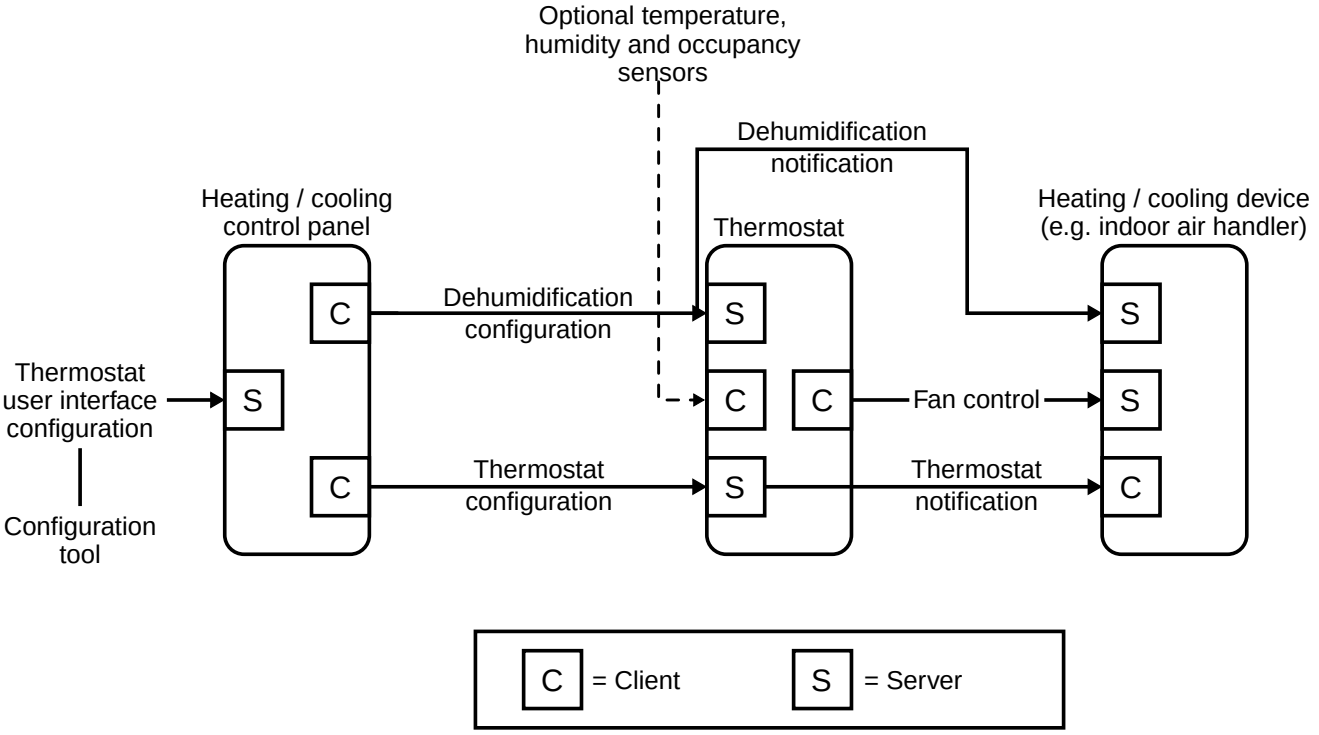
Table 34. Overview of the HVAC Clusters

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
0x0204	Thermostat User Interface Configuration	An interface for configuring the user interface of a thermostat (which MAY be remote from the thermostat)
0x0081	Valve Configuration and Control	An interface for configuring and controlling the functionality of a valve



Note: Device names are examples for illustration purposes only

Figure 12. Typical Usage of Pump Configuration and Control Cluster



Note: Device names are examples for illustration purposes only

Figure 13. Example Usage of the Thermostat and Related Clusters"

4.2. Pump Configuration and Control Cluster

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.

4.2.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table

below.

Revision	Description
1	Mandatory global ClusterRevision attribute added
2	All Hubs changes
3	New data model format and notation, added additional events
4	Added feature map

4.2.2. Classification

Hierarchy	Role	Scope	PICS Code
Base	Application	Endpoint	PCC

4.2.3. Cluster ID

ID	Name
0x0200	Pump Configuration and Control

4.2.4. Features

This cluster SHALL support the FeatureMap bitmap attribute as defined below.

Bit	Code	Feature	Conformance	Summary
0	**PRSCONST**	ConstantPressure	O.a+	Supports operating in constant pressure mode
1	**PRSCOMP**	CompensatedPressure	O.a+	Supports operating in compensated pressure mode
2	**FLW**	ConstantFlow	O.a+	Supports operating in constant flow mode
3	**SPD**	ConstantSpeed	O.a+	Supports operating in constant speed mode
4	**TEMP**	ConstantTemperature	O.a+	Supports operating in constant temperature mode

Bit	Code	Feature	Conformance	Summary
5	**AUTO**	Automatic	O	Supports operating in automatic mode
6	**LOCAL**	LocalOperation	O	Supports operating using local settings

4.2.5. Dependencies

Where external pressure, flow, and temperature measurements are processed by this cluster (see ControlMode attribute), these are provided by a Pressure Measurement cluster, a Flow Measurement cluster, and a Temperature Measurement client cluster, 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.

Note that control of the pump setpoint is not included in this cluster – the On/Off and Level Control clusters (see Typical Usage of Pump Configuration and Control Cluster) 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 OperationMode attribute for detailed description of the operation and control of the pump.).

4.2.6. Data Types

4.2.6.1. PumpStatusBitmap Type

This data type is derived from map16.

Bit	Name	Summary
0	**DeviceFault**	A fault related to the system or pump device is detected.
1	**SupplyFault**	A fault related to the supply to the pump is detected.
2	**SpeedLow**	Setpoint is too low to achieve.
3	**SpeedHigh**	Setpoint is too high to achieve.
4	**LocalOverride**	Device control is overridden by hardware, such as an external STOP button or via a local HMI.
5	**Running**	Pump is currently running
6	**RemotePressure**	A remote pressure sensor is used as the sensor for the regulation of the pump.

Bit	Name	Summary
7	**RemoteFlow**	A remote flow sensor is used as the sensor for the regulation of the pump.
8	**RemoteTemperature**	A remote temperature sensor is used as the sensor for the regulation of the pump.

4.2.6.1.1. DeviceFault Bit

If this bit is set, it MAY correspond to an event in the range 2-16, see Events.

4.2.6.1.2. SupplyFault Bit

If this bit is set, it MAY correspond to an event in the range 0-1 or 13, see Events.

4.2.6.1.3. LocalOverride Bit

While this bit is set, the EffectiveOperationMode is adjusted to Local. Any request changing OperationMode SHALL generate a FAILURE error status until LocalOverride is cleared on the physical device. When LocalOverride is cleared, the device SHALL return to the operation mode set in OperationMode.

4.2.6.1.4. RemotePressure Bit

If this bit is set, EffectiveControlMode is ConstantPressure and the setpoint for the pump is interpreted as a percentage of the range of the remote sensor ([MinMeasuredValue – MaxMeasuredValue]).

4.2.6.1.5. RemoteFlow Bit

If this bit is set, EffectiveControlMode is ConstantFlow, and the setpoint for the pump is interpreted as a percentage of the range of the remote sensor ([MinMeasuredValue – MaxMeasuredValue]).

4.2.6.1.6. RemoteTemperature Bit

If this bit is set, EffectiveControlMode is ConstantTemperature, and the setpoint for the pump is interpreted as a percentage of the range of the remote sensor ([MinMeasuredValue – MaxMeasuredValue])

4.2.6.2. OperationModeEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**Normal**	The pump is controlled by a setpoint, as defined by a connected remote sensor or by the ControlMode attribute.	M
1	**Minimum**	This value sets the pump to run at the minimum possible speed it can without being stopped.	SPD
2	**Maximum**	This value sets the pump to run at its maximum possible speed.	SPD
3	**Local**	This value sets the pump to run with the local settings of the pump, regardless of what these are.	LOCAL

4.2.6.2.1. Normal Value

If the pump is running in this operation mode the setpoint is an internal variable which MAY be controlled between 0% and 100%, e.g., by means of the Level Control cluster

4.2.6.3. ControlModeEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**ConstantSpeed**	The pump is running at a constant speed.	SPD
1	**ConstantPressure**	The pump will regulate its speed to maintain a constant differential pressure over its flanges.	PRSCONST
2	**ProportionalPressure**	The pump will regulate its speed to maintain a constant differential pressure over its flanges.	PRSCOMP

Value	Name	Summary	Conformance
3	**ConstantFlow**	The pump will regulate its speed to maintain a constant flow through the pump.	FLW
5	**ConstantTemperature**	The pump will regulate its speed to maintain a constant temperature.	TEMP
7	**Automatic**	The operation of the pump is automatically optimized to provide the most suitable performance with respect to comfort and energy savings.	AUTO

4.2.6.3.1. ConstantSpeed Value

The setpoint is interpreted as a percentage of the range derived from the [MinConstSpeed – MaxConstSpeed] attributes.

4.2.6.3.2. ConstantPressure Value

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.

4.2.6.3.3. ProportionalPressure Value

The setpoint is interpreted as a percentage of the range derived of the [MinCompPressure – MaxCompPressure] attributes. The internal setpoint will be lowered (compensated) dependent on the flow in the pump (lower flow ⇒ lower internal setpoint).

4.2.6.3.4. ConstantFlow Value

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.

4.2.6.3.5. ConstantTemperature Value

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.

4.2.6.3.6. Automatic Value

This behavior is manufacturer defined. The pump can be stopped by setting the setpoint of the level control cluster to 0, or 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.

4.2.7. Attributes

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0000	**MaxPressure**	int16	all	X F	null	R V	M
0x0001	**MaxSpeed**	uint16	all	X F	null	R V	M
0x0002	**MaxFlow**	uint16	all	X F	null	R V	M
0x0003	**MinConstPressure**	int16	all	X F	null	R V	PRSCONST, [AUTO]
0x0004	**MaxConstPressure**	int16	all	X F	null	R V	PRSCONST, [AUTO]
0x0005	**MinCompPressure**	int16	all	X F	null	R V	PRSCOMP, [AUTO]
0x0006	**MaxCompPressure**	int16	all	X F	null	R V	PRSCOMP, [AUTO]
0x0007	**MinConstSpeed**	uint16	all	X F	null	R V	SPD, [AUTO]
0x0008	**MaxConstSpeed**	uint16	all	X F	null	R V	SPD, [AUTO]
0x0009	**MinConstFlow**	uint16	all	X F	null	R V	FLW, [AUTO]
0x000A	**MaxConstFlow**	uint16	all	X F	null	R V	FLW, [AUTO]
0x000B	**MinConstTemp**	int16	min -27315	X F	null	R V	TEMP, [AUTO]
0x000C	**MaxConstTemp**	int16	min -27315	X F	null	R V	TEMP, [AUTO]
0x0010	**PumpStatus**	PumpStatusBitmap	desc	P	0	R V	O
0x0011	**EffectiveOperationMode**	OperationModeEnum	desc	N	desc	R V	M
0x0012	**EffectiveControlMode**	ControlModeEnum	desc	N	desc	R V	M

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0013	**Capacity**	int16	all	X P	null	R V	M
0x0014	**Speed**	uint16	all	X	null	R V	O
0x0015	**LifetimeRunningHours**	uint24	all	X N	0	RW VM	O
0x0016	**Power**	uint24	all	X	null	R V	O
0x0017	**LifetimeEnergyConsumed**	uint32	all	X N	0	RW VM	O
0x0020	**OperationMode**	OperationModeEnum	desc	N	0	RW VM	M
0x0021	**ControlMode**	ControlModeEnum	desc	N	0	RW VM	O
0x0022	**AlarmMask**	map16	desc	N	0	R V	D

4.2.7.1. MaxPressure Attribute

This 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.

Valid range is -3,276.7 kPa to 3,276.7 kPa (steps of 0.1 kPa).
This attribute SHALL be null if the value is invalid.

4.2.7.2. MaxSpeed Attribute

This 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.

Valid range is 0 to 65,534 RPM (steps of 1 RPM).
This attribute SHALL be null if the value is invalid.

4.2.7.3. MaxFlow Attribute

This 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.

Valid range is 0 m^3/h to 6,553.4 m^3/h (steps of 0.1 m^3/h).
This attribute SHALL be null if the value is invalid.

4.2.7.4. MinConstPressure Attribute

This attribute specifies the minimum pressure the pump can achieve when it is working with the ControlMode attribute set to ConstantPressure.

Valid range is –3,276.7 kPa to 3,276.7 kPa (steps of 0.1 kPa).
This attribute SHALL be null if the value is invalid.

4.2.7.5. MaxConstPressure Attribute

This attribute specifies the maximum pressure the pump can achieve when it is working with the ControlMode attribute set to ConstantPressure.

Valid range is –3,276.7 kPa to 3,276.7 kPa (steps of 0.1 kPa).
This attribute SHALL be null if the value is invalid.

4.2.7.6. MinCompPressure Attribute

This attribute specifies the minimum compensated pressure the pump can achieve when it is working with the ControlMode attribute set to ProportionalPressure.

Valid range is –3,276.7 kPa to 3,276.7 kPa (steps of 0.1 kPa).
This attribute SHALL be null if the value is invalid.

4.2.7.7. MaxCompPressure Attribute

This attribute specifies the maximum compensated pressure the pump can achieve when it is working with the ControlMode attribute set to ProportionalPressure.

Valid range is –3,276.7 kPa to 3,276.7 kPa (steps of 0.1 kPa).
This attribute SHALL be null if the value is invalid.

4.2.7.8. MinConstSpeed Attribute

This attribute specifies the minimum speed the pump can achieve when it is working with the ControlMode attribute set to ConstantSpeed.

Valid range is 0 to 65,534 RPM (steps of 1 RPM).
This attribute SHALL be null if the value is invalid.

4.2.7.9. MaxConstSpeed Attribute

This attribute specifies the maximum speed the pump can achieve when it is working with the ControlMode attribute set to ConstantSpeed.

Valid range is 0 to 65,534 RPM (steps of 1 RPM).
This attribute SHALL be null if the value is invalid.

4.2.7.10. MinConstFlow Attribute

This attribute specifies the minimum flow the pump can achieve when it is working with the Con-

trolMode attribute set to ConstantFlow.

Valid range is 0 m^3/h to 6,553.4 m^3/h (steps of 0.1 m^3/h).
This attribute SHALL be null if the value is invalid.

4.2.7.11. MaxConstFlow Attribute

This attribute specifies the maximum flow the pump can achieve when it is working with the ControlMode attribute set to ConstantFlow.

Valid range is 0 m^3/h to 6,553.4 m^3/h (steps of 0.1 m^3/h).
This attribute SHALL be null if the value is invalid.

4.2.7.12. MinConstTemp Attribute

This attribute specifies the minimum temperature the pump can maintain in the system when it is working with the ControlMode attribute set to ConstantTemperature.

Valid range is –273.15 °C to 327.67 °C (steps of 0.01 °C).
This attribute SHALL be null if the value is invalid.

4.2.7.13. MaxConstTemp Attribute

This attribute specifies the maximum temperature the pump can maintain in the system when it is working with the ControlMode attribute set to ConstantTemperature.

MaxConstTemp SHALL be greater than or equal to MinConstTemp

Valid range is –273.15 °C to 327.67 °C (steps of 0.01 °C).
This attribute SHALL be null if the value is invalid.

4.2.7.14. PumpStatus Attribute

This attribute specifies the activity status of the pump functions as listed in PumpStatusBitmap. 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.

4.2.7.15. EffectiveOperationMode Attribute

This attribute specifies current effective operation mode of the pump as defined in OperationModeEnum.

The value of the EffectiveOperationMode attribute is the same as the OperationMode attribute, unless one of the following points are true:

- The pump is physically set to run with the local settings
- The LocalOverride bit in the PumpStatus attribute is set,

See OperationMode and ControlMode attributes for a detailed description of the operation and control of the pump.

4.2.7.16. EffectiveControlMode Attribute

This attribute specifies the current effective control mode of the pump as defined in ControlModeEnum.

This attribute contains the control mode that currently applies to the pump. It will have the value of the ControlMode attribute, unless one of the following points are true:

- The ControlMode attribute is set to Automatic. In this case, the value of the EffectiveControlMode SHALL match the behavior of the pump.
- A remote sensor is used as the sensor for regulation of the pump. In this case, EffectiveControlMode will display ConstantPressure, ConstantFlow or ConstantTemperature if the remote sensor is a pressure sensor, a flow sensor or a temperature sensor respectively, regardless of the value of the ControlMode attribute.

In case the ControlMode attribute is not included on the device and no remote sensors are connected, the value of the EffectiveControlMode SHALL match the vendor-specific behavior of the pump.

See OperationMode and ControlMode attributes for detailed a description of the operation and control of the pump.

4.2.7.17. Capacity Attribute

This 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.

If the value is not available (the measurement or estimation of the speed is done in the pump), this attribute will indicate the null value.

Valid range is 0 % to 163.835% (0.005 % granularity). Although this attribute is a signed value, values of capacity less than zero have no physical meaning.

4.2.7.18. Speed Attribute

This attribute specifies the actual speed of the pump measured in RPM. It is updated dynamically as the speed of the pump changes.

If the value is not available (the measurement or estimation of the speed is done in the pump), this attribute will indicate the null value.

Valid range is 0 to 65.534 RPM.

4.2.7.19. LifetimeRunningHours Attribute

This 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 indicate the null value.

Valid range is 0 to 16,777,214 hrs.

4.2.7.20. Power Attribute

This attribute specifies the actual power consumption of the pump in Watts. The value of this 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 indicate the null value.

Valid range is 0 to 16,777,214 Watts.

4.2.7.21. LifetimeEnergyConsumed Attribute

This 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.

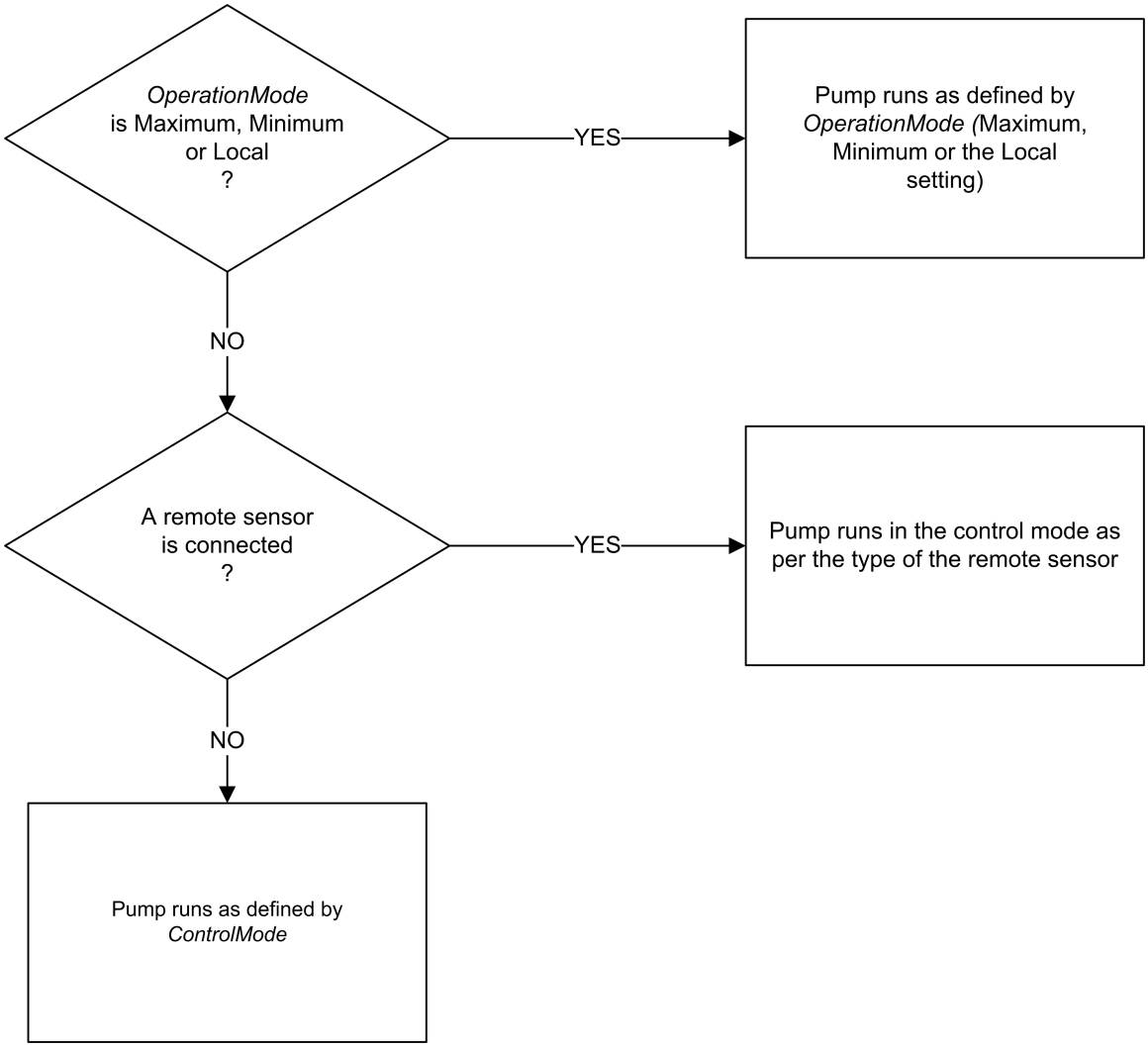
Valid range is 0 kWh to 4,294,967,294 kWh.
This attribute SHALL be null if the value is unknown.

4.2.7.22. OperationMode Attribute

This attribute specifies the operation mode of the pump as defined in OperationModeEnum.

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 below:

Priority Scheme of Pump Operation and Control



If this attribute is Maximum, Minimum or Local, the OperationMode attribute decides how the pump is operated.

If this 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 this 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 this attribute is vendor-specific.

In the case a device does not support a specific operation mode, the write interaction to this attribute with an unsupported operation mode value SHALL be ignored and a response containing the status of CONSTRAINT_ERROR SHALL be returned.

4.2.7.23. ControlMode Attribute

This attribute specifies the control mode of the pump as defined in ControlModeEnum.

See the OperationMode attribute for a 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.

In the case a device does not support a specific control mode, the write interaction to this attribute with an unsupported control mode value SHALL be ignored and a response containing the status of CONSTRAINT_ERROR SHALL be returned.

4.2.8. Events

ID	Name	Priority	Access	Conformance
0x00	**SupplyVoltageLow**	INFO	V	O
0x01	**SupplyVoltageHigh**	INFO	V	O
0x02	**PowerMissingPhase**	INFO	V	O
0x03	**SystemPressureLow**	INFO	V	O
0x04	**SystemPressureHigh**	INFO	V	O
0x05	**DryRunning**	CRITICAL	V	O
0x06	**MotorTemperatureHigh**	INFO	V	O
0x07	**PumpMotorFatalFailure**	CRITICAL	V	O
0x08	**ElectronicTemperatureHigh**	INFO	V	O
0x09	**PumpBlocked**	CRITICAL	V	O
0x0A	**SensorFailure**	INFO	V	O
0x0B	**ElectronicNonFatalFailure**	INFO	V	O
0x0C	**ElectronicFatalFailure**	CRITICAL	V	O
0x0D	**GeneralFault**	INFO	V	O
0x0E	**Leakage**	INFO	V	O
0x0F	**AirDetection**	INFO	V	O
0x10	**TurbineOperation**	INFO	V	O

4.3. Thermostat Cluster

This cluster provides an interface to the functionality of a thermostat.

4.3.1. Revision History

The global *ClusterRevision* attribute value SHALL be the highest revision number in the table below.

Revision	Description
1	Mandatory global *ClusterRevision* attribute added; fixed some defaults; CCB 1823, 1480
2	CCB 1981 2186 2249 2250 2251; NFR Thermostat Setback
3	CCB 2477 2560 2773 2777 2815 2816 3029
4	All Hubs changes
5	New data model format and notation, added FeatureMap, collapsed attribute sets, clarified edge cases around limits, default value of xxxSetpointLimit now respects AbsxxxSetpointLimit
6	Introduced the LTNE feature and adapted text (spec issue #5778)

4.3.2. Classification

Hierarchy	Role	Scope	PICS Code
Base	Application	Endpoint	TSTAT

4.3.3. Cluster ID

ID	Name
0x0201	Thermostat

4.3.4. Features

This cluster SHALL support the FeatureMap bitmap attribute as defined below.

Bit	Code	Feature	Conformance	Summary
0	**HEAT**	Heating	AUTO, O.a+	Thermostat is capable of managing a heating device

Bit	Code	Feature	Conformance	Summary
1	**COOL**	Cooling	AUTO, O.a+	Thermostat is capable of managing a cooling device
2	**OCC**	Occupancy	O	Supports Occupied and Unoccupied setpoints
3	**SCH**	ScheduleConfiguration	[Zigbee], D	Supports remote configuration of a weekly schedule of setpoint transitions
4	**SB**	Setback	O	Supports configurable setback (or span)
5	**AUTO**	AutoMode	O	Supports a System Mode of Auto
6	**LTNE**	LocalTemperatureNotExposed	O	Thermostat does not expose the LocalTemperature Value in the LocalTemperature attribute

4.3.4.1. LocalTemperatureNotExposed Feature

This feature indicates that the Calculated Local Temperature used internally is unavailable to report externally, for example due to the temperature control being done by a separate subsystem which does not offer a view into the currently measured temperature, but allows setpoints to be provided.

4.3.5. Units of Temperature

Temperatures within this cluster are represented by types using units of degree Celsius.

The temperature data type used throughout this cluster is defined in the Derived Data Types section of the Data Model.

The following temperature-related data types are also defined in and used throughout this cluster:

- TemperatureDifference
- SignedTemperature Type
- UnsignedTemperature Type

While temperature values MUST be transferred over the air using these types, this does not limit

how Thermostats may display or store temperature values. Thermostats which display temperature values SHOULD follow the recommendations in Conversion of Temperature Values for Display.

CAUTION	*Calculations with temperature attributes* Where calculations or comparisons are performed, attribute values must be converted to a common type. In many cases, it is not sufficient to simply use the integer representation as the scaling from °C to integer value differs.

4.3.6. Setpoint Limits

There are a number of attributes which impose limits on setpoint values. This imposes constraints which MUST be maintained by any mechanism which modifies a limit or setpoint. Individual attribute descriptions detail the actions to be taken should a conflict arise while modifying the value.

User configurable limits must be within device limits:

- AbsMinHeatSetpointLimit <= MinHeatSetpointLimit <= MaxHeatSetpointLimit <= AbsMaxHeatSetpointLimit
- AbsMinCoolSetpointLimit <= MinCoolSetpointLimit <= MaxCoolSetpointLimit <= AbsMaxCoolSetpointLimit

Setpoints must be within user configurable limits:

- MinHeatSetpointLimit <= OccupiedHeatingSetpoint <= MaxHeatSetpointLimit
- MinCoolSetpointLimit <= OccupiedCoolingSetpoint <= MaxCoolSetpointLimit
- MinHeatSetpointLimit <= UnoccupiedHeatingSetpoint <= MaxHeatSetpointLimit
- MinCoolSetpointLimit <= UnoccupiedCoolingSetpoint <= MaxCoolSetpointLimit

and if, and only if, the AUTO feature is supported, a deadband must be maintained between Heating and Cooling setpoints and limits:

- AbsMinHeatSetpointLimit <= (AbsMinCoolSetpointLimit - MinSetpointDeadBand)
- AbsMaxHeatSetpointLimit <= (AbsMaxCoolSetpointLimit - MinSetpointDeadBand)
- MinHeatSetpointLimit <= (MinCoolSetpointLimit - MinSetpointDeadBand)
- MaxHeatSetpointLimit <= (MaxCoolSetpointLimit - MinSetpointDeadBand)
- OccupiedHeatingSetpoint <= (OccupiedCoolingSetpoint - MinSetpointDeadBand)
- UnoccupiedHeatingSetpoint <= (UnoccupiedCoolingSetpoint - MinSetpointDeadBand)

4.3.7. Dependencies

If the Alarms server cluster is supported on the same endpoint then the Alarms functionality is enabled and the AlarmMask attribute SHALL be supported. For remote temperature sensing, the Temperature Measurement client cluster MAY be included on the same endpoint. For occupancy sensing, the Occupancy Sensing client cluster MAY be included on the same endpoint.

4.3.8. Data Types

4.3.8.1. TemperatureDifference Type

This data type is derived from int16 and represents a temperature difference with a resolution of 0.01°C.

- *value* = (*temperature in °C*) x 100

- -4°C ⇒ -400
- 123.45°C ⇒ 12345

The full (non-null) range of -327.67°C to 327.67°C may be used.

4.3.8.2. SignedTemperature Type

This data type is derived from int8 and represents a temperature from -12.7°C to 12.7°C with a resolution of 0.1°C.

- *value* = (*temperature in °C*) x 10

- -4°C ⇒ -40
- 12.3°C ⇒ 123

This type is employed where compactness of representation is important and where the resolution and range are still satisfactory.

4.3.8.3. UnsignedTemperature Type

This data type is derived from uint8 and represents a temperature from 0°C to 25.5°C with a resolution of 0.1°C.

- *value* = (*temperature in °C*) x 10

- 4°C ⇒ 40
- 12.3°C ⇒ 123

This type is employed where compactness of representation is important and where the resolution and range are still satisfactory.

4.3.8.4. ACErrorCodeBitmap Type

This data type is derived from map32.

Bit	Name	Summary	Conformance
0	**CompressorFail**	Compressor Failure or Refrigerant Leakage	M
1	**RoomSensorFail**	Room Temperature Sensor Failure	M
2	**OutdoorSensorFail**	Outdoor Temperature Sensor Failure	M
3	**CoilSensorFail**	Indoor Coil Temperature Sensor Failure	M
4	**FanFail**	Fan Failure	M

4.3.8.5. AlarmCodeBitmap Type

This data type is derived from map8.

Bit	Name	Summary	Conformance
0	**Initialization**	Initialization failure. The device failed to complete initialization at power-up.	M
1	**Hardware**	Hardware failure	M
2	**SelfCalibration**	Self-calibration failure	M

4.3.8.6. HVACSystemTypeBitmap Type

This data type is derived from map8.

Bit	Name	Summary	Conformance
0..1	**CoolingStage**	Stage of cooling the HVAC system is using.	M
2..3	**HeatingStage**	Stage of heating the HVAC system is using.	M
4	**HeatingIsHeatPump**	Is the heating type Heat Pump.	M
5	**HeatingUsesFuel**	Does the HVAC system use fuel.	M

4.3.8.6.1. CoolingStage Bits

These bits SHALL indicate what stage of cooling the HVAC system is using.

- 00 = Cool Stage 1
- 01 = Cool Stage 2

- 10 = Cool Stage 3
- 11 = Reserved

4.3.8.6.2. HeatingStage Bits

These bits SHALL indicate what stage of heating the HVAC system is using.

- 00 = Heat Stage 1
- 01 = Heat Stage 2
- 10 = Heat Stage 3
- 11 = Reserved

4.3.8.6.3. HeatingIsHeatPump Bit

This bit SHALL indicate whether the HVAC system is conventional or a heat pump.

- 0 = Conventional
- 1 = Heat Pump

4.3.8.6.4. HeatingUsesFuel Bit

This bit SHALL indicate whether the HVAC system uses fuel.

- 0 = Does not use fuel
- 1 = Uses fuel

4.3.8.7. ProgrammingOperationModeBitmap Type

This data type is derived from map8.

Bit	Name	Summary	Conformance
0	**ScheduleActive**	Schedule programming mode. This enables any programmed weekly schedule configura-tions.	M
1	**AutoRecovery**	Auto/recovery mode	M
2	**Economy**	Economy/EnergyStar mode	M

4.3.8.8. RelayStateBitmap Type

This data type is derived from map16.

Bit	Name	Summary	Conformance
0	**Heat**	Heat State On	M

Bit	Name	Summary	Conformance
1	**Cool**	Cool State On	M
2	**Fan**	Fan State On	M
3	**HeatStage2**	Heat 2nd State On	M
4	**CoolStage2**	Cool 2nd State On	M
5	**FanStage2**	Fan 2nd State On	M
6	**FanStage3**	Fan 3rd Stage On	M

4.3.8.9. RemoteSensingBitmap Type

This data type is derived from map8.

Bit	Name	Summary	Conformance
0	**LocalTemperature**	Calculated Local Temperature is derived from a remote node	M
1	**OutdoorTemperature**	OutdoorTemperature is derived from a remote node	desc
2	**Occupancy**	Occupancy is derived from a remote node	OCC

4.3.8.9.1. OutdoorTemperature Bit

This bit SHALL be supported if the OutdoorTemperature attribute is supported.

4.3.8.10. ScheduleDayOfWeekBitmap Type

This data type is derived from map8.

Bit	Name	Summary	Conformance
0	**Sunday**	Sunday	M
1	**Monday**	Monday	M
2	**Tuesday**	Tuesday	M
3	**Wednesday**	Wednesday	M
4	**Thursday**	Thursday	M
5	**Friday**	Friday	M
6	**Saturday**	Saturday	M
7	**Away**	Away or Vacation	M

4.3.8.11. ScheduleModeBitmap Type

This data type is derived from map8.

Bit	Name	Summary	Conformance
0	**HeatSetpointPresent**	Adjust Heat Setpoint	M
1	**CoolSetpointPresent**	Adjust Cool Setpoint	M

4.3.8.12. ACCapacityFormatEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**BTUh**	British Thermal Unit per Hour	O

4.3.8.13. ACCompressorTypeEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**Unknown**	Unknown compressor type	O
1	**T1**	Max working ambient 43 °C	O
2	**T2**	Max working ambient 35 °C	O
3	**T3**	Max working ambient 52 °C	O

4.3.8.14. ACLouverPositionEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
1	**Closed**	Fully Closed	O
2	**Open**	Fully Open	O
3	**Quarter**	Quarter Open	O
4	**Half**	Half Open	O
5	**ThreeQuarters**	Three Quarters Open	O

4.3.8.15. ACRefrigerantTypeEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**Unknown**	Unknown Refrigerant Type	O
1	**R22**	R22 Refrigerant	O
2	**R410a**	R410a Refrigerant	O
3	**R407c**	R407c Refrigerant	O

4.3.8.16. ACTypeEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**Unknown**	Unknown AC Type	O
1	**CoolingFixed**	Cooling and Fixed Speed	O
2	**HeatPumpFixed**	Heat Pump and Fixed Speed	O
3	**CoolingInverter**	Cooling and Inverter	O
4	**HeatPumpInverter**	Heat Pump and Inverter	O

4.3.8.17. SetpointRaiseLowerModeEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**Heat**	Adjust Heat Setpoint	HEAT
1	**Cool**	Adjust Cool Setpoint	COOL
2	**Both**	Adjust Heat Setpoint and Cool Setpoint	HEAT | COOL

4.3.8.18. ControlSequenceOfOperationEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**CoolingOnly**	Heat and Emergency are not possible	[COOL]
1	**CoolingWithReheat**	Heat and Emergency are not possible	[COOL]

Value	Name	Summary	Conformance
2	**HeatingOnly**	Cool and precooling (see Terms) are not possible	[HEAT]
3	**HeatingWithReheat**	Cool and precooling are not possible	[HEAT]
4	**CoolingAndHeating**	All modes are possible	[HEAT & COOL]
5	**CoolingAndHeating-WithReheat**	All modes are possible	[HEAT & COOL]

NOTE

CoolingAndHeating

A thermostat indicating it supports CoolingAndHeating (or CoolingAndHeatingWithReheat) SHOULD be able to request heating or cooling on demand and will usually support the Auto SystemMode.

Systems which support cooling **or** heating, requiring external intervention to change modes or where the whole building must be in the same mode, SHOULD report CoolingOnly or HeatingOnly based on the current capability.

4.3.8.19. SetpointChangeSourceEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**Manual**	Manual, user-initiated setpoint change via the thermostat	O
1	**Schedule**	Schedule/internal programming-initiated setpoint change	[SCH]
2	**External**	Externally-initiated setpoint change (e.g., DRLC cluster command, attribute write)	O

4.3.8.20. StartOfWeekEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**Sunday**		M
1	**Monday**		M
2	**Tuesday**		M

Value	Name	Summary	Conformance
3	**Wednesday**		M
4	**Thursday**		M
5	**Friday**		M
6	**Saturday**		M

4.3.8.21. SystemModeEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**Off**	The Thermostat does not generate demand for Cooling or Heating	O
1	**Auto**	Demand is generated for either Cooling or Heating, as required	AUTO
3	**Cool**	Demand is only generated for Cooling	[COOL]
4	**Heat**	Demand is only generated for Heating	[HEAT]
5	**EmergencyHeat**	2nd stage heating is in use to achieve desired temperature	[HEAT]
6	**Precooling**	(see Terms)	[COOL]
7	**FanOnly**		O
8	**Dry**		O
9	**Sleep**		O

Table 35. Interpretation of Heat, Cool and Auto SystemModeEnum 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

4.3.8.22. ThermostatRunningModeEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**Off**	The Thermostat does not generate demand for Cooling or Heating	O
3	**Cool**	Demand is only generated for Cooling	[COOL]
4	**Heat**	Demand is only generated for Heating	[HEAT]

4.3.8.23. TemperatureSetpointHoldEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**SetpointHoldOff**	Follow scheduling program	M
1	**SetpointHoldOn**	Maintain current setpoint, regardless of schedule transitions	M

4.3.8.24. WeeklyScheduleTransitionStruct Type

This represents a single transition in a Thermostat schedule

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0	**TransitionTime**	uint16	max 1439				M
1	**HeatSetpoint**	temperature	all	X			M
2	**CoolSetpoint**	temperature	all	X			M

4.3.8.24.1. TransitionTime Field

This field SHALL represent the start time of the schedule transition during the associated day. The time will be represented by a 16 bits unsigned integer to designate the minutes since midnight. For example, 6am will be represented by 360 minutes since midnight and 11:30pm will be represented by 1410 minutes since midnight.

4.3.8.24.2. HeatSetpoint Field

This field SHALL represent the heat setpoint to be applied at this associated transition start time.

4.3.8.24.3. CoolSetpoint Field

This field SHALL represent the cool setpoint to be applied at this associated transition start time.

4.3.9. Attributes

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0000	**LocalTemperature**	temperature	all	XP	null	R V	M
0x0001	**OutdoorTemperature**	temperature	all	X	null	R V	O
0x0002	**Occupancy**	OccupancyBitmap	desc		1	R V	OCC
0x0003	**AbsMinHeatSetpointLimit**	temperature	desc	F	7°C	R V	[HEAT]
0x0004	**AbsMaxHeatSetpointLimit**	temperature	desc	F	30°C	R V	[HEAT]
0x0005	**AbsMinCoolSetpointLimit**	temperature	desc	F	16°C	R V	[COOL]
0x0006	**AbsMaxCoolSetpointLimit**	temperature	desc	F	32°C	R V	[COOL]
0x0007	**PICoolingDemand**	uint8	0% to 100%	P	-	R V	[COOL]
0x0008	**PIHeatingDemand**	uint8	0% to 100%	P	-	R V	[HEAT]
0x0009	**HVACSystemTypeConfiguration**	HVACSystemTypeBitmap	desc	N	0	R[W] VM	D

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0010	**LocalTemperatureCalibration**	SignedTemperatureType	-2.5°C to 2.5°C	N	0°C	RW VM	[!LTNE]
0x0011	**OccupiedCoolingSetpoint**	temperature	desc	NS	26°C	RW VO	COOL
0x0012	**OccupiedHeatingSetpoint**	temperature	desc	NS	20°C	RW VO	HEAT
0x0013	**UnoccupiedCoolingSetpoint**	temperature	desc	N	26°C	RW VO	COOL & OCC
0x0014	**UnoccupiedHeatingSetpoint**	temperature	desc	N	20°C	RW VO	HEAT & OCC
0x0015	**MinHeatSetpointLimit**	temperature	desc	N	AbsMinHeatSetpointLimit	RW VM	[HEAT]
0x0016	**MaxHeatSetpointLimit**	temperature	desc	N	AbsMaxHeatSetpointLimit	RW VM	[HEAT]
0x0017	**MinCoolSetpointLimit**	temperature	desc	N	AbsMinCoolSetpointLimit	RW VM	[COOL]
0x0018	**MaxCoolSetpointLimit**	temperature	desc	N	AbsMaxCoolSetpointLimit	RW VM	[COOL]
0x0019	**MinSetpointDeadBand**	SignedTemperatureType	0°C to 2.5°C	N	2.5°C	R[W] VM	AUTO
0x001A	**RemoteSensing**	RemoteSensingBitmap	00000xxx	N	0	RW VM	O

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x001B	**ControlSequenceOfOperation**	ControlSequenceOfOperationEnum	desc	N	4	RW VM	M
0x001C	**SystemMode**	SystemModeEnum	desc	NS	1	RW VM	M
0x001D	**AlarmMask**	AlarmCodeBitmap	desc		0	R V	[Zigbee]
0x001E	**ThermostatRunningMode**	ThermostatRunningModeEnum	desc		0	R V	[AUTO]
0x0020	**StartOfWeek**	StartOfWeekEnum	desc	F	–	R V	SCH
0x0021	**NumberOfWeeklyTransitions**	uint8	all	F	0	R V	SCH
0x0022	**NumberOfDailyTransitions**	uint8	all	F	0	R V	SCH
0x0023	**TemperatureSetpointHold**	TemperatureSetpointHoldEnum	desc	N	0	RW VM	O
0x0024	**TemperatureSetpointHoldDuration**	uint16	max 1440	NX	null	RW VM	O
0x0025	**ThermostatProgrammingOperationMode**	ProgrammingOperationModeBitmap	desc	P	0	RW VM	O
0x0029	**ThermostatRunningState**	RelayStateBitmap	desc		-	R V	O

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0030	**SetpointChangeSource**	SetpointChangeSourceEnum	desc		0	R V	O
0x0031	**SetpointChangeAmount**	TemperatureDifference	all	X	null	R V	O
0x0032	**SetpointChangeSourceTimestamp**	utc	all		0	R V	O
0x0034	**OccupiedSetback**	UnsignedTemperature Type	OccupiedSetbackMin to OccupiedSetbackMax	XN	null	RW VM	SB
0x0035	**OccupiedSetbackMin**	UnsignedTemperature Type	max OccupiedSetbackMax	XF	null	R V	SB
0x0036	**OccupiedSetbackMax**	UnsignedTemperature Type	OccupiedSetbackMin to 25.4°C	XF	null	R V	SB
0x0037	**UnoccupiedSetback**	UnsignedTemperature Type	UnoccupiedSetbackMin to UnoccupiedSetbackMax	XN	null	RW VM	SB & OCC
0x0038	**UnoccupiedSetbackMin**	UnsignedTemperature Type	max UnoccupiedSetbackMax	XF	null	R V	SB & OCC
0x0039	**UnoccupiedSetbackMax**	UnsignedTemperature Type	UnoccupiedSetbackMin to 25.4°C	XF	null	R V	SB & OCC

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x003A	**EmergencyHeatDelta**	UnsignedTemperature Type	all	N	25.5°C	RW VM	O
0x0040	**ACType**	ACTypeEnum	desc	N	0	RW VM	O
0x0041	**ACCapacity**	uint16	all	N	0	RW VM	O
0x0042	**ACRefrigerantType**	ACRefrigerantTypeEnum	desc	N	0	RW VM	O
0x0043	**ACCompressorType**	ACCompressorTypeEnum	desc	N	0	RW VM	O
0x0044	**ACErrorCode**	ACErrorCodeBitmap	all		0	RW VM	O
0x0045	**ACLouverPosition**	ACLouverPositionEnum	desc	N	0	RW VM	O
0x0046	**ACCoilTemperature**	temperature	all	X	null	R V	O
0x0047	**ACCapacityFormat**	ACCapacityFormatEnum	desc	N	0	RW VM	O

4.3.9.1. Scene Table Extensions

If the Scenes Management server cluster is implemented on the same endpoint, the following attributes SHALL be part of the ExtensionFieldSetStruct of the Scene Table. If the implicit form of the ExtensionFieldSetStruct is used, the order of the attributes in the AttributeValueList is in the given order, i.e., the attribute listed as 1 is added first:

1. OccupiedCoolingSetpoint
2. OccupiedHeatingSetpoint
3. SystemMode

4.3.9.2. Calculated Local Temperature

The local temperature SHALL be calculated from the *measured temperature*, including any adjustments applied by LocalTemperatureCalibration attribute (if any) as follows:

Calculated Local Temperature = (*measured temperature*) + LocalTemperatureCalibration

The *measured temperature* value may be local, or remote, depending on the value of the RemoteSensing attribute when supported.

All setpoint attributes in the Thermostat cluster SHALL be triggered based off this calculated value (i.e., measured temperature and any calibration offset).

If the LocalTemperatureNotExposed feature is present, the behavior of the thermostat SHALL be that the equipment's temperature control uses the calculated local temperature even if that value is not reported in the LocalTemperature attribute.

4.3.9.3. LocalTemperature Attribute

This attribute SHALL indicate the current Calculated Local Temperature, when available.

- If the LTNE feature is not supported:
  - If the LocalTemperatureCalibration is invalid or currently unavailable, the attribute SHALL report null.
  - If the LocalTemperatureCalibration is valid, the attribute SHALL report that value.
- Otherwise, if the LTNE feature is supported, there is no feedback externally available for the LocalTemperatureCalibration. In that case, the LocalTemperature attribute SHALL always report null.

4.3.9.4. OutdoorTemperature Attribute

This attribute SHALL indicate the outdoor temperature, as measured locally or remotely (over the network).

4.3.9.5. Occupancy Attribute

This attribute SHALL indicate whether the heated/cooled space is occupied or not, as measured locally or remotely (over the network).

4.3.9.6. AbsMinHeatSetpointLimit Attribute

This attribute SHALL indicate the absolute minimum level that the heating setpoint MAY be set to. This is a limitation imposed by the manufacturer.

Refer to Setpoint Limits for constraints

4.3.9.7. AbsMaxHeatSetpointLimit Attribute

This attribute SHALL indicate the absolute maximum level that the heating setpoint MAY be set to. This is a limitation imposed by the manufacturer.

Refer to Setpoint Limits for constraints

4.3.9.8. AbsMinCoolSetpointLimit Attribute

This attribute SHALL indicate the absolute minimum level that the cooling setpoint MAY be set to. This is a limitation imposed by the manufacturer.

Refer to Setpoint Limits for constraints

4.3.9.9. AbsMaxCoolSetpointLimit Attribute

This attribute SHALL indicate the absolute maximum level that the cooling setpoint MAY be set to. This is a limitation imposed by the manufacturer.

Refer to Setpoint Limits for constraints

4.3.9.10. PICoolingDemand Attribute

This attribute SHALL indicate 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 cooling device.

4.3.9.11. PIHeatingDemand Attribute

This attribute SHALL indicate 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 heating device.

4.3.9.12. HVACSystemTypeConfiguration Attribute

This attribute SHALL indicate the HVAC system type controlled by the thermostat. If the thermostat uses physical DIP switches to set these parameters, this information SHALL be available read-only from the DIP switches. If these parameters are set via software, there SHALL be read/write access in order to provide remote programming capability.

4.3.9.13. LocalTemperatureCalibration Attribute

This attribute SHALL indicate the offset the Thermostat server SHALL make to the measured temperature (locally or remotely) to adjust the Calculated Local Temperature prior to using, displaying or reporting it.

The purpose of this attribute is to adjust the calibration of the Thermostat server per the user’s preferences (e.g., to match if there are multiple servers displaying different values for the same HVAC area) or compensate for variability amongst temperature sensors.

If a Thermostat client attempts to write LocalTemperatureCalibration attribute to an unsupported value (e.g., out of the range supported by the Thermostat server), the Thermostat server SHALL respond with a status of SUCCESS and set the value of LocalTemperatureCalibration to the upper or lower limit reached.

4.3.9.14. OccupiedCoolingSetpoint Attribute

This attribute SHALL indicate the cooling mode setpoint when the room is occupied.

Refer to Setpoint Limits for constraints. If an attempt is made to set this attribute such that these constraints are violated, a response with the status code CONSTRAINT_ERROR SHALL be returned.

If the occupancy status of the room is unknown, this attribute SHALL be used as the cooling mode setpoint.

4.3.9.15. OccupiedHeatingSetpoint Attribute

This attribute SHALL indicate the heating mode setpoint when the room is occupied.

Refer to Setpoint Limits for constraints. If an attempt is made to set this attribute such that these constraints are violated, a response with the status code CONSTRAINT_ERROR SHALL be returned.

If the occupancy status of the room is unknown, this attribute SHALL be used as the heating mode setpoint.

4.3.9.16. UnoccupiedCoolingSetpoint Attribute

This attribute SHALL indicate the cooling mode setpoint when the room is unoccupied.

Refer to Setpoint Limits for constraints. If an attempt is made to set this attribute such that these constraints are violated, a response with the status code CONSTRAINT_ERROR SHALL be returned.

If the occupancy status of the room is unknown, this attribute SHALL NOT be used.

4.3.9.17. UnoccupiedHeatingSetpoint Attribute

This attribute SHALL indicate the heating mode setpoint when the room is unoccupied.

Refer to Setpoint Limits for constraints. If an attempt is made to set this attribute such that these constraints are violated, a response with the status code CONSTRAINT_ERROR SHALL be returned.

If the occupancy status of the room is unknown, this attribute SHALL NOT be used.

4.3.9.18. MinHeatSetpointLimit Attribute

This attribute SHALL indicate the minimum level that the heating setpoint MAY be set to.

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.

Refer to Setpoint Limits for constraints. If an attempt is made to set this attribute to a value which conflicts with setpoint values then those setpoints SHALL be adjusted by the minimum amount to permit this attribute to be set to the desired value. If an attempt is made to set this attribute to a value which is not consistent with the constraints and cannot be resolved by modifying setpoints then a response with the status code CONSTRAINT_ERROR SHALL be returned.

4.3.9.19. MaxHeatSetpointLimit Attribute

This attribute SHALL indicate the maximum level that the heating setpoint MAY be set to.

Refer to Setpoint Limits for constraints. If an attempt is made to set this attribute to a value which conflicts with setpoint values then those setpoints SHALL be adjusted by the minimum amount to permit this attribute to be set to the desired value. If an attempt is made to set this attribute to a value which is not consistent with the constraints and cannot be resolved by modifying setpoints then a response with the status code CONSTRAINT_ERROR SHALL be returned.

4.3.9.20. MinCoolSetpointLimit Attribute

This attribute SHALL indicate the minimum level that the cooling setpoint MAY be set to.

Refer to Setpoint Limits for constraints. If an attempt is made to set this attribute to a value which conflicts with setpoint values then those setpoints SHALL be adjusted by the minimum amount to permit this attribute to be set to the desired value. If an attempt is made to set this attribute to a value which is not consistent with the constraints and cannot be resolved by modifying setpoints then a response with the status code CONSTRAINT_ERROR SHALL be returned.

4.3.9.21. MaxCoolSetpointLimit Attribute

This attribute SHALL indicate the maximum level that the cooling setpoint MAY be set to.

Refer to Setpoint Limits for constraints. If an attempt is made to set this attribute to a value which conflicts with setpoint values then those setpoints SHALL be adjusted by the minimum amount to permit this attribute to be set to the desired value. If an attempt is made to set this attribute to a value which is not consistent with the constraints and cannot be resolved by modifying setpoints then a response with the status code CONSTRAINT_ERROR SHALL be returned.

4.3.9.22. MinSetpointDeadBand Attribute

On devices which support the AUTO feature, this attribute SHALL indicate the minimum difference between the Heat Setpoint and the Cool Setpoint.

Refer to Setpoint Limits for constraints. If an attempt is made to set this attribute to a value which conflicts with setpoint values then those setpoints SHALL be adjusted by the minimum amount to permit this attribute to be set to the desired value. If an attempt is made to set this attribute to a value which is not consistent with the constraints and cannot be resolved by modifying setpoints then a response with the status code CONSTRAINT_ERROR SHALL be returned.

4.3.9.23. RemoteSensing Attribute

This attribute SHALL indicate when the local temperature, outdoor temperature and occupancy are being sensed by remote networked sensors, rather than internal sensors.

If the LTNE feature is present in the server, the LocalTemperature RemoteSensing bit value SHALL always report a value of 0.

If the LocalTemperature RemoteSensing bit is written with a value of 1 when the LTNE feature is present, the write SHALL fail and the server SHALL report a CONSTRAINT_ERROR.

4.3.9.24. ControlSequenceOfOperation Attribute

This attribute SHALL indicate the overall operating environment of the thermostat, and thus the possible system modes that the thermostat can operate in.

4.3.9.25. SystemMode Attribute

This attribute SHALL indicate the current operating mode of the thermostat. Its value SHALL be limited by the ControlSequenceOfOperation attribute.

4.3.9.26. AlarmMask Attribute

This attribute SHALL indicate whether each of the alarms in AlarmCodeBitmap is enabled.

When the Alarms cluster is implemented on a device, and one of the alarm conditions included in AlarmCodeBitmap occurs, an alarm notification is generated, with the alarm code field set as listed in AlarmCodeBitmap.

4.3.9.27. ThermostatRunningMode Attribute

This attribute SHALL indicate the running mode of the thermostat. This attribute uses the same values as SystemModeEnum but can only be Off, Cool or Heat. This attribute is intended to provide additional information when the thermostat's system mode is in auto mode.

4.3.9.28. StartOfWeek Attribute

This attribute SHALL indicate the day of the week that this thermostat considers to be the start of week for weekly setpoint scheduling.

This attribute MAY be able to be used as the base to determine if the device supports weekly scheduling by reading the attribute. Successful response means that the weekly scheduling is supported.

4.3.9.29. NumberOfWeeklyTransitions Attribute

This attribute SHALL indicate how many weekly schedule transitions the thermostat is capable of handling.

4.3.9.30. NumberOfDailyTransitions Attribute

This attribute SHALL indicate how many daily schedule transitions the thermostat is capable of handling.

4.3.9.31. TemperatureSetpointHold Attribute

This attribute SHALL indicate the temperature hold status on the thermostat. If hold status is on, the thermostat SHOULD maintain the temperature setpoint for the current mode until a system mode change. If hold status is off, the thermostat SHOULD follow the setpoint transitions specified by its internal scheduling program. If the thermostat supports setpoint hold for a specific duration, it SHOULD also implement the TemperatureSetpointHoldDuration attribute.

4.3.9.32. TemperatureSetpointHoldDuration Attribute

This attribute SHALL indicate the period in minutes for which a setpoint hold is active. Thermostats that support hold for a specified duration SHOULD implement this attribute. The null value indicates the field is unused. All other values are reserved.

4.3.9.33. ThermostatProgrammingOperationMode Attribute

This attribute SHALL indicate the operational state of the thermostat's programming. The thermostat SHALL modify its programming operation when this attribute is modified by a client and update this attribute when its programming operation is modified locally by a user. The thermostat MAY support more than one active ProgrammingOperationModeBitmap. For example, the thermostat MAY operate simultaneously in Schedule Programming Mode and Recovery Mode.

Thermostats which contain a schedule MAY use this attribute to control how that schedule is used, even if they do not support the ScheduleConfiguration feature.

When ScheduleActive is not set, the setpoint is altered only by manual up/down changes at the thermostat or remotely, not by internal schedule programming.

NOTE	Modifying the ScheduleActive bit does not clear or delete previous weekly schedule programming configurations.

4.3.9.34. ThermostatRunningState Attribute

This attribute SHALL indicate the current relay state of the heat, cool, and fan relays.

Unimplemented outputs SHALL be treated as if they were Off.

4.3.9.35. SetpointChangeSource Attribute

This attribute SHALL indicate the source of the current active OccupiedCoolingSetpoint or OccupiedHeatingSetpoint (i.e., who or what determined the current setpoint).

This attribute enables service providers to determine whether changes to setpoints were initiated due to occupant comfort, scheduled programming or some other source (e.g., electric utility or other service provider). Because automation services MAY initiate frequent setpoint changes, this attribute clearly differentiates the source of setpoint changes made at the thermostat.

4.3.9.36. SetpointChangeAmount Attribute

This attribute SHALL indicate the delta between the current active OccupiedCoolingSetpoint or OccupiedHeatingSetpoint and the previous active setpoint. This attribute is meant to accompany the SetpointChangeSource attribute; devices implementing SetpointChangeAmount SHOULD also implement SetpointChangeSource.

The null value indicates that the previous setpoint was unknown.

4.3.9.37. SetpointChangeSourceTimestamp Attribute

This attribute SHALL indicate the time in UTC at which the SetpointChangeAmount attribute change was recorded.

4.3.9.38. OccupiedSetback Attribute

This attribute SHALL indicate the amount that the Thermostat server will allow the Calculated Local Temperature to float above the OccupiedCoolingSetpoint (i.e., OccupiedCoolingSetpoint + OccupiedSetback) or below the OccupiedHeatingSetpoint setpoint (i.e., OccupiedHeatingSetpoint – OccupiedSetback) before initiating a state change to bring the temperature back to the user's desired setpoint. This attribute is sometimes also referred to as the "span."

The purpose of this attribute is to allow remote configuration of the span between the desired setpoint and the measured temperature to help prevent over-cycling and reduce energy bills, though this may result in lower comfort on the part of some users.

The null value indicates the attribute is unused.

If the Thermostat client attempts to write OccupiedSetback to a value greater than OccupiedSetbackMax, the Thermostat server SHALL set its OccupiedSetback value to OccupiedSetbackMax and SHALL send a Write Attribute Response command with a Status Code field enumeration of SUCCESS response.

If the Thermostat client attempts to write OccupiedSetback to a value less than OccupiedSetbackMin, the Thermostat server SHALL set its OccupiedSetback value to OccupiedSetbackMin and SHALL send a Write Attribute Response command with a Status Code field enumeration of SUCCESS response.

4.3.9.39. OccupiedSetbackMin Attribute

This attribute SHALL indicate the minimum value that the Thermostat server will allow the OccupiedSetback attribute to be configured by a user.

The null value indicates the attribute is unused.

4.3.9.40. OccupiedSetbackMax Attribute

This attribute SHALL indicate the maximum value that the Thermostat server will allow the OccupiedSetback attribute to be configured by a user.

The null value indicates the attribute is unused.

4.3.9.41. UnoccupiedSetback Attribute

This attribute SHALL indicate the amount that the Thermostat server will allow the Calculated Local Temperature to float above the UnoccupiedCoolingSetpoint (i.e., UnoccupiedCoolingSetpoint + UnoccupiedSetback) or below the UnoccupiedHeatingSetpoint setpoint (i.e., UnoccupiedHeatingSetpoint - UnoccupiedSetback) before initiating a state change to bring the temperature back to the user's desired setpoint. This attribute is sometimes also referred to as the "span."

The purpose of this attribute is to allow remote configuration of the span between the desired set-point and the measured temperature to help prevent over-cycling and reduce energy bills, though this may result in lower comfort on the part of some users.

The null value indicates the attribute is unused.

If the Thermostat client attempts to write UnoccupiedSetback to a value greater than UnoccupiedSetbackMax, the Thermostat server SHALL set its UnoccupiedSetback value to UnoccupiedSetbackMax and SHALL send a Write Attribute Response command with a Status Code field enumeration of SUCCESS response.

If the Thermostat client attempts to write UnoccupiedSetback to a value less than UnoccupiedSetbackMin, the Thermostat server SHALL set its UnoccupiedSetback value to UnoccupiedSetbackMin and SHALL send a Write Attribute Response command with a Status Code field enumeration of SUCCESS response.

4.3.9.42. UnoccupiedSetbackMin Attribute

This attribute SHALL indicate the minimum value that the Thermostat server will allow the UnoccupiedSetback attribute to be configured by a user.

The null value indicates the attribute is unused.

4.3.9.43. UnoccupiedSetbackMax Attribute

This attribute SHALL indicate the maximum value that the Thermostat server will allow the UnoccupiedSetback attribute to be configured by a user.

The null value indicates the attribute is unused.

4.3.9.44. EmergencyHeatDelta Attribute

This attribute SHALL indicate the delta between the Calculated Local Temperature and the OccupiedHeatingSetpoint or UnoccupiedHeatingSetpoint attributes at which the Thermostat server will operate in emergency heat mode.

If the difference between the Calculated Local Temperature and OccupiedCoolingSetpoint or UnoccupiedCoolingSetpoint is greater than or equal to the EmergencyHeatDelta and the Thermostat server's SystemMode attribute is in a heating-related mode, then the Thermostat server SHALL immediately switch to the SystemMode attribute value that provides the highest stage of heating (e.g., emergency heat) and continue operating in that running state until the OccupiedHeatingSetpoint value is reached. For example:

- Calculated Local Temperature = 10.0°C
- OccupiedHeatingSetpoint = 16.0°C
- EmergencyHeatDelta = 2.0°C

⇒ OccupiedHeatingSetpoint - Calculated Local Temperature ≥? EmergencyHeatDelta

⇒ 16°C - 10°C ≥? 2°C

⇒ TRUE >>> Thermostat server changes its SystemMode to operate in 2nd stage or emergency heat mode

The purpose of this attribute is to provide Thermostat clients the ability to configure rapid heating when a setpoint is of a specified amount greater than the measured temperature. This allows the heated space to be quickly heated to the desired level set by the user.

4.3.9.45. ACType Attribute

This attribute SHALL indicate the type of Mini Split ACTypeEnum of Mini Split AC is defined depending on how Cooling and Heating condition is achieved by Mini Split AC.

4.3.9.46. ACCapacity Attribute

This attribute SHALL indicate capacity of Mini Split AC in terms of the format defined by the ACCapacityFormat attribute

4.3.9.47. ACRefrigerantType Attribute

This attribute SHALL indicate type of refrigerant used within the Mini Split AC.

4.3.9.48. ACCompressorType Attribute

This attribute SHALL indicate the type of compressor used within the Mini Split AC.

4.3.9.49. ACErrorCode Attribute

This attribute SHALL indicate the type of errors encountered within the Mini Split AC.

4.3.9.50. ACLouverPosition Attribute

This attribute SHALL indicate the position of Louver on the AC.

4.3.9.51. ACCoilTemperature Attribute

This attribute SHALL indicate the temperature of the AC coil, as measured locally or remotely (over the network).

4.3.9.52. ACCapacityFormat Attribute

This attribute SHALL indicate the format for the ACCapacity attribute.

4.3.10. Commands

ID	Name	Direction	Response	Access	Conformance
0x00	**SetpointRaiseLower**	client ⇒ server	Y	O	M
0x01	**SetWeeklySchedule**	client ⇒ server	Y	M	SCH
0x02	**GetWeeklySchedule**	client ⇒ server	GetWeeklyScheduleResponse	O	SCH
0x00	**GetWeeklyScheduleResponse**	client ⇐ server	N		SCH
0x03	**ClearWeeklySchedule**	client ⇒ server	Y	M	SCH
0x04	**GetRelayStatusLog**	client ⇒ server	GetRelayStatusLogResponse	O	[Zigbee]
0x01	**GetRelayStatusLogResponse**	client ⇐ server	N		GetRelayStatusLog

4.3.10.1. SetpointRaiseLower Command

Upon receipt, the attributes for the indicated setpoint(s) SHALL have the amount specified in the Amount field added to them. If the resulting value is outside the limits imposed by MinCoolSetpointLimit, MaxCoolSetpointLimit, MinHeatSetpointLimit and MaxHeatSetpointLimit, the value is clamped to those limits. This is not considered an error condition.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**Mode**	SetpointRaiseLowerModeEnum	desc			M
1	**Amount**	int8	all			M

4.3.10.2. Mode Field

The field SHALL specify which setpoints are to be adjusted.

4.3.10.2.1. Heat Bit

If the server does not support the HEAT feature then it SHALL respond with INVALID_ARGUMENT. If the server supports the AUTO feature and the resulting setpoint would be invalid solely due to MinSetpointDeadBand then the Cooling setpoint SHALL be increased sufficiently to maintain the deadband.

4.3.10.2.2. Cool Bit

If the server does not support the COOL feature then it SHALL respond with INVALID_ARGUMENT. If the server supports the AUTO feature and the resulting setpoint would be invalid solely due to MinSetpointDeadBand then the Heating setpoint SHALL be decreased sufficiently to maintain the deadband.

4.3.10.2.3. Both Bit

The client MAY indicate Both regardless of the server feature support. The server SHALL only adjust the setpoint that it supports and not respond with an error.

4.3.10.3. Amount Field

This field SHALL indicate the amount (possibly negative) that should be added to the setpoint(s), in steps of 0.1°C.

4.3.10.4. SetWeeklySchedule Command

Upon receipt, the weekly schedule for updating setpoints SHALL be stored in the thermostat and SHOULD begin at the time of receipt. A status code SHALL be sent in response.

When a command is received that requires a total number of transitions greater than the device supports, the status of the response SHALL be INSUFFICIENT_SPACE.

When any of the setpoints sent in the sequence is out of range (AbsMin/MaxSetPointLimit), or when the Mode for Sequence field includes a mode not supported by the device, the status of the response SHALL be CONSTRAINT_ERROR and no setpoints from the entire sequence SHOULD be used.

When an overlapping transition is detected, the status of the response SHALL be FAILURE.

When a device which does not support multiple days in a command receives a command with more than one bit set in the DayOfWeekForSequence field, or when a device which does not support multiple modes in a command receives a command with more than one bit set in the ModeForSequence field, or when the contents of the Transitions field does not agree with NumberOfTransitionsForSequence, DayOfWeekForSequence or ModeForSequence, the status of the response SHALL be INVALID_COMMAND.

When the transitions could be added successfully, the status of the response SHALL be SUCCESS.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**NumberOfTransitionsForSequence**	uint8	all			M
1	**DayOfWeekForSequence**	Schedule-DayOfWeek-Bitmap	desc			M

ID	Name	Type	Constraint	Quality	Default	Conformance
2	**ModeForSequence**	ScheduleModeBitmap	desc			M
3	**Transitions**	list[WeeklyScheduleTransitionStruct]	max 10			M

The set weekly schedule command is used to update the thermostat weekly setpoint schedule from a management system. If the thermostat already has a weekly setpoint schedule programmed, then it SHOULD replace each daily setpoint set as it receives the updates from the management system. For example, if the thermostat has 4 setpoints for every day of the week and is sent a Set Weekly Schedule command with one setpoint for Saturday then the thermostat SHOULD remove all 4 setpoints for Saturday and replace those with the updated setpoint but leave all other days unchanged. If the schedule is larger than what fits in one frame or contains more than 10 transitions, the schedule SHALL then be sent using multiple Set Weekly Schedule Commands.

4.3.10.4.1. NumberOfTransitionsForSequence Field

This field SHALL indicate how many individual transitions to expect for this sequence of commands. If a device supports more than 10 transitions in its schedule they can send this by sending more than 1 “Set Weekly Schedule” command, each containing the separate information that the device needs to set.

4.3.10.4.2. DayOfWeekForSequence Field

This field SHALL represent the day of the week at which all the transitions within the payload of the command SHOULD be associated to. This field is a bitmap and therefore the associated setpoint could overlap onto multiple days (you could set one transition time for all “week days” or whatever combination of days the implementation requests).

Each setpoint transition will begin with the day of week for this transition. There can be up to 10 transitions for each command.

4.3.10.4.3. ModeForSequence Field

This field SHALL indicate how the application decodes the setpoint fields of each transition in the Transitions list.

If the HeatSetpointPresent bit is On, the HeatSetpoint field SHALL NOT be null in **every** entry of the Transitions list.

If the HeatSetpointPresent bit is Off, the HeatSetpoint field SHALL be null in **every** entry of the Transitions list.

If the CoolSetpointPresent bit is On, the CoolSetpoint field SHALL NOT be null in **every** entry of the Transitions list.

If the CoolSetpointPresent bit is Off, the CoolSetpoint field SHALL be null in **every** entry of the Transitions list.

At least one of the bits in the Mode For Sequence byte SHALL be on.

Both bits must be respected, even if the HEAT or COOL feature is not supported, to ensure the command is decoded and handled correctly.

4.3.10.4.4. Transitions Field

This field SHALL contain the list of setpoint transitions used to update the specified daily schedules

4.3.10.5. GetWeeklySchedule Command

Upon receipt, the unit SHOULD send in return the Get Weekly Schedule Response command. The Days to Return and Mode to Return fields are defined as bitmask for the flexibility to support multiple days and multiple modes within one command. If thermostat cannot handle incoming command with multiple days and/or multiple modes within one command, it SHALL send default response of INVALID_COMMAND in return.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**DaysToReturn**	Schedule-DayOfWeek-Bitmap	desc			M
1	**ModeToReturn**	Schedule-ModeBitmap	desc			M

4.3.10.5.1. DaysToReturn Field

This field SHALL indicate the number of days the client would like to return the setpoint values for and could be any combination of single days or the entire week.

4.3.10.5.2. ModeToReturn Field

This field SHALL indicate the mode the client would like to return the set point values for and could be any combination of heat only, cool only or heat & cool.

4.3.10.6. GetWeeklyScheduleResponse Command

This command has the same payload format as the Set Weekly Schedule.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**NumberOfTransitionsForSequence**	uint8	all			M

ID	Name	Type	Constraint	Quality	Default	Conformance
1	**DayOfWeekForSequence**	Schedule-DayOfWeek-Bitmap	desc			M
2	**ModeForSequence**	Schedule-ModeBitmap	desc			M
3	**Transitions**	list[WeeklySchedule-Transition-Struct]	max 10			M

4.3.10.7. ClearWeeklySchedule Command

This command is used to clear the weekly schedule. The Clear weekly schedule has no payload.

Upon receipt, all transitions currently stored SHALL be cleared and a default response of SUCCESS SHALL be sent in response. There are no error responses to this command.

4.3.10.8. GetRelayStatusLog Command

This command is used to query the thermostat internal relay status log. This command has no payload.

Upon receipt, the unit SHALL respond with Relay Status Log command if the relay status log feature is supported on the unit.

The log storing order is First in First Out (FIFO) when the log is generated and stored into the Queue.

The first record in the log (i.e., the oldest) one, is the first to be replaced when there is a new record and there is no more space in the log. Thus, the newest record will overwrite the oldest one if there is no space left.

The log storing order is Last In First Out (LIFO) when the log is being retrieved from the Queue by a client device.

Once the "Get Relay Status Log Response" frame is sent by the Server, the "Unread Entries" attribute SHOULD be decremented to indicate the number of unread records that remain in the queue.

If the "Unread Entries" attribute reaches zero and the Client sends a new "Get Relay Status Log Request", the Server MAY send one of the following items as a response:

i. Resend the last Get Relay Status Log Response

   or

ii. Generate new log record at the time of request and send Get Relay Status Log Response with the new data

For both cases, the "Unread Entries" attribute will remain zero.

4.3.10.9. GetRelayStatusLogResponse Command

This command is sent from the thermostat cluster server in response to the Get Relay Status Log. After the Relay Status Entry is sent over the air to the requesting client, the specific entry will be cleared from the thermostat internal log.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**TimeOfDay**	uint16	max 1439			M
1	**RelayStatus**	RelayStateBitmap	desc			M
2	**LocalTemperature**	temperature	all	X		M
3	**HumidityInPercentage**	uint8	0% to 100%	X		M
4	**SetPoint**	temperature	all			M
5	**UnreadEntries**	uint16	all			M

4.3.10.9.1. TimeOfDay Field

This field SHALL indicate the sample time of the day, in minutes since midnight, when the relay status was captured for this associated log entry. For example, 6am will be represented by 360 minutes since midnight and 11:30pm will be represented by 1410 minutes since midnight.

4.3.10.9.2. RelayStatus Field

This field SHALL indicate the relay status for thermostat when the log is captured. Each bit represents one relay used by the thermostat. If the bit is on, the associated relay is on and active. Each thermostat manufacturer can create its own mapping between the bitmap and the associated relay.

4.3.10.9.3. LocalTemperature Field

This field SHALL indicate the LocalTemperature when the log is captured. The null value indicates that LocalTemperature was invalid or unavailable.

4.3.10.9.4. Humidity Field

This field SHALL indicate the humidity as a percentage when the log was captured. The null value indicates that the humidity value was invalid or unknown.

4.3.10.9.5. Setpoint Field

This field SHALL indicate the target setpoint temperature when the log is captured.

4.3.10.9.6. UnreadEntries Field

This field SHALL indicate the number of unread entries within the thermostat internal log system.

4.4. Fan Control Cluster

This cluster specifies an interface to control the speed of a fan.

4.4.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Revision	Description
1	Global mandatory ClusterRevision attribute added
2	New data model format and notation; Percent, speed and motion settings; General cleanup
3	Addition of AirflowDirection and Step command
4	Change conformance for FanModeSequenceEnum

4.4.2. Classification

Hierarchy	Role	Scope	PICS Code
Base	Application	Endpoint	FAN

4.4.3. Cluster ID

ID	Name
0x0202	Fan Control

4.4.4. Features

This cluster SHALL support the FeatureMap bitmap attribute as defined below.

Bit	Code	Feature	Summary
0	**SPD**	MultiSpeed	0-SpeedMax Fan Speeds
1	**AUT**	Auto	Automatic mode supported for fan speed
2	**RCK**	Rocking	Rocking movement supported

Bit	Code	Feature	Summary
3	**WND**	Wind	Wind emulation supported
4	**STEP**	Step	Step command supported
5	**DIR**	AirflowDirection	Airflow Direction attribute is supported

4.4.4.1. MultiSpeed Feature

Legacy Fan Control cluster revision 0-1 defined 3 speeds (low, medium and high) plus automatic speed control but left it up to the implementer to decide what was supported. Therefore, it is assumed that legacy client implementations are capable of determining, from the server, the number of speeds supported between 1, 2, or 3, and whether automatic speed control is supported.

The MultiSpeed feature includes new attributes that support a running fan speed value from 0 to SpeedMax, which has a maximum of 100.

See Speed Rules for more details.

4.4.5. Data Types

4.4.5.1. RockBitmap Type

Bit	Name	Summary	Conformance
0	**RockLeftRight**	Indicate rock left to right	M
1	**RockUpDown**	Indicate rock up and down	M
2	**RockRound**	Indicate rock around	M

4.4.5.2. WindBitmap Type

Bit	Name	Summary	Conformance
0	**SleepWind**	Indicate sleep wind	M
1	**NaturalWind**	Indicate natural wind	M

4.4.5.2.1. SleepWind Value

The fan speed, based on current settings, SHALL gradually slow down to a final minimum speed. For this process, the sequence, speeds and duration are MS.

4.4.5.2.2. NaturalWind Value

The fan speed SHALL vary to emulate natural wind. For this setting, the sequence, speeds and duration are MS.

4.4.5.3. StepDirectionEnum Type

Value	Name	Summary	Conformance
0	**Increase**	Step moves in increasing direction	M
1	**Decrease**	Step moves in decreasing direction	M

4.4.5.4. AirflowDirectionEnum Type

Value	Name	Summary	Conformance
0	**Forward**	Airflow is in the forward direction	M
1	**Reverse**	Airflow is in the reverse direction	M

4.4.5.5. FanModeEnum Type

Value	Name	Summary	Conformance
0	**Off**	Fan is off	M
1	**Low**	Fan using low speed	desc
2	**Medium**	Fan using medium speed	desc
3	**High**	Fan using high speed	M
4	**On**		D
5	**Auto**	Fan is using auto mode	AUT
6	**Smart**	Fan is using smart mode	D

4.4.5.5.1. Low Value

If the fan supports 2 or more speeds, the Low value SHALL be supported.

The Low value SHALL be supported if and only if the FanModeSequence attribute value is less than 4.

4.4.5.5.2. Medium Value

If the fan supports 3 or more speeds, the Medium value SHALL be supported.

The Medium value SHALL be supported if and only if the FanModeSequence attribute value is 0 or 2.

4.4.5.6. FanModeSequenceEnum Type

Value	Name	Summary	Conformance
0	**Off/Low/Med/High**	Fan is capable of off, low, medium and high modes	[!AUT].a
1	**Off/Low/High**	Fan is capable of off, low and high modes	[!AUT].a
2	**Off/Low/Med/High/Auto**	Fan is capable of off, low, medium, high and auto modes	[AUT].a
3	**Off/Low/High/Auto**	Fan is capable of off, low, high and auto modes	[AUT].a
4	**Off/High/Auto**	Fan is capable of off, high and auto modes	[AUT].a
5	**Off/High**	Fan is capable of off and high modes	[!AUT].a

4.4.6. Attributes

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0000	**FanMode**	FanModeEnum	0 to 6	N	0	RW VO	M
0x0001	**FanModeSequence**	FanModeSequenceEnum	0 to 5	N	MS	R[W] VO	Zigbee
0x0001	**FanModeSequence**	FanModeSequenceEnum	0 to 5	F	MS	R V	M
0x0002	**PercentSetting**	percent	0 to 100	X	0	RW VO	M
0x0003	**PercentCurrent**	percent	0 to 100		desc	R V	M
0x0004	**SpeedMax**	uint8	1 to 100	F	MS	R V	SPD
0x0005	**SpeedSetting**	uint8	0 to SpeedMax	X	0	RW VO	SPD

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0006	**SpeedCurrent**	uint8	0 to SpeedMax	P	desc	R V	SPD
0x0007	**RockSupport**	RockBitmap	desc	F	0	R V	RCK
0x0008	**RockSetting**	RockBitmap	desc	P	0	RW VO	RCK
0x0009	**WindSupport**	WindBitmap	desc	F	0	R V	WND
0x000A	**WindSetting**	WindBitmap	desc	P	0	RW VO	WND
0x000B	**AirflowDirection**	AirflowDirectionEnum	desc	P	0	RW VO	DIR

4.4.6.1. FanMode Attribute

This attribute SHALL indicate the current speed mode of the fan. This attribute MAY be written by the client to request a different fan mode. A server SHALL return INVALID_IN_STATE to indicate that the fan is not in a state where the FanMode can be changed to the requested value. A server MAY have FanMode values that it can never be set to. For example, where this cluster appears on the same or another endpoint as other clusters with a system dependency, for example the Thermostat cluster, attempting to set the FanMode attribute of this cluster to Off may not be allowed by the system.

This attribute SHALL be set to one of the values in FanModeEnum.

When the FanMode attribute is successfully written to, the PercentSetting and SpeedSetting (if present) attributes SHALL be set to appropriate values, as defined by the Percent Rules and Speed Rules respectively, unless otherwise specified below.

When the FanMode attribute is set to any given mode, the PercentCurrent and SpeedCurrent (if present) SHALL indicate the actual currently operating fan speed, unless otherwise specified below.

4.4.6.1.1. Off Value

Setting the attribute value to Off SHALL set the values of these attributes to 0 (zero):

- PercentSetting
- PercentCurrent
- SpeedSetting (if present)
- SpeedCurrent (if present)

4.4.6.1.2. Auto Value

Setting the attribute value to Auto SHALL set the values of these attributes to null:

- PercentSetting
- SpeedSetting (if present)

These attributes SHALL continue to indicate the current state of the fan while this attribute value is Auto:

- PercentCurrent
- SpeedCurrent (if present)

4.4.6.1.3. On Value

If a client attempts to write a value of On, the attribute SHALL be set to High.

4.4.6.1.4. Smart Value

If a client attempts to write a value of Smart and the AUT feature is supported, the attribute SHALL be set to Auto, otherwise the attribute SHALL be set to High.

4.4.6.2. FanModeSequence Attribute

This attribute indicates the fan speed ranges that SHALL be supported.

4.4.6.3. PercentSetting Attribute

This attribute SHALL indicate the speed setting for the fan. This attribute MAY be written by the client to indicate a new fan speed. If the client writes null to this attribute, the attribute value SHALL NOT change. A server SHALL return INVALID_IN_STATE to indicate that the fan is not in a state where the PercentSetting can be changed to the requested value.

If this is successfully written to 0, the server SHALL set the FanMode attribute value to Off.

4.4.6.3.1. Percent Rules

It is up to the server implementation to map between ranges of the PercentSetting attribute and FanMode attribute enumerated values. Percent values are split into ranges, each range corresponding to a supported FanMode attribute value. Percent ranges SHALL NOT overlap. All percent values in the High speed range SHALL be higher than all percent values in the Medium and Low speed ranges, if supported. All percent values in the Medium speed range SHALL be higher than all percent values in the Low speed range. If the client sets the FanMode attribute to Low, Medium or High, the server SHALL set the PercentSetting attribute to a value within the corresponding range. If the client sets the PercentSetting attribute, the server SHALL set the FanMode attribute to Low, Medium or High, based on the percent value being in the corresponding range.

If the MultiSpeed feature is supported, the calculation of SpeedSetting or SpeedCurrent (speed) from a percent value change for PercentSetting or PercentCurrent respectively (percent) SHALL hold true:

- speed = ceil(SpeedMax * (percent * 0.01))

> For example: If the SpeedMax attribute is 42 (42 speed fan) and PercentSetting is changed to 25, then SpeedSetting and SpeedCurrent become 11 (rounding up 10.5).

4.4.6.4. PercentCurrent Attribute

This attribute SHALL indicate the actual currently operating fan speed, or zero to indicate that the fan is off. There MAY be a temporary mismatch between the value of this attribute and the value of the PercentSetting attribute due to other system requirements that would not allow the fan to operate at the requested setting. See Percent Rules for more details.

4.4.6.5. SpeedMax Attribute

This attribute SHALL indicate that the fan has one speed (value of 1) or the maximum speed, if the fan is capable of multiple speeds.

4.4.6.6. SpeedSetting Attribute

This attribute SHALL indicate the speed setting for the fan. This attribute MAY be written by the client to indicate a new fan speed. If the client writes null to this attribute, the attribute value SHALL NOT change. A server SHALL return INVALID_IN_STATE to indicate that the fan is not in a state where the SpeedSetting can be changed to the requested value.

If this is successfully written to 0, the server SHALL set the FanMode attribute value to Off. Please see the Speed Rules for details on other values.

4.4.6.6.1. Speed Rules

It is up to the server implementation to map between ranges of the SpeedSetting attribute and FanMode attribute enumerated values. Speed values are split into ranges, each range corresponding to a FanMode attribute value. Speed ranges SHALL NOT overlap. All speed values in the High speed range SHALL be higher than all speed values in the Medium and Low speed ranges, if supported. All speed values in the Medium speed range SHALL be higher than all speed values in the Low speed range. If the client sets the FanMode attribute to Low, Medium or High, the server SHALL set the SpeedSetting attribute to a value within the corresponding range. If the client sets the SpeedSetting attribute, the server SHALL set the FanMode attribute to Low, Medium or High, based on the speed value being in the corresponding range.

This calculation for the value of PercentSetting or PercentCurrent (percent) from a speed value change for SpeedSetting or SpeedCurrent respectively (speed) SHALL hold true:

- percent = floor(speed/SpeedMax * 100)

> For example: If the SpeedMax attribute is 42 (42 speed fan) and SpeedSetting attribute is changed to 11, then PercentSetting and PercentCurrent become 26.

4.4.6.7. SpeedCurrent Attribute

This attribute SHALL indicate the actual currently operating fan speed, or zero to indicate that the fan is off. There MAY be a temporary mismatch between the value of this attribute and the value of the SpeedSetting attribute due to other system requirements that would not allow the fan to operate at the requested setting.

4.4.6.8. RockSupport Attribute

This attribute is a bitmap that indicates what rocking motions the server supports.

4.4.6.9. RockSetting Attribute

This attribute is a bitmap that indicates the current active fan rocking motion settings. Each bit SHALL only be set to 1, if the corresponding bit in the RockSupport attribute is set to 1, otherwise a status code of CONSTRAINT_ERROR SHALL be returned.

If a combination of supported bits is set by the client, and the server does not support the combination, the lowest supported single bit in the combination SHALL be set and active, and all other bits SHALL indicate zero.

> For example: If RockUpDown and RockRound are both set, but this combination is not possible, then only RockUpDown becomes active.

4.4.6.10. WindSupport Attribute

This attribute is a bitmap that indicates what wind modes the server supports. At least one wind mode bit SHALL be set.

4.4.6.11. WindSetting Attribute

This attribute is a bitmap that indicates the current active fan wind feature settings. Each bit SHALL only be set to 1, if the corresponding bit in the WindSupport attribute is set to 1, otherwise a status code of CONSTRAINT_ERROR SHALL be returned.

If a combination of supported bits is set by the client, and the server does not support the combination, the lowest supported single bit in the combination SHALL be set and active, and all other bits SHALL indicate zero.

> For example: If Sleep Wind and Natural Wind are set, but this combination is not possible, then only Sleep Wind becomes active.

4.4.6.12. AirflowDirection Attribute

This attribute SHALL indicate the current airflow direction of the fan. This attribute MAY be written by the client to indicate a new airflow direction for the fan. This attribute SHALL be set to one of the values in the AirflowDirectionEnum table.

4.4.7. Commands

ID	Name	Direction	Response	Access	Conformance
0x00	**Step**	client ⇒ server	Y	O	STEP

4.4.7.1. Step Command

This command speeds up or slows down the fan, in steps, without the client having to know the fan speed. This command supports, for example, a user operated wall switch, where the user provides the feedback or control to stop sending this command when the proper speed is reached. The step speed values are implementation specific. How many step speeds are implemented is implementation specific.

This command supports these fields:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**Direction**	StepDirectionEnum			Increase	M
1	**Wrap**	bool			false	O
2	**LowestOff**	bool			true	O

4.4.7.1.1. Direction Field

This field SHALL indicate whether the fan speed increases or decreases to the next step value.

4.4.7.1.2. Wrap Field

This field SHALL indicate if the fan speed wraps between highest and lowest step value.

4.4.7.1.3. LowestOff Field

This field SHALL indicate that the fan being off (speed value 0) is included as a step value.

4.4.7.1.4. When Generated

The client sends this command to speed the fan up or down in a step by step fashion.

4.4.7.1.5. Effect Upon Receipt

- This command SHALL be executed even if the fan speed is not currently at an implemented step value.
- If the Direction field is Increase,
  - If the fan speed is lower than the highest step value, the fan speed SHALL change to the lowest step value that is higher than the current fan speed.
  - Else if Wrap is TRUE, the fan speed SHALL change to the lowest step value.
  - Else the fan speed SHALL change to (or remain at) the highest step value.

- If the Direction field is Decrease,
  - If the fan speed is higher than the lowest step value, the fan speed SHALL change to the highest step value that is lower than the current fan speed.
  - Else if Wrap is TRUE, the fan speed SHALL change to the highest step value.
  - Else the fan speed SHALL change to (or remain at) the lowest step value.
- Although the effect of the Step command is implementation specific, the effect on receipt of the Step command SHALL adhere to the conformance of the affected attributes.

4.5. Thermostat User Interface Configuration Cluster

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.

4.5.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Revision	Description
1	Global mandatory ClusterRevision attribute added
2	New data model format and notation, added "Conversion of Temperature Values for Display" section

4.5.2. Classification

Hierarchy	Role	Scope	PICS Code
Base	Application	Endpoint	TSUIC

4.5.3. Cluster ID

ID	Name
0x0204	Thermostat User Interface Configuration

4.5.4. Conversion of Temperature Values for Display

See the Temperature Conversion section in the Data Model for unit conversion between Fahrenheit and Celsius.

4.5.5. Data Types

4.5.5.1. TemperatureDisplayModeEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**Celsius**	Temperature displayed in °C	M
1	**Fahrenheit**	Temperature displayed in °F	M

4.5.5.2. KeypadLockoutEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**NoLockout**	All functionality available to the user	M
1	**Lockout1**	Level 1 reduced functionality	M
2	**Lockout2**	Level 2 reduced functionality	M
3	**Lockout3**	Level 3 reduced functionality	M
4	**Lockout4**	Level 4 reduced functionality	M
5	**Lockout5**	Least functionality available to the user	M

The interpretation of the various levels is device-dependent.

4.5.5.3. ScheduleProgrammingVisibilityEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**ScheduleProgrammingPermitted**	Local schedule programming functionality is enabled at the thermostat	M
1	**ScheduleProgrammingDenied**	Local schedule programming functionality is disabled at the thermostat	M

4.5.6. Attributes

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0000	**TemperatureDisplayMode**	TemperatureDisplayModeEnum	desc		Celsius	RW VO	M
0x0001	**KeypadLockout**	KeypadLockoutEnum	desc		NoLockout	RW VM	M
0x0002	**ScheduleProgrammingVisibility**	ScheduleProgrammingVisibilityEnum	desc		ScheduleProgrammingPermitted	RW VM	O

4.5.6.1. TemperatureDisplayMode Attribute

This attribute SHALL indicate the units of the temperature displayed on the thermostat screen.

4.5.6.2. KeypadLockout Attribute

This attribute SHALL indicate the level of functionality that is available to the user via the keypad.

4.5.6.3. ScheduleProgrammingVisibility Attribute

This attribute is used to hide the weekly schedule programming functionality or menu on a thermostat from a user to prevent local user programming of the weekly schedule. The schedule programming MAY still be performed via a remote interface, and the thermostat MAY operate in schedule programming mode.

This attribute is designed to prevent local tampering with or disabling of schedules that MAY have been programmed by users or service providers via a more capable remote interface. The programming schedule SHALL continue to run even though it is not visible to the user locally at the thermostat.

4.6. Valve Configuration and Control Cluster

This cluster is used to configure a valve.

4.6.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Revision	Description
1	Initial Release

4.6.2. Classification

Hierarchy	Role	Scope	PICS Code
Base	Application	Endpoint	VALCC

4.6.3. Cluster ID

ID	Name
0x0081	Valve Configuration and Control

4.6.4. Features

This cluster SHALL support the FeatureMap bitmap attribute as defined below.

Bit	Code	Feature	Conformance	Summary
0	**TS**	TimeSync	desc	UTC time is used for time indications
1	**LVL**	Level	O	Device supports setting the specific position of the valve

4.6.4.1. TimeSync Feature

This feature SHALL indicate that the valve uses Time Synchronization and UTC time to indicate duration and auto close time.

This feature SHALL NOT be supported unless the device supports the Time Synchronization cluster.

4.6.4.2. Level Feature

This feature SHALL indicate that the valve is capable of being adjusted to a specific position, as a percentage, of its full range of motion.

4.6.5. Data Types

4.6.5.1. ValveFaultBitmap Type

This data type is derived from map16.

Bit	Name	Summary	Conformance
0	**GeneralFault**	Unspecified fault detected	M
1	**Blocked**	Valve is blocked	M
2	**Leaking**	Valve has detected a leak	M
3	**NotConnected**	No valve is connected to controller	M
4	**ShortCircuit**	Short circuit is detected	M
5	**CurrentExceeded**	The available current has been exceeded	M

4.6.5.2. ValveStateEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**Closed**	Valve is in closed position	M
1	**Open**	Valve is in open position	M
2	**Transitioning**	Valve is transitioning between closed and open positions or between levels	M

4.6.6. Status Codes

4.6.6.1. StatusCodeEnum Type

This data type is derived from enum8.

Value	Name	Summary
0x02	**FailureDueToFault**	The requested action could not be performed due to a fault on the valve.

4.6.7. Attributes

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0000	**OpenDuration**	elapsed-s	min 1	X	null	R V	M

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0001	**DefaultOpenDuration**	elapsed-s	min 1	XN	null	RW VO	M
0x0002	**AutoCloseTime**	epoch-us	all	X	null	R V	TS
0x0003	**RemainingDuration**	elapsed-s	all	XQ	null	R V	M
0x0004	**CurrentState**	ValveStateEnum	all	X	null	R V	M
0x0005	**TargetState**	ValveStateEnum	all	X	null	R V	M
0x0006	**CurrentLevel**	percent	all	X	null	R V	LVL
0x0007	**TargetLevel**	percent	all	X	null	R V	LVL
0x0008	**DefaultOpenLevel**	percent	1 to 100	N	100	RW VO	[LVL]
0x0009	**ValveFault**	ValveFaultBitmap	all		0	R V	O
0x000A	**LevelStep**	uint8	1 to 50	F	1	R V	[LVL]

4.6.7.1. OpenDuration Attribute

This attribute SHALL indicate the total duration, in seconds, for which the valve will remain open for this current opening.

A value of null SHALL indicate the duration is not set, meaning that the valve will remain open until closed by the user or some other automation.

4.6.7.2. DefaultOpenDuration Attribute

This attribute SHALL indicate the default duration, in seconds, for which the valve will remain open, if the OpenDuration field is not present in the Open command.

A value of null SHALL indicate the duration is not set, meaning that the valve will remain open until closed by the user or some other automation.

4.6.7.3. AutoCloseTime Attribute

This attribute SHALL indicate the UTC time when the valve will close, depending on value of the OpenDuration attribute.

This attribute SHALL be null:

- When OpenDuration is null, or
- When the valve does not have a synchronized UTCTime in the Time Synchronization cluster, or
- When the valve is closed.

When the value of this attribute is earlier or equal to the current UTC time, the valve SHALL automatically transition to its closed position. The behavior of transitioning to the closed position, SHALL match the behavior described in the Close command.

If this attribute is not null and the Time Synchronization cluster receives a SetUTCTime command, modifying the current UTC time of the device, the value of this attribute SHALL be adjusted to match the new UTC time plus the value of the RemainingDuration attribute.

4.6.7.4. RemainingDuration Attribute

This attribute SHALL indicate the remaining duration, in seconds, until the valve closes.

This attribute SHALL be null:

- When OpenDuration is null, or
- When the valve is closed.

The value of this attribute SHALL only be reported in the following cases:

- When it changes from null to any other value and vice versa, or
- When it changes to 0, or
- When it increases, or
- When the closing time changes.

Meaning that clients SHOULD NOT rely on the reporting of this attribute in order to keep track of the remaining duration, due to this attribute not being reported during regular countdown.

When reading this attribute it SHALL return the remaining duration, in seconds, until the valve closes.

When the value of this attribute counts down to 0, the valve SHALL automatically transition to its closed position. The behavior of transitioning to the closed position SHALL match the behavior described in the Close command.

4.6.7.5. CurrentState Attribute

This attribute SHALL indicate the current state of the valve.

A value of null SHALL indicate that the current state is not known.

4.6.7.6. TargetState Attribute

This attribute SHALL indicate the target state, while changing the state, of the valve.

A value of null SHALL indicate that no target position is set, since the change in state is either done or failed.

4.6.7.7. CurrentLevel Attribute

This attribute SHALL indicate the current level of the valve as a percentage value, between fully closed and fully open. During a transition from one level to another level, the valve SHOULD keep this attribute updated to the best of its ability, in order to represent the actual level of the valve during the movement.

A value of 100 percent SHALL indicate the fully open position.

A value of 0 percent SHALL indicate the fully closed position.

A value of null SHALL indicate that the current state is not known.

4.6.7.8. TargetLevel Attribute

This attribute SHALL indicate the target level of the valve as a percentage value, between fully closed and fully open.

The interpretation of the percentage value is the same as for the CurrentLevel attribute.

A value of null SHALL indicate that no target position is set, since the change of level is either done or failed.

4.6.7.9. DefaultOpenLevel Attribute

This attribute SHALL indicate the default value used for the TargetLevel attribute, when a valve transitions from the closed to the open state, caused by an Open command, if a TargetLevel field is not present in the Open command.

If the LevelStep attribute is present and the value of a write interaction to this attribute field is not 100, the value SHALL be a supported value as defined by the LevelStep attribute, such that (Value received in the write interaction) % (Value of LevelStep attribute) equals 0. If the resulting value is not 0, the requested DefaultOpenLevel value is considered an unsupported value and a CONSTRAINT_ERROR status SHALL be returned.

4.6.7.10. ValveFault Attribute

This attribute SHALL indicate any faults registered by the valve.

4.6.7.11. LevelStep Attribute

This attribute SHALL indicate the step size the valve can support.

The step size defined by this attribute is counted from 0 and the final step towards 100 MAY be different than what is defined in this attribute. For example, if the value of this attribute is 15, it results in these target values being supported; 0, 15, 30, 45, 60, 75, 90 and 100.

The values of 0 and 100 SHALL always be supported, regardless of the value of this attribute.

4.6.8. Commands

ID	Name	Direction	Response	Access	Conformance
0x00	**Open**	client ⇒ server	Y	O	M
0x01	**Close**	client ⇒ server	Y	O	M

4.6.8.1. Open Command

This command is used to set the valve to its open position.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**OpenDuration**	elapsed-s	min 1	X		O
1	**TargetLevel**	percent	min 1			[LVL]

4.6.8.1.1. OpenDuration Field

This field SHALL indicate the duration that the valve will remain open for this specific Open command.

A value of null SHALL indicate the duration is not set, meaning that the valve will remain open until closed by the user or some other automation.

4.6.8.1.2. TargetLevel Field

This field SHALL indicate the target level used for this specific Open command.

4.6.8.1.3. Effect on Receipt

If the device has registered a fault, that prevents it from performing the requested action, the command SHALL be ignored and a FAILURE_DUE_TO_FAULT status SHALL be returned.

The device SHALL set the TargetState attribute to the Open value and set the CurrentState attribute to the Transitioning value.

If the OpenDuration field is present, the value of the OpenDuration attribute SHALL be set to the value of the OpenDuration field.
If the OpenDuration field is not present, the value of OpenDuration attribute SHALL be set to the value of the DefaultOpenDuration attribute.

If the OpenDuration attribute is null, it SHALL indicate that there is no auto close defined for the current Open action and the device SHALL set the RemainingDuration attribute to null. If the device supports the TimeSync feature, the device SHALL set the AutoCloseTime attribute to null.
If the OpenDuration attribute is not null, it indicates that an auto close duration is defined for the current open action and the device SHALL set the value of the RemainingDuration attribute equal to the value of the OpenDuration attribute. If the device supports the TimeSync feature, the device SHALL set the AutoCloseTime attribute to the UTC value of the time when command was received

plus the value of the OpenDuration attribute.

If the LevelStep attribute and the TargetLevel field are both present and the value of the TargetLevel field is not 100, the value of the TargetLevel field SHALL be a supported value as defined by the LevelStep attribute, such that (Value of TargetLevel field) % (Value of LevelStep attribute) equals 0. If the resulting value is not 0, the requested TargetLevel value is considered an unsupported value and a CONSTRAINT_ERROR status SHALL be returned.

If the device supports the Level feature, the TargetLevel attribute SHALL be set to the value of the TargetLevel field, if present. If the TargetLevel field is not present, the TargetLevel attribute SHALL be set to the value of the DefaultOpenLevel attribute, if implemented. If the DefaultOpenLevel attribute is not present, the TargetLevel attribute SHALL be set to 100.

When the relevant target and duration attributes have been set, the device SHALL start the movement towards the target value and start the countdown of the RemainingDuration attribute. If the device supports the Level feature, the device SHALL update the CurrentLevel attribute at the start of the movement, and SHOULD update it as appropriate during movement, especially if it is slow. When the movement is complete, the device SHALL set the CurrentState attribute to the Open value.

4.6.8.2. Close Command

This command is used to set the valve to its closed position.

4.6.8.2.1. Effect on Receipt

If the device has registered a fault that causes it to prevent the valve to perform the requested action, the command SHALL be ignored and a FAILURE_DUE_TO_FAULT status SHALL be returned.

The OpenDuration and RemainingDuration attribute SHALL be set to null.

If the device supports the TimeSync feature, the AutoCloseTime attribute SHALL be set to null.

The device SHALL set the TargetState attribute to the Closed value and set the CurrentState attribute to the Transitioning value.

If the device supports the Level feature, it SHALL set the TargetLevel attribute to 0.

When the relevant target attributes have been set, the device SHALL start the movement towards the target value. If the device supports the Level feature, the device SHALL update the CurrentLevel attribute, accordingly during the movement. When the movement is complete, the device SHALL set the CurrentState attribute to the Closed value.

4.6.9. Events

ID	Name	Priority	Access	Conformance
0x00	**ValveStateChanged**	INFO	V	O
0x01	**ValveFault**	INFO	V	O

4.6.9.1. ValveStateChanged Event

This event SHALL be generated when the valve state changed. For level changes, after the end of movement, for state changes when the new state has been reached.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**ValveState**	ValveStateEnum	all			M
1	**ValveLevel**	percent	all			LVL

4.6.9.1.1. ValveState Field

This field SHALL indicate the new state of the valve.

4.6.9.1.2. ValveLevel Field

This field SHALL indicate the new level of the valve.

4.6.9.2. ValveFault Event

This event SHALL be generated when the valve registers or clears a fault, e.g. not being able to transition to the requested target level or state.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**ValveFault**	ValveFaultBitmap	all			M

4.6.9.2.1. ValveFault Field

This field SHALL indicate the value of the ValveFault attribute, at the time this event is generated.

Chapter 5. Closures

The Cluster Library is made of individual chapters such as this one. References between chapters are made using a *X.Y* notation where *X* is the chapter and *Y* is the sub-section within that chapter.

5.1. General Description

5.1.1. Introduction

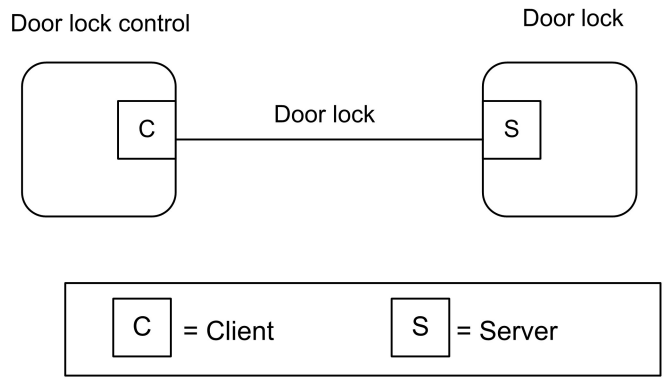
The clusters specified in this document are for use typically in applications involving closures (e.g., shades, windows, doors), but MAY be used in any application domain.

5.1.2. Cluster List

This section lists the closures specific clusters as specified in this chapter.

Table 36. Overview of the Closures Clusters

Cluster ID	Cluster Name	Description
0x0101	Door Lock	An interface to a generic way to secure a door
0x0102	Window Covering	Commands and attributes for controlling a window covering



Note: Device names are examples for illustration purposes only

Figure 14. Typical Usage of the Closures Clusters

5.2. Door Lock Cluster

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.

5.2.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Revision	Description
1	Mandatory global ClusterRevision attribute added; CCB 1811 1812 1821
2	CCB 2430
3	CCB 2629 2630
4	All Hubs changes and added feature map
5	
6	New data model format and notation. Added User features. General cleanup of functionality
7	Added support for European door locks (unbolt feature)

5.2.2. Classification

Hierarchy	Role	Scope	PICS Code
Base	Application	Endpoint	DRLK

5.2.3. Cluster ID

ID	Name
0x0101	Door Lock

5.2.4. Features

This cluster SHALL support the FeatureMap bitmap attribute as defined below.

Bit	Code	Feature	Conformance	Summary
0	**PIN**	PINCredential	O	Lock supports PIN credentials (via keypad, or over-the-air)
1	**RID**	RFIDCredential	O	Lock supports RFID credentials
2	**FGP**	FingerCredentials	P, O	Lock supports finger related credentials (fingerprint, finger vein)
3	**LOG**	Logging	O	Lock supports local/on-lock logging when Events are not supported

Bit	Code	Feature	Conformance	Summary
4	**WDSCH**	WeekDayAccessSchedules	O	Lock supports week day user access schedules
5	**DPS**	DoorPositionSensor	O	Lock supports a door position sensor that indicates door's state
6	**FACE**	FaceCredentials	P, O	Lock supports face related credentials (face, iris, retina)
7	**COTA**	CredentialOverTheAirAccess	O	PIN codes over-the-air supported for lock/unlock operations
8	**USR**	User	[PIN | RID | FGP | FACE]	Lock supports the user commands and database
9	**NOT**	Notification	O	Operation and Programming Notifications
10	**YDSCH**	YearDayAccessSchedules	O	Lock supports year day user access schedules
11	**HDSCH**	HolidaySchedules	O	Lock supports holiday schedules
12	**UBOLT**	Unbolting	O	Lock supports unbolting

5.2.4.1. PINCredential Feature

If the User Feature is also supported then any PIN Code stored in the lock SHALL be associated with a User.

A lock MAY support multiple credential types so if the User feature is supported the UserType, UserStatus and Schedules are all associated with a User index and not directly with a PIN index. A User index may have several credentials associated with it.

5.2.4.2. RFIDCredential Feature

If the User Feature is also supported then any RFID credential stored in the lock SHALL be associated with a User.

A lock MAY support multiple credential types so if the User feature is supported the UserType, UserStatus and Schedules are all associated with a User index and not directly with a RFID index. A User

Index may have several credentials associated with it.

5.2.4.3. FingerCredentials Feature

Currently the cluster only defines the metadata format for notifications when a fingerprint/ finger vein credential is used to access the lock and doesn't describe how to create fingerprint/finger vein credentials. If the Users feature is also supported then the User that a fingerprint/finger vein is associated with can also have its UserType, UserStatus and Schedule modified.

A lock MAY support multiple credential types so if the User feature is supported the UserType, UserStatus and Schedules are all associated with a User index and not directly with a Finger index. A User Index may have several credentials associated with it.

5.2.4.4. Logging Feature

If Events are not supported the logging feature SHALL replace the Event reporting structure. If Events are supported the logging feature SHALL NOT be supported.

5.2.4.5. WeekDayAccessSchedules Feature

If the User feature is supported then Week Day Schedules are applied to a User and not a credential.

Week Day Schedules are used to restrict access to a specified time window on certain days of the week. The schedule is repeated each week. When a schedule is cleared this clears the access restrictions and grants unrestricted access to the user. The lock MAY automatically adjust the UserType when a schedule is created or cleared.

5.2.4.6. DoorPositionSensor Feature

If this feature is supported this indicates that the lock has the ability to determine the position of the door which is separate from the state of the lock.

5.2.4.7. FaceCredentials Feature

Currently the cluster only defines the metadata format for notifications when a face recognition, iris, or retina credential is used to access the lock and doesn't describe how to create face recognition, iris, or retina credentials. If the Users feature is also supported then the User that a face recognition, iris, or retina credential is associated with can also have its UserType, UserStatus and Schedule modified.

A lock MAY support multiple credential types so if the User feature is supported the UserType, UserStatus and Schedules are all associated with a User and not directly with a credential.

5.2.4.8. CredentialOverTheAirAccess Feature

If this feature is supported then the lock supports the ability to verify a credential provided in a lock/unlock command. Currently the cluster only supports providing the PIN credential to the lock/unlock commands. If this feature is supported then the PIN Credential feature SHALL also be supported.

5.2.4.9. User Feature

If the User Feature is supported then a lock employs a User database. A User within the User database is used to associate credentials and schedules to single user record within the lock. This also means the UserType and UserStatus fields are associated with a User and not a credential.

5.2.4.10. Notification Feature

This is a feature used before support of events. This feature supports notification commands and masks used to filter these notifications.

5.2.4.11. YearDayAccessSchedules Feature

If the User feature is supported then Year Day Schedules are applied to a User and not a credential.

Year Day Schedules are used to restrict access to a specified date and time window. When a schedule is cleared this clears the access restrictions and grants unrestricted access to the user. The lock MAY automatically adjust the UserType when a schedule is created or cleared.

5.2.4.12. HolidaySchedules Feature

This feature is used to setup Holiday Schedule in the lock device. A Holiday Schedule sets a start and stop end date/time for the lock to use the specified operating mode set by the Holiday Schedule.

5.2.4.13. Unbolting Feature

Locks that support this feature differentiate between unbolting and unlocking. The Unbolt Door command retracts the bolt without pulling the latch. The Unlock Door command fully unlocks the door by retracting the bolt and briefly pulling the latch. While the latch is pulled, the lock state changes to Unlatched. Locks without unbolting support don't differentiate between unbolting and unlocking and perform the same operation for both commands.

5.2.5. Recommended steps for creating a new User

It is RECOMMENDED that the Administrator query the door lock for what users already exist in its database to find an available UserIndex for creating a new User (see GetUser command).

1. Use SetUser command with an available UserIndex to set the user record fields as applicable.
2. Use SetCredential command with same UserIndex to add one or more credentials as applicable.
3. Use SetWeekDaySchedule command or SetYearDaySchedule command with same UserIndex to add one or more schedule restrictions as applicable.

5.2.6. Data Types

5.2.6.1. DaysMaskBitmap Type

This data type is derived from map8.

This bitmap SHALL indicate the days of the week the Week Day schedule applies for.

Bit	Name	Summary
0	**Sunday**	Schedule is applied on Sunday
1	**Monday**	Schedule is applied on Monday
2	**Tuesday**	Schedule is applied on Tuesday
3	**Wednesday**	Schedule is applied on Wednesday
4	**Thursday**	Schedule is applied on Thursday
5	**Friday**	Schedule is applied on Friday
6	**Saturday**	Schedule is applied on Saturday

5.2.6.2. CredentialRulesBitmap Type

This data type is derived from map8.

Bit	Name	Summary
0	**Single**	Only one credential is required for lock operation
1	**Dual**	Any two credentials are required for lock operation
2	**Tri**	Any three credentials are required for lock operation

5.2.6.3. OperatingModesBitmap Type

This data type is derived from map16.

Bit	Name	Summary	Conformance
0	**Normal**	Normal operation mode	M
1	**Vacation**	Vacation operation mode	O
2	**Privacy**	Privacy operation mode	O
3	**NoRemoteLockUnlock**	No remote lock and unlock operation mode	M
4	**Passage**	Passage operation mode	O

5.2.6.4. ConfigurationRegisterBitmap Type

This data type is derived from map16.

Bit	Name	Summary
0	**LocalProgramming**	The state of local programming functionality
1	**KeypadInterface**	The state of the keypad interface
2	**RemoteInterface**	The state of the remote interface
5	**SoundVolume**	Sound volume is set to Silent value
6	**AutoRelockTime**	Auto relock time it set to 0
7	**LEDSettings**	LEDs is disabled

5.2.6.4.1. LocalProgramming Bit

This bit SHALL indicate the state related to local programming:

- 0 = Local programming is disabled
- 1 = Local programming is enabled

5.2.6.4.2. KeypadInterface Bit

This bit SHALL indicate the state related to keypad interface:

- 0 = Keypad interface is disabled
- 1 = Keypad interface is enabled

5.2.6.4.3. RemoteInterface Bit

This bit SHALL indicate the state related to remote interface:

- 0 = Remote interface is disabled
- 1 = Remote interface is enabled

5.2.6.4.4. SoundVolume Bit

This bit SHALL indicate the state related to sound volume:

- 0 = Sound volume value is 0 (Silent)
- 1 = Sound volume value is equal to something other than 0

5.2.6.4.5. AutoRelockTime Bit

This bit SHALL indicate the state related to auto relock time:

- 0 = Auto relock time value is 0
- 1 = Auto relock time value is equal to something other than 0

5.2.6.4.6. LEDSettings Bit

This bit SHALL indicate the state related to LED settings:

- 0 = LED settings value is 0 (NoLEDSignal)
- 1 = LED settings value is equal to something other than 0

5.2.6.5. LocalProgrammingFeaturesBitmap Type

This data type is derived from map8.

Bit	Name	Summary
0	**AddUsersCredentialsSchedules**	The state of the ability to add users, credentials or schedules on the device
1	**ModifyUsersCredentialsSchedules**	The state of the ability to modify users, credentials or schedules on the device
2	**ClearUsersCredentialsSchedules**	The state of the ability to clear users, credentials or schedules on the device
3	**AdjustSettings**	The state of the ability to adjust settings on the device

5.2.6.5.1. AddUsersCredentialsSchedules Bit

This bit SHALL indicate whether the door lock is able to add Users/Credentials/Schedules locally:

- 0 = This ability is disabled
- 1 = This ability is enabled

5.2.6.5.2. ModifyUsersCredentialsSchedules Bit

This bit SHALL indicate whether the door lock is able to modify Users/Credentials/Schedules locally:

- 0 = This ability is disabled
- 1 = This ability is enabled

5.2.6.5.3. ClearUsersCredentialsSchedules Bit

This bit SHALL indicate whether the door lock is able to clear Users/Credentials/Schedules locally:

- 0 = This ability is disabled
- 1 = This ability is enabled

5.2.6.5.4. AdjustSettings Bit

This bit SHALL indicate whether the door lock is able to adjust lock settings locally:

- 0 = This ability is disabled
- 1 = This ability is enabled

5.2.6.6. AlarmMaskBitmap Type

This data type is derived from map16.

Bit	Name	Summary	Conformance
0	**LockJammed**	Locking Mechanism Jammed	M
1	**LockFactoryReset**	Lock Reset to Factory Defaults	O
2	**N/A**	Reserved	O
3	**LockRadioPowerCycled**	RF Module Power Cycled	O
4	**WrongCodeEntryLimit**	Tamper Alarm - wrong code entry limit	PIN | RID
5	**FrontEscutcheonRemoved**	Tamper Alarm - front escutcheon removed from main	O
6	**DoorForcedOpen**	Forced Door Open under Door Locked Condition	O

5.2.6.7. EventMaskBitmap Type

This data type is derived from map16.

Bit	Name	Summary
0	**Bit0**	State of bit 0
1	**Bit1**	State of bit 1
2	**Bit2**	State of bit 2
3	**Bit3**	State of bit 3
4	**Bit4**	State of bit 4
5	**Bit5**	State of bit 5
6	**Bit6**	State of bit 6
7	**Bit7**	State of bit 7
8	**Bit8**	State of bit 8
9	**Bit9**	State of bit 9
10	**Bit10**	State of bit 10

Bit	Name	Summary
11	**Bit11**	State of bit 11
12	**Bit12**	State of bit 12
13	**Bit13**	State of bit 13
14	**Bit14**	State of bit 14
15	**Bit15**	State of bit 15

5.2.6.7.1. Bits

These bits SHALL indicate whether a given bit is being masked.

- 0 = This bit mask is disabled
- 1 = This bit mask is enabled

5.2.6.8. AlarmCodeEnum Type

This data type is derived from enum8.

This enumeration SHALL indicate the alarm type.

Value	Name	Summary	Conformance
0	**LockJammed**	Locking Mechanism Jammed	M
1	**LockFactoryReset**	Lock Reset to Factory Defaults	O
3	**LockRadioPowerCycled**	Lock Radio Power Cycled	O
4	**WrongCodeEntryLimit**	Tamper Alarm - wrong code entry limit	[USR]
5	**FrontEsceutcheonRemoved**	Tamper Alarm - front escutcheon removed from main	O
6	**DoorForcedOpen**	Forced Door Open under Door Locked Condition	[DPS]
7	**DoorAjar**	Door ajar	[DPS]
8	**ForcedUser**	Force User SOS alarm	[USR]

5.2.6.9. CredentialRuleEnum Type

This data type is derived from enum8.

This enumeration SHALL indicate the credential rule that can be applied to a particular user.

Value	Name	Summary	Conformance
0	**Single**	Only one credential is required for lock operation	USR
1	**Dual**	Any two credentials are required for lock operation	[USR]
2	**Tri**	Any three credentials are required for lock operation	[USR]

5.2.6.10. CredentialTypeEnum Type

This data type is derived from enum8.

This enumeration SHALL indicate the credential type.

Value	Name	Summary	Conformance
0	**ProgrammingPIN**	Programming PIN code credential type	O
1	**PIN**	PIN code credential type	PIN
2	**RFID**	RFID identifier credential type	RID
3	**Fingerprint**	Fingerprint identifier credential type	FGP
4	**FingerVein**	Finger vein identifier credential type	FGP
5	**Face**	Face identifier credential type	FACE

5.2.6.11. DataOperationTypeEnum Type

This data type is derived from enum8.

This enumeration SHALL indicate the data operation performed.

Value	Name	Summary	Conformance
0	**Add**	Data is being added or was added	M
1	**Clear**	Data is being cleared or was cleared	M

Value	Name	Summary	Conformance
2	**Modify**	Data is being modified or was modified	M

5.2.6.12. DoorStateEnum Type

This data type is derived from enum8.

This enumeration SHALL indicate the current door state.

Value	Name	Summary	Conformance
0	**DoorOpen**	Door state is open	DPS
1	**DoorClosed**	Door state is closed	DPS
2	**DoorJammed**	Door state is jammed	[DPS]
3	**DoorForcedOpen**	Door state is currently forced open	[DPS]
4	**DoorUnspecifiedError**	Door state is invalid for unspecified reason	[DPS]
5	**DoorAjar**	Door state is ajar	[DPS]

5.2.6.13. LockDataTypeEnum Type

This data type is derived from enum8.

This enumeration SHALL indicate the data type that is being or has changed.

Value	Name	Summary	Conformance
0	**Unspecified**	Unspecified or manufacturer specific lock user data added, cleared, or modified.	O
1	**ProgrammingCode**	Lock programming PIN code was added, cleared, or modified.	O
2	**UserIndex**	Lock user index was added, cleared, or modified.	M
3	**WeekDaySchedule**	Lock user week day schedule was added, cleared, or modified.	WDSCH
4	**YearDaySchedule**	Lock user year day schedule was added, cleared, or modified.	YDSCH

Value	Name	Summary	Conformance
5	**HolidaySchedule**	Lock holiday schedule was added, cleared, or modified.	HDSCH
6	**PIN**	Lock user PIN code was added, cleared, or modified.	PIN
7	**RFID**	Lock user RFID code was added, cleared, or modified.	RID
8	**Fingerprint**	Lock user fingerprint was added, cleared, or modified.	FGP
9	**FingerVein**	Lock user finger-vein information was added, cleared, or modified.	FGP
10	**Face**	Lock user face information was added, cleared, or modified.	FACE

5.2.6.14. LockOperationTypeEnum Type

This data type is derived from enum8.

This enumeration SHALL indicate the type of Lock operation performed.

Value	Name	Summary	Conformance
0	**Lock**	Lock operation	M
1	**Unlock**	Unlock operation	M
2	**NonAccessUserEvent**	Triggered by keypad entry for user with User Type set to Non Access User	O
3	**ForcedUserEvent**	Triggered by using a user with UserType set to Forced User	O
4	**Unlatch**	Unlatch operation	M

5.2.6.15. OperationErrorEnum Type

This data type is derived from enum8.

This enumeration SHALL indicate the error cause of the Lock/Unlock operation performed.

Value	Name	Summary	Conformance
0	**Unspecified**	Lock/unlock error caused by unknown or unspecified source	O
1	**InvalidCredential**	Lock/unlock error caused by invalid PIN, RFID, fingerprint or other credential	USR
2	**DisabledUserDenied**	Lock/unlock error caused by disabled USER or credential	M
3	**Restricted**	Lock/unlock error caused by schedule restriction	WDSCH | YDSCH
4	**InsufficientBattery**	Lock/unlock error caused by insufficient battery power left to safely actuate the lock	O

5.2.6.16. OperatingModeEnum Type

This data type is derived from enum8.

This enumeration SHALL indicate the lock operating mode.

Value	Name	Summary	Conformance
0	**Normal**		M
1	**Vacation**		O
2	**Privacy**		O
3	**NoRemoteLockUnlock**		M
4	**Passage**		O

The table below shows the operating mode and which interfaces are enabled, if supported, for each mode.

Name	Keypad*	Remote*	RFID*
Normal	Y	Y	Y
Vacation	N	Y	N
Privacy	N	N	N
NoRemoteLockUnlock	Y	N	Y
Passage	N/A	N/A	N/A

* Interface Operational: Yes, No or N/A

NOTE	For modes that disable the remote interface, the door lock SHALL respond to Lock, Unlock, Toggle, and Unlock with Timeout commands with a response status Failure and not take the action requested by those commands. The door lock SHALL NOT disable the radio or otherwise unbind or leave the network. It SHALL still respond to all other commands and requests.

5.2.6.16.1. Normal Value

The lock operates normally. All interfaces are enabled.

5.2.6.16.2. Vacation Value

Only remote interaction is enabled. The keypad SHALL only be operable by the master user.

5.2.6.16.3. Privacy Value

This mode is only possible if the door is locked. Manual unlocking changes the mode to Normal operating mode. All external interaction with the door lock is disabled. This mode is intended to be used so that users, presumably inside the property, will have control over the entrance.

5.2.6.16.4. NoRemoteLockUnlock Value

This mode only disables remote interaction with the lock. This does not apply to any remote proprietary means of communication. It specifically applies to the Lock, Unlock, Toggle, and Unlock with Timeout Commands.

5.2.6.16.5. Passage Value

The lock is open or can be opened or closed at will without the use of a Keypad or other means of user validation (e.g. a lock for a business during work hours).

5.2.6.17. OperationSourceEnum Type

This data type is derived from enum8.

This enumeration SHALL indicate the source of the Lock/Unlock or user change operation performed.

Value	Name	Summary	Conformance
0	**Unspecified**	Lock/unlock operation came from unspecified source	O
1	**Manual**	Lock/unlock operation came from manual operation (key, thumbturn, handle, etc).	O

Value	Name	Summary	Conformance
2	**ProprietaryRemote**	Lock/unlock operation came from proprietary remote source (e.g. vendor app/cloud)	O
3	**Keypad**	Lock/unlock operation came from keypad	O
4	**Auto**	Lock/unlock operation came from lock automatically (e.g. relock timer)	O
5	**Button**	Lock/unlock operation came from lock button (e.g. one touch or button)	O
6	**Schedule**	Lock/unlock operation came from lock due to a schedule	HDSCH
7	**Remote**	Lock/unlock operation came from remote node	M
8	**RFID**	Lock/unlock operation came from RFID card	RID
9	**Biometric**	Lock/unlock operation came from biometric source (e.g. face, fingerprint/fingervein)	[USR]

5.2.6.18. UserStatusEnum Type

This data type is derived from enum8.

This enumeration SHALL indicate what the status is for a specific user ID.

Value	Name	Summary	Conformance
0	**Available**	The user ID is available	M
1	**OccupiedEnabled**	The user ID is occupied and enabled	M
3	**OccupiedDisabled**	The user ID is occupied and disabled	O

5.2.6.19. UserTypeEnum Type

This data type is derived from enum8.

This enumeration SHALL indicate what the type is for a specific user ID.

Value	Name	Summary	Conformance
0	**UnrestrictedUser**	The user ID type is unrestricted	M
1	**YearDayScheduleUser**	The user ID type is schedule	O
2	**WeekDayScheduleUser**	The user ID type is schedule	O
3	**ProgrammingUser**	The user ID type is programming	O
4	**NonAccessUser**	The user ID type is non access	O
5	**ForcedUser**	The user ID type is forced	[USR]
6	**DisposableUser**	The user ID type is disposable	[USR]
7	**ExpiringUser**	The user ID type is expiring	[USR]
8	**ScheduleRestrictedUser**	The user ID type is schedule restricted	WDSCH | YDSCH
9	**RemoteOnlyUser**	The user ID type is remote only	USR & COTA & PIN

5.2.6.19.1. UnrestrictedUser Value

This value SHALL indicate the user has access 24/7 provided proper PIN or RFID is supplied (e.g., owner).

5.2.6.19.2. YearDayScheduleUser Value

This value SHALL indicate the user has the ability to open lock within a specific time period (e.g., guest).

5.2.6.19.3. WeekDayScheduleUser Value

This value SHALL indicate the user has the ability to open lock based on specific time period within a reoccurring weekly schedule (e.g., cleaning worker).

5.2.6.19.4. ProgrammingUser Value

This value SHALL indicate the user has the ability to both program and operate the door lock. This user can manage the users and user schedules. In all other respects this user matches the unrestricted (default) user. ProgrammingUser is the only user that can disable the user interface (keypad, remote, etc...).

5.2.6.19.5. NonAccessUser Value

This value SHALL indicate the user is recognized by the lock but does not have the ability to open the lock. This user will only cause the lock to generate the appropriate event notification to any bound devices.

5.2.6.19.6. ForcedUser Value

This value SHALL indicate the user has the ability to open lock but a ForcedUser LockOperationType and ForcedUser silent alarm will be emitted to allow a notified Node to alert emergency services or contacts on the user account when used.

5.2.6.19.7. DisposableUser Value

This value SHALL indicate the user has the ability to open lock once after which the lock SHALL change the corresponding user record UserStatus value to OccupiedDisabled automatically.

5.2.6.19.8. ExpiringUser Value

This value SHALL indicate the user has the ability to open lock for ExpiringUserTimeout attribute minutes after the first use of the PIN code, RFID code, Fingerprint, or other credential. After ExpiringUserTimeout minutes the corresponding user record UserStatus value SHALL be set to OccupiedDisabled automatically by the lock. The lock SHALL persist the timeout across reboots such that the ExpiringUserTimeout is honored.

5.2.6.19.9. ScheduleRestrictedUser Value

This value SHALL indicate the user access is restricted by Week Day and/or Year Day schedule.

5.2.6.19.10. RemoteOnlyUser Value

This value SHALL indicate the user access and PIN code is restricted to remote lock/unlock commands only. This type of user might be useful for regular delivery services or voice assistant unlocking operations to prevent a PIN code credential created for them from being used at the keypad. The PIN code credential would only be provided over-the-air for the lock/unlock commands.

5.2.6.20. LockStateEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**NotFullyLocked**	Lock state is not fully locked	M
1	**Locked**	Lock state is fully locked	M
2	**Unlocked**	Lock state is fully unlocked	M

Value	Name	Summary	Conformance
3	**Unlatched**	Lock state is fully unlocked and the latch is pulled	O

5.2.6.21. LockTypeEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**DeadBolt**	Physical lock type is dead bolt	M
1	**Magnetic**	Physical lock type is magnetic	M
2	**Other**	Physical lock type is other	M
3	**Mortise**	Physical lock type is mortise	M
4	**Rim**	Physical lock type is rim	M
5	**LatchBolt**	Physical lock type is latch bolt	M
6	**CylindricalLock**	Physical lock type is cylindrical lock	M
7	**TubularLock**	Physical lock type is tubular lock	M
8	**InterconnectedLock**	Physical lock type is interconnected lock	M
9	**DeadLatch**	Physical lock type is dead latch	M
10	**DoorFurniture**	Physical lock type is door furniture	M
11	**Eurocylinder**	Physical lock type is euro cylinder	M

5.2.6.22. LEDSettingEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**NoLEDSignal**	Never use LED for signalization	M

Value	Name	Summary	Conformance
1	**NoLEDSignalAccessAllowed**	Use LED signalization except for access allowed events	M
2	**LEDSignalAll**	Use LED signalization for all events	M

5.2.6.23. SoundVolumeEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**Silent**	Silent Mode	M
1	**Low**	Low Volume	M
2	**High**	High Volume	M
3	**Medium**	Medium Volume	M

5.2.6.24. EventTypeEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**Operation**	Event type is operation	M
1	**Programming**	Event type is programming	M
2	**Alarm**	Event type is alarm	M

5.2.6.25. EventSourceEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**Keypad**	Event source is keypad	M
1	**Remote**	Event source is remote	M
2	**Manual**	Event source is manual	M
3	**RFID**	Event source is RFID	M
255	**Indeterminate**	Event source is unknown	M

5.2.6.25.1. OperationEventCodeEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**UnknownOrMfgSpecific**	Event code is unknown	O
1	**Lock**	Event code is lock	O
2	**Unlock**	Event code is unlock	O
3	**LockFailureInvalidPINorRFID**	Event code is lock failure due to invalid PIN or RFID	O
4	**LockFailureInvalidSchedule**	Event code is lock failure due to invalid schedule	O
5	**UnlockFailureInvalidPINorRFID**	Event code is unlock failure due to invalid PIN or RFID	O
6	**UnlockFailureInvalidSchedule**	Event code is unlock failure due to invalid schedule	O
7	**OneTouchLock**	Event code is one touch lock	O
8	**KeyLock**	Event code is key lock	O
9	**KeyUnlock**	Event code is key unlock	O
10	**AutoLock**	Event code is auto lock	O
11	**ScheduleLock**	Event code is schedule lock	WDSCH|YDSCH
12	**ScheduleUnlock**	Event code is schedule unlock	WDSCH|YDSCH
13	**ManualLock**	Event code is manual lock (Key or Thumbturn)	O
14	**ManualUnlock**	Event code is manual unlock (Key or Thumbturn)	O
15	**NonAccessUserOperationEvent**	Event code is non access user operation	O

5.2.6.25.2. ProgrammingEventCodeEnum

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**UnknownOrMfgSpecific**	Event code is unknown	O
1	**ProgrammingCodeChanged**	Event code is code changed	O
2	**PINCodeAdded**	Event code is PIN added	O
3	**PINCodeCleared**	Event code is PIN cleared	O
4	**PINCodeChanged**	Event code is PIN changed	O
5	**RFIDCodeAdded**	Event code is RFID added	O
6	**RFIDCodeCleared**	Event code is RFID cleared	O

5.2.6.26. CredentialStruct Type

This struct SHALL indicate the credential types and their corresponding indices (if any) for the event or user record.

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0	**CredentialType**	CredentialTypeEnum	all				M
1	**CredentialIndex**	uint16	all		0		M

5.2.6.26.1. CredentialType Field

This field SHALL indicate the credential field used to authorize the lock operation.

5.2.6.26.2. CredentialIndex Field

This field SHALL indicate the index of the specific credential used to authorize the lock operation in the list of credentials identified by CredentialType (e.g. PIN, RFID, etc.). This field SHALL be set to 0 if CredentialType is ProgrammingPIN or does not correspond to a list that can be indexed into.

5.2.7. Status Codes

5.2.7.1. StatusCodeEnum Type

This data type is derived from enum8.

Value	Name	Summary
0x02	**DUPLICATE**	Entry would cause a duplicate credential/ID.
0x03	**OCCUPIED**	Entry would replace an occupied slot.

5.2.7.2. DUPLICATE Code

The provided User ID, PIN or RFID code or other credential is a duplicate of an existing entry.

5.2.7.3. OCCUPIED Code

The provided User ID, Credential ID, or entry location is already occupied. The entry might need to be deleted or a different ID or location chosen.

5.2.8. PIN/RFID Code Format

The PIN/RFID codes defined in this specification are all octet strings.

All value in the PIN/RFID code SHALL be ASCII encoded regardless if the PIN/RFID codes are number or characters. For example, code of “1, 2, 3, 4” SHALL be represented as 0x31, 0x32, 0x33, 0x34.

5.2.9. Attributes

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0000	**LockState**	LockStateEnum	desc	X P S		R V	M
0x0001	**LockType**	LockTypeEnum	desc			R V	M
0x0002	**ActuatorEnabled**	bool	all			R V	M
0x0003	**DoorState**	DoorStateEnum	desc	X P		R V	DPS
0x0004	**DoorOpenEvents**	uint32	all			RW VM	[DPS]
0x0005	**DoorClosedEvents**	uint32	all			RW VM	[DPS]
0x0006	**OpenPeriod**	uint16	all			RW VM	[DPS]

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0010	**NumberOfLogRecordsSupported**	uint16	all	F	0	R V	LOG
0x0011	**NumberOfTotalUsersSupported**	uint16	all	F	0	R V	USR
0x0012	**NumberOfPINUsersSupported**	uint16	all	F	0	R V	PIN
0x0013	**NumberOfRFIDUsersSupported**	uint16	all	F	0	R V	RID
0x0014	**NumberOfWeekDaySchedulesSupportedPerUser**	uint8	all	F	0	R V	WDSCH
0x0015	**NumberOfYearDaySchedulesSupportedPerUser**	uint8	all	F	0	R V	YDSCH
0x0016	**NumberOfHolidaySchedulesSupported**	uint8	all	F	0	R V	HDSCH
0x0017	**MaxPINCodeLength**	uint8	all	F	MS	R V	PIN
0x0018	**MinPINCodeLength**	uint8	all	F	MS	R V	PIN

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0019	**MaxRFIDCodeLength**	uint8	all	F	MS	R V	RID
0x001A	**MinRFIDCodeLength**	uint8	all	F	MS	R V	RID
0x001B	**CredentialRulesSupport**	CredentialRulesBitmap	all	F	1	R V	USR
0x001C	**NumberOfCredentialsSupportedPerUser**	uint8	all	F	0	R V	USR
0x0020	**EnableLogging**	bool	all	P	0	R[W] VA	LOG
0x0021	**Language**	string	max 3	P	MS	R[W] VM	O
0x0022	**LEDSettings**	LEDSettingEnum	all	P	0	R[W] VM	O
0x0023	**AutoRelockTime**	uint32	all	P	MS	R[W] VM	O
0x0024	**SoundVolume**	SoundVolumeEnum	all	P	0	R[W] VM	O
0x0025	**OperatingMode**	OperatingModeEnum	desc	P	0	R[W] VM	M
0x0026	**SupportedOperatingModes**	OperatingModesBitmap	all	F	0xFFF6	R V	M
0x0027	**DefaultConfigurationRegister**	ConfigurationRegisterBitmap	all	P	0	R V	O
0x0028	**EnableLocalProgramming**	bool	all	P	1	R[W] VA	O

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0029	**EnableOne-TouchLocking**	bool	all	P	0	RW VM	O
0x002A	**EnableInsideStatusLED**	bool	all	P	0	RW VM	O
0x002B	**EnablePrivacyModeButton**	bool	all	P	0	RW VM	O
0x002C	**LocalProgrammingFeatures**	LocalProgrammingFeatures Bitmap	all	P	0	R[W] VA	O
0x0030	**WrongCodeEntryLimit**	uint8	1 to 255	P	MS	R[W] VA	PIN | RID
0x0031	**UserCodeTemporaryDisableTime**	uint8	1 to 255	P	MS	R[W] VA	PIN | RID
0x0032	**SendPINOverTheAir**	bool	all	P	0	R[W] VA	[!USR & PIN]
0x0033	**RequirePINforRemoteOperation**	bool	all	P	0	R[W] VA	COTA & PIN
0x0034	**SecurityLevel**				0		D
0x0035	**ExpiringUserTimeout**	uint16	1 to 2880	P	MS	R[W] VA	[USR]
0x0040	**AlarmMask**	AlarmMaskBitmap	all	P	0xFFFF	RW VA	O
0x0041	**KeypadOperationEventMask**	EventMaskBitmap	all	P	0xFFFF	RW VA	[NOT & PIN]

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0042	**RemoteOperationEventMask**	EventMaskBitmap	all	P	0xFFFF	RW VA	[NOT]
0x0043	**ManualOperationEventMask**	EventMaskBitmap	all	P	0xFFFF	RW VA	[NOT]
0x0044	**RFIDOperationEventMask**	EventMaskBitmap	all	P	0xFFFF	RW VA	[NOT & RID]
0x0045	**KeypadProgrammingEventMask**	EventMaskBitmap	all	P	0xFFFF	RW VA	[NOT & PIN]
0x0046	**RemoteProgrammingEventMask**	EventMaskBitmap	all	P	0xFFFF	RW VA	[NOT]
0x0047	**RFIDProgrammingEventMask**	EventMaskBitmap	all	P	0xFFFF	RW VA	[NOT & RID]

AutoRelockTime, OperatingMode and SupportedOperatingModes represent mandatory fields that were previously not present or optional. Implementers should be aware that older devices may not implement them.

5.2.9.1. Scene Table Extensions

If the Scenes Management server cluster is implemented on the same endpoint, the following attribute SHALL be part of the ExtensionFieldSetStruct of the Scene Table.

- LockState

When the LockState attribute is part of a Scene table, the attribute is treated as a writable command; that is:

- Setting the LockState to Locked SHALL command the lock to lock.
- Setting the LockState to Unlocked SHALL command the lock to unlock.
- If a lock supports the Unbolting feature, setting the LockState to Unlocked SHALL unlock the door without pulling the latch.

- Setting the LockState attribute to Unlatched SHALL command the lock to unlock and pull the latch.
- Trying to write the LockState attribute to Not Fully Locked SHALL be ignored and not change the state of the lock.

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.

5.2.9.2. LockState Attribute

This attribute may be NULL if the lock hardware does not currently know the status of the locking mechanism. For example, a lock may not know the LockState status after a power cycle until the first lock actuation is completed.

The Not Fully Locked value is used by a lock to indicate that the state of the lock is somewhere between Locked and Unlocked so it is only partially secured. For example, a deadbolt could be partially extended and not in a dead latched state.

5.2.9.3. LockType Attribute

This attribute SHALL indicate the type of door lock as defined in LockTypeEnum.

5.2.9.4. ActuatorEnabled Attribute

This attribute SHALL indicate if the lock is currently able to (Enabled) or not able to (Disabled) process remote Lock, Unlock, or Unlock with Timeout commands.

This attribute has the following possible values:

Boolean Value	Summary
0	Disabled
1	Enabled

5.2.9.5. DoorState Attribute

This attribute SHALL indicate the current door state as defined in DoorStateEnum.

This attribute SHALL be null only if an internal error prevents the retrieval of the current door state.

5.2.9.6. DoorOpenEvents Attribute

This attribute SHALL hold the number of door open events that have occurred since it was last zeroed.

5.2.9.7. DoorClosedEvents Attribute

This attribute SHALL hold the number of door closed events that have occurred since it was last zeroed.

5.2.9.8. OpenPeriod Attribute

This attribute SHALL hold the number of minutes the door has been open since the last time it transitioned from closed to open.

5.2.9.9. NumberOfLogRecordsSupported Attribute

This attribute SHALL indicate the number of available log records.

5.2.9.10. NumberOfTotalUsersSupported Attribute

This attribute SHALL indicate the number of total users supported by the lock.

5.2.9.11. NumberOfPINUsersSupported Attribute

This attribute SHALL indicate the number of PIN users supported.

5.2.9.12. NumberOfRFIDUsersSupported Attribute

This attribute SHALL indicate the number of RFID users supported.

5.2.9.13. NumberOfWeekDaySchedulesSupportedPerUser Attribute

This attribute SHALL indicate the number of configurable week day schedule supported per user.

5.2.9.14. NumberOfYearDaySchedulesSupportedPerUser Attribute

This attribute SHALL indicate the number of configurable year day schedule supported per user.

5.2.9.15. NumberOfHolidaySchedulesSupported Attribute

This attribute SHALL indicate the number of holiday schedules supported for the entire door lock device.

5.2.9.16. MaxPINCodeLength Attribute

This attribute SHALL indicate the maximum length in bytes of a PIN Code on this device.

5.2.9.17. MinPINCodeLength Attribute

This attribute SHALL indicate the minimum length in bytes of a PIN Code on this device.

5.2.9.18. MaxRFIDCodeLength Attribute

This attribute SHALL indicate the maximum length in bytes of a RFID Code on this device. The value depends on the RFID code range specified by the manufacturer, if media anti-collision identifiers (UID) are used as RFID code, a value of 20 (equals 10 Byte ISO 14443A UID) is recommended.

5.2.9.19. MinRFIDCodeLength Attribute

This attribute SHALL indicate the minimum length in bytes of a RFID Code on this device. The value depends on the RFID code range specified by the manufacturer, if media anti-collision identifiers (UID) are used as RFID code, a value of 8 (equals 4 Byte ISO 14443A UID) is recommended.

5.2.9.20. CredentialRulesSupport Attribute

This attribute SHALL contain a bitmap with the bits set for the values of CredentialRuleEnum supported on this device.

5.2.9.21. NumberOfCredentialsSupportedPerUser Attribute

This attribute SHALL indicate the number of credentials that could be assigned for each user.

Depending on the value of NumberOfRFIDUsersSupported and NumberOfPINUsersSupported it may not be possible to assign that number of credentials for a user.

For example, if the device supports only PIN and RFID credential types, NumberOfCredentialsSupportedPerUser is set to 10, NumberOfPINUsersSupported is set to 5 and NumberOfRFIDUsersSupported is set to 3, it will not be possible to actually assign 10 credentials for a user because maximum number of credentials in the database is 8.

5.2.9.22. EnableLogging Attribute

This attribute SHALL enable/disable event logging.

When event logging is enabled, all event messages are stored on the lock for retrieval. Logging events can be, but are not limited to, Tamper Alarm, Lock, Unlock, AutoRelock, User Code Added, User Code Cleared, Schedule Added, and Schedule Cleared. For a full detail of all the possible alarms and events, please refer to the full list in the Alarm and Event Masks Attribute Set.

5.2.9.23. Language Attribute

This attribute SHALL indicate the language for the on-screen or audible user interface using a 2-byte language code from ISO-639-1.

5.2.9.24. LEDSettings Attribute

This attribute SHALL indicate the settings for the LED support, as defined by LEDSettingEnum.

5.2.9.25. AutoRelockTime Attribute

This attribute SHALL indicate the number of seconds to wait after unlocking a lock before it automatically locks again. 0=disabled. If set, unlock operations from any source will be timed. For one time unlock with timeout use the specific command.

5.2.9.26. SoundVolume Attribute

This attribute SHALL indicate the sound volume on a door lock as defined by SoundVolumeEnum.

5.2.9.27. OperatingMode Attribute

This attribute SHALL indicate the current operating mode of the lock as defined in OperatingModeEnum.

5.2.9.28. SupportedOperatingModes Attribute

This attribute SHALL contain a bitmap with all operating bits of the OperatingMode attribute supported by the lock. All operating modes NOT supported by a lock SHALL be set to one. The value of the OperatingMode enumeration defines the related bit to be set.

5.2.9.29. DefaultConfigurationRegister Attribute

This attribute SHALL represent the default configurations as they are physically set on the device (example: hardware dip switch setting, etc...) and represents the default setting for some of the attributes within this cluster (for example: LED, Auto Lock, Sound Volume, and Operating Mode attributes).

This is a read-only attribute and is intended to allow clients to determine what changes MAY need to be made without having to query all the included attributes. It MAY be beneficial for the clients to know what the device's original settings were in the event that the device needs to be restored to factory default settings.

If the Client device would like to query and modify the door lock server's operating settings, it SHOULD send read and write attribute requests to the specific attributes.

For example, the Sound Volume attribute default value is Silent Mode. However, it is possible that the current Sound Volume is High Volume. Therefore, if the client wants to query/modify the current Sound Volume setting on the server, the client SHOULD read/write to the Sound Volume attribute.

5.2.9.29.1. LocalProgramming Bit

This bit SHALL indicate the default value of the EnableLocalProgramming attribute.

5.2.9.29.2. KeypadInterface Bit

This bit SHALL indicate the default state of the keypad interface.

5.2.9.29.3. RemoteInterface Bit

This bit SHALL indicate the default state of the remote interface.

5.2.9.29.4. SoundVolume Bit

This bit SHALL indicate the default value of SoundVolume attribute.

5.2.9.29.5. AutoRelockTime Bit

This bit SHALL indicate the default value of AutoRelockTime attribute.

5.2.9.29.6. LEDSettings Bit

This bit SHALL indicate the default value of LEDSettings attribute.

5.2.9.30. EnableLocalProgramming Attribute

This attribute SHALL enable/disable local programming on the door lock of certain features (see LocalProgrammingFeatures attribute). If this value is set to TRUE then local programming is enabled on the door lock for all features. If it is set to FALSE then local programming is disabled on the door lock for those features whose bit is set to 0 in the LocalProgrammingFeatures attribute. Local programming SHALL be enabled by default.

5.2.9.31. EnableOneTouchLocking Attribute

This attribute SHALL enable/disable the ability to lock the door lock with a single touch on the door lock.

5.2.9.32. EnableInsideStatusLED Attribute

This attribute SHALL enable/disable an inside LED that allows the user to see at a glance if the door is locked.

5.2.9.33. EnablePrivacyModeButton Attribute

This attribute SHALL enable/disable a button inside the door that is used to put the lock into privacy mode. When the lock is in privacy mode it cannot be manipulated from the outside.

5.2.9.34. LocalProgrammingFeatures Attribute

This attribute SHALL indicate the local programming features that will be disabled when EnableLocalProgramming attribute is set to False. If a door lock doesn't support disabling one aspect of local programming it SHALL return CONSTRAINT_ERROR during a write operation of this attribute. If the EnableLocalProgramming attribute is set to True then all local programming features SHALL be enabled regardless of the bits set to 0 in this attribute.

The features that can be disabled from local programming are defined in LocalProgrammingFeaturesBitmap.

5.2.9.35. WrongCodeEntryLimit Attribute

This attribute SHALL indicate the number of incorrect Pin codes or RFID presentment attempts a user is allowed to enter before the lock will enter a lockout state. The value of this attribute is compared to all failing forms of credential presentation, including Pin codes used in an Unlock Command when RequirePINforRemoteOperation is set to true. Valid range is 1-255 incorrect attempts. The lockout state will be for the duration of UserCodeTemporaryDisableTime. If the attribute accepts writes and an attempt to write the value 0 is made, the device SHALL respond with CONSTRAINT_ERROR.

The lock MAY reset the counter used to track incorrect credential presentations as required by internal logic, environmental events, or other reasons. The lock SHALL reset the counter if a valid credential is presented.

5.2.9.36. UserCodeTemporaryDisableTime Attribute

This attribute SHALL indicate the number of seconds that the lock shuts down following wrong code entry. Valid range is 1-255 seconds. Device can shut down to lock user out for specified amount of time. (Makes it difficult to try and guess a PIN for the device.) If the attribute accepts writes and an attempt to write the attribute to 0 is made, the device SHALL respond with CONSTRAINT_ERROR.

5.2.9.37. SendPINOverTheAir Attribute

This attribute SHALL indicate the door locks ability to send PINs over the air. If the attribute is True it is ok for the door lock server to send PINs over the air. This attribute determines the behavior of the server's TX operation. If it is false, then it is not ok for the device to send PIN in any messages over the air.

The PIN field within any door lock cluster message SHALL keep the first octet unchanged and masks the actual code by replacing with 0xFF. For example (PIN "1234"): If the attribute value is True, 0x04 0x31 0x32 0x33 0x34 SHALL be used in the PIN field in any door lock cluster message payload. If the attribute value is False, 0x04 0xFF 0xFF 0xFF 0xFF SHALL be used.

5.2.9.38. RequirePINForRemoteOperation Attribute

This attribute SHALL indicate if the door lock requires an optional PIN. If this attribute is set to True, the door lock server requires that an optional PINs be included in the payload of remote lock operation events like Lock, Unlock, Unlock with Timeout and Toggle in order to function.

5.2.9.39. ExpiringUserTimeout Attribute

This attribute SHALL indicate the number of minutes a PIN, RFID, Fingerprint, or other credential associated with a user of type ExpiringUser SHALL remain valid after its first use before expiring. When the credential expires the UserStatus for the corresponding user record SHALL be set to OccupiedDisabled.

5.2.9.40. AlarmMask Attribute

This attribute is only supported if the Alarms cluster is on the same endpoint. The alarm mask is used to turn on/off alarms for particular functions. Alarms for an alarm group are enabled if the associated alarm mask bit is set. Each bit represents a group of alarms. Entire alarm groups can be turned on or off by setting or clearing the associated bit in the alarm mask.

This mask DOES NOT apply to the Events mechanism of this cluster.

Alarm Code Value	AlarmMaskBitmap Bit
0	0
1	1
2	2
3	3
4	4

Alarm Code Value	AlarmMaskBitmap Bit
5	5
6	6

5.2.9.41. KeypadOperationEventMask Attribute

Event mask used to turn on and off the transmission of keypad operation events. This mask DOES NOT apply to the storing of events in the event log. This mask only applies to the Operation Event Notification Command.

This mask DOES NOT apply to the Events mechanism of this cluster.

This attribute has the following possible values:

Event Source	Operation Event Code	Bit	Summary
0	0	0	Unknown or manufacturer-specific keypad operation event
0	1	1	Lock, source: keypad
0	2	2	Unlock, source: keypad
0	3	3	Lock, source: keypad, error: invalid PIN
0	4	4	Lock, source: keypad, error: invalid schedule
0	5	5	Unlock, source: keypad, error: invalid code
0	6	6	Unlock, source: keypad, error: invalid schedule
0	15	7	Non-Access User operation event, source keypad.

5.2.9.42. RemoteOperationEventMask Attribute

Event mask used to turn on and off the transmission of remote operation events. This mask DOES NOT apply to the storing of events in the event log. This mask only applies to the Operation Event Notification Command.

This mask DOES NOT apply to the Events mechanism of this cluster.

This attribute has the following possible values:

Event Source	Operation Event Code	Bit	Summary
1	0	0	Unknown or manufacturer-specific remote operation event
1	1	1	Lock, source: remote
1	2	2	Unlock, source: remote
1	3	3	Lock, source: remote, error: invalid code
1	4	4	Lock, source: remote, error: invalid schedule
1	5	5	Unlock, source: remote, error: invalid code
1	6	6	Unlock, source: remote, error: invalid schedule

5.2.9.43. ManualOperationEventMask Attribute

Event mask used to turn on and off manual operation events. This mask DOES NOT apply to the storing of events in the event log. This mask only applies to the Operation Event Notification Command.

This mask DOES NOT apply to the Events mechanism of this cluster.

This attribute has the following possible values:

Event Source	Operation Event Code	Bit	Summary
2	0	0	Unknown or manufacturer-specific manual operation event
2	1	1	Thumbturn Lock
2	2	2	Thumbturn Unlock
2	7	3	One Touch Lock
2	8	4	Key Lock
2	9	5	Key Unlock
2	10	6	Auto lock
2	11	7	Schedule Lock
2	12	8	Schedule Unlock
2	13	9	Manual Lock (Key or Thumbturn)
2	14	10	Manual Unlock (Key or Thumbturn)

5.2.9.44. RFIDOperationEventMask Attribute

Event mask used to turn on and off RFID operation events. This mask DOES NOT apply to the storing of events in the event log. This mask only applies to the Operation Event Notification Command.

This mask DOES NOT apply to the Events mechanism of this cluster.

This attribute has the following possible values:

Event Source	Operation Event Code	Bit	Summary
3	0	0	Unknown or manufacturer-specific keypad operation event
3	1	1	Lock, source: RFID
3	2	2	Unlock, source: RFID
3	3	3	Lock, source: RFID, error: invalid RFID ID
3	4	4	Lock, source: RFID, error: invalid schedule
3	5	5	Unlock, source: RFID, error: invalid RFID ID
3	6	6	Unlock, source: RFID, error: invalid schedule

5.2.9.45. KeypadProgrammingEventMask Attribute

Event mask used to turn on and off keypad programming events. This mask DOES NOT apply to the storing of events in the event log. This mask only applies to the Programming Event Notification Command.

This mask DOES NOT apply to the Events mechanism of this cluster.

This attribute has the following possible values:

Event Source	Program Event Code	Bit	Summary
0	0	0	Unknown or manufacturer-specific keypad programming event

Event Source	Program Event Code	Bit	Summary
0	1	1	Programming PIN code changed, source: keypad User ID: programming user ID. PIN: default or programming PIN code if codes can be sent over the air per attribute. User type: default User Status: default
0	2	2	PIN added, source: keypad User ID: user ID that was added. PIN: code that was added (if codes can be sent over the air per attribute.) User type: default or type added. User Status: default or status added.
0	3	3	PIN cleared, source: keypad User ID: user ID that was cleared. PIN: code that was cleared (if codes can be sent over the air per attribute.) User type: default or type cleared. User Status: default or status cleared.

Event Source	Program Event Code	Bit	Summary
0	4	4	PIN changed Source: keypad User ID: user ID that was changed PIN: code that was changed (if codes can be sent over the air per attribute.) User type: default or type changed. User Status: default or status changed.

5.2.9.46. RemoteProgrammingEventMask Attribute

Event mask used to turn on and off remote programming events. This mask DOES NOT apply to the storing of events in the event log. This mask only applies to the Programming Event Notification Command.

This mask DOES NOT apply to the Events mechanism of this cluster.

This attribute has the following possible values:

Event Source	Program Event Code	Bit	Summary
1	0	0	Unknown or manufacturer-specific remote programming event.
1	2	2	PIN added, source remote Same as keypad source above
1	3	3	PIN cleared, source remote Same as keypad source above.

Event Source	Program Event Code	Bit	Summary
1	4	4	PIN changed Source remote Same as keypad source above
1	5	5	RFID code added, Source remote
1	6	6	RFID code cleared, Source remote

5.2.9.47. RFIDProgrammingEventMask Attribute

Event mask used to turn on and off RFID programming events. This mask DOES NOT apply to the storing of events in the event log. This mask only applies to the Programming Event Notification Command.

This mask DOES NOT apply to the Events mechanism of this cluster.

This attribute has the following possible values:

Event Source	Program Event Code	Bit	Summary
3	0	0	Unknown or manufacturer-specific keypad programming event
3	5	5	ID Added, Source: RFID User ID: user ID that was added. ID: ID that was added (if codes can be sent over the air per attribute.) User Type: default or type added. User Status: default or status added.

Event Source	Program Event Code	Bit	Summary
3	6	6	ID cleared, Source: RFID User ID: user ID that was cleared. ID: ID that was cleared (if codes can be sent over the air per attribute.) User Type: default or type cleared. User Status: default or status cleared.

This mask DOES NOT apply to the Events mechanism of this cluster.

5.2.10. Commands

ID	Name	Direction	Response	Access	Conformance
0x00	**LockDoor**	client ⇒ server	Y	O T	M
0x01	**UnlockDoor**	client ⇒ server	Y	O T	M
0x02	**Toggle**	client ⇒ server	Y	O T	X
0x03	**UnlockWith-Timeout**	client ⇒ server	Y	O T	O
0x04	**GetLogRecord**	client ⇒ server	GetLogRecor-dResponse	M	LOG
0x04	**GetLogRecor-dResponse**	client ⇐ server	N		LOG
0x05	**SetPINCode**	client ⇒ server	Y	A T	!USR & PIN
0x06	**GetPINCode**	client ⇒ server	GetPINCodeRe-sponse	A	!USR & PIN
0x06	**GetPIN-CodeResponse**	client ⇐ server	N		!USR & PIN
0x07	**ClearPINCode**	client ⇒ server	Y	A T	!USR & PIN
0x08	**ClearAllPIN-Codes**	client ⇒ server	Y	A T	!USR & PIN
0x09	**SetUserStatus**	client ⇒ server	Y	A	!USR & (PIN | RID | FGP)

ID	Name	Direction	Response	Access	Conformance
0x0A	**GetUserStatus**	client ⇒ server	GetUserStatus-Response	A	!USR & (PIN | RID | FGP)
0x0A	**GetUserStatusResponse**	client ⇐ server	N		!USR
0x0B	**SetWeekDaySchedule**	client ⇒ server	Y	A	WDSCH
0x0C	**GetWeekDaySchedule**	client ⇒ server	GetWeekDaySchedule-Response	A	WDSCH
0x0C	**GetWeekDayScheduleResponse**	client ⇐ server	N		WDSCH
0x0D	**ClearWeekDaySchedule**	client ⇒ server	Y	A	WDSCH
0x0E	**SetYearDaySchedule**	client ⇒ server	Y	A	YDSCH
0x0F	**GetYearDaySchedule**	client ⇒ server	GetYearDaySchedule-Response	A	YDSCH
0x0F	**GetYearDayScheduleResponse**	client ⇐ server	N		YDSCH
0x10	**ClearYearDaySchedule**	client ⇒ server	Y	A	YDSCH
0x11	**SetHolidaySchedule**	client ⇒ server	Y	A	HDSCH
0x12	**GetHolidaySchedule**	client ⇒ server	GetHolidayScheduleResponse	A	HDSCH
0x12	**GetHolidayScheduleResponse**	client ⇐ server	N		HDSCH
0x13	**ClearHolidaySchedule**	client ⇒ server	Y	A	HDSCH
0x14	**SetUserType**	client ⇒ server	Y	A	!USR & (PIN | RID | FGP)
0x15	**GetUserType**	client ⇒ server	GetUserTypeResponse	A	!USR & (PIN | RID | FGP)
0x15	**GetUserTypeResponse**	client ⇐ server	N		!USR

ID	Name	Direction	Response	Access	Conformance
0x16	**SetRFIDCode**	client ⇒ server	Y	A T	!USR & RID
0x17	**GetRFIDCode**	client ⇒ server	GetRFID-CodeResponse	A	!USR & RID
0x17	**GetRFID-CodeResponse**	client ⇐ server	N		!USR & RID
0x18	**ClearRFID-Code**	client ⇒ server	Y	A T	!USR & RID
0x19	**ClearAllRFID-Codes**	client ⇒ server	Y	A T	!USR & RID
0x1A	**SetUser**	client ⇒ server	Y	A T	USR
0x1B	**GetUser**	client ⇒ server	GetUserRe-sponse	A	USR
0x1C	**GetUserRe-sponse**	client ⇐ server	N		USR
0x1D	**ClearUser**	client ⇒ server	Y	A T	USR
0x20	**Oper-atingEventNo-tification**	client ⇐ server	N		[NOT]
0x21	**Program-mingEventNo-tification**	client ⇐ server	N		[NOT]
0x22	**SetCredential**	client ⇒ server	SetCredential-Response	A T	USR
0x23	**SetCredential-Response**	client ⇐ server	N		USR
0x24	**GetCredential-Status**	client ⇒ server	GetCredential-StatusResponse	A	USR
0x25	**GetCredential-StatusRe-sponse**	client ⇐ server	N		USR
0x26	**ClearCreden-tial**	client ⇒ server	Y	A T	USR
0x27	**UnboltDoor**	client ⇒ server	Y	O T	UBOLT

5.2.10.1. LockDoor Command

This command causes the lock device to lock the door. This command includes an optional code for the lock. The door lock MAY require a PIN depending on the value of the RequirePINForRemoteOp-eration attribute.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**PINCode** **PIN/RFID Code†**	octstr				[COTA & PIN]

† The PIN/RFID Code is an obsolete field name, use PINCode instead.

5.2.10.1.1. PINCode Field

If the RequirePINforRemoteOperation attribute is True then PINCode field SHALL be provided and the door lock SHALL NOT grant access if it is not provided.

If the PINCode field is provided, the door lock SHALL verify PINCode before granting access regardless of the value of RequirePINForRemoteOperation attribute.

When the PINCode field is provided an invalid PIN will count towards the WrongCodeEntryLimit and the UserCodeTemporaryDisableTime will be triggered if the WrongCodeEntryLimit is exceeded. The lock SHALL ignore any attempts to lock/unlock the door until the UserCodeTemporaryDisableTime expires.

5.2.10.2. UnlockDoor Command

This command causes the lock device to unlock the door. This command includes an optional code for the lock. The door lock MAY require a code depending on the value of the RequirePINForRemoteOperation attribute.

NOTE	If the attribute AutoRelockTime is supported the lock will transition to the locked state when the auto relock time has expired.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**PINCode** **PIN/RFID Code†**	octstr				[COTA & PIN]

† The PIN/RFID Code is an obsolete field name, use PINCode instead.

5.2.10.2.1. PINCode Field

See PINCode field.

5.2.10.3. UnlockWithTimeout Command

This command causes the lock device to unlock the door with a timeout parameter. After the time in seconds specified in the timeout field, the lock device will relock itself automatically. This timeout parameter is only temporary for this message transition and overrides the default relock time

as specified in the AutoRelockTime attribute. If the door lock device is not capable of or does not want to support temporary Relock Timeout, it SHOULD NOT support this optional command.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**Timeout**	uint16				M
1	**PINCode** **PIN/RFID Code†**	octstr				[COTA & PIN]

† The PIN/RFID Code is an obsolete field name, use PINCode instead.

5.2.10.3.1. Timeout Field

This field SHALL indicate the timeout in seconds to wait before relocking the door lock. This value is independent of the AutoRelockTime attribute value.

5.2.10.3.2. PINCode Field

See PINCode field.

5.2.10.4. GetLogRecord Command

Request a log record. Log number is between 1 – [Number of Log Records Supported attribute]. If log number 0 is requested then the most recent log entry is returned.

Log record format: The log record format is defined in the description of the GetLogRecordResponse command.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**LogIndex**	uint16	desc			M

5.2.10.5. GetLogRecordResponse Command

Returns the specified log record. If an invalid log entry ID was requested, it is set to 0 and the most recent log entry will be returned.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**LogEntryID**	uint16	desc			M
1	**Timestamp**	epoch-s	all			M
2	**EventType**	EventTypeEnum	all			M
3	**Source**	EventSourceEnum	all			M

ID	Name	Type	Constraint	Quality	Default	Conformance
4	**EventID**	uint8	desc			M
5	**UserID**	uint16	desc			M
6	**PIN**	octstr				M

5.2.10.5.1. LogEntryID Field

This field SHALL indicate the index into the log table where this log entry is stored. If the log entry requested is 0, the most recent log is returned with the appropriate log entry ID.

5.2.10.5.2. Timestamp Field

This field SHALL indicate the timestamp for all events and alarms on the door lock in Epoch Time in Seconds with local time offset based on the local timezone and DST offset on the day of the event.

5.2.10.5.3. EventType Field

This field SHALL indicate the type of event that took place on the door lock, as defined in EventTypeEnum.

5.2.10.5.4. Source Field

This field SHALL indicate the source value as defined in EventSourceEnum.

If the EventType is 2 (Alarm) then the source SHOULD be, but does not have to be 255 (Indeterminate).

5.2.10.5.5. EventID Field

This field SHALL indicate the type of event that took place on the door lock depending on the event code table provided for a given event type and source. See Operation Event Codes.

5.2.10.5.6. UserID Field

This field SHALL indicate the ID of the user who generated the event on the door lock if one is available. Otherwise, the value is 0xFFFF.

5.2.10.5.7. PIN Field

This field SHALL indicate the PIN code or RFID code that was used to create the event on the door lock if one is available.

5.2.10.6. SetPINCode Command

Set a PIN Code into the lock.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**UserID**	uint16	desc			M
1	**UserStatus**	UserStatusEnum	desc	X	OccupiedEnabled	M
2	**UserType**	UserTypeEnum	all	X	UnrestrictedUser	M
3	**PIN**	octstr				M

Return status is a global status code or a cluster-specific status code from the Status Codes table and SHALL be one of the following values:

Name	Summary
SUCCESS	Setting PIN code was successful.
FAILURE	Setting PIN code failed.
CONSTRAINT_ERROR	Setting PIN code failed because User ID requested was out of range.
DUPLICATE	Setting PIN code failed because it would create a duplicate PIN code.

5.2.10.6.1. UserID Field

This field SHALL indicate the user ID. The value of the UserID field SHALL be between 0 and the value of the NumberOfPINUsersSupported attribute.

5.2.10.6.2. UserStatus Field

This field SHALL indicate the user status. Only the values 1 (Occupied/Enabled) and 3 (Occupied/Disabled) are allowed for UserStatus.

5.2.10.7. GetPINCode Command

Retrieve a PIN Code.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**UserID**	uint16	desc			M

5.2.10.7.1. UserID Field

This field SHALL indicate the user ID. The value of the UserID field SHALL be between 0 and the value of the NumberOfPINUsersSupported attribute.

5.2.10.8. GetPINCodeResponse Command

Returns the PIN for the specified user ID.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**UserID**	uint16	desc			M
1	**UserStatus**	UserStatusEnum	desc	X	Available	M
2	**UserType**	UserTypeEnum	desc	X		M
3	**PINCode**	octstr		X	empty	M

If the requested UserID is valid and the Code doesn't exist, Get RFID Code Response SHALL have the following format:

UserID = requested User ID

UserStatus = 0 (Available)

UserType = Null (Not supported)

PINCode = 0 (zero length)

If the requested UserID is invalid, send Default Response with an error status. The error status SHALL be equal to CONSTRAINT_ERROR when User_ID is less than the max number of users supported, and NOT_FOUND if greater than or equal to the max number of users supported.

5.2.10.9. ClearPINCode Command

Clear a PIN code or all PIN codes.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**PINSlotIndex** **User ID†**	uint16	1 to NumberOfPINUsersSupported, 0xFFFE			M

† The User ID is an obsolete field name, use PINSlotIndex instead.

For each PIN Code cleared whose user doesn't have a RFID Code or other credential type, then corresponding user record's UserStatus value SHALL be set to Available, and UserType value SHALL be set to UnrestrictedUser and all schedules SHALL be cleared.

5.2.10.9.1. PINSlotIndex Field

This field SHALL specify a valid PIN code slot index or 0xFFFE to indicate all PIN code slots SHALL be cleared.

5.2.10.10. ClearAllPINCodes Command

Clear out all PINs on the lock.

NOTE	On the server, the clear all PIN codes command SHOULD have the same effect as the ClearPINCode command with respect to the setting of user status, user type and schedules.

5.2.10.11. SetUserStatus Command

Set the status of a user ID.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**UserID**	uint16	desc			M
1	**UserStatus**	UserStatusEnum	desc			M

5.2.10.11.1. UserID Field

This field SHALL indicate the user ID. The value of the UserID field SHALL be between 0 and the value of the NumberOfPINUsersSupported attribute.

5.2.10.11.2. UserStatus Field

UserStatus value of Available is not allowed. In order to clear a user id, the ClearUser Command SHALL be used. For user status value please refer to UserStatusEnum.

5.2.10.12. GetUserStatus Command

Get the status of a user.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**UserID**	uint16	desc			M

5.2.10.12.1. UserID Field

This field SHALL indicate the user ID. The value of the UserID field SHALL be between 0 and the value of the NumberOfPINUsersSupported attribute.

5.2.10.13. GetUserStatusResponse Command

Returns the user status for the specified user ID.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**UserID**	uint16	desc			M

ID	Name	Type	Constraint	Quality	Default	Conformance
1	**UserStatus**	UserStatusEnum	all			M

5.2.10.13.1. UserID Field

This field SHALL indicate the user ID provided in the request.

5.2.10.13.2. UserStatus Field

This field SHALL indicate the current status of the requested user ID.

5.2.10.14. SetWeekDaySchedule Command

Set a weekly repeating schedule for a specified user.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**WeekDayIndex** **Schedule ID†**	uint8	1 to NumberOfWeekDaySchedulesSupportedPerUser			M
1	**UserIndex** **User ID†**	uint16	1 to NumberOfTotalUsersSupported			M
2	**DaysMask**	DaysMaskBitmap				M
3	**StartHour**	uint8	0 to 23			M
4	**StartMinute**	uint8	0 to 59			M
5	**EndHour**	uint8	0 to 23			M
6	**EndMinute**	uint8	0 to 59			M

† The Schedule ID and User ID are obsolete field names, use WeekDayIndex and UserIndex instead, respectively.

The associated UserType MAY be changed to ScheduleRestrictedUser by the lock when a Week Day schedule is set.

Return status SHALL be one of the following values:

Name	Summary
SUCCESS	Setting Week Day schedule was successful.

Name	Summary
FAILURE	Some unexpected internal error occurred setting Week Day schedule.
INVALID_COMMAND	One or more fields violates constraints or is invalid. Door lock is unable to set Week Day schedule for more than one day in DaysMask map (e.g. need to use separate schedules for each day).

5.2.10.14.1. WeekDayIndex Field

This field SHALL indicate the index of the Week Day schedule.

5.2.10.14.2. UserIndex Field

This field SHALL indicate the user ID.

5.2.10.14.3. DaysMask Field

This field SHALL indicate which week days the schedule is active.

5.2.10.14.4. StartHour Field

This field SHALL indicate the starting hour for the Week Day schedule.

5.2.10.14.5. StartMinute Field

This field SHALL indicate the starting minute for the Week Day schedule.

5.2.10.14.6. EndHour Field

This field SHALL indicate the ending hour for the Week Day schedule. EndHour SHALL be equal to or greater than StartHour.

5.2.10.14.7. EndMinute Field

This field SHALL indicate the ending minute for the Week Day schedule. If EndHour is equal to StartHour then EndMinute SHALL be greater than StartMinute.

If the EndHour is equal to 23 and the EndMinute is equal to 59 the Lock SHALL grant access to the user up until 23:59:59.

5.2.10.15. GetWeekDaySchedule Command

Retrieve the specific weekly schedule for the specific user.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**WeekDayIndex** **Schedule ID†**	uint8	1 to NumberOfWeekDaySchedulesSupportedPerUser			M
1	**UserIndex** **User ID†**	uint16	1 to NumberOfTotalUsersSupported			M

† The Schedule ID and User ID are obsolete field names, use WeekDayIndex and UserIndex instead, respectively.

5.2.10.16. GetWeekDayScheduleResponse Command

Returns the weekly repeating schedule data for the specified schedule index.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**WeekDayIndex** **Schedule ID†**	uint8	1 to NumberOfWeekDaySchedulesSupportedPerUser			M
1	**UserIndex** **User ID†**	uint16	1 to NumberOfTotalUsersSupported			M
2	**Status**	enum8	desc		SUCCESS	M
3	**DaysMask**	DaysMaskBitmap				O
4	**StartHour**	uint8	0 to 23			O
5	**StartMinute**	uint8	0 to 59			O
6	**EndHour**	uint8	0 to 23			O
7	**EndMinute**	uint8	0 to 59			O

† The Schedule ID and User ID are obsolete field names, use WeekDayIndex and UserIndex instead, respectively.

5.2.10.16.1. WeekDayIndex Field

This field SHALL indicate the index of the Week Day schedule.

5.2.10.16.2. UserIndex Field

This field SHALL indicate the user ID.

5.2.10.16.3. Status Field

Status SHALL be one of the following values:

- SUCCESS if both WeekDayIndex and UserIndex are valid and there is a corresponding schedule entry.
- INVALID_COMMAND if either WeekDayIndex and/or UserIndex values are not within valid range
- NOT_FOUND if no corresponding schedule entry found for WeekDayIndex.
- NOT_FOUND if no corresponding user entry found for UserIndex.

If this field is SUCCESS, the optional fields for this command SHALL be present. For other (error) status values, only the fields up to the status field SHALL be present.

5.2.10.16.4. StartHour Field

This field SHALL indicate the starting hour for the Week Day schedule.

5.2.10.16.5. StartMinute Field

This field SHALL indicate the starting minute for the Week Day schedule.

5.2.10.16.6. EndHour Field

This field SHALL indicate the ending hour for the Week Day schedule. EndHour SHALL be equal to or greater than StartHour.

5.2.10.16.7. EndMinute Field

This field SHALL indicate the ending minute for the Week Day schedule. If EndHour is equal to StartHour then EndMinute SHALL be greater than StartMinute.

5.2.10.17. ClearWeekDaySchedule Command

Clear the specific weekly schedule or all weekly schedules for the specific user.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**WeekDayIndex** **Schedule ID†**	uint8	1 to NumberOfWeekDaySchedulesSupportedPerUser, 0xFE			M

ID	Name	Type	Constraint	Quality	Default	Conformance
1	**UserIndex** **User ID†**	uint16	1 to NumberOfTotalUsersSupported			M

† The Schedule ID and User ID are obsolete field names, use WeekDayIndex and UserIndex instead, respectively.

Return status SHALL be one of the following values:

Name	Summary
SUCCESS	Clearing the requested WeekDaySchedule was successful.
FAILURE	Some unexpected internal error occurred clearing Week Day schedule.
INVALID_COMMAND	One or more fields violates constraints or is invalid.

5.2.10.17.1. WeekDayIndex Field

This field SHALL indicate the Week Day schedule index to clear or 0xFE to clear all Week Day schedules for the specified user.

5.2.10.17.2. UserIndex Field

This field SHALL indicate the user ID.

5.2.10.18. SetYearDaySchedule Command

Set a time-specific schedule ID for a specified user.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**YearDayIndex** **Schedule ID†**	uint8	1 to NumberOfYearDaySchedulesSupportedPerUser			M
1	**UserIndex** **User ID†**	uint16	1 to NumberOfTotalUsersSupported			M
2	**LocalStartTime**	epoch-s	all			M

ID	Name	Type	Constraint	Quality	Default	Conformance
3	**LocalEndTime**	epoch-s	all			M

† The Schedule ID and User ID are obsolete field names, use YearDayIndex and UserIndex instead, respectively.

The associated UserType MAY be changed to ScheduleRestrictedUser by the lock when a Year Day schedule is set.

Return status SHALL be one of the following values:

Name	Summary
SUCCESS	Setting Year Day schedule was successful.
FAILURE	Some unexpected internal error occurred setting Year Day schedule.
INVALID_COMMAND	One or more fields violates constraints or is invalid.

5.2.10.18.1. YearDayIndex Field

This field SHALL indicate the index of the Year Day schedule.

5.2.10.18.2. UserIndex Field

This field SHALL indicate the user ID.

5.2.10.18.3. LocalStartTime Field

This field SHALL indicate the starting time for the Year Day schedule in Epoch Time in Seconds with local time offset based on the local timezone and DST offset on the day represented by the value.

5.2.10.18.4. LocalEndTime Field

This field SHALL indicate the ending time for the Year Day schedule in Epoch Time in Seconds with local time offset based on the local timezone and DST offset on the day represented by the value. LocalEndTime SHALL be greater than LocalStartTime.

5.2.10.19. GetYearDaySchedule Command

Retrieve the specific year day schedule for the specific schedule and user indexes.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**YearDayIndex** **Schedule ID†**	uint8	1 to NumberOfYearDaySchedulesSupportedPerUser			M
1	**UserIndex** **User ID†**	uint16	1 to NumberOfTotalUsersSupported			M

† The Schedule ID and User ID are obsolete field names, use YearDayIndex and UserIndex instead, respectively.

5.2.10.20. GetYearDayScheduleResponse Command

Returns the year day schedule data for the specified schedule and user indexes.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**YearDayIndex** **Schedule ID†**	uint8	1 to NumberOfYearDaySchedulesSupportedPerUser			M
1	**UserIndex** **User ID†**	uint16	1 to NumberOfTotalUsersSupported			M
2	**Status**	enum8	desc		SUCCESS	M
2	**LocalStartTime**	epoch-s	all			O
3	**LocalEndTime**	epoch-s	all			O

† The Schedule ID and User ID are obsolete field names, use YearDayIndex and UserIndex instead, respectively.

5.2.10.20.1. YearDayIndex Field

This field SHALL indicate the index of the Year Day schedule.

5.2.10.20.2. UserIndex Field

This field SHALL indicate the user ID.

5.2.10.20.3. Status Field

Status SHALL be one of the following values:

- SUCCESS if both YearDayIndex and UserIndex are valid and there is a corresponding schedule entry.
- INVALID_COMMAND if either YearDayIndex and/or UserIndex values are not within valid range
- NOT_FOUND if no corresponding schedule entry found for YearDayIndex.
- NOT_FOUND if no corresponding user entry found for UserIndex.

If this field is SUCCESS, the optional fields for this command SHALL be present. For other (error) status values, only the fields up to the status field SHALL be present.

5.2.10.20.4. LocalStartTime Field

This field SHALL indicate the starting time for the Year Day schedule in Epoch Time in Seconds with local time offset based on the local timezone and DST offset on the day represented by the value. This SHALL be null if the schedule is not set for the YearDayIndex and UserIndex provided.

5.2.10.20.5. LocalEndTime Field

This field SHALL indicate the ending time for the Year Day schedule in Epoch Time in Seconds with local time offset based on the local timezone and DST offset on the day represented by the value. LocalEndTime SHALL be greater than LocalStartTime. This SHALL be null if the schedule is not set for the YearDayIndex and UserIndex provided.

5.2.10.21. ClearYearDaySchedule Command

Clears the specific year day schedule or all year day schedules for the specific user.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**YearDayIndex** **Schedule ID†**	uint8	1 to NumberOfYearDaySchedulesSupportedPerUser, 0xFE			M
1	**UserIndex** **User ID†**	uint16	1 to NumberOfTotalUsersSupported			M

† The Schedule ID and User ID are obsolete field names, use YearDayIndex and UserIndex instead, respectively.

Return status SHALL be one of the following values:

Name	Summary
SUCCESS	Clearing the requested YearDaySchedule was successful.
FAILURE	Some unexpected internal error occurred clearing Year Day schedule.
INVALID_COMMAND	One or more fields violates constraints or is invalid.

5.2.10.21.1. YearDayIndex Field

This field SHALL indicate the Year Day schedule index to clear or 0xFE to clear all Year Day schedules for the specified user.

5.2.10.21.2. UserIndex Field

This field SHALL indicate the user ID.

5.2.10.22. SetHolidaySchedule Command

Set the holiday Schedule by specifying local start time and local end time with respect to any Lock Operating Mode.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**HolidayIndex** **Holiday Schedule ID†**	uint8	1 to NumberOfHolidayScheduleSupported			M
1	**LocalStartTime**	epoch-s	all			M
2	**LocalEndTime**	epoch-s	all			M
3	**OperatingMode**	OperatingModeEnum	all			M

† The Holiday Schedule ID is an obsolete field name, use HolidayIndex instead.

Return status SHALL be one of the following values:

Name	Summary
SUCCESS	Setting Holiday schedule was successful.
FAILURE	Some unexpected internal error occurred setting Holiday schedule.

Name	Summary
INVALID_COMMAND	One or more fields violates constraints or is invalid.

5.2.10.22.1. HolidayIndex Field

This field SHALL indicate the index of the Holiday schedule.

5.2.10.22.2. LocalStartTime Field

This field SHALL indicate the starting time for the Holiday Day schedule in Epoch Time in Seconds with local time offset based on the local timezone and DST offset on the day represented by the value.

5.2.10.22.3. LocalEndTime Field

This field SHALL indicate the ending time for the Holiday Day schedule in Epoch Time in Seconds with local time offset based on the local timezone and DST offset on the day represented by the value. LocalEndTime SHALL be greater than LocalStartTime.

5.2.10.22.4. OperatingMode Field

This field SHALL indicate the operating mode to use during this Holiday schedule start/end time.

5.2.10.23. GetHolidaySchedule Command

Get the holiday schedule for the specified index.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**HolidayIndex** **Holiday Schedule ID†**	uint8	1 to NumberOfHolidaySchedulesSupported			M

† The Holiday Schedule ID is an obsolete field name, use HolidayIndex instead.

5.2.10.24. GetHolidayScheduleResponse Command

Returns the Holiday Schedule Entry for the specified Holiday ID.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**HolidayIndex** **Holiday Schedule ID†**	uint8	1 to NumberOfHolidaySchedulesSupported			M
1	**Status**	enum8	desc		SUCCESS	M
2	**LocalStartTime**	epoch-s	all	X		O
3	**Local End Time**	epoch-s	all	X		O
4	**OperatingMode**	OperatingModeEnum	all	X		O

† The Holiday Schedule ID is an obsolete field name, use HolidayIndex instead.

5.2.10.24.1. HolidayIndex Field

This field SHALL indicate the index of the Holiday schedule.

5.2.10.24.2. Status Field

Status SHALL be one of the following values:

- FAILURE if the attribute NumberOfHolidaySchedulesSupported is zero.
- SUCCESS if the HolidayIndex is valid and there is a corresponding schedule entry.
- INVALID_COMMAND if the HolidayIndex is not within valid range
- NOT_FOUND if the HolidayIndex is within the valid range, however, there is not corresponding schedule entry found.

If this field is SUCCESS, the optional fields for this command SHALL be present. For other (error) status values, only the fields up to the status field SHALL be present.

5.2.10.24.3. LocalStartTime Field

This field SHALL indicate the starting time for the Holiday schedule in Epoch Time in Seconds with local time offset based on the local timezone and DST offset on the day represented by the value. This SHALL be null if the schedule is not set for the HolidayIndex provided.

5.2.10.24.4. LocalEndTime Field

This field SHALL indicate the ending time for the Holiday schedule in Epoch Time in Seconds with local time offset based on the local timezone and DST offset on the day represented by the value. LocalEndTime SHALL be greater than LocalStartTime. This SHALL be null if the schedule is not set for the HolidayIndex provided.

5.2.10.24.5. OperatingMode Field

This field SHALL indicate the operating mode to use during this Holiday schedule start/end time. This SHALL be null if the schedule is not set for the HolidayIndex provided.

5.2.10.25. ClearHolidaySchedule Command

Clears the holiday schedule or all holiday schedules.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**HolidayIndex** **Holiday Schedule ID†**	uint8	1 to NumberOfHolidaySchedulesSupported, 0xFE			M

† The Holiday Schedule ID is an obsolete field name, use HolidayIndex instead.

5.2.10.25.1. HolidayIndex Field

This field SHALL indicate the Holiday schedule index to clear or 0xFE to clear all Holiday schedules.

5.2.10.26. SetUserType Command

Set the user type for a specified user.

For user type value please refer to User Type Value.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**UserID**	uint16	desc			M
1	**UserType**	UserTypeEnum	all			M

Return status SHALL be one of the following values:

Name	Summary
SUCCESS	Setting User Type was successful.
FAILURE	Some unexpected internal error occurred setting User Type.

Name	Summary
INVALID_COMMAND	One or more fields violates constraints or is invalid. `Door lock is unable to switch from restricted to unrestricted user (e.g. need to clear schedules to switch).`

5.2.10.26.1. UserID Field

This field SHALL indicate the user ID.

5.2.10.26.2. UserType Field

This field SHALL indicate the user type.

If UserType is currently YearDayScheduleUser, WeekDayScheduleUser, or ScheduleRestrictedUser and the new UserType is UnrestrictedUser then all existing Year Day and/or Week Day schedules SHALL be ignored or disabled (if this transition is supported by the door lock). If UserType is ScheduleRestrictedUser and the new UserType is ScheduleRestrictedUser then all existing Year Day and/or Week Day schedules SHALL be applied or enabled.

5.2.10.27. GetUserType Command

Retrieve the user type for a specific user.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**UserID**	uint16	desc			M

5.2.10.28. GetUserTypeResponse Command

Returns the user type for the specified user ID. If the requested User ID is invalid, send Default Response with an error status equal to FAILURE.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**UserID**	uint16	desc			M
1	**UserType**	UserType-Enum	all			M

5.2.10.29. SetRFIDCode Command

Set an ID for RFID access into the lock.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**UserID**	uint16	desc			M
1	**UserStatus**	UserStatusEnum	desc	X	OccupiedEnabled	M
2	**UserType**	UserTypeEnum	desc	X	UnrestrictedUser	M
3	**RFIDCode**	octstr				M

Return status is a global status code or a cluster-specific status code from the Status Codes table and SHALL be one of the following values:

Name	Summary
SUCCESS	Setting RFID code was successful.
FAILURE	Setting RFID code failed.
CONSTRAINT_ERROR	Setting RFID code failed because User ID requested was out of range.
DUPLICATE	Setting RFID code failed because it would create a duplicate RFID code.

5.2.10.29.1. UserID Field

This field SHALL indicate the user ID.

The value of the UserID field SHALL be between 0 and the value of the NumberOfRFIDUsersSupported attribute.

5.2.10.29.2. UserStatus Field

This field SHALL indicate what the status is for a specific user ID. The values are according to “Set PIN” while not all are supported.

Only the values 1 (Occupied/Enabled) and 3 (Occupied/Disabled) are allowed for UserStatus.

5.2.10.29.3. UserType Field

The values are the same as used for SetPINCode command.

5.2.10.30. GetRFIDCode Command

Retrieve an RFID code.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**UserID**	uint16	desc			M

5.2.10.30.1. UserID Field

This field SHALL indicate the user ID.

The value of the UserID field SHALL be between 0 and the value of the NumberOfRFIDUsersSupported attribute.

5.2.10.31. GetRFIDCodeResponse Command

Returns the RFID code for the specified user ID.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**UserID**	uint16	desc			M
1	**UserStatus**	UserStatusEnum	desc	X	Available	M
2	**UserType**	UserTypeEnum	desc	X		M
3	**RFIDCode**	octstr		X	empty	M

If the requested User ID is valid and the Code doesn't exist, Get RFID Code Response SHALL have the following format:

User ID = requested User ID

UserStatus = 0 (available)

UserType = 0xFF (not supported)

RFID Code = 0 (zero length)

If requested User ID is invalid, send Default Response with an error status. The error status SHALL be equal to CONSTRAINT_ERROR when User_ID is less than the max number of users supported, and NOT_FOUND if greater than or equal to the max number of users supported.

5.2.10.32. ClearRFIDCode Command

Clear an RFID code or all RFID codes.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**RFIDSlotIndex** **User ID†**	uint16	1 to NumberOfRFIDUsersSupported, 0xFFFE			M

† The User ID is an obsolete field name, use RFIDSlotIndex instead.

For each RFID Code cleared whose user doesn't have a PIN Code or other credential type, then the corresponding user record's UserStatus value SHALL be set to Available, and UserType value SHALL be set to UnrestrictedUser and all schedules SHALL be cleared.

5.2.10.32.1. RFIDSlotIndex Field

This field SHALL indicate a valid RFID code slot index or 0xFFFE to indicate all RFID code slots SHALL be cleared.

5.2.10.33. ClearAllRFIDCodes Command

Clear out all RFIDs on the lock. If you clear all RFID codes and this user didn't have a PIN code, the user status has to be set to "0 Available", the user type has to be set to the default value, and all schedules which are supported have to be set to the default values.

5.2.10.34. SetUser Command

Set user into the lock.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**OperationType**	DataOperationTypeEnum	Add, Modify			M
1	**UserIndex**	uint16	1 to NumberOfTotalUsersSupported			M
2	**UserName**	string	max 10	X	empty	M
3	**UserUniqueID**	uint32	all	X	0xFFFFFFFF	M
4	**UserStatus**	UserStatusEnum	OccupiedEnabled, OccupiedDisabled	X	OccupiedEnabled	M
5	**UserType**	UserTypeEnum	UnrestrictedUser, NonAccessUser, ForcedUser, DisposableUser, ExpiringUser, ScheduleRestrictedUser, RemoteOnlyUser	X	UnrestrictedUser	M

ID	Name	Type	Constraint	Quality	Default	Conformance
6	**CredentialRule**	CredentialRuleEnum		X	Single	M

Fields used for different use cases:

Use Case	Description
Create a new user record	• OperationType SHALL be set to Add. • UserIndex value SHALL be set to a user record with UserType set to Available. • UserName MAY be null causing new user record to use empty string for UserName otherwise UserName SHALL be set to the value provided in the new user record. • UserUniqueID MAY be null causing new user record to use 0xFFFFFFFF for UserUniqueID otherwise UserUniqueID SHALL be set to the value provided in the new user record. • UserStatus MAY be null causing new user record to use OccupiedEnabled for UserStatus otherwise UserStatus SHALL be set to the value provided in the new user record. • UserType MAY be null causing new user record to use UnrestrictedUser for UserType otherwise UserType SHALL be set to the value provided in the new user record. • CredentialRule MAY be null causing new user record to use Single for CredentialRule otherwise CredentialRule SHALL be set to the value provided in the new user record. CreatorFabricIndex and LastModifiedFabricIndex in the new user record SHALL be set to the accessing fabric index. A LockUserChange event SHALL be generated after successfully creating a new user.

Use Case	Description
Modify an existing user record	• OperationType SHALL be set to Modify. • UserIndex value SHALL be set for a user record with UserType NOT set to Available. • UserName SHALL be null if modifying a user record that was not created by the accessing fabric. • INVALID_COMMAND SHALL be returned if UserName is not null and the accessing fabric index doesn't match the CreatorFabricIndex in the user record otherwise UserName SHALL be set to the value provided in the user record. • UserUniqueID SHALL be null if modifying the user record that was not created by the accessing fabric. • INVALID_COMMAND SHALL be returned if UserUniqueID is not null and the accessing fabric index doesn't match the CreatorFabricIndex in the user record otherwise UserUniqueID SHALL be set to the value provided in the user record. • UserStatus MAY be null causing no change to UserStatus in user record otherwise UserStatus SHALL be set to the value provided in the user record. • UserType MAY be null causing no change to UserType in user record otherwise UserType SHALL be set to the value provided in the user record. • CredentialRule MAY be null causing no change to CredentialRule in user record otherwise CredentialRule SHALL be set to the value provided in the user record. CreatorFabricIndex SHALL NOT be changed in the user record. LastModifiedFabricIndex in the new user record SHALL be set to the accessing fabric index. A LockUserChange event SHALL be generated after successfully modifying a user.

Return status is a global status code or a cluster-specific status code from the Status Codes table and

SHALL be one of the following values:

- SUCCESS, if setting User was successful.
- FAILURE, if some unexpected internal error occurred setting User.
- OCCUPIED, if OperationType is Add and UserIndex points to an occupied slot.
- INVALID_COMMAND, if one or more fields violate constraints or are invalid or if OperationType is Modify and UserIndex points to an available slot.

5.2.10.34.1. OperationType Field

This field SHALL indicate the type of operation.

5.2.10.34.2. UserIndex Field

This field SHALL indicate the user ID.

5.2.10.34.3. UserName Field

This field SHALL contain a string to use as a human readable identifier for the user.

If UserName is null then:

- If the OperationType is Add, the UserName in the resulting user record SHALL be set to an empty string.
- If the OperationType is Modify, the UserName in the user record SHALL NOT be changed from the current value.

If UserName is not null, the UserName in the user record SHALL be set to the provided value.

5.2.10.34.4. UserUniqueID Field

This field SHALL indicate the fabric assigned number to use for connecting this user to other users on other devices from the fabric's perspective.

If UserUniqueID is null then:

- If the OperationType is Add, the UserUniqueID in the resulting user record SHALL be set to default value specified above.
- If the OperationType is Modify, the UserUniqueID in the user record SHALL NOT be changed from the current value.

If UserUniqueID is not null, the UserUniqueID in the user record SHALL be set to the provided value.

5.2.10.34.5. UserStatus Field

This field SHALL indicate the UserStatus to assign to this user when created or modified.

If UserStatus is null then:

- If the OperationType is Add, the UserStatus in the resulting user record SHALL be set to default value specified above.
- If the OperationType is Modify, the UserStatus in the user record SHALL NOT be changed from the current value.

If UserStatus is not null, the UserStatus in the user record SHALL be set to the provided value.

5.2.10.34.6. UserType Field

This field SHALL indicate the UserType to assign to this user when created or modified.

If UserType is null then:

- If the OperationType is Add, the UserType in the resulting user record SHALL be set to default value specified above.
- If the OperationType is Modify, the UserType in the user record SHALL NOT be changed from the current value.

If UserType is not null, the UserType in the user record SHALL be set to the provided value.

5.2.10.34.7. CredentialRule Field

This field SHALL indicate the CredentialRule to use for this user.

The valid CredentialRule enumeration values depends on the bits in the CredentialRulesBitmap map. Each bit in the map identifies a valid CredentialRule that can be used.

If CredentialRule is null then:

- If the OperationType is Add, the CredentialRule in the resulting user record SHALL be set to default value specified above.
- If the OperationType is Modify, the CredentialRule in the user record SHALL NOT be changed from the current value.

If CredentialRule is not null, the CredentialRule in the user record SHALL be set to the provided value.

5.2.10.35. GetUser Command

Retrieve user.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**UserIndex**	uint16	1 to NumberOfTotalUsersSupported			M

An InvokeResponse command SHALL be sent with an appropriate error (e.g. FAILURE, INVALID_-

COMMAND, etc.) as needed otherwise the GetUserResponse Command SHALL be sent implying a status of SUCCESS.

5.2.10.36. GetUserResponse Command

Returns the user for the specified UserIndex.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**UserIndex**	uint16	1 to NumberOfTotalUsersSupported			M
1	**UserName**	string	max 10	X	empty	M
2	**UserUniqueID**	uint32	all	X	0	M
3	**UserStatus**	UserStatusEnum	all	X	Available	M
4	**UserType**	UserTypeEnum	all	X	UnrestrictedUser	M
5	**CredentialRule**	CredentialRuleEnum	desc	X	Single	M
6	**Credentials**	list[CredentialStruct]	0 to NumberOfCredentialsSupportedPerUser	X		M
7	**CreatorFabricIndex**	fabric-idx	all	X		M
8	**LastModifiedFabricIndex**	fabric-idx	all	X		M
9	**NextUserIndex**	uint16	1 to NumberOfTotalUsersSupported	X		M

If the requested UserIndex is valid and the UserStatus is Available for the requested UserIndex then UserName, UserUniqueID, UserStatus, UserType, CredentialRule, Credentials, CreatorFabricIndex, and LastModifiedFabricIndex SHALL all be null in the response.

5.2.10.36.1. UserIndex Field

This field SHALL indicate the user ID.

5.2.10.36.2. UserName Field

This field SHALL contain a string to use as a human readable identifier for the user.

5.2.10.36.3. UserUniqueID Field

See UserUniqueID field.

5.2.10.36.4. UserStatus Field

This field SHALL indicate the UserStatus assigned to the user when created or modified.

5.2.10.36.5. UserType Field

This field SHALL indicate the UserType assigned to this user when created or modified.

5.2.10.36.6. CredentialRule Field

This field SHALL indicate the CredentialRule set for this user.

5.2.10.36.7. Credentials Field

This field SHALL contain a list of credentials for this user.

5.2.10.36.8. CreatorFabricIndex Field

This field SHALL indicate the user's creator fabric index. CreatorFabricIndex SHALL be null if UserStatus is set to Available or when the creator fabric cannot be determined (for example, when user was created outside the Interaction Model) and SHALL NOT be null otherwise. This value SHALL be set to 0 if the original creator fabric was deleted.

5.2.10.36.9. LastModifiedFabricIndex Field

This field SHALL indicate the user's last modifier fabric index. LastModifiedFabricIndex SHALL be null if UserStatus is set to Available or when the modifier fabric cannot be determined (for example, when user was modified outside the Interaction Model) and SHALL NOT be null otherwise. This value SHALL be set to 0 if the last modifier fabric was deleted.

5.2.10.36.10. NextUserIndex Field

This field SHALL indicate the next occupied UserIndex in the database which is useful for quickly identifying occupied user slots in the database. This SHALL NOT be null if there is at least one occupied entry after the requested UserIndex in the User database and SHALL be null if there are no more occupied entries.

5.2.10.37. ClearUser Command

Clears a user or all Users.

ID	name	Type	Constraint	Quality	Default	Conformance
0	**UserIndex**	uint16	1 to NumberOfTotalUsersSupported, 0xFFFE			M

For each user to clear, all associated credentials (e.g. PIN, RFID, fingerprint, etc.) SHALL be cleared and the user entry values SHALL be reset to their default values (e.g. UserStatus SHALL be Available, UserType SHALL be UnrestrictedUser) and all associated schedules SHALL be cleared.

A LockUserChange event with the provided UserIndex SHALL be generated after successfully clearing users.

5.2.10.37.1. UserIndex Field

This field SHALL specify a valid User index or 0xFFFE to indicate all user slots SHALL be cleared.

5.2.10.38. OperationEventNotification Command

The door lock server sends out operation event notification when the event is triggered by the various event sources. The specific operation event will only be sent out if the associated bitmask is enabled in the various EventMasks attributes.

All events are optional.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**OperationEventSource**	EventSourceEnum	desc			M
1	**OperationEventCode**	OperationEventCodeEnum	desc			M
2	**UserID**	uint16	desc			M
3	**PIN**	octstr				M
4	**LocalTime**	epoch-s	all			M
5	**Data**	string				O

5.2.10.38.1. KeypadOperationEvent Notifications

KeypadOperationEvent notification feature is enabled by setting the associated bitmasks in the KeypadOperationEventMask attribute.

5.2.10.38.2. RemoteOperationEvent Notifications

RemoteOperationEvent notification feature is enabled by setting the associated bitmasks in the RemoteOperationEventMask attribute.

5.2.10.38.3. ManualOperationEvent Notifications

ManualOperationEvent notification feature is enabled by setting the associated bitmasks in the ManualOperationEventMask attribute.

5.2.10.38.4. RFIDOperationEvent Notifications

RFIDOperationEvent notification feature is enabled by setting the associated bitmasks in the RFIDOperationEventMask attribute.

5.2.10.38.5. OperationEventSources Field

This field SHALL indicate where the event was triggered from.

5.2.10.38.6. OperationEventCodes Field

The door lock optionally sends out notifications (if they are enabled) whenever there is a significant operational event on the lock. When combined with a source from the Event Source table above, the following operational event codes constitute an event on the door lock that can be both logged and sent to a bound device using the Operation Event Notification command.

Not all operation event codes are applicable to each of the event source. The following table marks each event code with “A” if the event code is applicable to the event source.

Name	**Applicable**			
	Keypad	**Remote**	**Manual**	**RFID**
UnknownOrMfgSpecific	A	A	A	A
Lock	A	A	A	A
Unlock	A	A	A	A
LockFailureInvalidPINorRFID	A	A		A
LockFailureInvalidSchedule	A	A		A
UnlockFailureInvalidPINorRFID	A	A		A
UnlockFailureInvalidSchedule	A	A		A
OneTouchLock			A	
KeyLock			A	
KeyUnlock			A	

Name	Applicable			
AutoLock			A	
ScheduleLock			A	
ScheduleUnlock			A	
Manual Lock (Key or Thumb-turn)			A	
Manual Unlock (Key or Thumb-turn)			A	
Non-access User Operation Event	A			

5.2.10.38.7. UserID Field

This field SHALL indicate the UserID who performed the event.

5.2.10.38.8. PIN Field

This field SHALL indicate the PIN that is associated with the UserID who performed the event.

5.2.10.38.9. LocalTime Field

This field SHALL indicate the time when the event was triggered in Epoch Time in Seconds with local time offset based on the local timezone and DST offset on the day represented by the value. If time is not supported, the field SHALL be populated with default not used value 0xFFFFFFFF.

5.2.10.38.10. Data Field

This field SHALL indicate that the operation event notification command contains a variable string, which can be used to pass data associated with a particular event. Generally this field will be left empty. However, manufacturer can choose to use this field to store/display manufacturer-specific information.

5.2.10.39. ProgrammingEventNotification Command

The door lock server sends out a programming event notification whenever a programming event takes place on the door lock.

As with operational events, all programming events can be turned on and off by flipping bits in the associated event mask.

The programming event notification command includes an optional string of data that can be used by the manufacturer to pass some manufacturer-specific information if that is required.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**ProgramEventSource**	EventSourceEnum	desc			M
1	**ProgramEventCode**	ProgrammingEventCodeEnum	desc			M
2	**UserID**	uint16	desc			M
3	**PIN**	octstr				M
4	**UserType**	UserTypeEnum	desc			M
5	**UserStatus**	UserStatusEnum	desc			M
6	**LocalTime**	epoch-s	all			M
7	**Data**	string				O

5.2.10.39.1. ProgramEventSource Field

This field SHALL indicate where the event was triggered from.

5.2.10.39.2. ProgrammingEventCode Field

This field SHALL indicate the code for the triggered event.

The door lock optionally sends out notifications (if they are enabled) whenever there is a significant programming event on the lock. When combined with a source from the Event Source table above, the following programming event codes constitute an event on the door lock that can be both logged and sent to a bound device using the ProgrammingEventNotification command.

Not all event codes are applicable to each of the event source. The following table marks each event code with “A” if the event code is applicable to the event source.

Name	Keypad	Remote	RFID
UnknownOrMfgSpecific	A	A	A
ProgrammingCodeChanged	A		
PINCodeAdded	A	A	
PINCodeCleared	A	A	
PINCodeChanged	A	A	
RFIDCodeAdded			A
RFIDCodeCleared			A

5.2.10.39.3. KeypadProgrammingEvent Notifications

KeypadProgrammingEvent notification feature is enabled by setting the associated bitmasks in the KeypadProgrammingEventMask attribute.

5.2.10.39.4. RemoteProgrammingEvent Notifications

RemoteProgrammingEvent notification feature is enabled by setting the associated bitmasks in the RemoteProgrammingEventMask attribute.

5.2.10.39.5. RFIDProgrammingEvent Notifications

RFIDProgrammingEvent notification feature is enabled by setting the associated bitmasks in the RFIDProgrammingEventMask attribute.

5.2.10.39.6. UserID Field

This field SHALL indicate the UserID who performed the event

5.2.10.39.7. PIN Field

This field SHALL indicate the PIN that is associated with the UserID who performed the event

5.2.10.39.8. UserType Field

This field SHALL indicate the UserType that is associated with the UserID who performed the event

5.2.10.39.9. UserStatus Field

This field SHALL indicate the UserStatus that is associated with the UserID who performed the event

5.2.10.39.10. LocalTime Field

This field SHALL indicate the time when the event was triggered in Epoch Time in Seconds with local time offset based on the local timezone and DST offset on the day represented by the value. If time is not supported, the field SHALL be populated with default not used value 0xFFFFFFFF.

5.2.10.39.11. Data Field

This field MAY contain a variable string, which can be used to pass data associated with a particular event. Generally this field will be left empty. However, manufacturer can choose to use this field to store/display manufacturer-specific information.

5.2.10.40. SetCredential Command

Set a credential (e.g. PIN, RFID, Fingerprint, etc.) into the lock for a new user, existing user, or ProgrammingUser.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**OperationType**	DataOperationTypeEnum	Add, Modify			M
1	**Credential**	CredentialStruct				M
2	**CredentialData**	octstr	desc			M
3	**UserIndex**	uint16	1 to NumberOfTotalUsersSupported	X		M
4	**UserStatus**	UserStatusEnum	OccupiedEnabled, OccupiedDisabled	X	OccupiedEnabled	M
5	**UserType**	UserTypeEnum	UnrestrictedUser, ProgrammingUser, NonAccessUser, ForcedUser, DisposableUser, ExpiringUser, RemoteOnlyUser	X	UnrestrictedUser	M

Fields used for different use cases:

Use Case	Description
Create a new credential and a new user record	• OperationType SHALL be set to Add. • UserIndex SHALL be set to null and the lock will find a user record with a UserStatus value of Available and associate its UserIndex with the CredentialIndex in CredentialStruct provided. • CredentialIndex in CredentialStruct SHALL be for an unoccupied credential slot. • UserStatus MAY be null. If it is null, the new user record SHALL have UserStatus set to OccupiedEnabled. Otherwise the new user record SHALL have UserStatus set to the provided value. • UserType MAY be null. If it is null, the new user record SHALL have UserType set to UnrestrictedUser. Otherwise the new user record SHALL have UserType set to the provided value. • UserType SHALL NOT be set to ProgrammingUser for this use case. CreatorFabricIndex and LastModifiedFabricIndex in new user and credential records SHALL be set to the accessing fabric index. A LockUserChange event SHALL be generated after successfully creating a new credential and a new user. The UserIndex of this LockUserChange event SHALL be the UserIndex that was used to create the user. The DataIndex of this LockUserChange event SHALL be the CredentialIndex that was used to create the credential.

Use Case	Description
Add a new credential to existing user record	• OperationType SHALL be set to Add. • UserIndex SHALL NOT be null and SHALL NOT already be associated with the CredentialIndex in CredentialStruct provided otherwise INVALID_COMMAND status response SHALL be returned. • INVALID_COMMAND SHALL be returned if the accessing fabric index doesn't match the CreatorFabricIndex in the user record pointed to by UserIndex. • CredentialIndex in CredentialStruct provided SHALL be for an available credential slot. • UserStatus SHALL be null. • UserType SHALL be null. CreatorFabricIndex SHALL NOT be changed in the user record. LastModifiedFabricIndex in the user record SHALL be set to the accessing fabric index. CreatorFabricIndex and LastModifiedFabricIndex in the new credential record SHALL be set to the accessing fabric index. A LockUserChange event SHALL be generated after successfully adding a new credential.

Use Case	Description
Modify credential for an existing user record	• OperationType SHALL be set to Modify. • UserIndex value SHALL already be associated with the CredentialIndex in CredentialStruct provided otherwise INVALID_COMMAND status response SHALL be returned. • INVALID_COMMAND SHALL be returned if the accessing fabric index doesn't match the CreatorFabricIndex in the user record pointed to by UserIndex. • INVALID_COMMAND SHALL be returned if the accessing fabric index doesn't match the CreatorFabricIndex in the credential record pointed to by the CredentialIndex field value of the Credential parameter. • CredentialIndex in CredentialStruct provided SHALL be for an occupied credential slot • UserStatus SHALL be null. • UserType SHALL be null. CreatorFabricIndex SHALL NOT be changed in user and credential records. LastModifiedFabricIndex in user and credential records SHALL be set to the accessing fabric index. A LockUserChange event SHALL be generated after successfully modifying a credential.

Use Case	Description
Modify credential for a Programming User	• OperationType SHALL be set to Modify. • UserIndex SHALL be null. • INVALID_COMMAND SHALL be returned if the accessing fabric index doesn't match the CreatorFabricIndex in the credential record pointed to by the CredentialIndex field value of the Credential parameter. • CredentialType in CredentialStruct SHALL be set to ProgrammingPIN. • CredentialIndex in CredentialStruct SHALL be 0. • UserStatus SHALL be null. • UserType SHALL be set to ProgrammingUser. CreatorFabricIndex SHALL NOT be changed in the credential record. LastModifiedFabricIndex in the credential record SHALL be set to the accessing fabric index. A LockUserChange event SHALL be generated after successfully modifying a ProgrammingUser PIN code.

5.2.10.40.1. OperationType Field

This field SHALL indicate the set credential operation type requested.

5.2.10.40.2. Credential Field

This field SHALL contain a credential structure that contains the CredentialTypeEnum and the credential index (if applicable or 0 if not) to set.

5.2.10.40.3. CredentialData Field

This field SHALL indicate the credential data to set for the credential being added or modified. The length of the credential data SHALL conform to the limits of the CredentialType specified in the Credential structure otherwise an INVALID_COMMAND status SHALL be returned in the SetCredentialResponse command.

5.2.10.40.4. UserIndex Field

This field SHALL indicate the user index to the user record that corresponds to the credential being added or modified. This SHALL be null if OperationType is add and a new credential and user is being added at the same time.

5.2.10.40.5. UserStatus Field

This field SHALL indicate the user status to use in the new user record if a new user is being created. This SHALL be null if OperationType is Modify. This MAY be null when adding a new credential and user.

5.2.10.40.6. UserType Field

This field SHALL indicate the user type to use in the new user record if a new user is being created. This SHALL be null if OperationType is Modify. This MAY be null when adding a new credential and user.

5.2.10.41. SetCredentialResponse Command

Returns the status for setting the specified credential.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**Status**	status	desc			M
1	**UserIndex**	uint16	1 to NumberOfTotalUsersSupported	X	0	M
2	**NextCredentialIndex**	uint16	desc	X		O

5.2.10.41.1. Status Field

Status comes from the Status Codes table and SHALL be one of the following values:

- SUCCESS, if setting user credential was successful.
- FAILURE, if some unexpected internal error occurred setting user credential.
- OCCUPIED, if OperationType is Add and CredentialIndex in Credential structure points to an occupied slot.
- OCCUPIED, if OperationType is Modify and CredentialIndex in Credential structure does not match the CredentialIndex that is already associated with the provided UserIndex.
- DUPLICATE, if CredentialData provided is a duplicate of another credential with the same CredentialType (e.g. duplicate PIN code).
- RESOURCE_EXHAUSTED, if OperationType is Add and the user referred to by UserIndex already has NumberOfCredentialsSupportedPerUser credentials associated.
- INVALID_COMMAND, if one or more fields violate constraints or are invalid.
- INVALID_COMMAND, if the CredentialIndex in the Credential provided exceeds the number of credentials of the provided CredentialType supported by the lock.
- INVALID_COMMAND, if OperationType is Modify and UserIndex points to an available slot.

5.2.10.41.2. UserIndex Field

This field SHALL indicate the user index that was created with the new credential. If the status being returned is not success then this SHALL be null. This SHALL be null if OperationType was Modify; if the OperationType was Add and a new User was created this SHALL NOT be null and SHALL provide the UserIndex created. If the OperationType was Add and an existing User was associated with the new credential then this SHALL be null.

5.2.10.41.3. NextCredentialIndex Field

This field SHALL indicate the next available index in the database for the credential type set, which is useful for quickly identifying available credential slots in the database. This SHALL NOT be null if there is at least one available entry after the requested credential index in the corresponding database and SHALL be null if there are no more available entries. The NextCredentialIndex reported SHALL NOT exceed the maximum number of credentials for a particular credential type.

5.2.10.42. GetCredentialStatus Command

Retrieve the status of a particular credential (e.g. PIN, RFID, Fingerprint, etc.) by index.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**Credential**	CredentialStruct				M

An InvokeResponse command SHALL be sent with an appropriate error (e.g. FAILURE, INVALID_COMMAND, etc.) as needed otherwise the GetCredentialStatusResponse command SHALL be sent implying a status of SUCCESS.

5.2.10.42.1. Credential Field

This field SHALL contain a credential structure that contains the CredentialTypeEnum and the credential index (if applicable or 0 if not) to retrieve the status for.

5.2.10.43. GetCredentialStatusResponse Command

Returns the status for the specified credential.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**CredentialExists**	bool	all			M
1	**UserIndex**	uint16	1 to NumberOfTotalUsersSupported	X		M
2	**CreatorFabricIndex**	fabric-idx	all	X		M

ID	Name	Type	Constraint	Quality	Default	Conformance
3	**LastModifiedFabricIndex**	fabric-idx	all	X		M
4	**NextCredentialIndex**	uint16	desc	X		O

5.2.10.43.1. CredentialExists Field

This field SHALL indicate if the requested credential type and index exists and is populated for the requested user index.

5.2.10.43.2. UserIndex Field

This field SHALL indicate the credential's corresponding user index value if the credential exists. If CredentialType requested was ProgrammingPIN then UserIndex SHALL be null; otherwise, UserIndex SHALL be null if CredentialExists is set to False and SHALL NOT be null if CredentialExists is set to True.

5.2.10.43.3. CreatorFabricIndex Field

This field SHALL indicate the credential's creator fabric index. CreatorFabricIndex SHALL be null if CredentialExists is set to False or when the creator fabric cannot be determined (for example, when credential was created outside the Interaction Model) and SHALL NOT be null otherwise. This value SHALL be set to 0 if the original creator fabric was deleted.

5.2.10.43.4. LastModifiedFabricIndex Field

This field SHALL indicate the credential's last modifier fabric index. LastModifiedFabricIndex SHALL be null if CredentialExists is set to False or when the modifier fabric cannot be determined (for example, when credential was modified outside the Interaction Model) and SHALL NOT be null otherwise. This value SHALL be set to 0 if the last modifier fabric was deleted.

5.2.10.43.5. NextCredentialIndex Field

This field SHALL indicate the next occupied index in the database for the credential type requested, which is useful for quickly identifying occupied credential slots in the database. This SHALL NOT be null if there is at least one occupied entry after the requested credential index in the corresponding database and SHALL be null if there are no more occupied entries. The NextCredentialIndex reported SHALL NOT exceed the maximum number of credentials for a particular credential type.

5.2.10.44. ClearCredential Command

Clear one, one type, or all credentials except ProgrammingPIN credential.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**Credential**	CredentialStruct	desc	X		M

Fields used for different use cases:

Use Case	Description
Clear a single credential	• CredentialType in Credential structure SHALL be set to the credential type to be cleared. • CredentialType in Credential structure SHALL NOT be set to ProgrammingPIN. • CredentialIndex in Credential structure SHALL be set to the credential index to be cleared. A LockUserChange event SHALL be generated after successfully clearing a credential.
Clear all credentials of one type	• CredentialType in Credential structure SHALL be set to the credential type to be cleared. • CredentialType in Credential structure SHALL NOT be set to ProgrammingPIN. • CredentialIndex in Credential structure SHALL be set to 0xFFFE to indicate all credentials of that type SHALL be cleared. A single LockUserChange event SHALL be generated after successfully clearing credentials. This event SHALL have DataIndex set to the CredentialIndex in the Credential structure.
Clear all credentials of all types	• Credential field SHALL be null. The ProgrammingPIN credential SHALL NOT be cleared. For each credential type cleared, a LockUserChange event with the corresponding LockDataType SHALL be generated. This event SHALL have DataIndex set to 0xFFFE.

For each credential cleared whose user doesn't have another valid credential, the corresponding user record SHALL be reset back to default values and its UserStatus value SHALL be set to Available and UserType value SHALL be set to UnrestrictedUser and all schedules SHALL be cleared. In

this case a LockUserChange event SHALL be generated for the user being cleared.

Return status SHALL be one of the following values:

Name	Summary
SUCCESS	Clearing the requested credential was successful.
FAILURE	Some unexpected internal error occurred clearing the requested credential.
INVALID_COMMAND	One or more fields violate constraints or are invalid.

5.2.10.44.1. Credential Field

This field SHALL contain a credential structure that contains the CredentialTypeEnum and the credential index (0xFFFE for all credentials or 0 if not applicable) to clear. This SHALL be null if clearing all credential types otherwise it SHALL NOT be null.

5.2.10.45. UnboltDoor Command

This command causes the lock device to unlock the door without pulling the latch. This command includes an optional code for the lock. The door lock MAY require a code depending on the value of the RequirePINForRemoteOperation attribute.

NOTE	If the attribute AutoRelockTime is supported, the lock will transition to the locked state when the auto relock time has expired.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**PINCode**	octstr				[COTA & PIN]

5.2.10.45.1. PINCode Field

See PINCode field.

5.2.11. Events

This cluster SHALL support these events:

ID	Name	Priority	Access	Conformance
0x00	DoorLockAlarm	CRITICAL	V	M
0x01	DoorStateChange	desc	V	DPS
0x02	LockOperation	desc	V	M
0x03	LockOperationError	desc	V	M

ID	Name	Priority	Access	Conformance
0x04	LockUserChange	INFO	V	USR

The Events specified in this cluster are not intended to define the user experience. The events are only intended to define the metadata format used to notify any nodes that have subscribed for updates.

If the DoorState reported in the DoorStateChange event is not DoorClosed then the priority SHALL be CRITICAL; otherwise it MAY be INFO.

If the LockOperationType reported in the LockOperation event is Unlock or ForcedUserEvent then the priority SHALL be CRITICAL; otherwise it MAY be INFO.

If the OperationError reported in the LockOperationError event is DisabledUserDenied or the LockOperationType is Lock the priority SHALL be CRITICAL; otherwise it MAY be INFO.

5.2.11.1. DoorLockAlarm Event

The door lock server provides several alarms which can be sent when there is a critical state on the door lock. The alarms available for the door lock server are listed in AlarmCodeEnum.

The data of this event SHALL contain the following information:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**AlarmCode**	AlarmCodeEnum	all			M

5.2.11.1.1. AlarmCode Field

This field SHALL indicate the alarm code of the event that has happened.

5.2.11.2. DoorStateChange Event

The door lock server sends out a DoorStateChange event when the door lock door state changes.

The data of this event SHALL contain the following information:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**DoorState**	DoorStateEnum	all			M

5.2.11.2.1. DoorState Field

This field SHALL indicate the new door state for this door event.

5.2.11.3. LockOperation Event

The door lock server sends out a LockOperation event when the event is triggered by the various lock operation sources.

The data of this event SHALL contain the following information:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**LockOperationType**	LockOperationTypeEnum	all			M
1	**OperationSource**	OperationSourceEnum	all			M
2	**UserIndex**	uint16	all	X		M
3	**FabricIndex**	fabric-idx	all	X		M
4	**SourceNode**	node-id	all	X		M
5	**Credentials**	list[CredentialStruct]	1 to NumberOfCredentialsSupportedPerUser	X		[USR]

- If the door lock server supports the Unbolt Door command, it SHALL generate a LockOperation event with LockOperationType set to Unlock after an Unbolt Door command succeeds.
- If the door lock server supports the Unbolting feature and an Unlock Door command is performed, it SHALL generate a LockOperation event with LockOperationType set to Unlatch when the unlatched state is reached and a LockOperation event with LockOperationType set to Unlock when the lock successfully completes the unlock → hold latch → release latch and return to unlock state operation.
- If the command fails during holding or releasing the latch but after passing the unlocked state, the door lock server SHALL generate a LockOperationError event with LockOperationType set to Unlatch and a LockOperation event with LockOperationType set to Unlock.
  - If it fails before reaching the unlocked state, the door lock server SHALL generate only a LockOperationError event with LockOperationType set to Unlock.
- Upon manual actuation, a door lock server that supports the Unbolting feature:
  - SHALL generate a LockOperation event of LockOperationType Unlatch when it is actuated from the outside.
  - MAY generate a LockOperation event of LockOperationType Unlatch when it is actuated from the inside.

5.2.11.3.1. LockOperationType Field

This field SHALL indicate the type of the lock operation that was performed.

5.2.11.3.2. OperationSource Field

This field SHALL indicate the source of the lock operation that was performed.

5.2.11.3.3. UserIndex Field

This field SHALL indicate the UserIndex who performed the lock operation. This SHALL be null if there is no user index that can be determined for the given operation source. This SHALL NOT be null if a user index can be determined. In particular, this SHALL NOT be null if the operation was associated with a valid credential.

5.2.11.3.4. FabricIndex Field

This field SHALL indicate the fabric index of the fabric that performed the lock operation. This SHALL be null if there is no fabric that can be determined for the given operation source. This SHALL NOT be null if the operation source is "Remote".

5.2.11.3.5. SourceNode Field

This field SHALL indicate the Node ID of the node that performed the lock operation. This SHALL be null if there is no Node associated with the given operation source. This SHALL NOT be null if the operation source is "Remote".

5.2.11.3.6. Credentials Field

This field SHALL indicate the list of credentials used in performing the lock operation. This SHALL be null if no credentials were involved.

5.2.11.4. LockOperationError Event

The door lock server sends out a LockOperationError event when a lock operation fails for various reasons.

The data of this event SHALL contain the following information:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**LockOperationType**	LockOperationType-Enum	all			M
1	**OperationSource**	OperationSourceEnum	all			M
2	**OperationError**	OperationErrorEnum	all			M

ID	Name	Type	Constraint	Quality	Default	Conformance
3	**UserIndex**	uint16	all	X		M
4	**FabricIndex**	fabric-idx	all	X		M
5	**SourceNode**	node-id	all	X		M
6	**Credentials**	list[CredentialStruct]	1 to NumberOfCredentialsSupportedPerUser	X		[USR]

5.2.11.4.1. LockOperationType Field

This field SHALL indicate the type of the lock operation that was performed.

5.2.11.4.2. OperationSource Field

This field SHALL indicate the source of the lock operation that was performed.

5.2.11.4.3. OperationError Field

This field SHALL indicate the lock operation error triggered when the operation was performed.

5.2.11.4.4. UserIndex Field

This field SHALL indicate the lock UserIndex who performed the lock operation. This SHALL be null if there is no user id that can be determined for the given operation source.

5.2.11.4.5. FabricIndex Field

This field SHALL indicate the fabric index of the fabric that performed the lock operation. This SHALL be null if there is no fabric that can be determined for the given operation source. This SHALL NOT be null if the operation source is "Remote".

5.2.11.4.6. SourceNode Field

This field SHALL indicate the Node ID of the node that performed the lock operation. This SHALL be null if there is no Node associated with the given operation source. This SHALL NOT be null if the operation source is "Remote".

5.2.11.4.7. Credentials Field

This field SHALL indicate the list of credentials used in performing the lock operation. This SHALL be null if no credentials were involved.

5.2.11.5. LockUserChange Event

The door lock server sends out a LockUserChange event when a lock user, schedule, or credential change has occurred.

The data of this event SHALL contain the following information:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**LockDataType**	LockDataTypeEnum	all			M
1	**DataOperationType**	DataOperationTypeEnum	all			M
2	**OperationSource**	OperationSourceEnum	Unspecified, Keypad, Remote			M
3	**UserIndex**	uint16	all	X		M
4	**FabricIndex**	fabric-idx	all	X		M
5	**SourceNode**	node-id	all	X		M
6	**DataIndex**	uint16	all	X		M

5.2.11.5.1. LockDataType Field

This field SHALL indicate the lock data type that was changed.

5.2.11.5.2. DataOperationType Field

This field SHALL indicate the data operation performed on the lock data type changed.

5.2.11.5.3. OperationSource Field

This field SHALL indicate the source of the user data change.

5.2.11.5.4. UserIndex Field

This field SHALL indicate the lock UserIndex associated with the change (if any). This SHALL be null if there is no specific user associated with the data operation. This SHALL be 0xFFFE if all users are affected (e.g. Clear Users).

5.2.11.5.5. FabricIndex Field

This field SHALL indicate the fabric index of the fabric that performed the change (if any). This SHALL be null if there is no fabric that can be determined to have caused the change. This SHALL NOT be null if the operation source is "Remote".

5.2.11.5.6. SourceNode Field

This field SHALL indicate the Node ID that performed the change (if any). The Node ID of the node that performed the change. This SHALL be null if there was no Node involved in the change. This SHALL NOT be null if the operation source is "Remote".

5.2.11.5.7. DataIndex Field

This field SHALL indicate the index of the specific item that was changed (e.g. schedule, PIN, RFID, etc.) in the list of items identified by LockDataType. This SHALL be null if the LockDataType does not correspond to a list that can be indexed into (e.g. ProgrammingUser). This SHALL be 0xFFFE if all indices are affected (e.g. ClearPINCode, ClearRFIDCode, ClearWeekDaySchedule, ClearYearDaySchedule, etc.).

5.3. Window Covering Cluster

The window covering cluster provides an interface for controlling and adjusting automatic window coverings such as drapery motors, automatic shades, curtains and blinds.

5.3.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Revision	Description
1	Mandatory global ClusterRevision attribute added; CCB 1994 1995 1996 1997 2086 2094 2095 2096 2097
2	CCB 2328
3	CCB 2477 2555 2845 3028
4	All Hubs changes with FeatureMap & OperationalStatus attribute
5	New data model format and notation. Created plus clarified PositionAware and AbsolutePosition features. General cleanup of functionality.

5.3.2. Classification

Hierarchy	Role	Scope	PICS Code
Base	Application	Endpoint	WNCV

5.3.3. Cluster ID

ID	Name
0x0102	Window Covering

5.3.4. Features

This cluster SHALL support the FeatureMap bitmap attribute as defined below.

Bit	Code	Feature	Conformance	Summary
0	**LF**	Lift	O.a+	Lift control and behavior for lifting/sliding window coverings
1	**TL**	Tilt	O.a+	Tilt control and behavior for tilting window coverings
2	**PA_LF**	PositionAwareLift	[LF]	Position aware lift control is supported.
3	**ABS**	AbsolutePosition	O	Absolute positioning is supported.
4	**PA_TL**	PositionAwareTilt	[TL]	Position aware tilt control is supported.

Due to backward compatibility reasons this feature map SHALL match the advertised Type Attribute Supported Features.

5.3.4.1. Lift Feature

The Lift feature applies to window coverings that lift up and down (e.g. for a roller shade, Up and Down is lift Open and Close) or slide left to right (e.g. for a sliding curtain, Left and Right is lift Open and Close).

5.3.4.2. Tilt Feature

The Tilt feature applies to window coverings with vertical or horizontal strips.

5.3.4.3. PositionAware Features

Relative positioning with *percent100ths* (min step 0.01%) attribute is mandatory, e.g. Max 10000 equals 100.00% and relative positioning with *percent* (min step 1%) attribute is for backward compatibility.

The *CurrentPosition* attributes SHALL always reflects the physical position of an actuator and the *TargetPosition* attribute SHALL reflect the requested position of an actuator once a positioning command is received.

5.3.4.3.1. PositionAwareLift Feature

Relative positioning for lift attributes and commands.

5.3.4.3.2. PositionAwareTilt Feature

Relative positioning for tilt attributes and commands.

5.3.4.4. AbsolutePosition Feature

The percentage attributes SHALL indicate the position as a percentage between the InstalledOpenLimits and InstalledClosedLimits attributes of the window covering starting at the open (0.00%).

As a general rule, absolute positioning (in centimeters or tenth of a degrees) SHOULD NOT be supported for new implementations.

5.3.5. Data Types

5.3.5.1. ConfigStatusBitmap Type

This data type is derived from map8.

Bit	Name	Summary
0	**Operational**	Device is operational.
1	**OnlineReserved**	Deprecated and reserved.
2	**LiftMovementReversed**	The lift movement is reversed.
3	**LiftPositionAware**	Supports the PositionAwareLift feature (PA_LF).
4	**TiltPositionAware**	Supports the PositionAwareTilt feature (PA_TL).
5	**LiftEncoderControlled**	Uses an encoder for lift.
6	**TiltEncoderControlled**	Uses an encoder for tilt.

5.3.5.1.1. Operational Bit

This bit SHALL indicate whether the window covering is operational for regular use:

- 0 = Not Operational
- 1 = Operational

5.3.5.1.2. LiftMovementReversed Bit

This bit SHALL indicate whether the lift movement is reversed:

- 0 = Lift movement is normal
- 1 = Lift movement is reversed

5.3.5.1.3. LiftPositionAware Bit

This bit SHALL indicate whether the window covering supports the PositionAwareLift feature:

- 0 = Lift control is not position aware
- 1 = Lift control is position aware (PA_LF)

5.3.5.1.4. TiltPositionAware Bit

This bit SHALL indicate whether the window covering supports the PositionAwareTilt feature:

- 0 = Tilt control is not position aware
- 1 = Tilt control is position aware (PA_TL)

5.3.5.1.5. LiftEncoderControlled Bit

This bit SHALL indicate whether a position aware controlled window covering is employing an encoder for positioning the height of the window covering:

- 0 = Timer Controlled
- 1 = Encoder Controlled

5.3.5.1.6. TiltEncoderControlled Bit

This bit SHALL indicate whether a position aware controlled window covering is employing an encoder for tilting the window covering:

- 0 = Timer Controlled
- 1 = Encoder Controlled

5.3.5.2. ModeBitmap Type

This data type is derived from map8.

Bit	Name	Summary
0	**MotorDirectionReversed**	Reverse the lift direction.
1	**CalibrationMode**	Perform a calibration.
2	**MaintenanceMode**	Freeze all motions for maintenance.
3	**LedFeedback**	Control the LEDs feedback.

5.3.5.2.1. MotorDirectionReversed Bit

This bit SHALL control the motor direction:

- 0 = Lift movement is normal
- 1 = Lift movement is reversed

5.3.5.2.2. CalibrationMode Bit

This bit SHALL set the window covering into calibration mode:

- 0 = Normal mode
- 1 = Calibration mode

5.3.5.2.3. MaintenanceMode Bit

This bit SHALL set the window covering into maintenance mode:

- 0 = Normal mode
- 1 = Maintenance mode

5.3.5.2.4. LedFeedback Bit

This bit SHALL control feedback LEDs:

- 0 = LEDs are off
- 1 = LEDs will display feedback

5.3.5.3. OperationalStatusBitmap Type

This data type is derived from map8.

Bit	Name	Summary
0..1	**Global**	Global operational state.
2..3	**Lift**	Lift operational state.
4..5	**Tilt**	Tilt operational state.

The OperationalStatusBitmap is using several internal operational state fields (composed of 2 bits) following this definition:

- 00b = Currently not moving
- 01b = Currently opening (e.g. moving from closed to open).
- 10b = Currently closing (e.g. moving from open to closed).
- 11b = Reserved

5.3.5.3.1. Global Bits

These bits SHALL indicate in which direction the covering is currently moving or if it has stopped. Global operational state SHALL always reflect the overall motion of the device.

5.3.5.3.2. Lift Bits

These bits SHALL indicate in which direction the covering's lift is currently moving or if it has stopped.

5.3.5.3.3. Tilt Bits

These bits SHALL indicate in which direction the covering's tilt is currently moving or if it has stopped.

5.3.5.4. SafetyStatusBitmap

This data type is derived from map16.

Bit	Name	Summary
0	**RemoteLockout**	Movement commands are ignored (locked out). e.g. not granted authorization, outside some time/date range.
1	**TamperDetection**	Tampering detected on sensors or any other safety equipment. Ex: a device has been forcedly moved without its actuator(s).
2	**FailedCommunication**	Communication failure to sensors or other safety equipment.
3	**PositionFailure**	Device has failed to reach the desired position. e.g. with position aware device, time expired before TargetPosition is reached.
4	**ThermalProtection**	Motor(s) and/or electric circuit thermal protection activated.
5	**ObstacleDetected**	An obstacle is preventing actuator movement.
6	**Power**	Device has power related issue or limitation e.g. device is running w/ the help of a backup battery or power might not be fully available at the moment.
7	**StopInput**	Local safety sensor (not a direct obstacle) is preventing movements (e.g. Safety EU Standard EN60335).
8	**MotorJammed**	Mechanical problem related to the motor(s) detected.
9	**HardwareFailure**	PCB, fuse and other electrics problems.
10	**ManualOperation**	Actuator is manually operated and is preventing actuator movement (e.g. actuator is disengaged/decoupled).
11	**Protection**	Protection is activated.

5.3.5.5. TypeEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**RollerShade**	RollerShade	LF
1	**RollerShade2Motor**	RollerShade - 2 Motor	LF
2	**RollerShadeExterior**	RollerShade - Exterior	LF
3	**RollerShadeExterior2-Motor**	RollerShade - Exterior - 2 Motor	LF
4	**Drapery**	Drapery (curtain)	LF
5	**Awning**	Awning	LF
6	**Shutter**	Shutter	LF | TL
7	**TiltBlindTiltOnly**	Tilt Blind - Tilt Only	TL
8	**TiltBlindLiftAndTilt**	Tilt Blind - Lift & Tilt	LF & TL
9	**ProjectorScreen**	Projector Screen	LF
255	**Unknown**	Unknown	O

5.3.5.6. EndProductTypeEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**RollerShade**	Simple Roller Shade	LF
1	**RomanShade**	Roman Shade	LF
2	**BalloonShade**	Balloon Shade	LF
3	**WovenWood**	Woven Wood	LF
4	**PleatedShade**	Pleated Shade	LF
5	**CellularShade**	Cellular Shade	LF
6	**LayeredShade**	Layered Shade	LF
7	**LayeredShade2D**	Layered Shade 2D	LF
8	**SheerShade**	Sheer Shade	LF & TL
9	**TiltOnlyInteriorBlind**	Tilt Only Interior Blind	TL
10	**InteriorBlind**	Interior Blind	LF & TL
11	**VerticalBlindStripCur-tain**	Vertical Blind, Strip Curtain	LF & TL
12	**InteriorVenetianBlind**	Interior Venetian Blind	LF & TL

Value	Name	Summary	Conformance
13	**ExteriorVenetian-Blind**	Exterior Venetian Blind	LF & TL
14	**LateralLeftCurtain**	Lateral Left Curtain	LF
15	**LateralRightCurtain**	Lateral Right Curtain	LF
16	**CentralCurtain**	Central Curtain	LF
17	**RollerShutter**	Roller Shutter	LF
18	**ExteriorVerti-calScreen**	Exterior Vertical Screen	LF
19	**AwningTerracePatio**	Awning Terrace (Patio)	LF
20	**AwningVerticalScreen**	Awning Vertical Screen	LF
21	**TiltOnlyPergola**	Tilt Only Pergola	LF | TL
22	**SwingingShutter**	Swinging Shutter	LF | TL
23	**SlidingShutter**	Sliding Shutter	LF | TL
255	**Unknown**	Unknown	O

5.3.6. Attributes

ID	Name	Type	Constraint	Quality	Default	Access	Confor-mance
0x0000	**Type**	TypeEnum	desc	F	0	R V	M
0x0001	**Physical-Clos-edLim-itLift**	uint16	all	F	0	R V	[LF & PA_LF & ABS]
0x0002	**Physical-Clos-edLimit-Tilt**	uint16	all	F	0	R V	[TL & PA_TL & ABS]
0x0003	**Current-Position-Lift**[1]	uint16	Installe-dOpenLim-itLift to Installed-ClosedLim-itLift	XN	null	R V	[LF & PA_LF & ABS]

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0004	**CurrentPositionTilt**[1]	uint16	InstalledOpenLimitTilt to InstalledClosedLimitTilt	XN	null	R V	[TL & PA_TL & ABS]
0x0005	**NumberOfActuationsLift**	uint16	all	N	0	R V	[LF]
0x0006	**NumberOfActuationsTilt**	uint16	all	N	0	R V	[TL]
0x0007	**ConfigStatus**	ConfigStatusBitmap	desc	N	desc	R V	M
0x0008	**CurrentPositionLiftPercentage**[1]	percent		XNSP	null	R V	[LF & PA_LF]
0x0009	**CurrentPositionTiltPercentage**[1]	percent	0 to 100	XNSP	null	R V	[TL & PA_TL]
0x000A	**OperationalStatus**	OperationalStatusBitmap	0b00xxxxxx	P	0	R V	M
0x000B	**TargetPositionLiftPercent100ths**[2]	percent100ths		XSP	null	R V	LF & PA_LF
0x000C	**TargetPositionTiltPercent100ths**[2]	percent100ths		XSP	null	R V	TL & PA_TL
0x000D	**EndProductType**	EndProductTypeEnum	desc	F	0	R V	M

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x000E	**CurrentPositionLiftPercent100ths**[1]	percent100ths	0 to 10000	XNP	null	R V	LF & PA_LF
0x000F	**CurrentPositionTiltPercent100ths**[1]	percent100ths	0 to 10000	XNP	null	R V	TL & PA_TL
0x0010	**InstalledOpenLimitLift**	uint16	0 to 65534	N	0	R V	LF & PA_LF & ABS
0x0011	**InstalledClosedLimitLift**	uint16	0 to 65534	N	65534	R V	LF & PA_LF & ABS
0x0012	**InstalledOpenLimitTilt**	uint16	0 to 65534	N	0	R V	TL & PA_TL & ABS
0x0013	**InstalledClosedLimitTilt**	uint16	0 to 65534	N	65534	R V	TL & PA_TL & ABS
0x0014	**VelocityLift**						D
0x0015	**AccelerationTimeLift**						D
0x0016	**DecelerationTimeLift**						D
0x0017	**Mode**	ModeBitmap	0b0000 xxxx	N	0	RW VM	M
0x0018	**IntermediateSetpointsLift**						D
0x0019	**IntermediateSetpointsTilt**						D

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x001A	**SafetyStatus**	SafetyStatusBitmap	desc	P	0	R V	O

NOTE

Nullable positions

1 - The null value indicates that the current position is unknown, e.g. calibration is needed.

2 - The null value indicates that the value is unavailable, e.g. no target position has been set.

5.3.6.1. Scene Table Extensions

If the Scenes Management server cluster is implemented on the same endpoint, the following attributes SHALL be part of the ExtensionFieldSetStruct of the Scene Table. If the implicit form of the ExtensionFieldSetStruct is used, the order of the attributes in the AttributeValueList is in the given order, i.e., the attribute listed as 1 is added first:

1. CurrentPositionLiftPercentage
2. CurrentPositionTiltPercentage
3. TargetPositionLiftPercent100ths
4. TargetPositionTiltPercent100ths

When a Percentage attribute is part of a Scene Table, the attribute is treated as a writeable command, that is, activate a motion (lift or tilt) on the window covering device to the percentage specified in the Scene Table over the specified transition time.

The device MAY treat the commands as linear transitions if appropriate or MAY accelerate and decelerate as it deems necessary.

For position aware devices, a percentage written by a scene to either the current or target lift/tilt attributes MUST be treated as a GoToLiftPercentage/GoToTiltPercentage command. Using the CurrentPosition Attribute results in writing the received percentage to the associated TargetPosition and activate the motion (lift or tilt) of the window covering device to the specified percentage.

For position unaware devices, a percentage of 0 is treated as a UpOrOpen command and a non-zero percentage is treated as an DownOrClose command and the device will ignore the transition time and transition as fast as appropriate for that device.

Attributes in the Scene Table that are not supported by the device (according to the FeatureMap attribute) SHALL be present in the scene table but ignored.

5.3.6.2. Type Attribute

This attribute SHALL identify the type of window covering.

5.3.6.3. PhysicalClosedLimitLift Attribute

This attribute SHALL indicate the maximum possible encoder position possible (Unit *cm*, centimeters) to position the height of the window covering lift.

5.3.6.4. PhysicalClosedLimitTilt Attribute

This attribute SHALL indicate the maximum possible encoder position possible (Unit *0.1°*, tenths of a degree) to position the angle of the window covering tilt.

5.3.6.5. CurrentPositionLift Attribute

This attribute SHALL indicate the actual lift position (Unit *cm*, centimeters) of the window covering from the fully-open position.

5.3.6.6. CurrentPositionTilt Attribute

This attribute SHALL indicate the actual tilt position (Unit *0.1°*, tenths of a degree) of the window covering from the fully-open position.

5.3.6.7. NumberOfActuationsLift Attribute

This attribute SHALL indicate the total number of lift/slide actuations applied to the window covering since the device was installed.

5.3.6.8. NumberOfActuationsTilt Attribute

This attribute SHALL indicate the total number of tilt actuations applied to the window covering since the device was installed.

5.3.6.9. ConfigStatus Attribute

This attribute specifies the configuration and status information of the window covering.

To change settings, devices SHALL write to the Mode attribute. The behavior causing the setting or clearing of each bit is vendor specific.

5.3.6.9.1. Operational Bit

The SafetyStatus & Mode attributes might affect this bit state.

5.3.6.9.2. LiftMovementReversed Bit

This bit identifies if the directions of the lift/slide movements have been reversed in order for commands (e.g: Open, Close, GoTos) to match the physical installation conditions

This bit can be adjusted by setting the MotorDirectionReversed bit in the Mode attribute.

5.3.6.9.3. LiftEncoderControlled Bit

This bit is ignored if the device does not support the PositionAwareLift feature (PA_LF).

5.3.6.9.4. TiltEncoderControlled Bit

This bit is ignored if the device does not support the PositionAwareTilt feature (PA_TL).

5.3.6.10. CurrentPositionLiftPercent100ths Attribute

This attribute SHALL indicate the actual position as a percentage with a minimal step of 0.01%. E.g Max 10000 equals 100.00%.

5.3.6.11. CurrentPositionTiltPercent100ths Attribute

This attribute SHALL indicate the actual position as a percentage with a minimal step of 0.01%. E.g Max 10000 equals 100.00%.

5.3.6.12. CurrentPositionLiftPercentage Attribute

This attribute SHALL indicate the actual position as a percentage from 0% to 100% with 1% default step. This attribute is equal to CurrentPositionLiftPercent100ths attribute divided by 100.

5.3.6.13. CurrentPositionTiltPercentage Attribute

This attribute SHALL indicate the actual position as a percentage from 0% to 100% with 1% default step. This attribute is equal to CurrentPositionTiltPercent100ths attribute divided by 100.

5.3.6.14. TargetPositionLiftPercent100ths Attribute

This attribute SHALL indicate the position where the window covering lift will go or is moving to as a percentage (Unit *0.01%*).

5.3.6.15. TargetPositionTiltPercent100ths Attribute

This attribute SHALL indicate the position where the window covering tilt will go or is moving to as a percentage (Unit *0.01%*).

5.3.6.16. OperationalStatus Attribute

This attribute SHALL indicate the currently ongoing operations and applies to all type of devices.

5.3.6.17. EndProductType Attribute

This attribute SHOULD provide more detail about the product type than can be determined from the main category indicated by the Type attribute.

The table below helps to match the EndProductType attribute with the Type attribute.

Value	Name	Indoor Outdoor	Indicative Dimension	Recommended Type Attribute
0	**RollerShade**	I	1D	RollerShade
1	**RomanShade**	I	1D	RollerShade
2	**BalloonShade**	I	1D	RollerShade

Value	Name	Indoor Outdoor	Indicative Dimension	Recommended Type Attribute
3	**WovenWood**	I	1D	RollerShade
4	**PleatedShade**	I	1D	RollerShade
5	**CellularShade**	I	1D	RollerShade
6	**LayeredShade**	I	1D	RollerShade
7	**LayeredShade2D**	I	2D	RollerShade2Motor
8	**SheerShade**	I	2D	TiltBlindLiftAndTilt
9	**TiltOnlyInteriorBlind**	I	1D	TiltBlindTiltOnly
10	**InteriorBlind**	I	2D	TiltBlindLiftAndTilt
11	**VerticalBlindStripCurtain**	I	2D	TiltBlindLiftAndTilt
12	**InteriorVenetianBlind**	I	2D	TiltBlindLiftAndTilt
13	**ExteriorVenetianBlind**	O	2D	TiltBlindLiftAndTilt
14	**LateralLeftCurtain**	I	1D	Drapery
15	**LateralRightCurtain**	I	1D	Drapery
16	**CentralCurtain**	I	1D	Drapery
17	**RollerShutter**	O	1D	RollerShadeExterior
18	**ExteriorVerticalScreen**	O	1D	RollerShadeExterior
19	**AwningTerracePatio**	O	1D	Awning
20	**AwningVerticalScreen**	O	1D	Awning
21	**TiltOnlyPergola**	O	1D	Shutter
22	**SwingingShutter**	O	1D	Shutter
23	**SlidingShutter**	O	1D	Shutter
255	**Unknown**			Unknown

5.3.6.18. InstalledOpenLimitLift Attribute

This attribute SHALL indicate the open limit for lifting the window covering whether position (in centimeters) is encoded or timed.

5.3.6.19. InstalledClosedLimitLift Attribute

This attribute SHALL indicate the closed limit for lifting the window covering whether position (in centimeters) is encoded or timed.

5.3.6.20. InstalledOpenLimitTilt Attribute

This attribute SHALL indicate the open limit for tilting the window covering whether position (in tenth of a degree) is encoded or timed.

5.3.6.21. InstalledClosedLimitTilt Attribute

This attribute SHALL indicate the closed limit for tilting the window covering whether position (in tenth of a degree) is encoded or timed.

5.3.6.22. Mode Attribute

The Mode attribute allows configuration of the window covering, such as: reversing the motor direction, placing the window covering into calibration mode, placing the motor into maintenance mode, disabling the network, and disabling status LEDs.

In the case a device does not support or implement a specific mode, e.g. the device has a specific installation method and reversal is not relevant or the device does not include a maintenance mode, any write interaction to the Mode attribute, with an unsupported mode bit or any out of bounds bits set, must be ignored and a response containing the status of CONSTRAINT_ERROR will be returned.

5.3.6.22.1. MotorDirectionReversed Bit

This bit SHALL control the LiftMovementReversed Bit bit in the ConfigStatus Attribute attribute.

5.3.6.22.2. CalibrationMode Bit

If in calibration mode, all commands (e.g: UpOrOpen, DownOrClose, GoTos) that can result in movement, could be accepted and result in a self-calibration being initiated before the command is executed.

In case the window covering does not have the ability or is not able to perform a self-calibration, the command SHOULD be ignored and a FAILURE status SHOULD be returned.

In a write interaction, setting this bit to 0, while the device is in calibration mode, is not allowed and SHALL generate a FAILURE error status. In order to leave calibration mode, the device must perform its calibration routine, either as a self-calibration or assisted by external tool(s), depending on the device/manufacturer implementation.

A manufacturer might choose to set the Operational Bit bit of the ConfigStatus Attribute attribute to

its not operational value, if applicable during calibration mode.

5.3.6.22.3. MaintenanceMode Bit

While in maintenance mode, all commands (e.g: UpOrOpen, DownOrClose, GoTos) or local inputs that can result in movement, must be ignored and respond with a BUSY status. Additionally, the Operational Bit bit of the ConfigStatus Attribute attribute should be set to its not operational value.

5.3.6.23. SafetyStatus Attribute

The SafetyStatus attribute reflects the state of the safety sensors and the common issues preventing movements. By default for nominal operation all flags are cleared (0). A device might support none, one or several bit flags from this attribute (all optional).

5.3.7. Commands

ID	Name	Direction	Response	Access	Conformance
0x00	**UpOrOpen**	client ⇒ server	Y	O	M
0x01	**DownOrClose**	client ⇒ server	Y	O	M
0x02	**StopMotion**	client ⇒ server	Y	O	M
0x04	**GoToLiftValue**	client ⇒ server	Y	O	[LF & ABS]
0x05	**GoToLiftPercentage**	client ⇒ server	Y	O	LF & PA_LF, [LF]
0x07	**GoToTiltValue**	client ⇒ server	Y	O	[TL & ABS]
0x08	**GoToTiltPercentage**	client ⇒ server	Y	O	TL & PA_TL, [TL]

5.3.7.1. UpOrOpen Command

Upon receipt of this command, the window covering will adjust its position so the physical lift/slide and tilt is at the maximum open/up position. This will happen as fast as possible. The server attributes SHALL be updated as follows:

if the PositionAware feature is supported:

- TargetPositionLiftPercent100ths attribute SHALL be set to 0.00%.
- TargetPositionTiltPercent100ths attribute SHALL be set to 0.00%.

The server positioning attributes will follow the movements, once the movement has successfully finished, the server attributes SHALL be updated as follows:

if the PositionAware feature is supported:

- CurrentPositionLiftPercent100ths attribute SHALL be 0.00%.
- CurrentPositionLiftPercentage attribute SHALL be 0%.
- CurrentPositionTiltPercent100ths attribute SHALL be 0.00%.

- CurrentPositionTiltPercentage attribute SHALL be 0%.

if the AbsolutePosition feature is supported:

- CurrentPositionLift attribute SHALL be equal to the InstalledOpenLimitLift attribute.
- CurrentPositionTilt attribute SHALL be equal to the InstalledOpenLimitTilt attribute.

5.3.7.2. DownOrClose Command

Upon receipt of this command, the window covering will adjust its position so the physical lift/slide and tilt is at the maximum closed/down position. This will happen as fast as possible. The server attributes supported SHALL be updated as follows:

if the PositionAware feature is supported:

- TargetPositionLiftPercent100ths attribute SHALL be set to 100.00%.
- TargetPositionTiltPercent100ths attribute SHALL be set to 100.00%.

The server positioning attributes will follow the movements, once the movement has successfully finished, the server attributes SHALL be updated as follows:

if the PositionAware feature is supported:

- CurrentPositionLiftPercent100ths attribute SHALL be 100.00%.
- CurrentPositionLiftPercentage attribute SHALL be 100%.
- CurrentPositionTiltPercent100ths attribute SHALL be 100.00%.
- CurrentPositionTiltPercentage attribute SHALL be 100%.

if the AbsolutePosition feature is supported:

- CurrentPositionLift attribute SHALL be equal to the InstalledClosedLimitLift attribute.
- CurrentPositionTilt attribute SHALL be equal to the InstalledClosedLimitTilt attribute.

5.3.7.3. StopMotion Command

Upon receipt of this command, the window covering will stop any adjusting to the physical tilt and lift/slide that is currently occurring. The server attributes supported SHALL be updated as follows:

- TargetPositionLiftPercent100ths attribute will be set to CurrentPositionLiftPercent100ths attribute value.
- TargetPositionTiltPercent100ths attribute will be set to CurrentPositionTiltPercent100ths attribute value.

5.3.7.4. GoToLiftValue Command

This command SHALL have the following data fields:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**LiftValue**	uint16	desc			M

5.3.7.4.1. LiftValue Field

This field SHALL specify the requested physical lift/slide position in unit cm (centimeters).

5.3.7.4.2. Effect on Receipt

Upon receipt of this command, the window covering will adjust the lift position to the value specified in the LiftValue field, as long as that value is not larger thanInstalledOpenLimitLift attribute and not smaller than InstalledClosedLimitLift attribute. The TargetPositionLiftPercent100ths attribute SHALL update its value accordingly.

If the value is out of bounds a response containing the status of CONSTRAINT_ERROR will be returned.

5.3.7.5. GoToLiftPercentage Command

This command SHALL have the following data fields:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**LiftPercentageValue**	percent	desc			O.a
1	**LiftPercent100thsValue**	percent100ths	desc			O.a

Upon receipt of this command, the server will adjust the window covering to the lift/slide percentage specified in the payload of this command.

If the command includes LiftPercent100thsValue, then TargetPositionLiftPercent100ths attribute SHALL be set to LiftPercent100thsValue. Otherwise the TargetPositionLiftPercent100ths attribute SHALL be set to LiftPercentageValue * 100.

If a client includes LiftPercent100thsValue in the command, the LiftPercentageValue SHALL be set to LiftPercent100thsValue / 100, so a legacy server which only supports LiftPercentageValue (not LiftPercent100thsValue) has a value to set the target position.

If the server does not support the PositionAware feature, then a zero percentage SHALL be treated as a UpOrOpen command and a non-zero percentage SHALL be treated as an DownOrClose command. If the device is only a tilt control device, then the command SHOULD be ignored and a UNSUPPORTED_COMMAND status SHOULD be returned.

5.3.7.6. GoToTiltValue Command

This command SHALL have the following data fields:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**TiltValue**	uint16	desc			M

5.3.7.6.1. TiltValue

This field SHALL specify the requested physical tilt position in unit *0.1°* (tenth of a degrees).

5.3.7.6.2. Effect on Receipt

Upon receipt of this command, the window covering will adjust the tilt position to the value specified in the TiltValue field, as long as that value is not larger than InstalledOpenLimitTilt attribute and not smaller than InstalledClosedLimitTilt attribute. The TargetPositionTiltPercent100ths attribute SHALL update its value accordingly.

If the tilt value is out of bounds a response containing the status of CONSTRAINT_ERROR will be returned.

5.3.7.7. GoToTiltPercentage Command

This command SHALL have the following data fields:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**TiltPercentageValue**	percent	desc			O.a
1	**TiltPercent100thsValue**	percent100ths	desc			O.a

Upon receipt of this command, the server will adjust the window covering to the tilt percentage specified in the payload of this command.

If the command includes TiltPercent100thsValue, then TargetPositionTiltPercent100ths attribute SHALL be set to TiltPercent100thsValue. Otherwise the TargetPositionTiltPercent100ths attribute SHALL be set to TiltPercentageValue * 100.

If a client includes TiltPercent100thsValue in the command, the TiltPercentageValue SHALL be set to TiltPercent100thsValue / 100, so a legacy server which only supports TiltPercentageValue (not TiltPercent100thsValue) has a value to set the target position.

If the server does not support the PositionAware feature, then a zero percentage SHALL be treated as a UpOrOpen command and a non-zero percentage SHALL be treated as an DownOrClose command. If the device is only a tilt control device, then the command SHOULD be ignored and a UNSUPPORTED_COMMAND status SHOULD be returned.

Chapter 6. Media

The Cluster Library is made of individual chapters such as this one. See Document Control in the Cluster Library for a list of all chapters and documents. References between chapters are made using a *X.Y* notation where *X* is the chapter and *Y* is the sub-section within that chapter. References to external documents are contained in Chapter 1 and are made using [*Rn*] notation.

6.1. General Description

6.1.1. Introduction

The clusters specified in this document are for use typically in applications involving media (e.g., Video Players, Content Apps, Speakers), but MAY be used in any application domain.

6.1.2. Cluster List

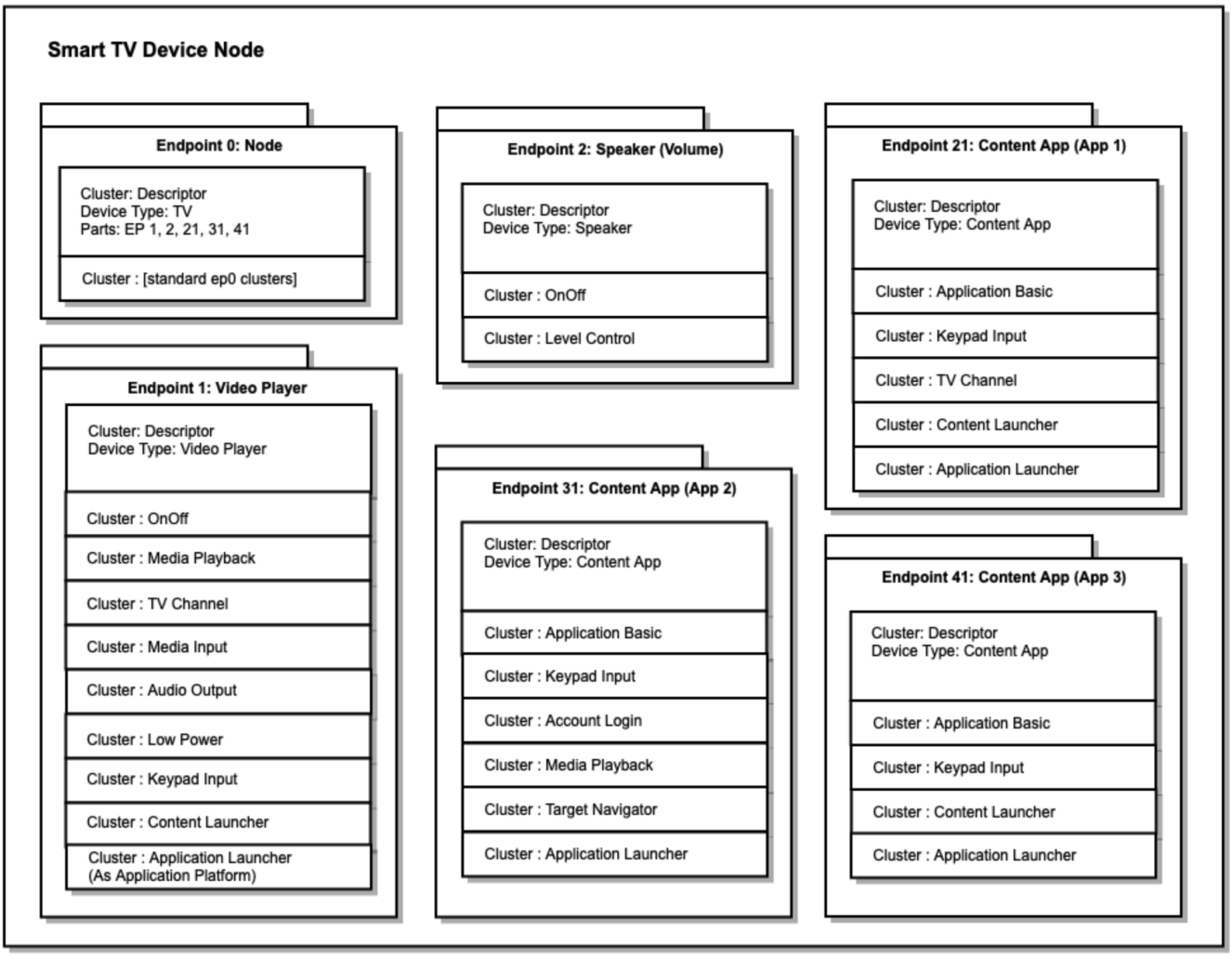
This section lists the media specific clusters as specified in this chapter.

Table 37. Overview of the Media Clusters

Cluster ID	Cluster Name	Description
0x050E	Account Login	This cluster provides an interface for facilitating user account login on an application or a node.
0x050D	Application Basic	Provides information about a Content App running on a Video Player device which is represented as an endpoint.
0x050C	Application Launcher	This cluster provides an interface for launching content on a Video Player device.
0x050B	Audio Output	This cluster provides an interface for controlling the Output on a Video Player device.
0x0504	Channel	This cluster provides an interface for controlling the current Channel on an endpoint.
0x0510	Content App Observer	This cluster provides an interface for sending targeted messages from a Content App to a Client of that Content App.

Cluster ID	Cluster Name	Description
0x050A	Content Launcher	This cluster provides an interface for launching content on a Video Player device or a Content App.
0x0509	Keypad Input	This cluster provides an interface for controlling a Video Player or a Content App using action commands such as UP, DOWN, and SELECT.
0x0507	Media Input	This cluster provides an interface for controlling the Input Selector on a Video Player device.
0x0506	Media Playback	This cluster provides an interface for controlling Media Playback (PLAY, PAUSE, etc) on a Video Player device.
0x0505	Target Navigator	This cluster provides an interface for UX navigation within a set of targets on a Video Player device or Content App endpoint.
0x050F	Content Control	This cluster provides an interface for controlling Content controls such as turning on/off Content Control feature on a Video Player device.

Example Usage of the Media Clusters



6.2. Account Login Cluster

This cluster provides commands that facilitate user account login on a Content App or a node. For example, a Content App running on a Video Player device, which is represented as an endpoint (see Device Type Library document), can use this cluster to help make the user account on the Content App match the user account on the Client.

Often a fabric administrator will facilitate commissioning of a Client (such as a Casting Video Client), and invoke commands on the AccountLogin cluster on the Content App associated with that client. Specifically:

1. GetSetupPIN in order to attempt to obtain the Passcode for commissioning.
2. Login in order to let the Content App know that commissioning has completed. The Content App can use information provided in this command in order to determine the user account associated with the client, and potentially assume that user account.
3. Logout in order to let the Content App know that client access has been removed, and potentially clear the current user account.

The cluster server for this cluster may be supported on each endpoint that represents a Content App on a Video Player device.

See Device Type Library document for details of how a Content App, represented as an endpoint on

the Video Player device, may implement the cluster server for this cluster to simplify account login for its users.

6.2.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Revision	Description
1	Initial Release
2	Add support for 8 character PIN code, Add Node to Login, Logout commands. Add LoggedOut event.

6.2.2. Classification

Hierarchy	Role	Scope	PICS Code
Base	Application	Endpoint	ALOGIN

6.2.3. Cluster ID

ID	Name
0x050E	Account Login

6.2.4. Commands

ID	Name	Direction	Response	Access	Conformance
0x00	**GetSetupPIN**	client ⇒ server	GetSetupPIN-Response	A F T	M
0x01	**GetSetupPIN-Response**	client ⇐ server	N	F	M
0x02	**Login**	client ⇒ server	Y	A F T	M
0x03	**Logout**	client ⇒ server	Y	O F T	M

6.2.4.1. GetSetupPIN Command

The purpose of this command is to determine if the active user account of the given Content App matches the active user account of a given Commissionee, and when it does, return a Setup PIN code which can be used for password-authenticated session establishment (PASE) with the Commissionee.

For example, a Video Player with a Content App Platform may invoke this command on one of its Content App endpoints to facilitate commissioning of a Phone App made by the same vendor as the Content App. If the accounts match, then the Content App may return a setup code that can be used

by the Video Player to commission the Phone App without requiring the user to physically input a setup code.

The account match is determined by the Content App using a method which is outside the scope of this specification and will typically involve a central service which is in communication with both the Content App and the Commissionee. The GetSetupPIN command is needed in order to provide the Commissioner/Admin with a Setup PIN when this Commissioner/Admin is operated by a different vendor from the Content App.

This method is used to facilitate Setup PIN exchange (for PASE) between Commissioner and Commissionee when the same user account is active on both nodes. With this method, the Content App satisfies proof of possession related to commissioning by requiring the same user account to be active on both Commissionee and Content App, while the Commissioner/Admin ensures user consent by prompting the user prior to invocation of the command.

Upon receipt of this command, the Content App checks if the account associated with the Temporary Account Identifier sent by the client is the same account that is active on itself. If the accounts are the same, then the Content App returns the GetSetupPIN Response which includes a Setup PIN that may be used for PASE with the Commissionee.

The Temporary Account Identifier for a Commissionee may be populated with the Rotating ID field of the client's commissionable node advertisement (see Rotating Device Identifier section in [MatterCore]) encoded as an octet string where the octets of the Rotating Device Identifier are encoded as 2-character sequences by representing each octet's value as a 2-digit hexadecimal number, using uppercase letters.

The Setup PIN is a character string so that it can accommodate different future formats, including alpha-numeric encodings. For a Commissionee it SHALL be populated with the Manual Pairing Code (see Manual Pairing Code section in [MatterCore]) encoded as a string (11 characters) or the Passcode portion of the Manual Pairing Code (when less than 11 characters) .

The server SHALL implement rate limiting to prevent brute force attacks. No more than 10 unique requests in a 10 minute period SHALL be allowed; a command response status of FAILURE should sent for additional commands received within the 10 minute period. Because access to this command is limited to nodes with Admin-level access, and the user is prompted for consent prior to Commissioning, there are in place multiple obstacles to successfully mounting a brute force attack. A Content App that supports this command SHALL ensure that the Temporary Account Identifier used by its clients is not valid for more than 10 minutes.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**TempAccountIdentifier**	string	16 to 100			M

6.2.4.1.1. TempAccountIdentifier Field

This field SHALL specify the client's Temporary Account Identifier. The length of this field SHALL be at least 16 characters to protect the account holder against password guessing attacks.

6.2.4.2. GetSetupPINResponse Command

This message is sent in response to the GetSetupPIN command, and contains the Setup PIN code, or null when the account identified in the request does not match the active account of the running Content App.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**SetupPIN**	string	desc			M

6.2.4.2.1. SetupPIN Field

This field SHALL provide the setup PIN code as a text string at least 8 characters in length or empty string to indicate that the accounts do not match.

NOTE	Newer cluster clients should be aware that AccountLogin cluster version 1 specified an 11 digit minimum length.

6.2.4.3. Login Command

The purpose of this command is to allow the Content App to assume the user account of a given Commissionee by leveraging the Setup PIN code input by the user during the commissioning process.

For example, a Video Player with a Content App Platform may invoke this command on one of its Content App endpoints after the commissioning has completed of a Phone App made by the same vendor as the Content App. The Content App may determine whether the Temporary Account Identifier maps to an account with a corresponding Setup PIN and, if so, it may automatically login to the account for the corresponding user. The end result is that a user performs commissioning of a Phone App to a Video Player by inputting the Setup PIN for the Phone App into the Video Player UX. Once commissioning has completed, the Video Player invokes this command to allow the corresponding Content App to assume the same user account as the Phone App.

The verification of Setup PIN for the given Temporary Account Identifier is determined by the Content App using a method which is outside the scope of this specification and will typically involve a central service which is in communication with both the Content App and the Commissionee. Implementations of such a service should impose aggressive time outs for any mapping of Temporary Account Identifier to Setup PIN in order to prevent accidental login due to delayed invocation.

Upon receipt, the Content App checks if the account associated with the client's Temp Account Identifier has a current active Setup PIN with the given value. If the Setup PIN is valid for the user account associated with the Temp Account Identifier, then the Content App MAY make that user account active.

The Temporary Account Identifier for a Commissionee may be populated with the Rotating ID field of the client's commissionable node advertisement encoded as an octet string where the octets of the Rotating Device Identifier are encoded as 2-character sequences by representing each octet's value as a 2-digit hexadecimal number, using uppercase letters.

The Setup PIN for a Commissionee may be populated with the Manual Pairing Code encoded as a string of decimal numbers (11 characters) or the Passcode portion of the Manual Pairing Code encoded as a string of decimal numbers (8 characters) .

The server SHALL implement rate limiting to prevent brute force attacks. No more than 10 unique requests in a 10 minute period SHALL be allowed; a command response status of FAILURE should sent for additional commands received within the 10 minute period. Because access to this command is limited to nodes with Admin-level access, and the user is involved when obtaining the SetupPIN, there are in place multiple obstacles to successfully mounting a brute force attack. A Content App that supports this command SHALL ensure that the Temporary Account Identifier used by its clients is not valid for more than 10 minutes.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**TempAccountIdentifier**	string	16 to 100			M
1	**SetupPIN**	string	min 8			M
2	**Node**	node-id				O

6.2.4.3.1. TempAccountIdentifier Field

This field SHALL specify the client's temporary account identifier.

6.2.4.3.2. SetupPIN Field

This field SHALL provide the setup PIN code as a text string at least 8 characters in length.

NOTE	Newer cluster clients should be aware that AccountLogin cluster version 1 specified an 11 digit minimum length.

6.2.4.3.3. Node Field

This optional field SHALL provide the Node ID of the Client. This field can be used by the Content App to keep track of Nodes which currently have access to it.

6.2.4.4. Logout Command

The purpose of this command is to instruct the Content App to clear the current user account. This command SHOULD be used by clients of a Content App to indicate the end of a user session.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**Node**	node-id				O

6.2.4.4.1. Node Field

This optional field SHALL provide the Node ID of the Client. This field can be used by the Content

App to keep track of Nodes which currently have access to it.

6.2.5. Events

ID	Name	Priority	Access	Conformance
0	LoggedOut	CRITICAL	A S	O

6.2.5.1. LoggedOut Event

This event can be used by the Content App to indicate that the current user has logged out. In response to this event, the Fabric Admin SHALL remove access to this Content App by the specified Node. If no Node is provided, then the Fabric Admin SHALL remove access to all non-Admin Nodes.

The data of this event SHALL contain the following information:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**Node**	node-id	all			O

6.2.5.1.1. Node

This field SHALL provide the Node ID corresponding to the user account that has logged out, if that Node ID is available. If it is NOT available, this field SHALL NOT be present in the event.

6.3. Application Basic Cluster

This cluster provides information about a Content App running on a Video Player device which is represented as an endpoint (see Device Type Library document).

The cluster server for this cluster should be supported on each endpoint that represents a Content App on a Video Player device. This cluster provides identification information about the Content App such as vendor and product.

6.3.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Revision	Description
1	Initial Release

6.3.2. Classification

Hierarchy	Role	Scope	PICS Code
Base	Application	Endpoint	APBSC

6.3.3. Cluster ID

ID	Name
0x050D	Application Basic

6.3.4. Data Types

6.3.4.1. ApplicationStatusEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**Stopped**	Application is not running.	M
1	**ActiveVisibleFocus**	Application is running, is visible to the user, and is the active target for input.	M
2	**ActiveHidden**	Application is running but not visible to the user.	M
3	**ActiveVisibleNotFocus**	Application is running and visible, but is not the active target for input.	M

6.3.4.2. ApplicationStruct Type

This indicates a global identifier for an Application given a catalog.

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0	**CatalogVendorID**	uint16	all				M
1	**ApplicationID**	string	all				M

6.3.4.2.1. CatalogVendorID Field

This field SHALL indicate the Connectivity Standards Alliance issued vendor ID for the catalog. The DIAL registry SHALL use value 0x0000.

It is assumed that Content App Platform providers (see Video Player Architecture section in [Matter-DevLib]) will have their own catalog vendor ID (set to their own Vendor ID) and will assign an ApplicationID to each Content App.

6.3.4.2.2. ApplicationID Field

This field SHALL indicate the application identifier, expressed as a string, such as "123456-5433", "PruneVideo" or "Company X". This field SHALL be unique within a catalog.

For the DIAL registry catalog, this value SHALL be the DIAL prefix.

6.3.5. Attributes

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0000	**VendorName**	string	max 32	F	empty	R V	O
0x0001	**VendorID**	vendor-id	all	F		R V	O
0x0002	**ApplicationName**	string	desc	F		R V	M
0x0003	**ProductID**	uint16	all	F		R V	O
0x0004	**Application**	ApplicationStruct	desc	F		R V	M
0x0005	**Status**	ApplicationStatusEnum	desc		MS	R V	M
0x0006	**ApplicationVersion**	string	max 32	F		R V	M
0x0007	**AllowedVendorList**	list[vendor-id]		F		R A	M

6.3.5.1. VendorName Attribute

This attribute SHALL specify a human readable (displayable) name of the vendor for the Content App.

6.3.5.2. VendorID Attribute

This attribute, if present, SHALL specify the Connectivity Standards Alliance assigned Vendor ID for the Content App.

6.3.5.3. ApplicationName Attribute

This attribute SHALL specify a human readable (displayable) name of the Content App assigned by the vendor. For example, "NPR On Demand". The maximum length of the ApplicationName attribute is 256 bytes of UTF-8 characters.

6.3.5.4. ProductID Attribute

This attribute, if present, SHALL specify a numeric ID assigned by the vendor to identify a specific Content App made by them. If the Content App is certified by the Connectivity Standards Alliance, then this would be the Product ID as specified by the vendor for the certification.

6.3.5.5. Application Attribute

This attribute SHALL specify a Content App which consists of an Application ID using a specified catalog.

6.3.5.6. Status Attribute

This attribute SHALL specify the current running status of the application.

6.3.5.7. ApplicationVersion Attribute

This attribute SHALL specify a human readable (displayable) version of the Content App assigned by the vendor. The maximum length of the ApplicationVersion attribute is 32 bytes of UTF-8 characters.

6.3.5.8. AllowedVendorList Attribute

This attribute is a list of vendor IDs. Each entry is a vendor-id.

6.4. Application Launcher Cluster

This cluster provides an interface for launching applications on a Video Player device such as a TV.

This cluster is supported on endpoints that can launch Applications, such as a Casting Video Player device with a Content App Platform. It supports identifying an Application by global identifier from a given catalog, and launching it. It also supports tracking the currently in-focus Application.

Depending on the support for the Application Platform feature, the cluster can either support launching the application corresponding to the endpoint on which the cluster is supported (AP feature not supported) or it can support launching any application (AP feature supported).

6.4.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Revision	Description
1	Initial Release

6.4.2. Classification

Hierarchy	Role	Scope	PICS Code
Base	Application	Endpoint	APPLAUNCHER

6.4.3. Cluster ID

ID	Name
0x050C	Application Launcher

6.4.4. Features

This cluster SHALL support the FeatureMap bitmap attribute as defined below.

Bit	Code	Feature	Summary
0	**AP**	ApplicationPlatform	Support for attributes and commands required for endpoint to support launching any application within the supported application catalogs

6.4.5. Data Types

6.4.5.1. StatusEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**Success**	Command succeeded	M
1	**AppNotAvailable**	Requested app is not available.	M
2	**SystemBusy**	Video platform unable to honor command.	M

6.4.5.2. ApplicationStruct Type

This indicates a global identifier for an Application given a catalog.

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0	**CatalogVendorID**	uint16	all				M

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
1	**ApplicationID**	string	all				M

6.4.5.2.1. CatalogVendorID Field

This field SHALL indicate the CSA-issued vendor ID for the catalog. The DIAL registry SHALL use value 0x0000.

Content App Platform providers will have their own catalog vendor ID (set to their own Vendor ID) and will assign an ApplicationID to each Content App.

6.4.5.2.2. ApplicationID Field

This field SHALL indicate the application identifier, expressed as a string, such as "PruneVideo" or "Company X". This field SHALL be unique within a catalog.

For the DIAL registry catalog, this value SHALL be the DIAL prefix (see [DIAL Registry]).

6.4.5.3. ApplicationEPStruct Type

This specifies an app along with its corresponding endpoint.

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0	**Application**	ApplicationStruct	all				M
1	**Endpoint**	endpoint-no	all		MS		O

6.4.6. Attributes

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0000	**CatalogList**	list[uint16]		N		R V	AP
0x0001	**CurrentApp**	ApplicationEPStruct	desc	X	null	R V	O

6.4.6.1. CatalogList Attribute

This attribute SHALL specify the list of supported application catalogs, where each entry in the list is the CSA-issued vendor ID for the catalog. The DIAL registry (see [DIAL Registry]) SHALL use value 0x0000.

It is expected that Content App Platform providers will have their own catalog vendor ID (set to their own Vendor ID) and will assign an ApplicationID to each Content App.

6.4.6.2. CurrentApp Attribute

This attribute SHALL specify the current in-focus application, identified using an Application ID, catalog vendor ID and the corresponding endpoint number when the application is represented by a Content App endpoint. A null SHALL be used to indicate there is no current in-focus application.

6.4.7. Commands

ID	Name	Direction	Response	Access	Conformance
0x00	**LaunchApp**	client ⇒ server	LauncherResponse	O	M
0x01	**StopApp**	client ⇒ server	LauncherResponse	O	M
0x02	**HideApp**	client ⇒ server	LauncherResponse	O	M
0x03	**LauncherResponse**	client ⇐ server	N		M

6.4.7.1. LaunchApp Command

Upon receipt of this command, the server SHALL launch the application with optional data. The application SHALL be either

- the specified application, if the Application Platform feature is supported;
- otherwise the application corresponding to the endpoint.

The endpoint SHALL launch and bring to foreground the requisite application if the application is not already launched and in foreground. The Status attribute SHALL be updated to ActiveVisibleFocus on the Application Basic cluster of the Endpoint corresponding to the launched application. The Status attribute SHALL be updated on any other application whose Status MAY have changed as a result of this command. The CurrentApp attribute, if supported, SHALL be updated to reflect the new application in the foreground.

This command returns a Launcher Response.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**Application**	Application-Struct	desc			AP
1	**Data**	octstr			MS	O

6.4.7.1.1. Application Field

This field SHALL specify the Application to launch.

6.4.7.1.2. Data Field

This field SHALL specify optional app-specific data to be sent to the app.

NOTE	This format and meaning of this value is proprietary and outside the specification. It provides a transition path for device makers that use other protocols (like DIAL) which allow for proprietary data. Apps that are not yet Matter aware can be launched via Matter, while retaining the existing ability to launch with proprietary data.

6.4.7.2. StopApp Command

Upon receipt of this command, the server SHALL stop the application if it is running. The application SHALL be either

- the specified application, if the Application Platform feature is supported;
- otherwise the application corresponding to the endpoint.

The Status attribute SHALL be updated to Stopped on the Application Basic cluster of the Endpoint corresponding to the stopped application. The Status attribute SHALL be updated on any other application whose Status MAY have changed as a result of this command.

This command returns a Launcher Response.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**Application**	Application-Struct	desc		MS	AP

6.4.7.2.1. Application Field

This field SHALL specify the Application to stop.

6.4.7.3. HideApp Command

Upon receipt of this command, the server SHALL hide the application. The application SHALL be either

- the specified application, if the Application Platform feature is supported;
- otherwise the application corresponding to the endpoint.

The endpoint MAY decide to stop the application based on manufacturer specific behavior or resource constraints if any. The Status attribute SHALL be updated to ActiveHidden or Stopped, depending on the action taken, on the Application Basic cluster of the Endpoint corresponding to the application on which the action was taken. The Status attribute SHALL be updated on any other

application whose Status MAY have changed as a result of this command.

This command returns a Launcher Response.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**Application**	Application-Struct	desc		MS	AP

6.4.7.3.1. Application Field

This field SHALL specify the Application to hide.

6.4.7.4. LauncherResponse Command

This command SHALL be generated in response to LaunchApp/StopApp/HideApp commands.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	Status	StatusEnum	all			M
1	Data	octstr			MS	O

6.4.7.4.1. Status Field

This field SHALL indicate the status of the command which resulted in this response.

6.4.7.4.2. Data Field

This field SHALL specify Optional app-specific data.

6.5. Audio Output Cluster

This cluster provides an interface for controlling the Output on a Video Player device such as a TV.

This cluster would be supported on a device with audio outputs like a Video Player device (Smart TV, TV Setup Top Box, Smart Speaker, etc).

This cluster provides the list of available outputs and provides commands for selecting and renaming them.

The cluster server for Audio Output is implemented by a device that has configurable audio output.

6.5.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Revision	Description
1	Initial Release

6.5.2. Classification

Hierarchy	Role	Scope	PICS Code
Base	Application	Endpoint	AUDIOOUTPUT

6.5.3. Cluster ID

ID	Name
0x050B	Audio Output

6.5.4. Features

This cluster SHALL support the FeatureMap bitmap attribute as defined below.

Bit	Code	Feature	Summary
0	**NU**	NameUpdates	Supports updates to output names

6.5.5. Data Types

6.5.5.1. OutputTypeEnum Type

This data type is derived from enum8.

The type of output, expressed as an enum, with the following values:

Value	Name	Summary	Conformance
0	**HDMI**	HDMI	M
1	**BT**		M
2	**Optical**		M
3	**Headphone**		M
4	**Internal**		M
5	**Other**		M

6.5.5.2. OutputInfoStruct Type

This contains information about an output.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**Index**	uint8	all			M
1	**OutputType**	OutputTypeEnum	desc			M
2	**Name**	string	all			M

6.5.5.2.1. Index Field

This field SHALL indicate the unique index into the list of outputs.

6.5.5.2.2. OutputType Field

This field SHALL indicate the type of output.

6.5.5.2.3. Name Field

The device defined and user editable output name, such as “Soundbar”, “Speakers”. This field may be blank, but SHOULD be provided when known.

6.5.6. Attributes

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0000	**OutputList**	list[OutputInfoStruct]				R V	M
0x0001	**CurrentOutput**	uint8	all			R V	M

6.5.6.1. OutputList Attribute

This attribute provides the list of outputs supported by the device.

6.5.6.2. CurrentOutput Attribute

This attribute contains the value of the index field of the currently selected OutputInfoStruct.

6.5.7. Commands

ID	Name	Direction	Response	Access	Conformance
0x00	**SelectOutput**	client ⇒ server	Y	O	M
0x01	**RenameOutput**	client ⇒ server	Y	M	NU

6.5.7.1. SelectOutput Command

Upon receipt, this SHALL change the output on the device to the output at a specific index in the Output List.

Note that when the current output is set to an output of type HDMI, adjustments to volume via a Speaker endpoint on the same node MAY cause HDMI volume up/down commands to be sent to the given HDMI output.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**Index**	uint8	all			M

6.5.7.1.1. Index Field

This SHALL indicate the index field of the OutputInfoStruct from the OutputList attribute in which to change to.

6.5.7.2. RenameOutput Command

Upon receipt, this SHALL rename the output at a specific index in the Output List.

Updates to the output name SHALL appear in the device's settings menus. Name updates MAY automatically be sent to the actual device to which the output connects.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**Index**	uint8	all			M
1	**Name**	string	all			M

6.6. Channel Cluster

This cluster provides an interface for controlling the current Channel on a device or endpoint.

This cluster server would be supported on Video Player devices or endpoints that allow Channel control such as a Content App. This cluster provides a list of available channels and provides commands for absolute and relative channel changes.

The cluster server for Channel is implemented by an endpoint that controls the current Channel.

6.6.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Revision	Description
1	Initial Release

Revision	Description
2	Add EG and RP features, Identifier and Type to Channel Info for Over-the-Top (OTT) channel support.

6.6.2. Classification

Hierarchy	Role	Scope	PICS Code
Base	Application	Endpoint	CHANNEL

6.6.3. Cluster ID

ID	Name
0x0504	Channel

6.6.4. Features

This cluster SHALL support the FeatureMap bitmap attribute as defined below.

Bit	Code	Feature	Summary
0	**CL**	ChannelList	Provides list of available channels.
1	**LI**	LineupInfo	Provides lineup info, which is a reference to an external source of lineup information.
2	**EG**	ElectronicGuide	Provides electronic program guide information.
3	**RP**	RecordProgram	Provides ability to record program.

6.6.5. Data Types

6.6.5.1. RecordingFlagBitmap Type

This data type is derived from map8.

Bit	Name	Summary	Conformance
0	**Scheduled**	The program is scheduled for recording.	M

Bit	Name	Summary	Conformance
1	**RecordSeries**	The program series is scheduled for record-ing.	M
2	**Recorded**	The program is recorded and available to be played.	M

6.6.5.2. LineupInfoTypeEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**MSO**	Multi System Operator	M

6.6.5.3. StatusEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**Success**	Command succeeded	M
1	**MultipleMatches**	More than one equal match for the Chan-nelInfoStruct passed in.	M
2	**NoMatches**	No matches for the ChannelInfoStruct passed in.	M

6.6.5.4. ChannelTypeEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**Satellite**	The channel is sourced from a satellite provider.	M
1	**Cable**	The channel is sourced from a cable provider.	M
2	**Terrestrial**	The channel is sourced from a terrestrial provider.	M
3	**OTT**	The channel is sourced from an OTT provider.	M

6.6.5.5. ChannelInfoStruct Type

This indicates a channel in a channel lineup.

While the major and minor numbers in the ChannelInfoStruct support use of ATSC channel format, a lineup MAY use other formats which can map into these numeric values.

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0	**MajorNumber**	uint16	all				M
1	**MinorNumber**	uint16	all				M
2	**Name**	string			empty		O
3	**CallSign**	string			empty		O
4	**AffiliateCallSign**	string			empty		O
5	**Identifier**	string			empty		O
6	**Type**	ChannelTypeEnum			empty		O

6.6.5.5.1. MajorNumber Field

This field SHALL indicate the channel major number value (for example, using ATSC format). When the channel number is expressed as a string, such as "13.1" or "256", the major number would be 13 or 256, respectively. This field is required but SHALL be set to 0 for channels such as over-the-top channels that are not represented by a major or minor number.

6.6.5.5.2. MinorNumber Field

This field SHALL indicate the channel minor number value (for example, using ATSC format). When the channel number is expressed as a string, such as "13.1" or "256", the minor number would be 1 or 0, respectively. This field is required but SHALL be set to 0 for channels such as over-the-top channels that are not represented by a major or minor number.

6.6.5.5.3. Name Field

This field SHALL indicate the marketing name for the channel, such as “The CW" or "Comedy Central". This field is optional, but SHOULD be provided when known.

6.6.5.5.4. CallSign Field

This field SHALL indicate the call sign of the channel, such as "PBS". This field is optional, but SHOULD be provided when known.

6.6.5.5.5. AffiliateCallSign Field

This field SHALL indicate the local affiliate call sign, such as "KCTS". This field is optional, but

SHOULD be provided when known.

6.6.5.5.6. Identifier Field

This SHALL indicate the unique identifier for a specific channel. This field is optional, but SHOULD be provided when MajorNumber and MinorNumber are not available.

6.6.5.5.7. Type Field

This SHALL indicate the type or grouping of a specific channel. This field is optional, but SHOULD be provided when known.

6.6.5.6. LineupInfoStruct Type

The Lineup Info allows references to external lineup sources like Gracenote. The combination of OperatorName, LineupName, and PostalCode MUST uniquely identify a lineup.

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0	OperatorName	string					M
1	LineupName	string			empty		O
2	PostalCode	string			empty		O
3	LineupInfoType	LineupInfoType Enum	desc				M

6.6.5.6.1. OperatorName Field

This field SHALL indicate the name of the operator, for example “Comcast”.

6.6.5.6.2. LineupName Field

This field SHALL indicate the name of the provider lineup, for example "Comcast King County". This field is optional, but SHOULD be provided when known.

6.6.5.6.3. PostalCode Field

This field SHALL indicate the postal code (zip code) for the location of the device, such as "98052". This field is optional, but SHOULD be provided when known.

6.6.5.6.4. LineupInfoType Field

This field SHALL indicate the type of lineup. This field is optional, but SHOULD be provided when known.

6.6.5.7. ProgramStruct Type

This indicates a program within an electronic program guide (EPG).

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0	**Identifier**	string	max 255				M
1	**Channel**	ChannelInfoStruct					M
2	**StartTime**	epoch-s					M
3	**EndTime**	epoch-s					M
4	**Title**	string	max 255				M
5	**Subtitle**	string	max 255		empty		O
6	**Description**	string	max 8192		empty		O
7	**AudioLanguages**	list[string]	max 10 [max 50]		empty		O
8	**Ratings**	list[string]	max 255		empty		O
9	**ThumbnailUrl**	string	max 8192		empty		O
10	**PosterArtUrl**	string	max 8192		empty		O
11	**DvbiUrl**	string	max 8192		empty		O
12	**ReleaseDate**	string	max 30		empty		O
13	**ParentalGuidanceText**	string	max 255		empty		O
14	**RecordingFlag**	RecordingFlagBitmap					RP
15	**SeriesInfo**	SeriesInfoStruct		X	null		O
16	**CategoryList**	list[ProgramCategoryStruct]	max 255		empty		O
17	**CastList**	list[ProgramCastStruct]	max 255		empty		O

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
18	**ExternalIDList**	list[AdditionalInfoStruct]	max 255		empty		O

6.6.5.7.1. Identifier Field

This field SHALL indicate a unique identifier for a program within an electronic program guide list. The identifier SHALL be unique across multiple channels.

6.6.5.7.2. Channel Field

This field SHALL indicate the channel associated to the program.

6.6.5.7.3. StartTime Field

This field SHALL indicate an epoch time in seconds indicating the start time of a program, as a UTC time. This field can represent a past or future value.

6.6.5.7.4. EndTime Field

This field SHALL indicate an epoch time in seconds indicating the end time of a program, as a UTC time. This field can represent a past or future value but SHALL be greater than the StartTime.

6.6.5.7.5. Title Field

This field SHALL indicate the title or name for the specific program. For example, “MCIS: Los Angeles”.

6.6.5.7.6. Subtitle Field

This field SHALL indicate the subtitle for the specific program. For example, “Maybe Today" which is an episode name for “MCIS: Los Angeles”. This field is optional but SHALL be provided if applicable and known.

6.6.5.7.7. Description Field

This field SHALL indicate the brief description for the specific program. For example, a description of an episode. This field is optional but SHALL be provided if known.

6.6.5.7.8. AudioLanguages Field

This field SHALL indicate the audio language for the specific program. The value is a string containing one of the standard Tags for Identifying Languages RFC 5646. This field is optional but SHALL be provided if known.

6.6.5.7.9. Ratings Field

This field SHALL be used for indicating the level of parental guidance recommended for of a particular program. This can be any rating system used in the country or region where the program is

broadcast. For example, in the United States “TV-PG” may contain material that parents can find not suitable for younger children but can be accepted in general for older children. This field is optional but SHALL be provided if known.

6.6.5.7.10. ThumbnailUrl Field

This field SHALL represent a url of a thumbnail that clients can use to render an image for the program.

6.6.5.7.11. PosterArtUrl Field

This field SHALL represent a url of a poster that clients can use to render an image for the program on the detail view.

6.6.5.7.12. DvbiUrl Field

This field SHALL represent the DVB-I url associated to the program.

6.6.5.7.13. ReleaseDate Field

This field SHALL be a string, in ISO 8601 format, representing the date on which the program was released. This field is optional but when provided, the year SHALL be provided as part of the string.

6.6.5.7.14. ParentalGuidanceText Field

This field SHALL represent a string providing additional information on the parental guidance. This field is optional.

6.6.5.7.15. RecordingFlag Field

This field SHALL represent the recording status of the program. This field is required if the RecordProgram feature is set.

6.6.5.7.16. SeriesInfo Field

This field SHALL represent the information of a series such as season and episode number. This field is optional but SHOULD be provided if the program represents a series and this information is available.

6.6.5.7.17. CategoryList Field

This field SHALL represent the category of a particular program. This field is optional but SHALL be provided if known.

6.6.5.7.18. CastList Field

This field SHALL represent a list of the cast or the crew on the program. A single cast member may have more than one role. This field is optional but SHALL be provided if known.

6.6.5.7.19. ExternalIDList Field

This field SHALL indicate the list of additional external content identifiers.

6.6.5.8. ProgramCategoryStruct Type

This object defines the category associated to a program.

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0	**Category**	string	max 256				M
1	**SubCategory**	string	max 256		empty		O

6.6.5.8.1. Category Field

This field SHALL represent the category or genre of the program. Ex. News.

6.6.5.8.2. SubCategory Field

This field SHALL represent the sub-category or sub-genre of the program. Ex. Local.

6.6.5.9. SeriesInfoStruct Type

This object provides the episode information related to a program.

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0	**Season**	string	max 256				M
1	**Episode**	string	max 256				M

6.6.5.9.1. Season Field

This field SHALL represent the season of the series associated to the program.

6.6.5.9.2. Episode Field

This field SHALL represent the episode of the program.

6.6.5.10. ProgramCastStruct Type

This object provides the cast information related to a program.

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0	**Name**	string	max 256				M
1	**Role**	string	max 256				M

6.6.5.10.1. Name Field

This field SHALL represent the name of the cast member.

6.6.5.10.2. Role Field

This field SHALL represent the role of the cast member. Ex. Actor, Director.

6.6.5.11. PageTokenStruct Type

This object defines the pagination structure.

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0	**Limit**	uint16	all		0		O
1	**After**	string	max 8192		empty		O
2	**Before**	string	max 8192		empty		O

6.6.5.11.1. Limit Field

This field SHALL indicate the maximum number of entries that should be retrieved from the program guide in a single response. It allows clients to specify the size of the paginated result set based on their needs.

6.6.5.11.2. After Field

This field SHALL indicate the cursor that pinpoints the start of the upcoming data page. In a Cursor-based pagination system, the field acts as a reference point, ensuring the set of results corresponds directly to the data following the specified cursor. In a Offset-based pagination system, the field, along with limit, indicate the offset from which entries in the program guide will be retrieved.

6.6.5.11.3. Before Field

This field SHALL indicate the cursor that pinpoints the end of the upcoming data page. In a Cursor-based pagination system, the field acts as a reference point, ensuring the set of results corresponds directly to the data preceding the specified cursor. In a Offset-based pagination system, the field, along with limit, indicate the offset from which entries in the program guide will be retrieved.

6.6.5.12. ChannelPagingStruct Type

This object defines the paging structure that includes the previous and next pagination tokens.

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0	**PreviousToken**	PageTokenStruct		X	null		O
1	**NextToken**	PageTokenStruct		X	null		O

6.6.5.12.1. PreviousToken Field

This field SHALL indicate the token to retrieve the preceding page. Absence of this field denotes the response as the initial page.

6.6.5.12.2. NextToken Field

This field SHALL indicate the token to retrieve the next page. Absence of this field denotes the response as the last page.

6.6.6. Attributes

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0000	**ChannelList**	list[ChannelInfoStruct]			empty	R V	CL
0x0001	**Lineup**	LineupInfoStruct	desc	X	null	R V	LI
0x0002	**CurrentChannel**	ChannelInfoStruct	desc	X	null	R V	O

6.6.6.1. ChannelList Attribute

This attribute SHALL provide the list of supported channels.

6.6.6.2. Lineup Attribute

This attribute SHALL identify the channel lineup using external data sources.

6.6.6.3. CurrentChannel Attribute

This attribute SHALL contain the current channel. When supported but a channel is not currently tuned to (if a content application is in foreground), the value of the field SHALL be null.

6.6.7. Commands

ID	Name	Direction	Response	Access	Quality	Conformance
0x00	**ChangeChannel**	client ⇒ server	ChangeChannelResponse	O		CL | LI
0x01	**ChangeChannelResponse**	client ⇐ server	N			CL | LI
0x02	**ChangeChannelByNumber**	client ⇒ server	Y	O		M
0x03	**SkipChannel**	client ⇒ server	Y	O		M

ID	Name	Direction	Response	Access	Quality	Conformance
0x04	**GetProgramGuide**	client ⇒ server	ProgramGuideResponse	O	L	EG
0x05	**ProgramGuideResponse**	client ⇐ server	N		L	EG
0x06	**RecordProgram**	client ⇒ server	Y	O	L	RP & EG
0x07	**CancelRecordProgram**	client ⇒ server	Y	O	L	RP & EG

6.6.7.1. ChangeChannel Command

Change the channel to the channel case-insensitive exact matching the value passed as an argument.

The match priority order SHALL be: Identifier, AffiliateCallSign, CallSign, Name, Number. In the match string, the Channel number should be presented in the "Major.Minor" format, such as "13.1".

Upon receipt, this SHALL generate a ChangeChannelResponse command.

Upon success, the CurrentChannel attribute, if supported, SHALL be updated to reflect the change.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**Match**	string				M

6.6.7.1.1. Match Field

This field SHALL contain a user-input string to match in order to identify the target channel.

6.6.7.2. ChangeChannelResponse Command

This command SHALL be generated in response to a ChangeChannel command.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**Status**	StatusEnum	desc			M
1	**Data**	string	any			O

6.6.7.2.1. Status Field

This field SHALL indicate the status of the command which resulted in this response.

6.6.7.2.2. Data Field

This field SHALL indicate Optional app-specific data.

6.6.7.3. ChangeChannelByNumber Command

Change the channel to the channel with the given Number in the *ChannelList* attribute.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**MajorNumber**	uint16	all			M
1	**MinorNumber**	uint16	all			M

6.6.7.3.1. MajorNumber Field

This field SHALL indicate the channel major number value (ATSC format) to which the channel should change.

6.6.7.3.2. MinorNumber Field

This field SHALL indicate the channel minor number value (ATSC format) to which the channel should change.

6.6.7.4. SkipChannel Command

This command provides channel up and channel down functionality, but allows channel index jumps of size *Count*.

When the value of the increase or decrease is larger than the number of channels remaining in the given direction, then the behavior SHALL be to return to the beginning (or end) of the channel list and continue. For example, if the current channel is at index 0 and count value of -1 is given, then the current channel should change to the last channel.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**Count**	int16	all			M

6.6.7.4.1. Count Field

This field SHALL indicate the number of steps to increase (Count is positive) or decrease (Count is negative) the current channel.

6.6.7.5. GetProgramGuide Command

This command retrieves the program guide. It accepts several filter parameters to return specific schedule and program information from a content app. The command shall receive in response a ProgramGuideResponse. Standard error codes SHALL be used when arguments provided are not

valid. For example, if StartTime is greater than EndTime, the status code INVALID_ACTION SHALL be returned.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**StartTime**	epoch-s				M
1	**EndTime**	epoch-s				M
2	**ChannelList**	list[ChannelInfoStruct]	max 255		empty	O
3	**PageToken**	PageTokenStruct		X	null	O
5	**RecordingFlag**	RecordingFlagBitmap		X	null	O
6	**ExternalIDList**	list[AdditionalInfoStruct]	max 255		empty	O
7	**Data**	octstr	max 8092		MS	O

6.6.7.5.1. StartTime Field

This field SHALL indicate the beginning of the time window for which program guide entries are to be retrieved, as a UTC time. Entries with a start time on or after this value will be included in the results.

6.6.7.5.2. EndTime Field

This field SHALL indicate the end of the time window for which program guide entries are to be retrieved, as a UTC time. Entries with an end time on or before this value will be included in the results. This field can represent a past or future value but SHALL be greater than the StartTime.

6.6.7.5.3. ChannelList Field

This field SHALL indicate the set of channels for which program guide entries should be fetched. By providing a list of channels in this field, the response will only include entries corresponding to the specified channels.

6.6.7.5.4. PageToken Field

This field SHALL indicate the pagination token used for managing pagination progression.

6.6.7.5.5. RecordingFlag Field

This field SHALL indicate the flags of the programs for which entries should be fetched.

6.6.7.5.6. ExternalIDList Field

This field SHALL indicate the list of additional external content identifiers.

6.6.7.5.7. Data Field

This field SHALL indicate Optional app-specific data.

6.6.7.6. ProgramGuideResponse Command

This command is a response to the GetProgramGuide command.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**Paging**	ChannelPagingStruct				M
1	**ProgramList**	list[ProgramStruct]			empty	M

6.6.7.6.1. Paging Field

This field SHALL indicate the necessary pagination attributes that define information for both the succeeding and preceding data pages.

6.6.7.6.2. ProgramList Field

This field SHALL indicate the list of programs.

6.6.7.7. RecordProgram Command

Record a specific program or series when it goes live. This functionality enables DVR recording features.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**ProgramIdentifier**	string	max 255			M
1	**ShouldRecordSeries**	bool				M
2	**ExternalIDList**	list[AdditionalInfoStruct]	max 255		empty	O
3	**Data**	octstr	max 8092		MS	O

6.6.7.7.1. ProgramIdentifier Field

This field SHALL indicate the program identifier for the program that should be recorded. This value is provided by the identifier field in ProgramStruct.

6.6.7.7.2. ShouldRecordSeries Field

This field SHALL indicate whether the whole series associated to the program should be recorded. For example, invoking record program on an episode with that flag set to true, the target should schedule record the whole series.

6.6.7.7.3. ExternalIDList Field

This field, if present, SHALL indicate the list of additional external content identifiers.

6.6.7.7.4. Data Field

This field, if present, SHALL indicate app-specific data.

6.6.7.8. CancelRecordProgram Command

Cancel recording for a specific program or series.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**ProgramIdentifier**	string	max 255			M
1	**ShouldRecordSeries**	bool				M
2	**ExternalIDList**	list[AdditionalInfoStruct]	max 255		empty	O
3	**Data**	octstr	max 8092		MS	O

6.6.7.8.1. ProgramIdentifier Field

This field SHALL indicate the program identifier for the program that should be cancelled from recording. This value is provided by the identifier field in ProgramStruct.

6.6.7.8.2. ShouldRecordSeries Field

This field SHALL indicate whether the whole series associated to the program should be cancelled from recording. For example, invoking record program on an episode with that flag set to true, the target should schedule record the whole series.

6.6.7.8.3. ExternalIDList Field

This field, if present, SHALL indicate the list of additional external content identifiers.

6.6.7.8.4. Data Field

This field, if present, SHALL indicate app-specific data.

6.7. Content Launcher Cluster

This cluster provides an interface for launching content on a Video Player device such as a Streaming Media Player, Smart TV or Smart Screen.

This cluster would be supported on a Video Player device or devices that can playback content, such as a Streaming Media Player, Smart TV or Smart Screen. This cluster supports playing back content referenced by URL. It supports finding content by type and global identifier, and either playing the content or displaying the search results.

The cluster server for Content Launcher is implemented by an endpoint that can launch content, such as a Video Player, or an endpoint representing a Content App on such a device.

When this cluster is implemented for an Content App Endpoint (Endpoint with type “Content App” and having an Application Basic cluster), the Video Player device SHALL launch the application when a client invokes the LaunchContent or LaunchURL commands.

6.7.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Revision	Description
1	Initial Release
2	Add Seasons, Episode, Any and CurrentContext to search, Added Text/Audio tracks support with PlaybackPreferences field

6.7.2. Classification

Hierarchy	Role	Scope	PICS Code
Base	Application	Endpoint	CONTENTLAUNCHER

6.7.3. Cluster ID

ID	Name
0x050A	Content Launcher

6.7.4. Features

This cluster SHALL support the FeatureMap bitmap attribute as defined below.

Bit	Code	Feature	Summary
0	**CS**	ContentSearch	Device supports content search (non-app specific)

Bit	Code	Feature	Summary
1	**UP**	URLPlayback	Device supports basic URL-based file playback
2	**AS**	AdvancedSeek	Enables clients to implement more advanced media seeking behavior in their user interface, such as for example a "seek bar".
3	**TT**	TextTracks	Device or app supports Text Tracks.
4	**AT**	AudioTracks	Device or app supports Audio Tracks.

6.7.5. Data Types

6.7.5.1. SupportedProtocolsBitmap Type

This data type is derived from map32.

Bit	Name	Summary
0	**DASH**	Device supports Dynamic Adaptive Streaming over HTTP (DASH)
1	**HLS**	Device supports HTTP Live Streaming (HLS)

6.7.5.2. StatusEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**Success**	Command succeeded	M
1	**URLNotAvailable**	Requested URL could not be reached by device.	M
2	**AuthFailed**	Requested URL returned 401 error code.	M
3	**TextTrackNotAvailable**	Requested Text Track (in PlaybackPreferences) not available	TT

Value	Name	Summary	Conformance
4	**AudioTrackNotAvailable**	Requested Audio Track (in PlaybackPreferences) not available	AT

6.7.5.3. ParameterEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**Actor**	Actor represents an actor credited in video media content; for example, “Gaby Hoffman”	M
1	**Channel**	Channel represents the identifying data for a television channel; for example, "PBS"	M
2	**Character**	A character represented in video media content; for example, “Snow White”	M
3	**Director**	A director of the video media content; for example, “Spike Lee”	M
4	**Event**	An event is a reference to a type of event; examples would include sports, music, or other types of events. For example, searching for "Football games" would search for a 'game' event entity and a 'football' sport entity.	M

Value	Name	Summary	Conformance
5	**Franchise**	A franchise is a video entity which can represent a number of video entities, like movies or TV shows. For example, take the fictional franchise "Intergalactic Wars" which represents a collection of movie trilogies, as well as animated and live action TV shows. This entity type was introduced to account for requests by customers such as "Find Intergalactic Wars movies", which would search for all 'Intergalactic Wars' programs of the MOVIE MediaType, rather than attempting to match to a single title.	M
6	**Genre**	Genre represents the genre of video media content such as action, drama or comedy.	M
7	**League**	League represents the categorical information for a sporting league; for example, "NCAA"	M
8	**Popularity**	Popularity indicates whether the user asks for popular content.	M
9	**Provider**	The provider (MSP) the user wants this media to be played on; for example, "Netflix".	M
10	**Sport**	Sport represents the categorical information of a sport; for example, football	M

Value	Name	Summary	Conformance
11	**SportsTeam**	SportsTeam represents the categorical information of a professional sports team; for example, "University of Washington Huskies"	M
12	**Type**	The type of content requested. Supported types are "Movie", "MovieSeries", "TVSeries", "TVSeason", "TVEpisode", "Trailer", "SportsEvent", "LiveEvent", and "Video"	M
13	**Video**	Video represents the identifying data for a specific piece of video content; for example, "Manchester by the Sea".	M
14	**Season**	Season represents the specific season number within a TV series.	O
15	**Episode**	Episode represents a specific episode number within a Season in a TV series.	O
16	**Any**	Represents a search text input across many parameter types or even outside of the defined param types.	O

6.7.5.4. MetricTypeEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**Pixels**	Dimensions defined in a number of Pixels	M
1	**Percentage**	Dimensions defined as a percentage	M

6.7.5.4.1. Pixels Value

This value is used for dimensions defined in a number of Pixels.

6.7.5.4.2. Percentage Value

This value is for dimensions defined as a percentage of the overall display dimensions. For example, if using a Percentage Metric type for a Width measurement of 50.0, against a display width of 1920 pixels, then the resulting value used would be 960 pixels (50.0% of 1920) for that dimension. Whenever a measurement uses this Metric type, the resulting values SHALL be rounded ("floored") towards 0 if the measurement requires an integer final value.

6.7.5.5. AdditionalInfoStruct Type

This object defines additional name=value pairs that can be used for identifying content.

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0	**Name**	string	max 256				M
1	**Value**	string	max 8192				M

6.7.5.5.1. Name Field

This field SHALL indicate the name of external id, ex. "musicbrainz".

6.7.5.5.2. Value Field

This field SHALL indicate the value for external id, ex. "ST0000000666661".

6.7.5.6. ParameterStruct Type

This object defines inputs to a search for content for display or playback.

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0	**Type**	ParameterEnum	all				M
1	**Value**	string	max 1024				M
2	**ExternalIDList**	list[AdditionalInfoStruct]	all		empty		O

6.7.5.6.1. Type Field

This field SHALL indicate the entity type.

6.7.5.6.2. Value Field

This field SHALL indicate the entity value, which is a search string, ex. “Manchester by the Sea”.

6.7.5.6.3. ExternalIDList Field

This field SHALL indicate the list of additional external content identifiers.

6.7.5.7. ContentSearchStruct Type

This object defines inputs to a search for content for display or playback.

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0	**ParameterList**	list[ParameterStruct]	all		0		M

6.7.5.7.1. ParameterList Field

This field SHALL indicate the list of parameters comprising the search. If multiple parameters are provided, the search parameters SHALL be joined with 'AND' logic. e.g. action movies with Tom Cruise will be represented as [{Actor: 'Tom Cruise'}, {Type: 'Movie'}, {Genre: 'Action'}]

6.7.5.8. DimensionStruct Type

This object defines dimension which can be used for defining Size of background images.

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0	**Width**	double			MS		M
1	**Height**	double			MS		M
2	**Metric**	MetricTypeEnum					M

6.7.5.8.1. Width Field

This field SHALL indicate the width using the metric defined in Metric

6.7.5.8.2. Height Field

This field SHALL indicate the height using the metric defined in Metric

6.7.5.8.3. Metric Field

This field SHALL indicate metric used for defining Height/Width.

6.7.5.9. StyleInformationStruct Type

This object defines style information which can be used by content providers to change the Media Player’s style related properties.

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0	**ImageURL**	string	max 8192		MS		O
1	**Color**	string	7,9		MS		O
2	**Size**	DimensionStruct			MS		O

6.7.5.9.1. ImageURL Field

This field SHALL indicate the URL of image used for Styling different Video Player sections like Logo, Watermark etc.

6.7.5.9.2. Color Field

This field SHALL indicate the color, in RGB or RGBA, used for styling different Video Player sections like Logo, Watermark, etc. The value SHALL conform to the 6-digit or 8-digit format defined for CSS sRGB hexadecimal color notation [https://www.w3.org/TR/css-color-4/#hex-notation]. Examples:

- #76DE19 for R=0x76, G=0xDE, B=0x19, A absent
- #76DE1980 for R=0x76, G=0xDE, B=0x19, A=0x80

6.7.5.9.3. Size Field

This field SHALL indicate the size of the image used for Styling different Video Player sections like Logo, Watermark etc.

6.7.5.10. BrandingInformationStruct Type

This object defines Branding Information which can be provided by the client in order to customize the skin of the Video Player during playback.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**ProviderName**	string	max 256			M
1	**Background**	StyleInformationStruct			MS	O
2	**Logo**	StyleInformationStruct			MS	O
3	**ProgressBar**	StyleInformationStruct			MS	O
4	**Splash**	StyleInformationStruct			MS	O
5	**WaterMark**	StyleInformationStruct			MS	O

6.7.5.10.1. ProviderName Field

This field SHALL indicate name of the provider for the given content.

6.7.5.10.2. Background Field

This field SHALL indicate background of the Video Player while content launch request is being processed by it. This background information MAY also be used by the Video Player when it is in idle state.

6.7.5.10.3. Logo Field

This field SHALL indicate the logo shown when the Video Player is launching. This is also used when the Video Player is in the idle state and Splash field is not available.

6.7.5.10.4. ProgressBar Field

This field SHALL indicate the style of progress bar for media playback.

6.7.5.10.5. Splash Field

This field SHALL indicate the screen shown when the Video Player is in an idle state. If this property is not populated, the Video Player SHALL default to logo or the provider name.

6.7.5.10.6. Watermark Field

This field SHALL indicate watermark shown when the media is playing.

6.7.5.11. PlaybackPreferencesStruct Type

PlaybackPreferencesStruct defines the preferences sent by the client to the receiver in the ContentLauncher LaunchURL or LaunchContent commands.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**PlaybackPosition**	uint64		X		AS
1	**TextTrack**	TrackPreferenceStruct		X		TT
2	**AudioTracks**	list[TrackPreferenceStruct]		X		AT

6.7.5.11.1. PlaybackPosition Field

This field SHALL indicate the preferred position (in milliseconds) in the media to launch playback from. In case the position falls in the middle of a frame, the server SHALL set the position to the beginning of that frame and set the `SampledPosition` attribute on the MediaPlayback cluster accordingly. A value of null SHALL indicate that playback position is not applicable for the current state of the media playback. (For example : Live media with no known duration and where seek is not sup-

ported).

6.7.5.11.2. TextTrack Field

This field SHALL indicate the user's preferred Text Track. A value of null SHALL indicate that the user did not specify a preferred Text Track on the client. In such a case, the decision to display and select a Text Track is up to the server.

6.7.5.11.3. AudioTracks Field

This field SHALL indicate the list of the user's preferred Audio Tracks. If the list contains multiple values, each AudioTrack must also specify a unique audioOutputIndex to play the track on. A value of null SHALL indicate that the user did not specify a preferred Audio Track on the client. In such a case, the decision to play and select an Audio Track is up to the server.

6.7.5.12. TrackPreferenceStruct Type

This structure defines Text/Audio Track preferences.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**LanguageCode**	string	max 32			M
1	**Characteristics**	list[CharacteristicEnum]		X	null	O
2	**AudioOutputIndex**	uint8		X		AT

6.7.5.12.1. LanguageCode Field

This field SHALL contain one of the standard Tags for Identifying Languages RFC 5646, which identifies the primary language used in the Track.

6.7.5.12.2. Characteristics Field

This field SHALL contain a list of enumerated CharacteristicEnum values that indicate a purpose, trait or feature associated with the Track. A value of null SHALL indicate that there are no Characteristics corresponding to the Track.

6.7.5.12.3. AudioOutputIndex Field

This field if present SHALL indicate the index of the OutputInfoStruct from the OutputList attribute (from the AudioOutput cluster) and indicates which audio output the Audio Track should be played on.

This field SHALL NOT be present if the track is not an audio track.

If the track is an audio track, this field MUST be present. A value of null SHALL indicate that the

server can choose the audio output(s) to play the Audio Track on.

6.7.6. Attributes

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0000	**AcceptHeader**	list[string]	max 100[max 1024]	N	empty	R V	UP
0x0001	**SupportedStreamingProtocols**	SupportedProtocolsBitmap		N	0	R V	UP

6.7.6.1. AcceptHeader Attribute

This attribute SHALL provide a list of content types supported by the Video Player or Content App in the form of entries in the HTTP "Accept" request header.

6.7.6.2. SupportedStreamingProtocols Attribute

This attribute SHALL provide information about supported streaming protocols.

6.7.7. Commands

ID	Name	Direction	Response	Access	Conformance
0x00	**LaunchContent**	client ⇒ server	LauncherResponse	O	CS
0x01	**LaunchURL**	client ⇒ server	LauncherResponse	O	UP
0x02	**LauncherResponse**	client ⇐ server	N		CS | UP

6.7.7.1. LaunchContent Command

Upon receipt, this SHALL launch the specified content with optional search criteria.

This command returns a Launch Response.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**Search**	ContentSearchStruct	desc			M
1	**AutoPlay**	bool	desc			M

ID	Name	Type	Constraint	Quality	Default	Conformance
2	**Data**	string			MS	O
3	**PlaybackPreferences**	PlaybackPreferencesStruct	all		MS	O
4	**UseCurrentContext**	bool	desc		TRUE	O

6.7.7.1.1. Search Field

This field SHALL indicate the content to launch.

6.7.7.1.2. AutoPlay Field

This field SHALL indicate whether to automatically start playing content, where:

- TRUE means best match should start playing automatically.
- FALSE means matches should be displayed on screen for user selection.

6.7.7.1.3. Data Field

This field, if present, SHALL indicate app-specific data.

6.7.7.1.4. PlaybackPreferences Field

This field, if present, SHALL indicate the user's preferred Text/AudioTracks and playbackPosition for the media, sent from the client to the server. If the server does not find an available track for the title being played exactly matching a Track requested here, in the list of available tracks, it may default to picking another track that closely matches the requested track. Alternately, it may go with user preferences set on the server side (it will use this option if these PlaybackPreferences are not specified). In the case of text tracks, that may mean that the subtitle text is not displayed at all. In the cases where the preferred Text/AudioTracks are not available, the server SHALL return the TextTrackNotAvailable and/or AudioTrackNotAvailable Status(es) in the LauncherResponse.

6.7.7.1.5. UseCurrentContext Field

This field, if present, SHALL indicate whether to consider the context of current ongoing activity on the receiver to fulfill the request. For example if the request only includes data in ContentSearch that specifies an Episode number, and UseCurrentContent is set to TRUE, if there is a TV series on going, the request refers to the specific episode of the ongoing season of the TV series. TRUE means current activity context MAY be considered FALSE means current activity context SHALL NOT be considered

6.7.7.2. LaunchURL Command

Upon receipt, this SHALL launch content from the specified URL.

The content types supported include those identified in the AcceptHeader and SupportedStreaming-

Protocols attributes.

A check SHALL be made to ensure the URL is secure (uses HTTPS).

When playing a video stream in response to this command, an indication (ex. visual) of the identity of the origin node of the video stream SHALL be provided. This could be in the form of a friendly name label which uniquely identifies the node to the user. This friendly name label is typically assigned by the Matter Admin (ex. TV) at the time of commissioning and, when it's a device, is often editable by the user. It might be a combination of a company name and friendly name, for example, "Acme" or "Acme Streaming Service on Alice's Phone".

This command returns a Launch Response.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**ContentURL**	string	any			M
1	**DisplayString**	string	any		MS	O
2	**BrandingInformation**	BrandingInformationStruct	any		MS	O
3	**PlaybackPreferences**	PlaybackPreferencesStruct	any		MS	O

6.7.7.2.1. ContentURL Field

This field SHALL indicate the URL of content to launch.

6.7.7.2.2. DisplayString Field

This field, if present, SHALL provide a string that MAY be used to describe the content being accessed at the given URL.

6.7.7.2.3. BrandingInformation Field

This field, if present, SHALL indicate the branding information that MAY be displayed when playing back the given content.

6.7.7.2.4. PlaybackPreferences Field

This field, if present, SHALL indicate the user's preferred Text/AudioTracks and playbackPosition for the media, sent from the client to the server. If the server does not find an available track for the title being played exactly matching a Track requested here, in the list of available tracks, it may default to picking another track that closely matches the requested track. Alternately, it may go with user preferences set on the server side (it will use this option if these PlaybackPreferences are not specified). In the case of text tracks, that may mean that the subtitle text is not displayed at all. In the cases where the preferred Text/AudioTracks are not available, the server SHALL return the TextTrackNotAvailable and/or AudioTrackNotAvailable Status(es) in the LauncherResponse.

6.7.7.3. LauncherResponse Command

This command SHALL be generated in response to LaunchContent and LaunchURL commands.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**Status**	StatusEnum	all			M
1	**Data**	string			MS	O

6.7.7.3.1. Status Field

This field SHALL indicate the status of the command which resulted in this response.

6.7.7.3.2. Data Field

This field SHALL indicate Optional app-specific data.

6.8. Keypad Input Cluster

This cluster provides an interface for key code based input and control on a device like a Video Player or an endpoint like a Content App. This may include text or action commands such as UP, DOWN, and SELECT.

This cluster would be supported on Video Player devices as well as devices that support remote control input from a keypad or remote. This cluster provides the list of supported keypad inputs and provides a command for sending them.

The cluster server for Keypad Input is implemented by a device that can receive keypad input, such as a Video Player, or an endpoint that can receive keypad input, such as a Content App.

The key codes used are those defined in the HDMI CEC specification (see HDMI).

Devices MAY understand a subset of these key codes. Feature flags are used to indicate a specific subset that is supported. Device MAY support additional codes beyond what is indicated in feature flags.

6.8.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Revision	Description
1	Initial Release

6.8.2. Classification

Hierarchy	Role	Scope	PICS Code
Base	Application	Endpoint	KEYPADINPUT

6.8.3. Cluster ID

ID	Name
0x0509	Keypad Input

6.8.4. Features

This cluster SHALL support the FeatureMap bitmap attribute as defined below.

Bit	Code	Feature	Summary
0	**NV**	NavigationKeyCodes	Supports UP, DOWN, LEFT, RIGHT, SELECT, BACK, EXIT, MENU
1	**LK**	LocationKeys	Supports CEC keys 0x0A (Settings) and 0x09 (Home)
2	**NK**	NumberKeys	Supports numeric input 0..9

6.8.5. Data Types

6.8.5.1. StatusEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**Success**	Succeeded	M
1	**UnsupportedKey**	Key code is not supported.	M
2	**InvalidKeyInCurrentState**	Requested key code is invalid in the context of the responder's current state.	M

6.8.5.2. CecKeyCodeEnum Type

This data type is derived from enum8.

Value	Name	Conformance
0x00	**Select**	M
0x01	**Up**	M
0x02	**Down**	M
0x03	**Left**	M

Value	Name	Conformance
0x04	**Right**	M
0x05	**RightUp**	M
0x06	**RightDown**	M
0x07	**LeftUp**	M
0x08	**LeftDown**	M
0x09	**RootMenu**	M
0x0A	**SetupMenu**	M
0x0B	**ContentsMenu**	M
0x0C	**FavoriteMenu**	M
0x0D	**Exit**	M
0x10	**MediaTopMenu**	M
0x11	**MediaContextSensitiveMenu**	M
0x1D	**NumberEntryMode**	M
0x1E	**Number11**	M
0x1F	**Number12**	M
0x20	**Number0OrNumber10**	M
0x21	**Numbers1**	M
0x22	**Numbers2**	M
0x23	**Numbers3**	M
0x24	**Numbers4**	M
0x25	**Numbers5**	M
0x26	**Numbers6**	M
0x27	**Numbers7**	M
0x28	**Numbers8**	M
0x29	**Numbers9**	M
0x2A	**Dot**	M
0x2B	**Enter**	M
0x2C	**Clear**	M
0x2F	**NextFavorite**	M
0x30	**ChannelUp**	M
0x31	**ChannelDown**	M
0x32	**PreviousChannel**	M
0x33	**SoundSelect**	M

Value	Name	Conformance
0x34	**InputSelect**	M
0x35	**DisplayInformation**	M
0x36	**Help**	M
0x37	**PageUp**	M
0x38	**PageDown**	M
0x40	**Power**	M
0x41	**VolumeUp**	M
0x42	**VolumeDown**	M
0x43	**Mute**	M
0x44	**Play**	M
0x45	**Stop**	M
0x46	**Pause**	M
0x47	**Record**	M
0x48	**Rewind**	M
0x49	**FastForward**	M
0x4A	**Eject**	M
0x4B	**Forward**	M
0x4C	**Backward**	M
0x4D	**StopRecord**	M
0x4E	**PauseRecord**	M
0x4F	**Reserved**	M
0x50	**Angle**	M
0x51	**SubPicture**	M
0x52	**VideoOnDemand**	M
0x53	**ElectronicProgramGuide**	M
0x54	**TimerProgramming**	M
0x55	**InitialConfiguration**	M
0x56	**SelectBroadcastType**	M
0x57	**SelectSoundPresentation**	M
0x60	**PlayFunction**	M
0x61	**PausePlayFunction**	M
0x62	**RecordFunction**	M
0x63	**PauseRecordFunction**	M

Value	Name	Conformance
0x64	**StopFunction**	M
0x65	**MuteFunction**	M
0x66	**RestoreVolumeFunction**	M
0x67	**TuneFunction**	M
0x68	**SelectMediaFunction**	M
0x69	**SelectAvInputFunction**	M
0x6A	**SelectAudioInputFunction**	M
0x6B	**PowerToggleFunction**	M
0x6C	**PowerOffFunction**	M
0x6D	**PowerOnFunction**	M
0x71	**F1Blue**	M
0x72	**F2Red**	M
0x73	**F3Green**	M
0x74	**F4Yellow**	M
0x75	**F5**	M
0x76	**Data**	M

6.8.6. Commands

ID	Name	Direction	Response	Access	Conformance
0x00	**SendKey**	client ⇒ server	SendKeyResponse	O	M
0x01	**SendKeyResponse**	client ⇐ server	N		M

6.8.6.1. SendKey Command

Upon receipt, this SHALL process a keycode as input to the media endpoint.

If a device has multiple media endpoints implementing this cluster, such as a casting video player endpoint with one or more content app endpoints, then only the endpoint receiving the command SHALL process the keycode as input. In other words, a specific content app endpoint SHALL NOT process a keycode received by a different content app endpoint.

If a second SendKey request with the same KeyCode value is received within 200 ms, then the endpoint will consider the first key press to be a press and hold. When such a repeat KeyCode value is not received within 200 ms, then the endpoint will consider the last key press to be a release.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**KeyCode**	CecKeyCodeEnum	all			M

6.8.6.1.1. KeyCode Field

This field SHALL indicate the key code to process.

6.8.6.2. SendKeyResponse Command

This command SHALL be generated in response to a SendKey command. The data for this command SHALL be as follows:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**Status**	StatusEnum	all			M

6.8.6.2.1. Status Field

This field SHALL indicate the status of the request.

6.9. Media Input Cluster

This cluster provides an interface for controlling the Input Selector on a media device such as a Video Player.

This cluster would be implemented on TV and other media streaming devices, as well as devices that provide input to or output from such devices.

This cluster provides the list of available inputs and provides commands for selecting and renaming them.

The cluster server for Media Input is implemented by a device that has selectable input, such as a Video Player device.

6.9.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Revision	Description
1	Initial Release

6.9.2. Classification

Hierarchy	Role	Scope	PICS Code
Base	Application	Endpoint	MEDIAINPUT

6.9.3. Cluster ID

ID	Name
0x0507	Media Input

6.9.4. Features

This cluster SHALL support the FeatureMap bitmap attribute as defined below.

Bit	Code	Feature	Summary
0	**NU**	NameUpdates	Supports updates to the input names

6.9.5. Data Types

6.9.5.1. InputTypeEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**Internal**	Indicates content not coming from a physical input.	M
1	**Aux**		M
2	**Coax**		M
3	**Composite**		M
4	**HDMI**		M
5	**Input**		M
6	**Line**		M
7	**Optical**		M
8	**Video**		M
9	**SCART**		M
10	**USB**		M
11	**Other**		M

6.9.5.2. InputInfoStruct Type

This contains information about an input.

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0	**Index**	uint8	all				M
1	**InputType**	InputTypeEnum	desc				M
2	**Name**	string					M
3	**Description**	string					M

6.9.5.2.1. Index Field

This field SHALL indicate the unique index into the list of Inputs.

6.9.5.2.2. InputType Field

This field SHALL indicate the type of input

6.9.5.2.3. Name Field

This field SHALL indicate the input name, such as “HDMI 1”. This field may be blank, but SHOULD be provided when known.

6.9.5.2.4. Description Field

This field SHALL indicate the user editable input description, such as “Living room Playstation”. This field may be blank, but SHOULD be provided when known.

6.9.6. Attributes

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0000	**InputList**	list[InputInfoStruct]				R V	M
0x0001	**CurrentInput**	uint8	all			R V	M

6.9.6.1. InputList Attribute

This attribute SHALL provide a list of the media inputs supported by the device.

6.9.6.2. CurrentInput Attribute

This attribute SHALL contain the value of the index field of the currently selected InputInfoStruct.

6.9.7. Commands

ID	Name	Direction	Response	Access	Conformance
0x00	**SelectInput**	client ⇒ server	Y	O	M
0x01	**ShowInputStatus**	client ⇒ server	Y	O	M
0x02	**HideInputStatus**	client ⇒ server	Y	O	M
0x03	**RenameInput**	client ⇒ server	Y	M	NU

6.9.7.1. SelectInput Command

Upon receipt, this command SHALL change the media input on the device to the input at a specific index in the Input List.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**Index**	uint8	all			M

6.9.7.1.1. Index Field

This field SHALL indicate the index field of the InputInfoStruct from the InputList attribute in which to change to.

6.9.7.2. ShowInputStatus Command

Upon receipt, this command SHALL display the active status of the input list on screen.

6.9.7.3. HideInputStatus Command

Upon receipt, this command SHALL hide the input list from the screen.

6.9.7.4. RenameInput Command

Upon receipt, this command SHALL rename the input at a specific index in the Input List.

Updates to the input name SHALL appear in the device's settings menus.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**Index**	uint8	all			M
1	**Name**	string	all			M

6.10. Media Playback Cluster

This cluster provides an interface for controlling Media Playback (PLAY, PAUSE, etc) on a media device such as a TV, Set-top Box, or Smart Speaker.

This cluster server would be supported on Video Player devices or endpoints that provide media playback, such as a Content App. This cluster provides an interface for controlling Media Playback.

6.10.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Revision	Description
1	Initial Release
2	Added Text/Audio tracks support, Audio-while-advancing (AA) feature, StateChanged event

6.10.2. Classification

Hierarchy	Role	Scope	PICS Code
Base	Application	Endpoint	MEDIAPLAYBACK

6.10.3. Cluster ID

ID	Name
0x0506	Media Playback

6.10.4. Features

This cluster SHALL support the FeatureMap bitmap attribute as defined below.

Bit	Code	Feature	Summary
0	**AS**	AdvancedSeek	Advanced media seeking
1	**VS**	VariableSpeed	Variable speed playback
2	**TT**	TextTracks	Text Tracks
3	**AT**	AudioTracks	Audio Tracks
4	**AA**	AudioAdvance	Can play audio during fast and slow playback speeds

6.10.4.1. AdvancedSeek Feature

This feature provides access to the time offset location within current playback media and allows for jumping to a specific location using time offsets. This enables clients to implement more advanced media seeking behavior in their user interface, for instance a "seek bar".

6.10.4.2. VariableSpeed Feature

This feature is for a device which supports variable speed playback on media that supports it.

6.10.4.3. TextTracks Feature

This feature is for a device or app that supports Text Tracks.

6.10.4.4. AudioTracks Feature

This feature is for a device or app that supports Audio Tracks.

6.10.4.5. AudioAdvance Feature

This feature is for a device or app that supports playing audio during fast and slow advance and rewind (e.g., while playback speed is not 1). A device that supports this feature MAY only support playing audio during certain speeds.

A cluster implementing AA SHALL implement AS.

6.10.5. Data Types

6.10.5.1. PlaybackStateEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**Playing**	Media is currently playing (includes FF and REW)	M
1	**Paused**	Media is currently paused	M
2	**NotPlaying**	Media is not currently playing	M
3	**Buffering**	Media is not currently buffering and playback will start when buffer has been filled	M

6.10.5.2. StatusEnum Type

Status Data Type is derived from enum8.

Value	Name	Summary	Conformance
0	**Success**	Succeeded	M

Value	Name	Summary	Conformance
1	**InvalidStateForCommand**	Requested playback command is invalid in the current playback state.	M
2	**NotAllowed**	Requested playback command is not allowed in the current playback state. For example, attempting to fast-forward during a commercial might return NotAllowed.	M
3	**NotActive**	This endpoint is not active for playback.	M
4	**SpeedOutOfRange**	The FastForward or Rewind Command was issued but the media is already playing back at the fastest speed supported by the server in the respective direction.	VS
5	**SeekOutOfRange**	The Seek Command was issued with a value of position outside of the allowed seek range of the media.	AS

6.10.5.3. CharacteristicEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**ForcedSubtitles**	Textual information meant for display when no other text representation is selected. It is used to clarify dialogue, alternate languages, texted graphics or location/person IDs that are not otherwise covered in the dubbed/localized audio.	M

Value	Name	Summary	Conformance
1	**DescribesVideo**	Textual or audio media component containing a textual description (intended for audio synthesis) or an audio description describing a visual component	M
2	**EasyToRead**	Simplified or reduced captions as specified in [United States Code Title 47 CFR 79.103(c)(9)].	M
3	**FrameBased**	A media characteristic that indicates that a track selection option includes frame-based content.	M
4	**MainProgram**	Main media component(s) which is/are intended for presentation if no other information is provided	M
5	**OriginalContent**	A media characteristic that indicates that a track or media selection option contains original content.	M
6	**VoiceOverTranslation**	A media characteristic that indicates that a track or media selection option contains a language translation and verbal interpretation of spoken dialog.	M
7	**Caption**	Textual media component containing transcriptions of spoken dialog and auditory cues such as sound effects and music for the hearing impaired.	M
8	**Subtitle**	Textual transcriptions of spoken dialog.	M

Value	Name	Summary	Conformance
9	**Alternate**	Textual media component containing transcriptions of spoken dialog and auditory cues such as sound effects and music for the hearing impaired.	M
10	**Supplementary**	Media content component that is supplementary to a media content component of a different media component type.	M
11	**Commentary**	Experience that contains a commentary (e.g. director's commentary) (typically audio)	M
12	**DubbedTranslation**	Experience that contains an element that is presented in a different language from the original (e.g. dubbed audio, translated captions)	M
13	**Description**	Textual or audio media component containing a textual description (intended for audio synthesis) or an audio description describing a visual component	M
14	**Metadata**	Media component containing information intended to be processed by application specific elements.	M
15	**EnhancedAudioIntelligibility**	Experience containing an element for improved intelligibility of the dialogue.	M

Value	Name	Summary	Conformance
16	**Emergency**	Experience that provides information, about a current emergency, that is intended to enable the protection of life, health, safety, and property, and may also include critical details regarding the emergency and how to respond to the emergency.	M
17	**Karaoke**	Textual representation of a songs' lyrics, usually in the same language as the associated song as specified in [SMPTE ST 2067-2].	M

6.10.5.4. PlaybackPositionStruct Type

This structure defines a playback position within a media stream being played.

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0	**UpdatedAt**	epoch-us	all				M
1	**Position**	uint64	all	X			M

6.10.5.4.1. UpdatedAt Field

This field SHALL indicate the time when the position was last updated.

6.10.5.4.2. Position Field

This field SHALL indicate the associated discrete position within the media stream, in milliseconds from the beginning of the stream, being associated with the time indicated by the UpdatedAt field. The Position SHALL not be greater than the duration of the media if duration is specified. The Position SHALL not be greater than the time difference between current time and start time of the media when start time is specified.

A value of null SHALL indicate that playback position is not applicable for the current state of the media playback (For example : Live media with no known duration and where seek is not supported).

6.10.5.5. TrackStruct Type

This structure defines a uniquely identifiable Text Track or Audio Track.

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0	**ID**	string	max 32				M
1	**TrackAttributes**	TrackAttributesStruct	all				M

6.10.5.5.1. ID Field

This field SHALL indicate the Identifier for the Track which is unique within the Track catalog. The Track catalog contains all the Text/Audio tracks corresponding to the main media content.

6.10.5.5.2. TrackAttributes Field

This field SHALL indicate the Attributes associated to the Track, like languageCode.

6.10.5.6. TrackAttributesStruct Type

This structure includes the attributes associated with a Text/Audio Track

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0	**LanguageCode**	string	max 32				M
1	**Characteristics**	list[CharacteristicEnum]	all	X	null		O
2	**DisplayName**	string	max 256	X	null		O

6.10.5.6.1. LanguageCode Field

The value is a String containing one of the standard Tags for Identifying Languages RFC 5646, which identifies the primary language used in the Track.

6.10.5.6.2. Characteristics Field

This is a list of enumerated CharacteristicEnum values that indicate a purpose, trait or feature associated with the Track. A value of null SHALL indicate that there are no Characteristics corresponding to the Track.

6.10.5.6.3. DisplayName Field

The value is a String containing a user displayable name for the Track. A value of null SHALL indicate that there is no DisplayName corresponding to the Track.

6.10.6. Attributes

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0000	**CurrentState**	PlaybackStateEnum	desc			R V	M
0x0001	**StartTime**	epoch-us	desc	X	null	R V	AS
0x0002	**Duration**	uint64	desc	X	null	R V	AS
0x0003	**SampledPosition**	PlaybackPositionStruct	desc	X	null	R V	AS
0x0004	**PlaybackSpeed**	single	desc		0	R V	AS
0x0005	**SeekRangeEnd**	uint64	desc	X	null	R V	AS
0x0006	**SeekRangeStart**	uint64	desc	X	null	R V	AS
0x0007	**ActiveAudioTrack**	TrackStruct	desc	X	null	R V	AT
0x0008	**AvailableAudioTracks**	list[TrackStruct]	desc	X	null	R V	AT
0x0009	**ActiveTextTrack**	TrackStruct	desc	X	null	R V	TT
0x000A	**AvailableTextTracks**	list[TrackStruct]	desc	X	null	R V	TT

6.10.6.1. CurrentState Attribute

This attribute SHALL indicate the current playback state of media.

During fast-forward, rewind, and other seek operations; this attribute SHALL be set to PLAYING.

6.10.6.2. StartTime Attribute

This attribute SHALL indicate the start time of the media, in case the media has a fixed start time (for example, live stream or television broadcast), or null when start time does not apply to the current media (for example, video-on-demand). This time is a UTC time. The client needs to handle conversion to local time, as required, taking in account time zone and possible local DST offset.

6.10.6.3. Duration Attribute

This attribute SHALL indicate the duration, in milliseconds, of the current media being played back

or null when duration is not applicable (for example, in live streaming content with no known duration). This attribute SHALL never be 0.

6.10.6.4. SampledPosition Attribute

This attribute SHALL indicate the position of playback (Position field) at the time (UpdateAt field) specified in the attribute. The client MAY use the SampledPosition attribute to compute the current position within the media stream based on the PlaybackSpeed, PlaybackPositionStruct.UpdatedAt and PlaybackPositionStruct.Position fields. To enable this, the SampledPosition attribute SHALL be updated whenever a change in either the playback speed or the playback position is triggered outside the normal playback of the media. The events which MAY cause this to happen include:

- Starting or resumption of playback
- Seeking
- Skipping forward or backward
- Fast-forwarding or rewinding
- Updating of playback speed as a result of explicit request, or as a result of buffering events

6.10.6.5. PlaybackSpeed Attribute

This attribute SHALL indicate the speed at which the current media is being played. The new PlaybackSpeed SHALL be reflected in this attribute whenever any of the following occurs:

- Starting of playback
- Resuming of playback
- Fast-forwarding
- Rewinding

The PlaybackSpeed SHALL reflect the ratio of time elapsed in the media to the actual time taken for the playback assuming no changes to media playback (for example buffering events or requests to pause/rewind/forward).

- A value for PlaybackSpeed of 1 SHALL indicate normal playback where, for example, playback for 1 second causes the media to advance by 1 second within the duration of the media.
- A value for PlaybackSpeed which is greater than 0 SHALL indicate that as playback is happening the media is currently advancing in time within the duration of the media.
- A value for PlaybackSpeed which is less than 0 SHALL indicate that as playback is happening the media is currently going back in time within the duration of the media.
- A value for PlaybackSpeed of 0 SHALL indicate that the media is currently not playing back. When the CurrentState attribute has the value of PAUSED, NOT_PLAYING or BUFFERING, the PlaybackSpeed SHALL be set to 0 to reflect that the media is not playing.

Following examples illustrate the PlaybackSpeed attribute values in various conditions.

Seconds of Media Played	Actual Time Taken in Seconds	Direction of playback	PlaybackSpeed
2	2	Forward	1.0
2	1	Forward	2.0
1	2	Forward	0.5
2	2	Reverse	-1.0
2	1	Reverse	-2.0
1	2	Reverse	-0.5

6.10.6.6. SeekRangeStart Attribute

This attribute SHALL indicate the earliest valid position to which a client MAY seek back, in milliseconds from start of the media. A value of Nas SHALL indicate that seeking backwards is not allowed.

6.10.6.7. SeekRangeEnd Attribute

This attribute SHALL indicate the furthest forward valid position to which a client MAY seek forward, in milliseconds from the start of the media. When the media has an associated StartTime, a value of null SHALL indicate that a seek forward is valid only until the current time within the media, using a position computed from the difference between the current time offset and StartTime, in milliseconds from start of the media, truncating fractional milliseconds towards 0. A value of Nas when StartTime is not specified SHALL indicate that seeking forward is not allowed.

6.10.6.8. ActiveAudioTrack Attribute

ActiveTrack refers to the Audio track currently set and being used for the streaming media. A value of null SHALL indicate that no Audio Track corresponding to the current media is currently being played.

6.10.6.9. AvailableAudioTracks Attribute

AvailableAudioTracks refers to the list of Audio tracks available for the current title being played. A value of null SHALL indicate that no Audio Tracks corresponding to the current media are selectable by the client.

6.10.6.10. ActiveTextTrack Attribute

ActiveTrack refers to the Text track currently set and being used for the streaming media. This can be nil. A value of null SHALL indicate that no Text Track corresponding to the current media is currently being displayed.

6.10.6.11. AvailableTextTracks Attribute

AvailableTextTracks refers to the list of Text tracks available for the current title being played. This can be an empty list. A value of null SHALL indicate that no Text Tracks corresponding to the current media are selectable by the client.

6.10.7. Commands

ID	Name	Direction	Response	Access	Conformance
0x00	**Play**	client ⇒ server	PlaybackResponse	O	M
0x01	**Pause**	client ⇒ server	PlaybackResponse	O	M
0x02	**Stop**	client ⇒ server	PlaybackResponse	O	M
0x03	**StartOver**	client ⇒ server	PlaybackResponse	O	O
0x04	**Previous**	client ⇒ server	PlaybackResponse	O	O
0x05	**Next**	client ⇒ server	PlaybackResponse	O	O
0x06	**Rewind**	client ⇒ server	PlaybackResponse	O	VS
0x07	**FastForward**	client ⇒ server	PlaybackResponse	O	VS
0x08	**SkipForward**	client ⇒ server	PlaybackResponse	O	O
0x09	**SkipBackward**	client ⇒ server	PlaybackResponse	O	O
0x0A	**PlaybackResponse**	client ⇐ server	N		M
0x0B	**Seek**	client ⇒ server	PlaybackResponse	O	AS
0x0C	**ActivateAudioTrack**	client ⇒ server	Y	O	AT
0x0D	**ActivateTextTrack**	client ⇒ server	Y	O	TT
0x0E	**DeactivateTextTrack**	client ⇒ server	Y	O	TT

6.10.7.1. Play Command

Upon receipt, this SHALL play media. If content is currently in a FastForward or Rewind state. Play SHALL return media to normal playback speed.

6.10.7.2. Pause Command

Upon receipt, this SHALL pause playback of the media.

6.10.7.3. Stop Command

Upon receipt, this SHALL stop playback of the media. User-visible outcome is context-specific. This MAY navigate the user back to the location from where the media was originally launched.

6.10.7.4. StartOver Command

Upon receipt, this SHALL Start Over with the current media playback item.

6.10.7.5. Previous Command

Upon receipt, this SHALL cause the handler to be invoked for "Previous". User experience is context-specific. This will often Go back to the previous media playback item.

6.10.7.6. Next Command

Upon receipt, this SHALL cause the handler to be invoked for "Next". User experience is context-specific. This will often Go forward to the next media playback item.

6.10.7.7. Rewind Command

Upon receipt, this SHALL start playback of the media backward in case the media is currently playing in the forward direction or is not playing. If the playback is already happening in the backwards direction receipt of this command SHALL increase the speed of the media playback backwards.

Different "rewind" speeds MAY be reflected on the media playback device based upon the number of sequential calls to this function and the capability of the device. This is to avoid needing to define every speed (multiple fast, slow motion, etc). If the PlaybackSpeed attribute is supported it SHALL be updated to reflect the new speed of playback. If the playback speed cannot be changed for the media being played(for example, in live streaming content not supporting seek), the status of NOT_ALLOWED SHALL be returned. If the playback speed has reached the maximum supported speed for media playing backwards, the status of SPEED_OUT_OF_RANGE SHALL be returned.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**AudioAdvanceUnmuted**	bool	all		false	AA

6.10.7.7.1. AudioAdvanceUnmuted Field

This field SHALL indicate whether audio should be unmuted by the player during rewind.

A value of true does not guarantee that audio can be heard by the user since the speaker may be muted, turned down to a low level and/or unplugged.

6.10.7.8. FastForward Command

Upon receipt, this SHALL start playback of the media in the forward direction in case the media is

currently playing in the backward direction or is not playing. If the playback is already happening in the forward direction receipt of this command SHALL increase the speed of the media playback.

Different "fast-forward" speeds MAY be reflected on the media playback device based upon the number of sequential calls to this function and the capability of the device. This is to avoid needing to define every speed (multiple fast, slow motion, etc). If the PlaybackSpeed attribute is supported it SHALL be updated to reflect the new speed of playback. If the playback speed cannot be changed for the media being played(for example, in live streaming content not supporting seek), the status of NOT_ALLOWED SHALL be returned. If the playback speed has reached the maximum supported speed for media playing forward, the status of SPEED_OUT_OF_RANGE SHALL be returned.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**AudioAdvanceUnmuted**	bool	all		false	AA

6.10.7.8.1. AudioAdvanceUnmuted Field

This field SHALL indicate whether audio should be unmuted by the player during fast forward.

A value of true does not guarantee that audio can be heard by the user since the speaker may be muted, turned down to a low level and/or unplugged.

6.10.7.9. SkipForward Command

Upon receipt, this SHALL Skip forward in the media by the given number of milliseconds, using the data as follows:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**DeltaPositionMilliseconds**	uint64	all			M

6.10.7.9.1. DeltaPositionMilliseconds Field

This field SHALL indicate the duration of the time span to skip forward in the media, in milliseconds. In case the resulting position falls in the middle of a frame, the server SHALL set the position to the beginning of that frame and set the SampledPosition attribute on the cluster accordingly. If the resultant position falls beyond the furthest valid position in the media the client MAY seek forward to, the position should be set to that furthest valid position. If the SampledPosition attribute is supported it SHALL be updated on the cluster accordingly.

6.10.7.10. SkipBackward Command

Upon receipt, this SHALL Skip backward in the media by the given number of milliseconds, using the data as follows:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**DeltaPositionMilliseconds**	uint64	all			M

6.10.7.10.1. DeltaPositionMilliseconds Field

This field SHALL indicate the duration of the time span to skip backward in the media, in milliseconds. In case the resulting position falls in the middle of a frame, the server SHALL set the position to the beginning of that frame and set the SampledPosition attribute on the cluster accordingly. If the resultant position falls before the earliest valid position to which a client MAY seek back to, the position should be set to that earliest valid position. If the SampledPosition attribute is supported it SHALL be updated on the cluster accordingly.

6.10.7.11. Seek Command

Upon receipt, this SHALL change the playback position in the media to the given position using data as follows:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**Position**	uint64	all			M

6.10.7.11.1. Position Field

This field SHALL indicate the position (in milliseconds) in the media to seek to. In case the position falls in the middle of a frame, the server SHALL set the position to the beginning of that frame and set the SampledPosition attribute on the cluster accordingly. If the position falls before the earliest valid position or beyond the furthest valid position to which a client MAY seek back or forward to respectively, the status of SEEK_OUT_OF_RANGE SHALL be returned and no change SHALL be made to the position of the playback.

6.10.7.12. PlaybackResponse Command

This command SHALL be generated in response to various Playback Commands. The data for this command SHALL be as follows:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**Status**	StatusEnum	desc			M
1	**Data**	string	any			O

6.10.7.12.1. Status Field

This field SHALL indicate the status of the command which resulted in this response.

6.10.7.12.2. Data Field

This field SHALL indicate Optional app-specific data.

6.10.7.13. ActivateAudioTrack Command

Upon receipt, the server SHALL set the active Audio Track to the one identified by the TrackID in the Track catalog for the streaming media. If the TrackID does not exist in the Track catalog, OR does not correspond to the streaming media OR no media is being streamed at the time of receipt of this command, the server will return an error status of INVALID_ARGUMENT.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**TrackID**	string	max 32			M
1	**AudioOutputIndex**	uint8	all	X		AT

6.10.7.13.1. TrackID Field

This field SHALL indicate the Audio Track to activate.

6.10.7.13.2. AudioOutputIndex Field

This value is the index field of the OutputInfoStruct from the OutputList attribute (from the AudioOutput cluster) and indicates which audio output the Audio Track should be played on. This field is absent for Text Tracks and only present for Audio Tracks. A value of null SHALL indicate that the server can choose the audio output(s) to play the Audio Track on.

6.10.7.14. ActivateTextTrack Command

Upon receipt, the server SHALL set the active Text Track to the one identified by the TrackID in the Track catalog for the streaming media. If the TrackID does not exist in the Track catalog, OR does not correspond to the streaming media OR no media is being streamed at the time of receipt of this command, the server SHALL return an error status of INVALID_ARGUMENT.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**TrackID**	string	max 32			M

6.10.7.14.1. TrackID Field

This field SHALL indicate the Text Track to activate.

6.10.7.15. DeactivateTextTrack Command

If a Text Track is active (i.e. being displayed), upon receipt of this command, the server SHALL stop displaying it.

6.10.8. Events

ID	Name	Priority	Access	Conformance
0x00	**StateChanged**	INFO	V	O

6.10.8.1. StateChanged Event

If supported, this event SHALL be generated when there is a change in any of the supported attributes of the Media Playback cluster. This event SHALL have the following data fields:

ID	Name	Type	Constraint	Quality	Default	Conformance
0x0000	**CurrentState**	PlaybackStateEnum	desc			M
0x0001	**StartTime**	epoch-us	desc			AS
0x0002	**Duration**	uint64	desc			AS
0x0003	**SampledPosition**	PlaybackPositionStruct	desc			AS
0x0004	**PlaybackSpeed**	single	desc			AS
0x0005	**SeekRangeEnd**	uint64	desc			AS
0x0006	**SeekRangeStart**	uint64	desc			AS
0x0007	**Data**	octstr	max 900			O
0x0008	**AudioAdvanceUnmuted**	bool	desc		false	AA

6.10.8.1.1. CurrentState Field

This field SHALL indicate the updated playback state as defined by the CurrentState attribute, and has the same constraint as that attribute.

6.10.8.1.2. StartTime Field

This field SHALL indicate the updated start time as defined by the StartTime attribute, and has the same constraint as that attribute.

6.10.8.1.3. Duration Field

This field SHALL indicate the updated duration as defined by the Duration attribute, and has the same constraint as that attribute.

6.10.8.1.4. SampledPosition Field

This field SHALL indicate the updated position of playback as defined by the SampledPosition attribute, and has the same constraint as that attribute.

6.10.8.1.5. PlaybackSpeed Field

This field SHALL indicate the updated speed at which the current media is being played as defined by the PlaybackSpeed attribute, and has the same constraint as that attribute.

6.10.8.1.6. SeekRangeStart Field

This field SHALL indicate the updated start of the seek range start as defined by the SeekRangeStart attribute, and has the same constraint as that attribute.

6.10.8.1.7. SeekRangeEnd Field

This field SHALL indicate the updated start of the seek range end as defined by the SeekRangeEnd attribute, and has the same constraint as that attribute.

6.10.8.1.8. Data Field

This field SHALL indicate Optional app-specific data.

6.10.8.1.9. AudioAdvanceUnmuted Field

This field SHALL indicate whether audio is unmuted by the player due to a FF or REW command. This field is only meaningful when the PlaybackSpeed is present and not equal to 0 (paused) or 1 (normal playback). Typically the value will be false (muted), however, some players will play audio during certain fast forward and rewind speeds, and in these cases, the value will be true (not muted).

A value of true does not guarantee that audio can be heard by the user since the speaker may be muted, turned down to a low level and/or unplugged.

6.11. Target Navigator Cluster

This cluster provides an interface for UX navigation within a set of targets on a device or endpoint.

This cluster would be supported on Video Player devices or devices with navigable user interfaces. This cluster would also be supported on endpoints with navigable user interfaces such as a Content App. It supports listing a set of navigation targets, tracking and changing the current target.

The cluster server for Target Navigator is implemented by endpoints on a device that support UX navigation.

When this cluster is implemented for a Content App endpoint, the Video Player device containing the endpoint SHALL launch the Content App when a client invokes the NavigateTarget command.

6.11.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Revision	Description
1	Initial Release
2	Add TargetUpdated event

6.11.2. Classification

Hierarchy	Role	Scope	PICS Code
Base	Application	Endpoint	TGTNAV

6.11.3. Cluster ID

ID	Name
0x0505	Target Navigator

6.11.4. Data Types

6.11.4.1. StatusEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**Success**	Command succeeded	M
1	**TargetNotFound**	Requested target was not found in the TargetList	M
2	**NotAllowed**	Target request is not allowed in current state.	M

6.11.4.2. TargetInfoStruct Type

This indicates an object describing the navigable target.

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0	**Identifier**	uint8	max 254				M
1	**Name**	string					M

6.11.4.2.1. Identifier Field

This field SHALL contain an unique id within the TargetList.

6.11.4.2.2. Name Field

This field SHALL contain a name string for the TargetInfoStruct.

6.11.5. Attributes

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0000	**TargetList**	list[TargetInfoStruct]				R V	M
0x0001	**CurrentTarget**	uint8	desc		0xFF	R V	O

6.11.5.1. TargetList Attribute

This attribute SHALL represent a list of targets that can be navigated to within the experience presented to the user by the Endpoint (Video Player or Content App). The list SHALL not contain any entries with the same Identifier in the TargetInfoStruct object.

6.11.5.2. CurrentTarget Attribute

This attribute SHALL represent the Identifier for the target which is currently in foreground on the corresponding Endpoint (Video Player or Content App), or 0xFF to indicate that no target is in the foreground.

When not 0xFF, the CurrentTarget SHALL be an Identifier value contained within one of the TargetInfoStruct objects in the TargetList attribute.

6.11.6. Commands

ID	Name	Direction	Response	Access	Conformance
0x00	**NavigateTarget**	client ⇒ server	NavigateTargetResponse	O	M
0x01	**NavigateTargetResponse**	client ⇐ server	N		M

6.11.6.1. NavigateTarget Command

Upon receipt, this SHALL navigation the UX to the target identified.

ID	Field	Type	Constraint	Quality	Default	Conformance
0	**Target**	uint8	all			M

ID	Field	Type	Constraint	Quality	Default	Conformance
1	**Data**	string			MS	O

6.11.6.1.1. Target Field

This field SHALL indicate the Identifier for the target for UX navigation. The Target SHALL be an Identifier value contained within one of the TargetInfoStruct objects in the TargetList attribute.

6.11.6.1.2. Data Field

This field SHALL indicate Optional app-specific data.

6.11.6.2. NavigateTargetResponse Command

This command SHALL be generated in response to NavigateTarget command.

ID	Field	Type	Constraint	Quality	Default	Conformance
0	**Status**	StatusEnum	all			M
1	**Data**	string	any		MS	O

6.11.6.2.1. Status Field

This field SHALL indicate the of the command.

6.11.6.2.2. Data Field

This field SHALL indicate Optional app-specific data.

6.11.7. Events

ID	Name	Priority	Access	Conformance
0x00	**TargetUpdated**	INFO	V	O

6.11.7.1. TargetUpdated Event

This event SHALL be generated when there is a change in either the active target or the list of available targets or both. This event SHALL have the following data fields:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**TargetList**	list[TargetInfoStruct]				O
1	**CurrentTarget**	uint8	desc		0xFF	O
2	**Data**	octstr	max 900			O

6.11.7.2. TargetList Field

This field SHALL indicate the updated target list as defined by the TargetList attribute if there is a change in the list of targets. Otherwise this field can be omitted from the event.

6.11.7.3. CurrentTarget Field

This field SHALL indicate the updated target that is in foreground as defined by the CurrentTarget attribute if supported (see CurrentTarget attribute for constraints).

6.11.7.4. Data Field

This field SHALL indicate Optional app-specific data.

6.12. Content App Observer Cluster

This cluster provides an interface for sending targeted commands to an Observer of a Content App on a Video Player device such as a Streaming Media Player, Smart TV or Smart Screen.

The cluster server for Content App Observer is implemented by an endpoint that communicates with a Content App, such as a Casting Video Client.

The cluster client for Content App Observer is implemented by a Content App endpoint.

A Content App is informed of the NodeId of an Observer when a binding is set on the Content App. For a Content App Platform, the binding is set by the platform when a CastingVideoClient is granted access to the Content App, and the CastingVideoClient supports the Content App Observer cluster. The Content App can then send the ContentAppMessage to the Observer (server cluster), and the Observer responds with a ContentAppMessageResponse.

The Data and EncodingHint fields of the ContentAppMessage and ContentAppMessageResponse contain content app-specific values, the format and interpretation of which is defined by the Content App vendor, analogous to the custom message features offered by other popular casting protocols. Standardized cluster and commands are used here rather than manufacturer-specific cluster and commands because of the role that the Content App Platform plays in creating the ACLs and Bindings on both sides of the communication between the Content App Observer endpoint and the Content App endpoint.

By using standard cluster and commands:

1. The Content App Platform is able to easily determine that a binding is needed on the Content App endpoint because it can recognize the Content App Observer cluster implemented by a client node.
2. The Content App Platform is able to easily identify commands that are allowed to be sent by the Content App to a client node because those commands use the Content App Observer cluster.
3. The Content App is able to easily determine that a node supports the Content App Observer cluster because it has received a binding which specifies the Content App Observer cluster.
4. The Casting Video Client is able to support a single cluster for receiving commands from any Content App and does not need to explicitly list every Content App it understands.

A Content App Observer SHOULD ignore the Data and EncodingHint field values in commands from a Content App it does not recognize. A Content App SHOULD ignore the Data field values in responses when the EncodingHint value is blank or not recognized.

6.12.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Revision	Description
1	Initial Release

6.12.2. Classification

Hierarchy	Role	Scope	PICS Code
Base	Application	Endpoint	APPOBSERVER

6.12.3. Cluster ID

ID	Name
0x0510	Content App Observer

6.12.4. Data Types

6.12.4.1. StatusEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**Success**	Command succeeded	M
1	**UnexpectedData**	Data field in command was not understood by the Observer	M

6.12.5. Commands

ID	Name	Direction	Response	Access	Conformance
0x00	**ContentAppMessage**	client ⇒ server	ContentAppMessageResponse	O	M
0x01	**ContentAppMessageResponse**	client ⇐ server	N		M

6.12.5.1. ContentAppMessage Command

Upon receipt, the data field MAY be parsed and interpreted. Message encoding is specific to the Content App. A Content App MAY when possible read attributes from the Basic Information Cluster on the Observer and use this to determine the Message encoding.

This command returns a ContentAppMessage Response.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**Data**	string	max 500			M
1	**EncodingHint**	string	max 100			O

6.12.5.1.1. Data Field

This field SHALL indicate content app-specific data.

6.12.5.1.2. EncodingHint Field

This optional field SHALL indicate a content app-specific hint to the encoding of the data.

6.12.5.2. ContentAppMessageResponse Command

This command SHALL be generated in response to ContentAppMessage command.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**Status**	StatusEnum	all			M
1	**Data**	string	max 500			O
2	**EncodingHint**	string	max 100			O

6.12.5.2.1. Status Field

This field SHALL indicate the status of the command which resulted in this response.

6.12.5.2.2. Data Field

This optional field SHALL indicate content app-specific data.

6.12.5.2.3. EncodingHint Field

This optional field SHALL indicate a content app-specific hint to the encoding of the data.

6.13. Content Control Cluster

This cluster is used for managing the content control (including "parental control") settings on a

media device such as a TV, or Set-top Box.

This cluster allows to configure content control settings by clients with the Management privilege. It is responsibility of the end product to enforce appropriate right access (for example, to prevent a child from disabling this feature).

NOTE	Support for Content Control cluster is provisional.

6.13.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Revision	Description
1	Initial Release

6.13.2. Classification

Hierarchy	Role	Scope	PICS Code
Base	Application	Endpoint	CONCON

6.13.3. Cluster ID

ID	Name	Conformance
0x050F	Content Control	P

6.13.4. Features

This cluster SHALL support the FeatureMap bitmap attribute as defined below.

Bit	Code	Feature	Summary
0	ST	ScreenTime	Supports managing screen time limits.
1	PM	PINManagement	Supports managing a PIN code which is used for restricting access to configuration of this feature.
2	BU	BlockUnrated	Supports managing content controls for unrated content.

Bit	Code	Feature	Summary
3	OCR	OnDemandContentRating	Supports managing content controls based upon rating threshold for on demand content.
4	SCR	ScheduledContentRating	Supports managing content controls based upon rating threshold for scheduled content.
5	BC	BlockChannels	Supports managing a set of channels that are prohibited.
6	BA	BlockApplications	Supports managing a set of applications that are prohibited.
7	BTW	BlockContentTimeWindow	Supports managing content controls based upon setting time window in which all contents and applications SHALL be blocked.

6.13.5. Data Types

6.13.5.1. RatingNameStruct Type

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0	**RatingName**	string	max 8				M
1	**RatingNameDesc**	string	max 64				O

6.13.5.1.1. RatingName Field

This field SHALL indicate the name of the rating level of the applied rating system. The applied rating system is dependent upon the region or country where the Node has been provisioned, and may vary from one country to another.

6.13.5.1.2. RatingNameDesc Field

This field SHALL specify a human readable (displayable) description for RatingName.

6.13.5.2. BlockChannelStruct Type

[options="header",valign="middle"]b

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0	**BlockChannelIndex**	uint16	all	X			M
1	**MajorNumber**	uint16	all				M
2	**MinorNumber**	uint16	all				M
3	**Identifier**	string	all				O

6.13.5.2.1. BlockChannelIndex Field

This field SHALL indicate a unique index value for a blocked channel. This value may be used to indicate one selected channel which will be removed from BlockChannelList attribute.

6.13.5.2.2. MajorNumber Field

This field SHALL indicate the channel major number value (for example, using ATSC format). When the channel number is expressed as a string, such as "13.1" or "256", the major number would be 13 or 256, respectively. This field is required but SHALL be set to 0 for channels such as over-the-top channels that are not represented by a major or minor number.

6.13.5.2.3. MinorNumber Field

This field SHALL indicate the channel minor number value (for example, using ATSC format). When the channel number is expressed as a string, such as "13.1" or "256", the minor number would be 1 or 0, respectively. This field is required but SHALL be set to 0 for channels such as over-the-top channels that are not represented by a major or minor number.

6.13.5.2.4. Identifier

This field SHALL indicate the unique identifier for a specific channel. This field is optional, but SHOULD be provided when MajorNumber and MinorNumber are not available.

6.13.5.3. AppInfoStruct Type

ID	**Name**	**Type**	**Constraint**	**Quality**	**Default**	**Access**	**Conformance**
0	**CatalogVendorID**	uint16	all				M
1	**ApplicationID**	string	all				M

6.13.5.3.1. CatalogVendorID Field

This field SHALL indicate the CSA-issued vendor ID for the catalog. The DIAL registry SHALL use value 0x0000.

Content App Platform providers will have their own catalog vendor ID (set to their own Vendor ID) and will assign an ApplicationID to each Content App.

6.13.5.3.2. ApplicationID field

This field SHALL indicate the application identifier, expressed as a string, such as "PruneVideo" or "Company X". This field SHALL be unique within a catalog.

6.13.5.4. TimeWindowStruct Type

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0	**TimeWindowIndex**	uint16	all	X			M
1	**DayOfWeek**	DayOfWeekBitmap	desc				M
2	**TimePeriod**	list[TimePeriodStruct]	desc				M

6.13.5.4.1. TimeWindowIndex Field

This field SHALL indicate a unique index of a specific time window. This value may be used to indicate a selected time window which will be removed from the BlockContentTimeWindow attribute.

6.13.5.4.2. DayOfWeek Field

This field SHALL indicate a day of week.

6.13.5.4.3. TimePeriod Field

This field SHALL indicate one or more discrete time periods.

6.13.5.5. DayOfWeekBitmap type

This data type is derived from enum8.

bit	Name	Summary
0	**Sunday**	Sunday
1	**Monday**	Monday
2	**Tuesday**	Tuesday
3	**Wednesday**	Wednesday

bit	Name	Summary
4	**Thursday**	Thursday
5	**Friday**	Friday
6	**Saturday**	Saturday

6.13.5.6. TimePeriodStruct type

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0	**StartHour**	uint8	0 to 23				M
1	**StartMinute**	uint8	0 to 59				M
2	**EndHour**	uint8	0 to 23				M
3	**EndMinute**	uint8	0 to 59				M

6.13.5.6.1. StartHour Field

This field SHALL indicate the starting hour.

6.13.5.6.2. StartMinute Field

This field SHALL indicate the starting minute.

6.13.5.6.3. EndHour Field

This field SHALL indicate the ending hour. EndHour SHALL be equal to or greater than StartHour

6.13.5.6.4. EndMinute Field

This field SHALL indicate the ending minute. If EndHour is equal to StartHour then EndMinute SHALL be greater than StartMinute. If the EndHour is equal to 23 and the EndMinute is equal to 59, all contents SHALL be blocked until 23:59:59.

6.13.6. Status Codes

6.13.6.1. StatusCodeEnum Type

This data type is derived from enum8.

Value	Name	Summary
0x02	**InvalidPINCode**	Provided PIN Code does not match the current PIN code.
0x03	**InvalidRating**	Provided Rating is out of scope of the corresponding Rating list.

Value	Name	Summary
0x04	**InvalidChannel**	Provided Channel(s) is invalid.
0x05	**ChannelAlreadyExist**	Provided Channel(s) already exists.
0x06	**ChannelNotExist**	Provided Channel(s) doesn't exist in BlockChannelList attribute.
0x07	**UnidentifiableApplication**	Provided Application(s) is not identified.
0x08	**ApplicationAlreadyExist**	Provided Application(s) already exists.
0x09	**ApplicationNotExist**	Provided Application(s) doesn't exist in BlockApplicationList attribute.
0x0A	**TimeWindowAlreadyExist**	Provided time Window already exists in BlockContentTimeWindow attribute.
0x0B	**TimeWindowNotExist**	Provided time window doesn't exist in BlockContentTimeWindow attribute.

6.13.7. Attributes

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0000	**Enabled**	bool	all			R V	M
0x0001	**OnDemandRatings**	list[RatingNameStruct]	all			R V	OCR
0x0002	**OnDemandRatingThreshold**	string	max 8			R V	OCR
0x0003	**ScheduledContentRatings**	list[RatingNameStruct]	all			R V	SCR
0x0004	**ScheduledContentRatingThreshold**	string	max 8			R V	SCR

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0005	**ScreenDailyTime**	elapsed-s	max 86400			R V	ST
0x0006	**RemainingScreenTime**	elapsed-s	max 86400			R V	ST
0x0007	**BlockUnrated**	bool	all			R V	BU
0x0008	**BlockChannelList**	list[BlockChannelStruct]	all			R V	BC
0x0009	**BlockApplicationList**	list[AppInfoStruct]	all			R V	BA
0x000A	**BlockContentTimeWindow**	list[TimeWindowStruct]	max 7			R V	BTW

6.13.7.1. Enabled Attribute

This attribute SHALL indicate whether the Content Control feature implemented on a media device is turned off (FALSE) or turned on (TRUE).

6.13.7.2. OnDemandRatings Attribute

This attribute SHALL provide the collection of ratings that are currently valid for this media device. The items should honor the metadata of the on-demand content (e.g. Movie) rating system for one country or region where the media device has been provisioned. For example, for the MPAA system, RatingName may be one value out of "G", "PG", "PG-13", "R", "NC-17".

The media device SHALL have a way to determine which rating system applies for the on-demand content and then populate this attribute. For example, it can do it through examining the Location attribute in the Basic Information cluster, and then determining which rating system applies.

The ratings in this collection SHALL be in order from a rating for the youngest viewers to the one for the oldest viewers. Each rating in the list SHALL be unique.

6.13.7.3. OnDemandRatingThreshold Attribute

This attribute SHALL indicate a threshold rating as a content filter which is compared with the rating for on-demand content. For example, if the on-demand content rating is greater than or equal to OnDemandRatingThreshold, for a rating system that is ordered from lower viewer age to higher viewer age, then on-demand content is not appropriate for the User and the Node SHALL prevent the playback of content.

This attribute SHALL be set to one of the values present in the OnDemandRatings attribute.

When this attribute changes, the device SHOULD make the user aware of any limits of this feature. For example, if the feature does not control content within apps, then the device should make this clear to the user when the attribute changes.

6.13.7.4. ScheduledContentRatings Attribute

This attribute SHALL indicate a collection of ratings which ScheduledContentRatingThreshold can be set to. The items should honor metadata of the scheduled content rating system for the country or region where the media device has been provisioned.

The media device SHALL have a way to determine which scheduled content rating system applies and then populate this attribute. For example, this can be done by examining the Location attribute in Basic Information cluster, and then determining which rating system applies.

The ratings in this collection SHALL be in order from a rating for the youngest viewers to the one for the oldest viewers. Each rating in the list SHALL be unique.

6.13.7.5. ScheduledContentRatingThreshold Attribute

This attribute SHALL indicate a threshold rating as a content filter which is used to compare with the rating for scheduled content. For example, if the scheduled content rating is greater than or equal to ScheduledContentRatingThreshold for a rating system that is ordered from lower viewer age to higher viewer age, then the scheduled content is not appropriate for the User and SHALL be blocked.

This attribute SHALL be set to one of the values present in the ScheduledContentRatings attribute.

When this attribute changes, the device SHOULD make the user aware of any limits of this feature. For example, if the feature does not control content within apps, then the device should make this clear to the user when the attribute changes.

6.13.7.6. ScreenDailyTime Attribute

This attribute SHALL indicate the amount of time (in seconds) which the User is allowed to spend watching TV within one day when the Content Control feature is activated.

6.13.7.7. RemainingScreenTime Attribute

This attribute SHALL indicate the remaining screen time (in seconds) which the User is allowed to spend watching TV for the current day when the Content Control feature is activated. When this value equals 0, the media device SHALL terminate the playback of content.

This attribute SHALL be updated when the AddBonusTime command is received and processed successfully (with the correct PIN).

6.13.7.8. BlockUnrated Attribute

This attribute SHALL indicate whether the playback of unrated content is allowed when the Content Control feature is activated. If this attribute equals FALSE, then playback of unrated content

SHALL be permitted. Otherwise, the media device SHALL prevent the playback of unrated content.

When this attribute changes, the device SHOULD make the user aware of any limits of this feature. For example, if the feature does not control content within apps, then the device should make this clear to the user when the attribute changes.

6.13.7.9. BlockChannelList Attribute

This attribute SHALL indicate a set of channels that SHALL be blocked when the Content Control feature is activated.

6.13.7.10. BlockApplicationList Attribute

This attribute SHALL indicate a set of applications that SHALL be blocked when the Content Control feature is activated.

6.13.7.11. BlockContentTimeWindow Attribute

This attribute SHALL indicate a set of periods during which the playback of content on media device SHALL be blocked when the Content Control feature is activated. The media device SHALL reject any request to play content during one period of this attribute. If it is entering any one period of this attribute, the media device SHALL block content which is playing and generate an event EnteringBlockContentTimeWindow. There SHALL not be multiple entries in this attribute list for the same day of week.

6.13.8. Commands

ID	Name	Direction	Response	Access	Conformance
0x00	**UpdatePIN**	client ⇒ server	Y	M T	PM
0x01	**ResetPIN**	client ⇒ server	ResetPINResponse	A T	PM
0x02	**ResetPINResponse**	client ⇐ server	N		PM
0x03	**Enable**	client ⇒ server	Y	M T	M
0x04	**Disable**	client ⇒ server	Y	M T	M
0x05	**AddBonusTime**	client ⇒ server	Y	O	ST
0x06	**SetScreenDailyTime**	client ⇒ server	Y	M	ST
0x07	**BlockUnratedContent**	client ⇒ server	Y	M	BU
0x08	**UnblockUnratedContent**	client ⇒ server	Y	M	BU

ID	Name	Direction	Response	Access	Conformance
0x09	**SetOnDemandRatingThreshold**	client ⇒ server	Y	M	OCR
0x0A	**SetScheduledContentRatingThreshold**	client ⇒ server	Y	M	SCR
0x0B	**AddBlockChannels**	client ⇒ server	Y	M	BC
0x0C	**RemoveBlockChannels**	client ⇒ server	Y	M	BC
0x0D	**AddBlockApplications**	client ⇒ server	Y	M	BA
0x0E	**RemoveBlockApplications**	client ⇒ server	Y	M	BA
0x0F	**SetBlockContentTimeWindow**	client ⇒ server	Y	M	BTW
0x10	**RemoveBlockContentTimeWindow**	client ⇒ server	Y	M	BTW

6.13.8.1. UpdatePIN Command

The purpose of this command is to update the PIN used for protecting configuration of the content control settings. Upon success, the old PIN SHALL no longer work.

The PIN is used to ensure that only the Node (or User) with the PIN code can make changes to the Content Control settings, for example, turn off Content Controls or modify the ScreenDailyTime. The PIN is composed of a numeric string of up to 6 human readable characters (displayable) .

Upon receipt of this command, the media device SHALL check if the OldPIN field of this command is the same as the current PIN. If the PINs are the same, then the PIN code SHALL be set to NewPIN. Otherwise a response with InvalidPINCode error status SHALL be returned.

The media device MAY provide a default PIN to the User via an out of band mechanism. For security reasons, it is recommended that a client encourage the user to update the PIN from its default value when performing configuration of the Content Control settings exposed by this cluster. The ResetPIN command can also be used to obtain the default PIN.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**OldPIN**	string	max 6			M

ID	Name	Type	Constraint	Quality	Default	Conformance
1	**NewPIN**	string	max 6			M

6.13.8.1.1. OldPIN Field

This field SHALL specify the original PIN. Once the UpdatePIN command is performed successfully, it SHALL be invalid.

6.13.8.1.2. NewPIN Field

This field SHALL indicate a new PIN for the Content Control feature.

6.13.8.2. ResetPIN Command

The purpose of this command is to reset the PIN.

If this command is executed successfully, a ResetPINResponse command with a new PIN SHALL be returned.

6.13.8.3. ResetPINResponse Command

This command SHALL be generated in response to a ResetPIN command. The data for this command SHALL be as follows:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**PINCode**	string	max 6			M

6.13.8.3.1. PINCode Field

This field SHALL indicate a new PIN of the Content Control feature.

6.13.8.4. Enable Command

The purpose of this command is to turn on the Content Control feature on a media device.

Upon receipt of the Enable command, the media device SHALL set the Enabled attribute to TRUE.

6.13.8.5. Disable Command

The purpose of this command is to turn off the Content Control feature on a media device.

On receipt of the Disable command, the media device SHALL set the Enabled attribute to FALSE.

6.13.8.6. AddBonusTime Command

The purpose of this command is to add the extra screen time for the user.

If a client with Operate privilege invokes this command, the media device SHALL check whether

the PINCode passed in the command matches the current PINCode value. If these match, then the RemainingScreenTime attribute SHALL be increased by the specified BonusTime value.

If the PINs do not match, then a response with InvalidPINCode error status SHALL be returned, and no changes SHALL be made to RemainingScreenTime.

If a client with Manage privilege or greater invokes this command, the media device SHALL ignore the PINCode field and directly increase the RemainingScreenTime attribute by the specified BonusTime value.

A server that does not support the PM feature SHALL respond with InvalidPINCode to clients that only have Operate privilege unless:

- It has been provided with the PIN value to expect via an out of band mechanism, and
- The client has provided a PINCode that matches the expected PIN value.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**PINCode**	string	max 6			O
1	**BonusTime**	elapsed-s	desc		300s	M

6.13.8.6.1. PINCode Field

This field SHALL indicate the PIN.

This field SHALL be optional for clients with Manage or greater privilege but SHALL be mandatory for clients with Operate privilege. The PIN provided in this field SHALL be used to guarantee that a client with Operate permission is allowed to invoke this command only if the PIN passed in this command is equal to the current PIN value.

6.13.8.6.2. BonusTime Field

This field SHALL indicate the amount of extra time (in seconds) to increase RemainingScreenTime. This field SHALL not exceed the remaining time of this day.

6.13.8.7. SetScreenDailyTime Command

The purpose of this command is to set the ScreenDailyTime attribute.

Upon receipt of the SetScreenDailyTime command, the media device SHALL set the ScreenDailyTime attribute to the ScreenTime value.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**ScreenTime**	elapsed-s	max 86400			M

6.13.8.7.1. ScreenTime Field

This field SHALL indicate the time (in seconds) which the User is allowed to spend watching TV on

this media device within one day.

6.13.8.8. BlockUnratedContent Command

The purpose of this command is to specify whether programs with no Content rating must be blocked by this media device.

Upon receipt of the BlockUnratedContent command, the media device SHALL set the BlockUnrated attribute to TRUE.

6.13.8.9. UnblockUnratedContent Command

The purpose of this command is to specify whether programs with no Content rating must be blocked by this media device.

Upon receipt of the UnblockUnratedContent command, the media device SHALL set the BlockUnrated attribute to FALSE.

6.13.8.10. SetOnDemandRatingThreshold Command

The purpose of this command is to set the OnDemandRatingThreshold attribute.

Upon receipt of the SetOnDemandRatingThreshold command, the media device SHALL check if the Rating field is one of values present in the OnDemandRatings attribute. If not, then a response with InvalidRating error status SHALL be returned.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**Rating**	string	max 8			M

6.13.8.10.1. Rating

This field indicates a threshold rating for filtering on-demand content. This field SHALL be set to one of the values present in the OnDemandRatings attribute

6.13.8.11. SetScheduledContentRatingThreshold Command

The purpose of this command is to set ScheduledContentRatingThreshold attribute.

Upon receipt of the SetScheduledContentRatingThreshold command, the media device SHALL check if the Rating field is one of values present in the ScheduledContentRatings attribute. If not, then a response with InvalidRating error status SHALL be returned.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**Rating**	string	max 8			M

6.13.8.11.1. Rating

This field indicates a threshold rating for filtering scheduled content. This field SHALL be set to one

of the values present in the ScheduledContentRatings attribute.

6.13.8.12. AddBlockChannels command

The purpose of this command is to set BlockChannelList attribute.

Upon receipt of the AddBlockChannels command, the media device SHALL check if the channels passed in this command are valid. If the channel is invalid, then a response with InvalidChannel error Status SHALL be returned.

If there is at least one channel in Channels field which is not in the BlockChannelList attribute, the media device SHALL process the request by adding these new channels into the BlockChannelList attribute and return a successful Status Response. During this process, the media device SHALL assign one unique index to BlockChannelIndex field for every channel passed in this command.

If all channels in Channel field already exist in the BlockChannelList attribute, then a response with ChannelAlreadyExist error Status SHALL be returned.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**Channels**	list[BlockChannelStruct]	all			M

6.13.8.12.1. Channels Field

This field indicates a set of channels that SHALL be blocked when the Content Control feature is activated. This field SHALL be set to values present in ChannelList attribute in the Channel cluster. The BlockChannelIndex field passed in this command SHALL be NULL.

6.13.8.13. RemoveBlockChannels command

The purpose of this command is to remove channels from the BlockChannelList attribute.

Upon receipt of the RemoveBlockChannels command, the media device SHALL check if the channels indicated by ChannelIndexes passed in this command are present in BlockChannelList attribute. If one or more channels indicated by ChannelIndexes passed in this command field are not present in the BlockChannelList attribute, then a response with ChannelNotExist error Status SHALL be returned.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**ChannelIndexes**	list[uint16]	all			M

6.13.8.13.1. ChannelIndexes Field

This field SHALL specify a set of indexes indicating Which channels SHALL be removed from the BlockChannelList attribute.

6.13.8.14. AddBlockApplications Command

The purpose of this command is to set applications to the BlockApplicationList attribute.

Upon receipt of the AddBlockApplications command, the media device SHALL check if the Applications passed in this command are installed. If there is an application in Applications field which is not identified by media device, then a response with UnidentifiableApplication error Status MAY be returned.

If there is one or more applications which are not present in BlockApplicationList attribute, the media device SHALL process the request by adding the new application to the BlockApplicationList attribute and return a successful Status Response.

If all applications in Applications field are already present in BlockApplicationList attribute, then a response with ApplicationAlreadyExist error Status SHALL be returned.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**Applications**	list[AppInfoStruct]	all			M

6.13.8.14.1. Applications Field

This field indicates a set of applications that SHALL be blocked when the Content Control feature is activated.

6.13.8.15. RemoveBlockApplications Command

The purpose of this command is to remove applications from the BlockApplicationList attribute.

Upon receipt of the RemoveBlockApplications command, the media device SHALL check if the applications passed in this command present in the BlockApplicationList attribute. If one or more applications in Applications field which are not present in the BlockApplicationList attribute, then a response with ApplicationNotExist error Status SHALL be returned.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**Applications**	list[AppInfoStruct]	all			M

6.13.8.15.1. Applications Field

This field indicates a set of applications which SHALL be removed from BlockApplicationList attribute.

6.13.8.16. SetBlockContentTimeWindow Command

The purpose of this command is to set the BlockContentTimeWindow attribute.

Upon receipt of the SetBlockContentTimeWindow command, the media device SHALL check if the

TimeWindowIndex field passed in this command is NULL. If the TimeWindowIndex field is NULL, the media device SHALL check if there is an entry in the BlockContentTimeWindow attribute which matches with the TimePeriod and DayOfWeek fields passed in this command. * If Yes, then a response with TimeWindowAlreadyExist error status SHALL be returned. * If No, then the media device SHALL assign one unique index for this time window and add it into the BlockContent-TimeWindow list attribute.

If the TimeWindowIndex field is not NULL and presents in the BlockContentTimeWindow attribute, the media device SHALL replace the original time window with the new time window passed in this command.

ID	Name	Type	Constraint	Quality	Default	Confor-mance
0	**TimeWin-dow**	TimeWin-dowStruct				M

6.13.8.16.1. TimeWindow Field

This field SHALL indicate a time window requested to set to the BlockContentTimeWindow attribute.

6.13.8.17. RemoveBlockContentTimeWindow Command

The purpose of this command is to remove the selected time windows from the BlockContent-TimeWindow attribute.

Upon receipt of the RemoveBlockContentTimeWindow command, the media device SHALL check if the time window index passed in this command presents in the BlockContentTimeWindow attribute.

If one or more time window indexes passed in this command are not present in BlockContent-TimeWindow attribute, then a response with TimeWindowNotExist error status SHALL be returned.

ID	Name	Type	Constraint	Quality	Default	Confor-mance
0	**TimeWin-dowIndexes**	list[uint16]				M

6.13.8.17.1. TimeWindowIndexes Field

This field SHALL specify a set of time window indexes indicating which time windows will be removed from the BlockContentTimeWindow attribute.

6.13.9. Events

ID	Name	Priority	Access	Conformance
0x00	**RemainingScreenTimeExpired**	INFO	V	ST
0x01	**EnteringBlockContentTimeWindow**	INFO	V	BTW

6.13.9.1. RemainingScreenTimeExpired Event

This event SHALL be generated when the RemainingScreenTime equals 0.

6.13.9.2. EnteringBlockContentTimeWindow Event

This event SHALL be generated when entering a period of blocked content as configured in the BlockContentTimeWindow attribute.

Chapter 7. Robots

The Cluster Library is made of individual chapters such as this one. See Document Control in the Cluster Library for a list of all chapters and documents. References between chapters are made using a *X.Y* notation where *X* is the chapter and *Y* is the sub-section within that chapter.

7.1. General Description

7.1.1. Introduction

The clusters specified in this section define the operation of robotic devices, such as Robotic Vacuum Cleaners (RVCs).

7.1.2. Cluster List

This section lists the RVC specific clusters as specified in this chapter.

Table 38. Overview of the RVC Clusters

Cluster ID	Cluster Name	Description
0x0054	RVC Run Mode	Commands and attributes for controlling the running mode of an RVC device.
0x0055	RVC Clean Mode	Commands and attributes for controlling the cleaning mode of an RVC device.
0x0061	RVC Operational State	Commands and attributes for monitoring and controlling the operational state of an RVC device.

7.2. RVC Run Mode Cluster

This cluster is derived from the Mode Base cluster to define specifics for Robotic Vacuum Cleaner devices. It also defines a namespace for the running modes of these devices.

7.2.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Revision	Description
1	Initial Release

Revision	Description
2	Add constraint about switching from non-Idle to non-Idle modes. ChangeToModeResponse command: StatusText must be provided for InvalidInMode status. Deprecate the OnMode attribute and the related feature map bit. Add the Mapping mode tag.

7.2.2. Classification

Hierarchy	Role	Scope	PICS Code
Mode Base	Application	Endpoint	RVCRUNM

7.2.3. Cluster ID

ID	Name
0x0054	RVC Run Mode

7.2.4. Feature Map

The table below lists the changes relative to the Mode Base cluster for the feature map bits. A blank field indicates no change.

Bit	Code	Feature	Conformance
0	**DEPONOFF**	OnOff	D

7.2.5. Data Types

7.2.5.1. ModeOptionStruct Type

The table below lists the changes relative to the Mode Base cluster for the fields of the ModeOptionStruct type. A blank field indicates no change.

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0	**Label**						M
1	**Mode**						M
2	**ModeTags**		1 to 8				M

At least one entry in the SupportedModes attribute SHALL include the Idle mode tag in the ModeTags field.

At least one entry in the SupportedModes attribute (different from the one above) SHALL include the Cleaning mode tag in the ModeTags field.

The Mapping, Cleaning, and Idle mode tags are mutually exclusive and SHALL NOT be used together in a mode's ModeTags.

7.2.6. Attributes

The table below lists the changes relative to the Mode Base cluster for the attributes. A blank field indicates no change.

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0000	**SupportedModes**						
0x0001	**CurrentMode**						
0x0002	**StartUpMode**						X
0x0003	**OnMode**						D

7.2.7. Derived Cluster Namespace

This namespace includes definitions for data associated exclusively with the derived cluster.

7.2.7.1. ChangeToModeResponse Command Namespace Definitions

The following table defines the derived cluster specific StatusCode values.

Status Code Value	Name
0x41	**Stuck**
0x42	**DustBinMissing**
0x43	**DustBinFull**
0x44	**WaterTankEmpty**
0x45	**WaterTankMissing**
0x46	**WaterTankLidOpen**
0x47	**MopCleaningPadMissing**
0x48	**BatteryLow**

7.2.7.2. Mode Tags

The following table defines the derived cluster specific ModeTag values.

Mode Tag Value	Name
0x4000	**Idle**
0x4001	**Cleaning**

Mode Tag Value	Name
0x4002	**Mapping**

7.2.7.2.1. Idle Tag

The device is not performing any of the main operations of the other modes. However, auxiliary actions, such as seeking the charger or charging, may occur.

For example, the device has completed cleaning, successfully or not, on its own or due to a command, or has not been asked to clean after a restart.

7.2.7.2.2. Cleaning Tag

The device was asked to clean so it may be actively running, or paused due to an error, due to a pause command, or for recharging etc. If currently paused and the device can resume it will continue to clean.

7.2.7.2.3. Mapping Tag

The device was asked to create a map of the space it is located in, so it may be actively running, or paused due to an error, due to a pause command, or for recharging etc. If currently paused and the device can resume, it will continue to map.

NOTE	this mode is intended to be used so the current space can be mapped by the device if the robot has not previously done that, or if the layout has substantially changed, for an optimal subsequent cleaning experience.

7.2.8. Mode Use

Starting a cleaning cycle SHALL be done by switching from a mode with the Idle mode tag to a mode with the Cleaning mode tag.

Stopping a cleaning cycle SHALL be done by switching from a mode with the Cleaning mode tag to a mode with the Idle mode tag.

Starting a mapping cycle SHALL be done by switching from a mode with the Idle mode tag to a mode with the Mapping mode tag.

Stopping a mapping cycle SHALL be done by switching from a mode with the Mapping mode tag to a mode with the Idle mode tag.

The RVC Run Mode cluster's SupportedModes attribute MAY include multiple modes with the Cleaning mode tag, and/or modes with the Mapping mode tag.

7.2.9. Mode Change Restrictions

Attempting to switch the RVC Run Mode from a mode without the Idle mode tag to another non-Idle mode SHALL NOT be allowed and the ChangeToModeResponse command SHALL have the StatusCode field set to the InvalidInMode value in that case.

7.3. RVC Clean Mode Cluster

This cluster is derived from the Mode Base cluster to define specifics for Robotic Vacuum Cleaner devices. It also defines a namespace for the cleaning type for these devices.

7.3.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Revision	Description
1	Initial Release
2	Add constraint about changing cleaning modes while the RVC Run Mode cluster is in a non-Idle mode. ChangeToModeResponse command: StatusText must be provided for InvalidInMode status. Deprecate the OnMode attribute and the related feature map bit.

7.3.2. Classification

Hierarchy	Role	Scope	PICS Code
Mode Base	Application	Endpoint	RVCCLEANM

7.3.3. Cluster ID

ID	Name
0x0055	RVC Clean Mode

7.3.4. Feature Map

The table below lists the changes relative to the Mode Base cluster for the feature map bits. A blank field indicates no change.

Bit	Code	Feature	Conformance
0	**DEPONOFF**	OnOff	D

7.3.5. Data Types

7.3.5.1. ModeOptionStruct Type

The table below lists the changes relative to the Mode Base cluster for the fields of the ModeOptionStruct type. A blank field indicates no change.

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0	**Label**						M
1	**Mode**						M
2	**ModeTags**		1 to 8				M

At least one entry in the SupportedModes attribute SHALL include the Vacuum and/or the Mop mode tag in the ModeTags field list.

7.3.6. Attributes

The table below lists the changes relative to the Mode Base cluster for the attributes. A blank field indicates no change.

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0000	**SupportedModes**						
0x0001	**CurrentMode**						
0x0002	**StartUpMode**						X
0x0003	**OnMode**						D

7.3.7. Derived Cluster Namespace

This namespace includes definitions for data associated exclusively with the derived cluster.

7.3.7.1. ChangeToModeResponse Command Namespace Definitions

The following table defines the derived cluster specific StatusCode values.

Status Code Value	Name
0x40	**CleaningInProgress**

7.3.7.2. Mode Tags

The following table defines the derived cluster specific ModeTag values.

Mode Tag Value	Name
0x4000	**DeepClean**
0x4001	**Vacuum**
0x4002	**Mop**

7.3.7.2.1. Deep Clean Tag

While in this mode, the device is optimizing for improved cleaning.

7.3.7.2.2. Vacuum Tag

The device's vacuuming feature is enabled in this mode.

7.3.7.2.3. Mop Tag

The device's mopping feature is enabled in this mode.

7.3.8. Mode Change Restrictions

This cluster SHALL NOT permit changing its mode while the RVC Run Mode cluster's CurrentMode attribute is set to a mode without the Idle mode tag. The ChangeToModeResponse command SHALL have the StatusCode field set to the InvalidInMode value if this restriction prevents a mode change.

7.3.9. Mode Examples

A few examples of modes and their mode tags are provided below.

For the "Turbo, Vacuum Only" mode, tags: 0x4000 (Deep Clean), 0x4001 (Vacuum).

For the "Mop Only" mode, tags: 0x4002 (Mop), 0x0003 (Low Noise).

For the "Rapid Vacuum and Mop" mode, tags: 0x0001 (Quick), 0x4001 (Vacuum), 0x4002 (Mop).

Note that the "Low Noise" and "Quick" mode tags are defined in the generic Mode Base cluster specification.

7.4. RVC Operational State Cluster

This cluster provides an interface for monitoring the operational state of a Robotic Vacuum Cleaner.

7.4.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Revision	Description
1	Initial release
2	The Pause and Resume commands are usable in all compatible states. Deprecate the Start and Stop commands. Add the GoHome command.

7.4.2. Classification

Hierarchy	Role	Scope	PICS Code
Operational State	Application	Endpoint	RVCOPSTATE

7.4.3. Cluster ID

ID	Name
0x0061	RVC Operational State

7.4.4. Data Types

7.4.4.1. OperationalStateEnum Type

This data type is derived from enum8.

The values defined herein are applicable to this derived cluster of Operational State only and are additional to the set of values defined in Operational State itself.

Value	Name	Summary	Conformance
0x40	**SeekingCharger**	The device is en route to the charging dock	M
0x41	**Charging**	The device is charging	M
0x42	**Docked**	The device is on the dock, not charging	M

RVC Pause Compatibility defines the compatibility of the states this cluster defines with the Pause command.

Table 39. RVC Pause Compatibility

State Value	State Name	Pause-Compatible	Notes
0x40	**SeekingCharger**	Y	
0x41	**Charging**	N	
0x42	**Docked**	N	

RVC Resume Compatibility defines the compatibility of the states this cluster defines with the Resume command.

Table 40. RVC Resume Compatibility

State Value	State Name	Resume-Compatible	Notes
0x40	**SeekingCharger**	N	
0x41	**Charging**	Y	
0x42	**Docked**	Y	

While in the Charging or Docked states, the device SHALL NOT attempt to resume unless it transitioned to those states while operating and can resume, such as, for example, if it is recharging while in a cleaning cycle. Else, if the operational state is Charging or Docked but there's no operation to resume or the operation can't be resumed, the device SHALL respond with an OperationalCommandResponse command with an ErrorStateID of CommandInvalidInState but take no further action.

7.4.4.2. ErrorStateEnum Type

This data type is derived from enum8.

The values defined herein are applicable to this derived cluster of Operational State only and are additional to the set of values defined in Operational State itself.

Value	Name	Summary	Conformance
0x40	**FailedToFindChargingDock**	The device has failed to find or reach the charging dock	M
0x41	**Stuck**	The device is stuck and requires manual intervention	M
0x42	**DustBinMissing**	The device has detected that its dust bin is missing	M
0x43	**DustBinFull**	The device has detected that its dust bin is full	M
0x44	**WaterTankEmpty**	The device has detected that its water tank is empty	M
0x45	**WaterTankMissing**	The device has detected that its water tank is missing	M
0x46	**WaterTankLidOpen**	The device has detected that its water tank lid is open	M
0x47	**MopCleaningPadMissing**	The device has detected that its cleaning pad is missing	M

7.4.5. Commands

The table below lists the changes relative to the Operational State cluster for the commands. A blank field indicates no change.

ID	Name	Direction	Response	Access	Conformance
0x00	**Pause**				

ID	Name	Direction	Response	Access	Conformance
0x01	**Stop**				D
0x02	**Start**				D
0x03	**Resume**				
0x04	**OperationalCommandResponse**				
0x80	**GoHome**	client ⇒ server	OperationalCommandResponse	O	P, O

7.4.5.1. GoHome Command

On receipt of this command, the device SHALL start seeking the charging dock, if possible in the current state of the device.

If this command is received when already in the SeekingCharger state the device SHALL respond with an OperationalCommandResponse command with an ErrorStateID of NoError but the command SHALL have no other effect.

A device that receives this command in any state which does not allow seeking the charger, such as Charging or Docked, SHALL respond with an OperationalCommandResponse command with an ErrorStateID of CommandInvalidInState and SHALL have no other effect.

Otherwise, on success:

- The OperationalState attribute SHALL be set to SeekingCharger.
- The device SHALL respond with an OperationalCommandResponse command with an ErrorStateID of NoError.

Chapter 8. Home Appliances

The Cluster Library is made of individual chapters such as this one. See Document Control in the Cluster Library for a list of all chapters and documents. References between chapters are made using a *X.Y* notation where *X* is the chapter and *Y* is the sub-section within that chapter.

8.1. General Description

8.1.1. Introduction

The clusters specified in this section are typically used in control of home appliances, such as laundry washers, refrigerators etc.

8.1.2. Cluster List

This section lists the home appliance specific clusters as specified in this chapter.

Table 41. Overview of the Home Appliance Clusters

Cluster ID	Cluster Name	Description
Common Clusters		
0x0056	Temperature Control	Commands and attributes for control of a temperature set point
Dishwasher Clusters		
0x0059	Dishwasher Mode	Commands and attributes for controlling a dishwasher
0x005D	Dishwasher Alarm	Alarm definitions for Dishwasher devices
Laundry Clusters (General)		
0x0051	Laundry Washer Mode	Commands and attributes for controlling a laundry washer or dryer
Laundry Dryer Clusters		
0x004A	Laundry Dryer Controls	Commands and attributes for the control of options on a device that does laundry drying
Laundry Washer Clusters		
0x0053	Laundry Washer Controls	Commands and attributes for the control of options on a device that does laundry washing
Microwave Oven Clusters		

Cluster ID	Cluster Name	Description
0x005E	Microwave Oven Mode	Modes associated with a Microwave Oven
0x005F	Microwave Oven Control	Commands and attributes for controlling a Microwave Oven
Oven Clusters		
0x0048	Oven Operational State	Operational states and phases associated with an Oven
0x0049	Oven Mode	Modes associated with an Oven
Refrigerator Clusters		
0x0052	Refrigerator And Temperature Controlled Cabinet Mode	Commands and attributes for controlling a refrigerator or a temperature controlled cabinet
0x0057	Refrigerator Alarm	Alarm definitions for Refrigerator devices

8.2. Temperature Control Cluster

This cluster provides an interface to the setpoint temperature on devices such as washers, refrigerators, and water heaters. The setpoint temperature is the temperature to which a device using this cluster would attempt to control to. This cluster does not provide access to the actual or physical temperature associated with any device using this cluster. Access to the physical temperature associated with a device using this cluster would be provided by other clusters as part of that devices device type definition.

The values and constraints of the attributes communicated to clients SHOULD match the controls on any physical interface on a device implementing this server. For example, the value of the Step attribute SHOULD match the incremental value by which the temperature setpoint can be changed on the physical device.

8.2.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Rev	Description
1	Initial release

8.2.2. Classification

Hierarchy	Role	Scope	PICS Code
Base	Application	Endpoint	TCTL

8.2.3. Cluster ID

ID	Name
0x0056	Temperature Control

8.2.4. Features

This cluster SHALL support the FeatureMap bitmap attribute as defined below.

Bit	Code	Feature	Conformance	Summary
0	**TN**	TemperatureNumber	O.a	Use actual temperature numbers
1	**TL**	TemperatureLevel	O.a	Use temperature levels
2	**STEP**	TemperatureStep	[TN]	Use step control with temperature numbers

8.2.4.1. TemperatureNumber Feature

For devices that use an actual temperature value for the temperature setpoint, such as some water heaters, the feature TN SHALL be used. Note that this cluster provides and supports temperatures in degrees Celsius via the `temperature` data type.

8.2.4.2. TemperatureLevel Feature

For devices that use vendor-specific temperature levels for the temperature setpoint, such as some washers, the feature TL SHALL be used.

8.2.4.3. TemperatureStep Feature

For devices that support discrete temperature setpoints that are larger than the temperature resolution imposed via the `temperature` data type, the Step feature MAY be used.

8.2.5. Attributes

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0000	**TemperatureSetpoint**	temperature	MinTemperature to MaxTemperature			R V	TN
0x0001	**MinTemperature**	temperature	max (MaxTemperature - 1)*	F		R V	TN

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0002	**MaxTemperature**	temperature	desc	F		R V	TN
0x0003	**Step**	temperature	max (MaxTemperature - MinTemperature)	F		R V	STEP
0x0004	**SelectedTemperatureLevel**	uint8	0 to 31			R V	TL
0x0005	**SupportedTemperatureLevels**	list[string]	max 32[max 16]			R V	TL

NOTE * See temperature data type, in the data model, for encoding units.

8.2.5.1. TemperatureSetpoint Attribute

This attribute SHALL represent the desired Temperature Setpoint on the device.

8.2.5.2. MinTemperature Attribute

This attribute SHALL represent the minimum temperature to which the TemperatureSetpoint attribute MAY be set.

8.2.5.3. MaxTemperature Attribute

This attribute SHALL represent the maximum temperature to which the TemperatureSetpoint attribute MAY be set.

If the Step attribute is supported, this attribute SHALL be such that MaxTemperature = MinTemperature + Step * n, where n is an integer and n > 0. If the Step attribute is not supported, this attribute SHALL be such that MaxTemperature > MinTemperature.

8.2.5.4. Step Attribute

This attribute SHALL represent the discrete value by which the TemperatureSetpoint attribute can be changed via the SetTemperature command.

> For example, if the value of MinTemperature is 25.00C (2500) and the Step value is 0.50C (50), valid values of the TargetTemperature field of the SetTemperature command would be 25.50C (2550), 26.00C (2600), 26.50C (2650), etc.

8.2.5.5. SelectedTemperatureLevel Attribute

This attribute SHALL represent the currently selected temperature level setting of the server. This attribute SHALL be the positional index of the list item in the SupportedTemperatureLevels list that represents the currently selected temperature level setting of the server.

8.2.5.6. SupportedTemperatureLevels Attribute

This attribute SHALL represent the list of supported temperature level settings that may be selected via the TargetTemperatureLevel field in the SetTemperature command. Each string is readable text that describes each temperature level setting in a way that can be easily understood by humans. For example, a washing machine can have temperature levels like "Cold", "Warm", and "Hot". Each string is specified by the manufacturer.

Each item in this list SHALL represent a unique temperature level. Each entry in this list SHALL have a unique value. The entries in this list SHALL appear in order of increasing temperature level with list item 0 being the setting with the lowest temperature level.

8.2.6. Commands

ID	Name	Direction	Response	Access	Conformance
0x00	**SetTemperature**	client ⇒ server	Y	O	M

8.2.6.1. SetTemperature Command

The SetTemperature command SHALL have the following data fields:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**TargetTemperature**	temperature	desc			TN
1	**TargetTemperatureLevel**	uint8	desc			TL

8.2.6.1.1. TargetTemperature Field

This field SHALL specify the desired temperature setpoint that the server is to be set to.

The TargetTemperature SHALL be from MinTemperature to MaxTemperature inclusive. If the Step attribute is supported, TargetTemperature SHALL be such that (TargetTemperature - MinTemperature) % Step == 0.

8.2.6.1.2. TargetTemperatureLevel Field

This field SHALL specify the index of the list item in the SupportedTemperatureLevels list that represents the desired temperature level setting of the server. The value of this field SHALL be between 0 and the length of the SupportedTemperatureLevels list -1.

8.2.6.1.3. Effect on Receipt

If the TargetTemperature or TargetTemperatureLevel fields of the command meet all constraints but the server is unable to execute the command at the time the command is received by the server (e.g. due to the design of a device it cannot accept a change in its temperature setting after it has begun operation), then the server SHALL respond with a status code of INVALID_IN_STATE, and discard the command.

If the TN feature is supported, on receipt of this command,

- If the value of the TargetTemperature field meets all constraints, the server SHALL set the TemperatureSetpoint attribute to the value of the TargetTemperature field and the response SHALL have a status code of SUCCESS.
- Otherwise (e.g. if the value of the TargetTemperature field falls outside of the constraints of the TemperatureSetpoint attribute or if the Step attribute is supported in the server and the value of the TargetTemperature field is such that (TargetTemperature - MinTemperature) % Step != 0), the status of the response SHALL be CONSTRAINT_ERROR and the value of the TemperatureSetpoint attribute SHALL remain unchanged.

If the TL feature is supported, on receipt of this command,

- If value of the TargetTemperatureLevel field meets all constraints, the server SHALL set its SelectedTemperatureLevel attribute to the value of TargetTemperatureLevel field and respond with status SUCCESS.
- Otherwise (e.g. if the value of the TargetTemperatureLevel field is out of bounds of the SupportedTemperatureLevels list), the status of the response SHALL be CONSTRAINT_ERROR, and the value of SelectedTemperatureLevel SHALL remain unchanged.

8.3. Dishwasher Mode Cluster

This cluster is derived from the Mode Base cluster, defining additional mode tags and namespaced enumerated values for dishwasher devices.

8.3.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Revision	Description
1	Initial Release
2	ChangeToModeResponse command: StatusText must be provided for InvalidInMode status

8.3.2. Classification

Hierarchy	Role	Scope	PICS Code
Mode Base	Application	Endpoint	DISHM

8.3.3. Cluster ID

ID	Name
0x0059	Dishwasher Mode

8.3.4. Data Types

8.3.4.1. ModeOptionStruct Type

The table below lists the changes relative to the Mode Base cluster for the fields of the ModeOptionStruct type. A blank field indicates no change.

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0	**Label**						M
1	**Mode**						M
2	**ModeTags**		1 to 8				M

At least one entry in the SupportedModes attribute SHALL include the Normal mode tag in the ModeTags field list.

8.3.5. Attributes

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0000	**SupportedModes**						M
0x0001	**CurrentMode**						M
0x0002	**StartUpMode**						P
0x0003	**OnMode**						P

8.3.5.1. StartUpMode Attribute

If this attribute is supported, the device SHOULD initially set this to one of the supported modes that has the Normal tag associated with it. See the Mode Base cluster specification for full details about the StartUpMode attribute.

8.3.6. Derived Cluster Namespace

This namespace includes definitions for data associated exclusively with the derived cluster.

8.3.6.1. Mode Tags

The following table defines the derived cluster specific ModeTag values.

Mode Tag Value	Name
0x4000	**Normal**
0x4001	**Heavy**
0x4002	**Light**

8.3.6.1.1. Normal Tag

The normal regime of operation.

8.3.6.1.2. Heavy Tag

Mode optimized for washing heavily-soiled dishes.

8.3.6.1.3. Light Tag

Mode optimized for light washing.

8.4. Dishwasher Alarm Cluster

This cluster is a derived cluster of the Alarm Base cluster.

8.4.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Revision	Description
1	Initial revision

8.4.2. Classification

Hierarchy	Role	Scope	PICS Code
Alarm Base	Application	Endpoint	DISHALM

8.4.3. Cluster ID

ID	Name
0x005D	Dishwasher Alarm

8.4.4. Data Types

8.4.4.1. AlarmBitmap Type

This data type is derived from map32.

Bit	Name	Summary	Conformance
0	**InflowError**	Water inflow is abnormal	P,O.a+
1	**DrainError**	Water draining is abnormal	P,O.a+
2	**DoorError**	Door or door lock is abnormal	O.a+
3	**TempTooLow**	Unable to reach normal temperature	P,O.a+
4	**TempTooHigh**	Temperature is too high	P,O.a+
5	**WaterLevelError**	Water level is abnormal	P,O.a+

8.5. Laundry Washer Mode Cluster

This cluster is derived from the Mode Base cluster, defining additional mode tags and namespaced enumerated values for laundry washer as well as laundry dryer devices.

8.5.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Revision	Description
1	Initial Release
2	ChangeToModeResponse command: StatusText must be provided for InvalidInMode status

8.5.2. Classification

Hierarchy	Role	Scope	PICS Code
Mode Base	Application	Endpoint	LWM

8.5.3. Cluster ID

ID	Name
0x0051	Laundry Washer Mode

8.5.4. Data Types

8.5.4.1. ModeOptionStruct Type

The table below lists the changes relative to the Mode Base cluster for the fields of the ModeOptionStruct type. A blank field indicates no change.

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0	**Label**						M
1	**Mode**						M
2	**ModeTags**		1 to 8				M

At least one entry in the SupportedModes attribute SHALL include the Normal mode tag in the ModeTags field list.

8.5.5. Attributes

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0000	**SupportedModes**						M
0x0001	**CurrentMode**						M
0x0002	**StartUpMode**						P
0x0003	**OnMode**						P

8.5.5.1. StartUpMode Attribute

If this attribute is supported, the device SHOULD initially set this to one of the supported modes that has the Normal tag associated with it. See the Mode Base cluster specification for full details about the StartUpMode attribute.

8.5.6. Derived Cluster Namespace

This namespace includes definitions for data associated exclusively with the derived cluster.

8.5.6.1. Mode Tags

The following table defines the derived cluster specific ModeTag values.

Mode Tag Value	Name
0x4000	**Normal**
0x4001	**Delicate**

Mode Tag Value	Name
0x4002	**Heavy**
0x4003	**Whites**

8.5.6.1.1. Normal Tag

The normal regime of operation.

8.5.6.1.2. Delicate Tag

Mode optimized for washing delicate garments.

8.5.6.1.3. Heavy Tag

Mode optimized for heavy washing.

8.5.6.1.4. Whites Tag

Mode optimized for stain removal on white fabrics.

8.5.7. Mode Examples

A few examples of Laundry modes and their mode tags are provided below.

- For the "Heavy Wash, Whites" mode, tags: 0x4002 (Heavy), 0x4003 (Whites).
- For the "Fast" mode, tags: 0x0001 (Quick), 0x4000 (Normal).

8.6. Laundry Washer Controls Cluster

This cluster provides a way to access options associated with the operation of a laundry washer device type.

8.6.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Rev	Description
1	Initial release

8.6.2. Classification

Hierarchy	Role	Scope	PICS Code
Base	Application	Endpoint	WASHERCTRL

8.6.3. Cluster ID

ID	Name
0x0053	Laundry Washer Controls

8.6.4. Features

This cluster SHALL support the FeatureMap bitmap attribute as defined below.

Bit	Code	Feature	Conformance	Summary
0	**SPIN**	Spin	O	Multiple spin speeds supported
1	**RINSE**	Rinse	O	Multiple rinse cycles supported

8.6.4.1. Spin Feature

This feature indicates multiple spin speeds are supported in at least one supported mode. Note that some modes may not support multiple spin speeds even if this feature is supported.

8.6.4.2. Rinse Feature

This feature indicates multiple rinse cycles are supported in at least one supported mode. Note that some modes may not support selection of the number of rinse cycles even if this feature is supported.

8.6.5. Data Types

8.6.5.1. NumberOfRinsesEnum Type

This data type is derived from enum8.

The NumberOfRinsesEnum provides a representation of the number of rinses that will be performed for a selected mode. NumberOfRinsesEnum is derived from enum8. It is up to the device manufacturer to determine the mapping between the enum values and the corresponding numbers of rinses.

Value	Name	Conformance	Description
0	**None**	RINSE	This laundry washer mode does not perform rinse cycles
1	**Normal**	RINSE	This laundry washer mode performs normal rinse cycles determined by the manufacturer

Value	Name	Conformance	Description
2	**Extra**	RINSE	This laundry washer mode performs an extra rinse cycle
3	**Max**	RINSE	This laundry washer mode performs the maximum number of rinse cycles determined by the manufacturer

8.6.6. Attributes

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0000	**SpinSpeeds**	list[string]	max 16[max 64]			R V	SPIN
0x0001	**SpinSpeedCurrent**	uint8	0 to 15	X	desc	RW VO	SPIN
0x0002	**NumberOfRinses**	NumberOfRinsesEnum	desc		1	RW VO	RINSE
0x0003	**SupportedRinses**	list[NumberOfRinsesEnum]	max 4			R V	RINSE

8.6.6.1. SpinSpeeds Attribute

This attribute indicates the list of spin speeds available to the appliance in the currently selected mode. The spin speed values are determined by the manufacturer. At least one spin speed value SHALL be provided in the SpinSpeeds list. The list of spin speeds MAY change depending on the currently selected Laundry Washer mode. For example, Quick mode might have a completely different list of SpinSpeeds than Delicates mode.

8.6.6.2. SpinSpeedCurrent Attribute

This attribute indicates the currently selected spin speed. It is the index into the SpinSpeeds list of the selected spin speed, as such, this attribute can be an integer between 0 and the number of entries in SpinSpeeds - 1. If a value is received that is outside of the defined constraints, a CONSTRAINT_ERROR SHALL be sent as the response. If a value is attempted to be written that doesn't match a valid index (e.g. an index of 5 when the list has 4 values), a CONSTRAINT_ERROR SHALL be sent as the response. If null is written to this attribute, there will be no spin speed for the selected cycle. If the value is null, there will be no spin speed on the current mode.

8.6.6.3. NumberOfRinses Attribute

This attribute represents how many times a rinse cycle SHALL be performed on a device for the current mode of operation. A value of None SHALL indicate that no rinse cycle will be performed. This value may be set by the client to adjust the number of rinses that are performed for the current mode of operation. If the device is not in a compatible state to accept the provided value, an INVALID_IN_STATE error SHALL be sent as the response.

8.6.6.4. SupportedRinses Attribute

This attribute represents the amount of rinses allowed for a specific mode. Each entry SHALL indicate a NumberOfRinsesEnum value that is possible in the selected mode on the device. The value of this attribute MAY change at runtime based on the currently selected mode. Each entry SHALL be distinct.

8.7. Refrigerator And Temperature Controlled Cabinet Mode Cluster

This cluster is derived from the Mode Base cluster, defining additional mode tags and namespaced enumerated values for refrigerator and temperature controlled cabinet devices.

8.7.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Revision	Description
1	Initial Release
2	ChangeToModeResponse command: StatusText must be provided for InvalidInMode status

8.7.2. Classification

Hierarchy	Role	Scope	PICS Code
Mode Base	Application	Endpoint	TCCM

8.7.3. Cluster ID

ID	Name
0x0052	Refrigerator And Temperature Controlled Cabinet Mode

8.7.4. Data Types

8.7.4.1. ModeOptionStruct Type

The table below lists the changes relative to the Mode Base cluster for the fields of the ModeOptionStruct type. A blank field indicates no change.

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0	**Label**						M
1	**Mode**						M
2	**ModeTags**		1 to 8				M

At least one entry in the SupportedModes attribute SHALL include the Auto mode tag in the ModeTags field list.

8.7.5. Attributes

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0000	**SupportedModes**						M
0x0001	**CurrentMode**						M
0x0002	**StartUpMode**						P
0x0003	**OnMode**						P

8.7.6. Derived Cluster Namespace

This namespace includes definitions for data associated exclusively with the derived cluster.

8.7.6.1. Mode Tags

The following table defines the derived cluster specific ModeTag values.

Mode Tag Value	Name
0x4000	**RapidCool**
0x4001	**RapidFreeze**

8.7.6.1.1. RapidCool Tag

This mode reduces the temperature rapidly, typically above freezing grade.

8.7.6.1.2. RapidFreeze Tag

This mode reduces the temperature rapidly, below freezing grade.

8.7.7. Mode Examples

A few examples of Refrigerator and Temperature Controlled Cabinet modes and their mode tags are provided below.

- For the "Normal" mode, tags: 0x0000 (Auto)
- For the "Energy Save" mode, tags: 0x0004 (LowEnergy)
- For the "Rapid Cool" mode, tags: 0x4000 (RapidCool)

8.8. Refrigerator Alarm Cluster

This cluster is a derived cluster of Alarm Base cluster.

8.8.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Revision	Description
1	Initial revision

8.8.2. Classification

Hierarchy	Role	Scope	PICS Code
Alarm Base	Application	Endpoint	REFALM

8.8.3. Cluster ID

ID	Name
0x0057	Refrigerator Alarm

8.8.4. Features

Bit	Code	Feature	Conformance	Summary
0	**RESET**	Reset	X	Supports the ability to reset alarms

8.8.5. Data Types

8.8.5.1. AlarmBitmap Type

This data type is derived from map32.

Bit	Name	Summary	Conformance
0	**DoorOpen**	The cabinet's door has been open for a vendor defined amount of time.	M

8.8.6. Attributes

8.8.6.1. Mask Attribute

If the generation of the alarm has not been suppressed at the device itself, then this attribute SHALL have these fixed values.

Bit	Name	Value
0	**DoorOpen**	1

This alarm SHALL be cleared only when the door is closed (manual action).

If the generation of the alarm is suppressed at the device itself, then bit 0 SHALL have a value of 0. It SHALL be re-set to 1 if the alarm is re-enabled at the device itself.

8.8.7. Commands

ID	Name	Direction	Response	Access	Conformance
0x01	**ModifyEnabledAlarms**				X

8.9. Laundry Dryer Controls Cluster

This cluster provides a way to access options associated with the operation of a laundry dryer device type.

8.9.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Rev	Description
1	Initial release

8.9.2. Classification

Hierarchy	Role	Scope	PICS Code
Base	Application	Endpoint	DRYERCTRL

8.9.3. Cluster Identifiers

Identifier	Name
0x004A	Laundry Dryer Controls

8.9.4. Data Types

8.9.4.1. DrynessLevelEnum Type

This data type is derived from enum8.

This enum provides a representation of the level of dryness that will be used while drying in a selected mode.

It is up to the device manufacturer to determine the mapping between the enum values and the corresponding temperature level.

Value	Name	Summary	Conformance
0	**Low**	Provides a low dryness level for the selected mode	M
1	**Normal**	Provides the normal level of dryness for the selected mode	M
2	**Extra**	Provides an extra dryness level for the selected mode	M
3	**Max**	Provides the max dryness level for the selected mode	M

8.9.5. Attributes

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0000	SupportedDrynessLevels	list[DrynessLevelEnum]	1 to 4			R	M
0x0001	SelectedDrynessLevel	DrynessLevelEnum	desc	X	desc	RW VO	M

8.9.5.1. SupportedDrynessLevels Attribute

This attribute SHALL indicate the list of supported dryness levels available to the appliance in the

currently selected mode. The dryness level values are determined by the manufacturer. At least one dryness level value SHALL be provided in the SupportedDrynessLevels list. The list of dryness levels MAY change depending on the currently-selected Laundry Dryer mode.

8.9.5.2. SelectedDrynessLevel Attribute

This attribute SHALL indicate the currently-selected dryness level and it SHALL be the index into the SupportedDrynessLevels list of the selected dryness level.

If an attempt is made to write this attribute with a value other than null or a value contained in SupportedDrynessLevels, a CONSTRAINT_ERROR response SHALL be sent as the response. If an attempt is made to write this attribute while the device is not in a state that supports modifying the dryness level, an INVALID_IN_STATE error SHALL be sent as the response. A value of null SHALL indicate that there will be no dryness level setting for the current mode.

8.10. Oven Cavity Operational State Cluster

This cluster provides an interface for monitoring the operational state of an oven.

8.10.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Rev	Description
1	Initial release

8.10.2. Classification

Hierarchy	Role	Scope	PICS Code
Operational State	Application	Endpoint	OVENOPSTATE

8.10.3. Cluster ID

Identifier	Name
0x0048	Oven Cavity Operational State

8.10.4. Attributes

8.10.4.1. PhaseList Attribute

As defined in the base cluster, this attribute indicates a list of names of different phases that the device can go through for the selected function or mode.

For this derived cluster, only these pre-defined strings may be used in the PhaseList attribute:

"pre-heating", "pre-heated", and "cooling down".

Other values SHALL NOT be used.

As defined in the base cluster, a null value indicates that the device does not present phases during its operation. When this attribute's value is null, the CurrentPhase attribute SHALL also be set to null.

8.11. Oven Mode Cluster

This cluster is derived from the Mode Base cluster, defining additional mode tags and namespaced enumerated values for oven devices.

8.11.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Revision	Description
1	Initial Release

8.11.2. Classification

Hierarchy	Role	Context	PICS Code
Mode Base	Application	Endpoint	OTCCM

8.11.3. Cluster Identifiers

Identifier	Name
0x0049	Oven Mode

8.11.4. Derived Cluster Namespace

This namespace includes definitions for data associated exclusively with the derived cluster.

8.11.4.1. Mode Tags

The following table defines the derived cluster specific ModeTag values.

Mode Tag Value	Name
0x4000	**Bake**
0x4001	**Convection**
0x4002	**Grill**
0x4003	**Roast**
0x4004	**Clean**
0x4005	**Convection Bake**

Mode Tag Value	Name
0x4006	**Convection Roast**
0x4007	**Warming**
0x4008	**Proofing**
0x4009	**Steam**

8.11.4.1.1. Bake Tag

This mode sets the device into baking mode for baking food items.

8.11.4.1.2. Convection Tag

This mode sets the device into convection mode which creates an airflow within the device during the cooking duration.

8.11.4.1.3. Grill Tag

This mode sets the device into grill mode for grilling food items. This is the same as Broil for many regions.

8.11.4.1.4. Roast Tag

This mode sets the device into roast mode for roasting food items.

8.11.4.1.5. Clean Tag

This mode sets the device into cleaning mode to clean the internal components of the appliance.

8.11.4.1.6. Convection Bake Tag

This mode sets the device into convection bake mode which creates an airflow within the device during the baking duration.

8.11.4.1.7. Convection Roast Tag

This mode sets the device into convection roast mode which creates an airflow within the device during the roasting duration.

8.11.4.1.8. Warming Tag

This mode sets the device into a warming mode which begins warming the cavity.

8.11.4.1.9. Proofing Tag

This mode sets the device into proofing mode which creates an environment ready for proofing.

8.11.5. Mode Constraints

The table below lists the changes relative to the Mode Base cluster for the fields of the ModeOptionStruct type. A blank field indicates no change.

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
2	**ModeTags**		1 to 8				

At least one entry in the SupportedModes attribute SHALL include the Bake mode tag in the ModeTags field list.

8.11.6. Mode Examples

A few examples of Oven modes and their mode tags are provided below.

- For the "Convection" mode, tags: 0x4001 (Convection)
- For the "Bake" mode, tags: 0x4000 (Bake)
- For the "Bake and Warm" mode, tags: 0x4000 (Bake), 0x4007 (Warming)
- For the "Convection Cook and Clean" mode, tags: 0x4001 (Convection), 0x4004 (Clean)

8.12. Microwave Oven Mode Cluster

This cluster is derived from the Mode Base cluster, defining additional mode tags and namespaced enumerated values for Microwave Oven devices.

8.12.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Revision	Description
1	Initial Release

8.12.2. Classification

Hierarchy	Role	Scope	PICS Code
Mode Base	Application	Endpoint	MWOM

8.12.3. Cluster ID

ID	Name
0x005E	Microwave Oven Mode

8.12.4. Attributes

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0000	**SupportedModes**						
0x0001	**CurrentMode**						
0x0002	**StartUpMode**						X
0x0003	**OnMode**						X

Exactly one entry in the SupportedModes attribute SHALL include the Normal mode tag in the ModeTags field.

The Normal and Defrost mode tags are mutually exclusive and SHALL NOT both be used together in a mode's ModeTags.

8.12.5. Commands

ID	Name	Direction	Response	Access	Conformance
0x00	**ChangeToMode**				X
0x01	**ChangeToModeResponse**				X

8.12.6. Derived Cluster Namespace

This namespace includes definitions for data associated exclusively with the derived cluster.

8.12.6.1. Mode Tags

The following table defines the derived cluster specific ModeTag values.

Mode Tag Value	Name	Summary
0x4000	**Normal**	The normal mode of operation
0x4001	**Defrost**	A mode optimized for defrosting foods

8.13. Microwave Oven Control Cluster

This cluster defines the requirements for the Microwave Oven Control cluster.

This cluster has dependencies with the Operational State and Microwave Oven Mode clusters. The Operational State cluster and the Microwave Oven Mode clusters, or derivatives of those clusters SHALL appear on the same endpoint as this cluster.

8.13.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Revision	Description
1	Initial Release

8.13.2. Classification

Hierarchy	Role	Scope	PICS Code
Base	Application	Endpoint	MWOCTRL

8.13.3. Cluster ID

ID	Name
0x005F	Microwave Oven Control

8.13.4. Features

This cluster SHALL support the FeatureMap bitmap attribute as defined below.

Bit	Code	Feature	Conformance	Summary
0	**PWRNUM**	PowerAsNumber	O.a	Power is specified as a unitless number or a percentage
1	**WATTS**	PowerInWatts	P, O.a	Power is specified in Watts
2	**PWRLMTS**	PowerNumberLimits	[PWRNUM]	Supports the limit attributes used with the PWRNUM feature

8.13.5. Attributes

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0000	**CookTime**	elapsed-s	1 to MaxCookTime		30	R V	M
0x0001	**MaxCookTime**	elapsed-s	1 to 86400	F	MS	R V	M

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0002	**PowerSetting**	uint8	desc		desc	R V	PWRNUM
0x0003	**MinPower**	uint8	1 to MaxPower	F	10	R V	PWRLMTS
0x0004	**MaxPower**	uint8	MinPower to 100	F	100	R V	PWRLMTS
0x0005	**PowerStep**	uint8	1 to (MaxPower - MinPower)	F	10	R V	PWRLMTS
0x0006	**SupportedWatts**	list[uint16]	1 to 10	F	MS	R V	P, WATTS
0x0007	**SelectedWattIndex**	uint8	desc		MS	R V	P, WATTS
0x0008	**WattRating**	uint16	all	F	MS	R V	O

8.13.5.1. CookTime Attribute

This attribute SHALL indicate the total cook time associated with the operation of the device.

This attribute SHALL remain unchanged during the operation of the oven unless the value is changed via a command or out-of-band action.

8.13.5.2. MaxCookTime Attribute

This attribute SHALL indicate the maximum value to which the CookTime attribute can be set.

8.13.5.3. PowerSetting Attribute

This attribute SHALL indicate the power level associated with the operation of the device.

If the MinPower, MaxPower, and PowerStep attributes are not supported, the minimum value of this attribute SHALL be 10, the maximum value of this attribute SHALL be 100, the value SHALL be in even multiples of 10, and the default value SHALL be 100.

If the MinPower, MaxPower, and PowerStep attributes are supported:

- The value of this attribute SHALL be between MinPower and MaxPower inclusive.
- The value of this attribute SHALL be an integer multiple of PowerStep.

8.13.5.4. MinPower Attribute

This attribute SHALL indicate the minimum power level that can be set on the server. The value of this attribute SHALL be less than or equal to the value of MaxPower. The value of this attribute

SHALL be an integer multiple of PowerStep.

8.13.5.5. MaxPower Attribute

This attribute SHALL indicate the maximum power level that can be set on the server. The value of this attribute SHALL be greater than or equal to the value of MinPower. The value of this attribute SHALL be an integer multiple of PowerStep.

8.13.5.6. PowerStep Attribute

This attribute SHALL indicate the increment of power that can be set on the server.

For example, if MinPower is 1, MaxPower is 10, and PowerSetting can be set to any integer between MinPower and MaxPower, PowerStep would be set to 1.

8.13.5.7. SupportedWatts Attribute

This attribute SHALL indicate the list of power levels (in W) supported by the server.

8.13.5.8. SelectedWattIndex Attribute

This attribute SHALL indicate the index into the list of SupportedWatts of the currently selected power setting.

The index SHALL be a valid index into the SupportedWatts list.

8.13.5.9. WattRating Attribute

This attribute SHALL indicate the rating, in Watts, of the microwave power of the oven.

Supporting this attribute can assist clients in suggesting cooking settings for various foods and beverages.

8.13.6. Commands

ID	Name	Direction	Response	Access	Conformance
0x00	**SetCookingParameters**	client ⇒ server	Y	O	M
0x01	**AddMoreTime**	client ⇒ server	Y	O	O

8.13.6.1. Command Responses Impacted By the Operational State Cluster

When the Operational State cluster or a cluster derived from it is included on the same endpoint as this cluster, the server MAY respond to commands defined in this cluster with an INVALID_IN_STATE response if the server is unable to accept those command due to restrictions imposed by the current operational state of the device or other factors.

8.13.6.2. SetCookingParameters Command

This command is used to set the cooking parameters associated with the operation of the device.

This command supports the following fields:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**CookMode**	uint8	desc		desc	O.a+
1	**CookTime**	elapsed-s	1 to MaxCookTime		30	O.a+
2	**PowerSetting**	uint8	MinPower to MaxPower		MaxPower	[PWRNUM].a+
3	**WattSettingIndex**	uint8	desc		MS	[WATTS].a+
4	**StartAfterSetting**	bool	all		false	O

8.13.6.2.1. CookMode Field

This field SHALL indicate the value to which the CurrentMode attribute of the Microwave Oven Mode cluster should be set. The value of this field SHALL be one from the list of SupportedModes from the Microwave Oven Mode cluster.

If this field is missing, the CurrentMode attribute SHALL be set to a mode having the Normal mode tag.

8.13.6.2.2. CookTime Field

This field SHALL indicate the CookTime associated with the operation of the device. The value of this field SHALL be subject to the constraints of the CookTime attribute of this cluster.

If this field is missing, the CookTime attribute SHALL be set to 30 seconds by the server.

8.13.6.2.3. PowerSetting Field

This field SHALL indicate the PowerSetting associated with the operation of the device. The value of this field SHALL be subject to the constraints of the PowerSetting attribute of this cluster. If the PowerSetting field does not conform to the constraints of the PowerSetting attribute, the server SHALL return a CONSTRAINT_ERROR status.

If this field is missing, the PowerSetting attribute SHALL be set to 100 if MaxPower is not supported by the server, otherwise it SHALL be set to MaxPower if the MaxPower attribute is supported by the server.

8.13.6.2.4. WattSettingIndex Field

This field SHALL indicate the value to which the SelectedWattIndex attribute is set. If the value of this field is greater than or equal to the length of the SupportedWatts attribute list, the server SHALL return a CONSTRAINT_ERROR status and the value of the SelectedWattIndex attribute SHALL be unchanged.

If this field is missing, the SelectedWattIndex attribute SHALL be set by the server to the index associated with the highest Watt setting for the selected CookMode.

8.13.6.2.5. StartAfterSetting Field

This field SHALL indicate whether or not oven operation SHALL be started when the command is received.

8.13.6.2.6. Effect on Receipt

If this command is received while the operational state of the server cannot support the command in that state, the server SHALL respond with an INVALID_IN_STATE response and the attributes and state SHALL remain unchanged. See Operational State Cluster for details on the operational states.

If this command is received and any fields sent with the command do not meet the constraints of any of the associated attributes (i.e. bearing the same name as the field or as described in the field description), the server SHALL respond with a response of CONSTRAINT_ERROR and the attributes and state SHALL remain unchanged.

If the StartAfterSetting field is present in the command but the Start command of the Operational State cluster or one of its derivatives on the same endpoint as this cluster is not supported, the server SHALL respond with a response of INVALID_COMMAND and the attributes and state SHALL remain unchanged.

Otherwise:

- The attributes associated with any included fields SHALL be set to the values of those fields.
- The attributes associated with any missing fields SHALL be set to the values as specified in descriptions of the missing fields.
- If the StartAfterSetting field is included:
  - If the value of StartAfterSetting is TRUE, oven operation SHALL start.
  - If the value of StartAfterSetting is FALSE, oven operation SHALL NOT start.

8.13.6.3. AddMoreTime Command

This command is used to add more time to the CookTime attribute of the server.

This command supports these fields:

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**TimeToAdd**	elapsed-s	1 to MaxCookTime			M

8.13.6.3.1. TimeToAdd Field

This field SHALL indicate the number of seconds to be added to the CookTime attribute.

8.13.6.3.2. Effect on Receipt

Upon receipt of this command, if the sum of the value of the TimeToAdd field and the current value of the CookTime attribute is greater than the MaxCookTime attribute, the server SHALL respond with a response of CONSTRAINT_ERROR and the command SHALL be ignored.

If this command is received while the operational state of the server cannot support the command in that state, the server SHALL respond with an INVALID_IN_STATE response. See Operational State Cluster for details on the operational states.

Otherwise, the server SHALL add the value of the TimeToAdd field to the value of the CookTime attribute of this cluster and the server SHALL add the value of the TimeToAdd field to the value of the CountdownTime attribute of the Operational State cluster if that cluster or a derivative is on the same endpoint as this cluster.

Chapter 9. Energy Management

The Cluster Library is made of individual chapters such as this one. See Document Control in the Cluster Library for a list of all chapters and documents. References between chapters are made using a *X.Y* notation where *X* is the chapter and *Y* is the sub-section within that chapter.

9.1. General Description

9.1.1. Introduction

The clusters specified in this chapter are for use typically in Energy Management applications with associated security controls at the application layer. These clusters may be used in any application domain.

9.1.2. Cluster List

This section lists the Energy Management specific clusters as specified in this chapter.

Table 42. Overview of the Energy Management Clusters

Cluster ID	Cluster Name	Description
0x009F	Device Energy Management Mode	Commands and attributes for setting the mode of devices with energy management capability
0x0098	Device Energy Management	Generic cluster for enabling power adjustment, sharing power forecasts and modifying power forecasts for devices
0x0099	Energy EVSE	Commands and attributes for controlling an EVSE
0x009D	Energy EVSE Mode	Commands and attributes for setting the mode of an EVSE
0x009B	Energy Preference	Attributes and commands for expressing user preferences around energy consumption

9.2. Device Energy Management Cluster

This cluster allows a client to manage the power draw of a device. An example of such a client could be an Energy Management System (EMS) which controls an Energy Smart Appliance (ESA).

In most deployments the EMS will be the client, and the ESA will host the Energy Management Cluster server.

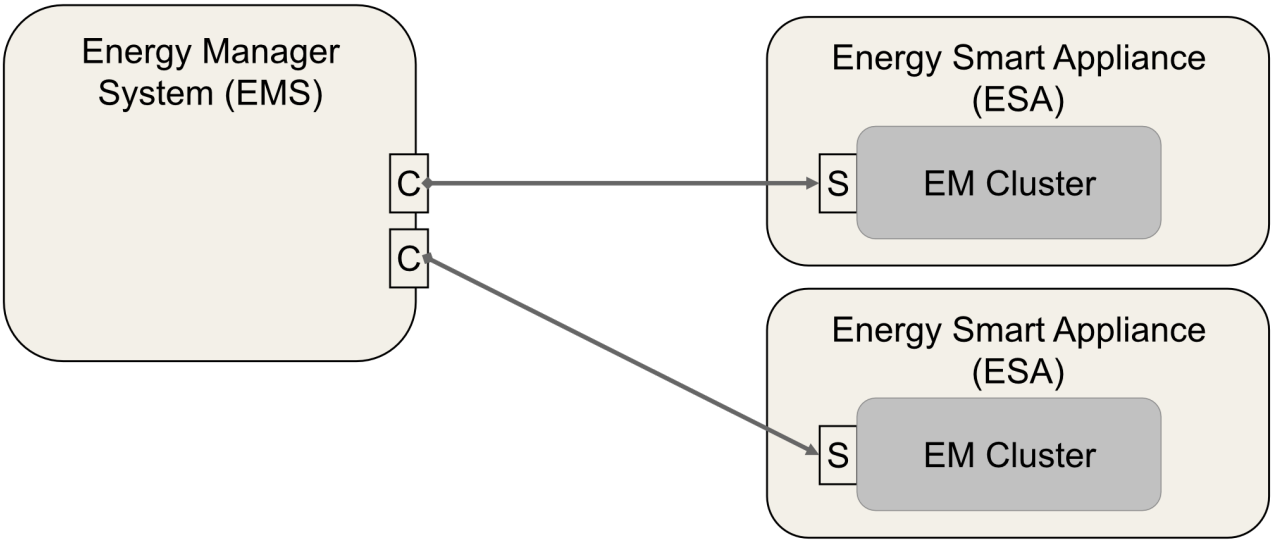


Figure 15. Example of the how an EMS is a client of multiple ESAs Energy Management clusters.

This cluster is intended to be generic in nature and could apply to any electrical load or generator (e.g. a Battery Electric Storage System - BESS, solar PV inverter, EVSE, HVAC, heat pump, hot water heater, white goods appliances etc).

It consists of the following areas which SHALL be supported by all devices implementing this cluster:

- Description of ESA and its capabilities & power limits (sometimes referred to as a nameplate)
- Current state of operation (including user opt-out, safety limitations / alarms)

There are some optional capabilities that some ESAs may be able to offer:

- Ability to control the load or generation
- Forecast data, including when it can be flexible (i.e. modify the power or time period)
- The ability to have their power profile adjusted by an EMS, and to provide an updated Forecast back to the EMS.

This allows the EMS to manage multiple home loads and where ESAs can be flexible, continuously optimizing the home energy to minimize cost, reduce CO2 impact, maximize self-consumption of solar PV and provide Demand Side Response (DSR) Grid services.

It is likely that the ESA may also use the Pricing Cluster to obtain incentive signals such as 'grid carbon intensity', 'time of use' or 'type of use' tariffs to schedule its operation to run at the cheapest and greenest times.

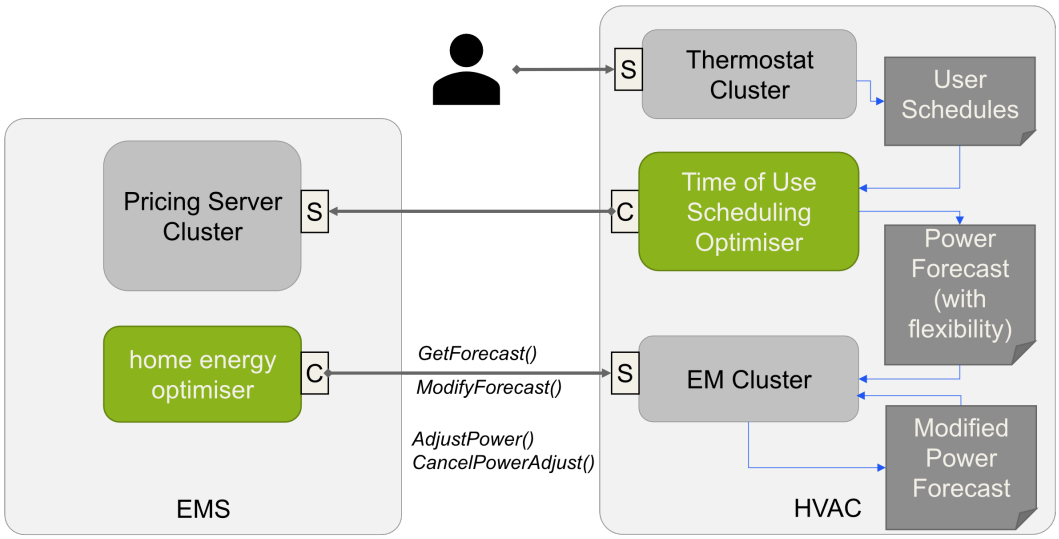


Figure 16. Example of the how an HVAC may use multiple clusters

NOTE Grid Services are market dependent and will use other protocols ([OpenADR] / [IEEE2030.5]) to communicate grid events to the EMS. These are outside the scope of Matter.

NOTE Different markets may follow different approaches, but the UK [PAS1878] and [EUCodeOfConduct] give examples of how ESAs may be mandated to support these features in the future.

NOTE Support of Device Energy Management Cluster is provisional.

9.2.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Revision	Description
1	Initial Release
2	Updates after 0.7 Ballot review
3	Updates to match cluster spec updates

9.2.2. Classification

Hierarchy	Role	Scope	PICS Code
Base	Application	Endpoint	DEM

9.2.3. Cluster ID

ID	Name	Conformance
0x0098	Device Energy Management	P

9.2.4. Features

This cluster SHALL support the FeatureMap bitmap attribute as defined below.

Bit	Code	Feature	Conformance	Summary
0	**PA**	PowerAdjustment	O.a+	Allows an EMS to make a temporary power adjustment (within the limits offered by the ESA).
1	**PFR**	PowerForecastReporting	(STA|PAU|FA|CON)&(!SFR),O.a+	Allows an ESA to advertise its indicative future power consumption vs time.
2	**SFR**	StateForecastReporting	(STA|PAU|FA|CON)&(!PFR),O.a+	Allows an ESA to advertise its indicative future state vs time.
3	**STA**	StartTimeAdjustment	O.a+	Allows an EMS to delay an ESA's planned operation.
4	**PAU**	Pausable	O.a+	Allows an EMS to pause an ESA's planned operation.
5	**FA**	ForecastAdjustment	P,O.a+	Allows an EMS to adjust an ESA's planned operation.
6	**CON**	ConstraintBasedAdjustment	P,O.a+	Allows an EMS to request constraints to an ESA's planned operation.

9.2.4.1. PowerAdjustment Feature

For Energy Smart Appliances (ESA) the definition of being 'smart' mandates that they can report their current power adjustment capability and have an EMS request a temporary adjustment. This may typically be to curtail power requirements during peak periods, but can also be used to turn on an ESA if there is excess renewable or local generation (Solar PV).

For example, a home may have solar PV which often produces more power than the home requires,

resulting in the excess power flowing into the grid. This excess power naturally fluctuates when clouds pass overhead and other loads in the home are switched on and off.

EVSE Example: An EMS may therefore be able to turn on the EVSE (if the vehicle is plugged in) and can start charging the vehicle, and periodically modify the charging power depending on PV generation and other home loads, so as to minimize import and export to the grid.

9.2.4.2. PowerForecastReporting Feature

For Energy Smart Appliances (ESA) the definition of being 'smart' implies that they can report their indicative forecast power demands or generation, to a greater or lesser extent. For some ESAs this is highly predictable (in terms of both power and time), in other appliances this is more challenging and only a basic level of forecast is possible.

Forecasts are defined from a current time, using a slot format, where the slot is akin to a relatively constant operating mode.

Washing machine example: a washing machine may have stages of a washing cycle: heating, tumbling, rinse and spin stages. At each stage, the approximate minimum and maximum power consumption may be known, as well as the duration of that stage.

In some circumstances the ESA may allow the stage to be delayed or paused (subject to safety and manufacturer's discretion and user preferences).

Typically, appliances with a heating element cannot have their power consumption adjusted and can only be paused or delayed.

Some ESAs may not be flexible other than a delayed cycle start (for example, once the washing cycle has been started then they run continuously until the cycle completes).

Appliances that only support the PowerForecastReporting and not any of the adjustment features may indicate that they are not flexible in the forecast slot format.

The PowerForecastReporting and the adjustment features aim to align to the [SAREF4ENER] ontology.

Inverter driven ESAs: some inverter driven ESAs can consume or generate a variable amount of power.

For example, a single phase EVSE can be adjusted in the range of 6-32Amps in 0.6 Amp steps in EU or on a hardwired 120V supply in the range of 6-15 Amps in US.

For example, a home battery may be adjusted to charge or discharge in steps of 1W.

For example, a heat pump may be able to modulate its compressor inverter between 20-100% of its rated power.

The ESA indicates its power adjustment range and its nominal power consumption as part of its Forecast.

9.2.4.3. StateForecastReporting Feature

Some ESAs do not know their actual power consumption, but do know the state of operation. Like the PowerForecastingReporting feature, this uses the same slot structure mechanism to indicate a change in state vs time.

An external observing EMS may have access to real-time meter readings, and could learn the typical power consumption based on the advertised internal state of the ESA.

To enable this capability, the ESA SHALL report its internal operational state using an manufacturer specific value.

Once the EMS has built a model of the state vs observed power consumption, it may request a forecast adjustment for particular times of the day, encouraging the ESA to use power at alternative times.

9.2.4.4. StartTimeAdjustment Feature

ESAs which support the Start Time Adjustment feature, allow an EMS to recommend a change to the start time of the energy transfer that the ESA has previously suggested it would use.

Washing machine example: A Washing Machine may have been set to start a wash cycle at 9pm when the variable tariff normally reduces.

However, the EMS is aware that a grid event has occurred, making it cheaper to run the cycle at a later time, but the washing machine is not aware of this.

The EMS first requests the Forecast data from each of its registered ESAs. It determines that the washing machine has a power profile suggesting it will start the wash cycle at 9pm, but the EMS now knows that the grid event means it will be cheaper to delay the start until 11pm.

The EMS can then optimize the cost by asking the washing machine to delay starting the wash cycle until 11pm.

It does this by sending a StartTimeAdjustRequest to the washing machine to request delaying the start of the washing cycle.

9.2.4.5. Pausable Feature

ESAs which support the Pausable feature, allow an EMS to recommend a pause in the middle of a forecast power profile that the ESA is currently using.

Washing machine example: A Washing Machine is in operation, and starting its water heating step.

However, the EMS becomes aware from the smart meter that the total home load on the grid is close to exceeding its allowed total grid load.

The EMS first requests the Forecast data from each of its registered ESAs. It determines that the washing machine has a power profile suggesting its current step in the wash cycle is using power to heat the water, but that this step can be paused.

The EMS can then reduce the grid load by asking the washing machine to pause the wash cycle for a short duration.

It does this by sending a PauseRequest to the washing machine to request pausing the current step of the forecast power usage for a period to allow other home loads to finish before resuming the washing cycle.

9.2.4.6. ForecastAdjustment Feature

ESAs which support the Forecast adjustment feature, allow an EMS to recommend a change to the start, duration and/or power level limits of the steps of the power profile that the ESA has previously suggested it would use.

Heat pump and Solar PV example: A heat pump may have the ability to heat hot water as well as heating the home. The heat pump scheduling system may have determined that the home will be unoccupied during the day, or that the indoor temperature is above the set-point and so it knows that it will not need to heat the home.

However, the hot water tank is likely to need to be reheated before the homeowner comes home in the evening. The heat pump is not aware that the property also has a solar PV inverter which is also an ESA that is communicating with the EMS.

The EMS first requests the Forecast data from each of its registered ESAs. It determines that the heat pump has a power profile suggesting it needs to heat hot water around 6pm. The solar PV inverter has forecast that it will generate 3.6kW of power during the middle of the day and into the afternoon before the sun goes down.

The EMS can then optimize the home considering other non-ESA loads and can ask the heat pump to heat the hot water around 3pm when it has forecast that excess solar power will be available.

It does this by sending a ModifyForecastRequest to the heat pump and asks the heat pump to expect to run at a lower power consumption (within the solar excess power) which requires the heat pump to run for a longer duration to achieve its required energy demand.

9.2.4.7. ConstraintBasedAdjustment Feature

ESAs which support the Constraint-Based Adjustment feature allow an EMS to inform the ESA of periods during which power usage should be modified (for example when the EMS has been made aware that the grid supplier has requested reduced energy usage due to overall peak grid demand) and may cause the ESA to modify the intended power profile has previously suggested it would use.

EVSE example: An EVSE scheduling system may have determined that the vehicle would be charged starting at a moderate rate at 1am, so that it has enough charge by the time it is needed later that morning.

However, the DSR service provider has informed the EMS that due to high forecast winds it is now forecast that there will be very cheap energy available from wind generation between 2am and 3am.

The EMS first requests the Forecast data from each of its registered ESAs. It determines that the

EVSE has a power profile suggesting it plans to start charging the vehicle at 1am.

The EMS can then try to reduce the cost of charging the EV by informing the EVSE of the desire to increase the charging between scheduled times.

It does this by sending a RequestConstraintBasedForecast to the EVSE and asks it to run at a higher NominalPower consumption during the constraint period, which may require it to decrease its charge rate outside the constraint period to achieve its required energy demand.

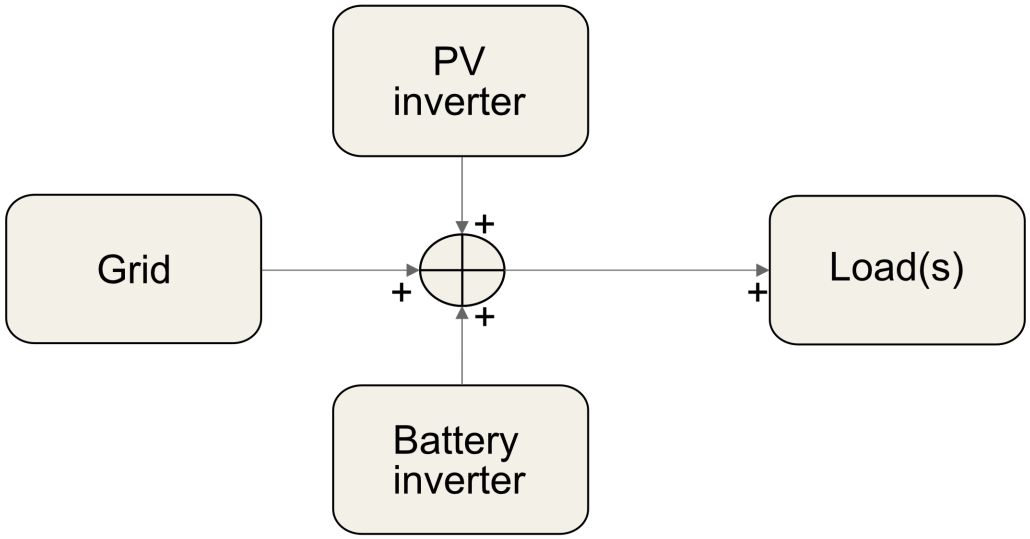
9.2.5. Dependencies

This cluster does not report electrical power and electrical energy. Devices that use this cluster SHALL also support the Electrical Power Measurement and optionally support the Electrical Energy Measurement cluster to allow an energy management system to perform its role. See Device Type library for device specific details.

9.2.6. Definitions

9.2.6.1. Power

Power is defined in the main specification (see Data Model) in units of milliwatts. It is a signed value, where positive values indicate the direction of current flow towards a load.



NOTE: Charging the battery implies a negative power wrt. the loads.

Figure 17. Power flows towards loads

Solar PV inverter example: A solar PV inverter is normally connected to the home loads, so its generation power will be indicated as a positive integer. Note that at night (when there is no solar production) the PV inverter will consume some standby power, and this will result in a small negative power reading.

Grid Power: Power from the grid typically flows towards the home loads, so positive power values are indicating that current is flowing into the home. Negative values indicate power is flowing back towards the grid.

EVSE example: An EVSE provides power to the EV (when charging) so a positive value indicates that the EV is charging, and negative value indicates that the EV is discharging.

BESS example: A battery storage inverter normally provides power to loads when discharging, so positive power indicates discharging (to a load), and negative power indicates charging.

Washing Machine example: A washing machine only consumes power (i.e. is a load), so it will always have a positive power value.

9.2.6.2. Energy

Energy is defined in the main specification (see Data Model). This is defined in units of mWh (milli-watt-hours). It is signed value, where positive values indicate the direction of current flow towards a load.

9.2.7. Data Types

9.2.7.1. CostTypeEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**Financial**	Financial cost	M
1	**GHGEmissions**	Grid CO2e grams cost	M
2	**Comfort**	Consumer comfort impact cost	M
3	**Temperature**	Temperature impact cost	M

9.2.7.1.1. Financial Value

This value SHALL indicate that the cost is related to the financial cost to provide the energy.

9.2.7.1.2. GHGEmissions Value

This value SHALL indicate that the cost is related to greenhouse gas emissions (in grams of CO2e).

9.2.7.1.3. Comfort Value

This value SHALL indicate that the cost is related to some abstract sense of comfort expressed by the consumer; a higher value indicates more discomfort. For example, a consumer may be more comfortable knowing that their EV is charged earlier in the day in case there is a sudden need to depart and drive to the hospital. Or the consumer may feel inconvenienced by the fact that they need to wait for the washing machine to finish its load so that they can use it again.

9.2.7.1.4. Temperature Value

This value SHALL indicate that the cost is related to the temperature of the home or water being at its setpoint. Some consumers may be more sensitive to being too hot or too cold.

This is expressed in degrees Celsius.

9.2.7.2. ESATypeEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**EVSE**	EV Supply Equipment	O
1	**SpaceHeating**	Space heating appliance	O
2	**WaterHeating**	Water heating appliance	O
3	**SpaceCooling**	Space cooling appliance	O
4	**SpaceHeatingCooling**	Space heating and cooling appliance	O
5	**BatteryStorage**	Battery Electric Storage System	O
6	**SolarPV**	Solar PV inverter	O
7	**FridgeFreezer**	Fridge / Freezer	O
8	**WashingMachine**	Washing Machine	O
9	**Dishwasher**	Dishwasher	O
10	**Cooking**	Cooking appliance	O
11	**HomeWaterPump**	Home water pump (e.g. drinking well)	O
12	**IrrigationWaterPump**	Irrigation water pump	O
13	**PoolPump**	Pool pump	O
255	**Other**	Other appliance type	O

9.2.7.3. ESAStateEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**Offline**	The ESA is not available to the EMS (e.g. start-up, maintenance mode)	M
1	**Online**	The ESA is working normally and can be controlled by the EMS	M

Value	Name	Summary	Conformance
2	**Fault**	The ESA has developed a fault and cannot provide service	M
3	**PowerAdjustActive**	The ESA is in the middle of a power adjustment event	PA
4	**Paused**	The ESA is currently paused by a client using the PauseRequest command	[FA]

9.2.7.4. OptOutStateEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**NoOptOut**	The user has not opted out of either local or grid optimizations	M
1	**LocalOptOut**	The user has opted out of local EMS optimizations only	M
2	**GridOptOut**	The user has opted out of grid EMS optimizations only	M
3	**OptOut**	The user has opted out of all external optimizations	M

9.2.7.5. CauseEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**NormalCompletion**	The ESA completed the power adjustment as requested	O
1	**Offline**	The ESA was set to offline	O
2	**Fault**	The ESA has developed a fault could not complete the adjustment	O

Value	Name	Summary	Conformance
3	**UserOptOut**	The user has disabled the ESA's flexibility capability	O
4	**Cancelled**	The adjustment was cancelled by a client	O

9.2.7.6. AdjustmentCauseEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**LocalOptimization**	The adjustment is to optimize the local energy usage	O
1	**GridOptimization**	The adjustment is to optimize the grid energy usage	O

9.2.7.7. ForecastUpdateReasonEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**InternalOptimization**	The update was due to internal ESA device optimization	O
1	**LocalOptimization**	The update was due to local EMS optimization	O
2	**GridOptimization**	The update was due to grid optimization	O

9.2.7.8. CostStruct Type

This indicates a generic mechanism for expressing cost to run an appliance, in terms of financial, GHG emissions, comfort value etc.

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0	**CostType**	CostTypeEnum	all		0		M
1	**Value**	int32	all		0		M
2	**DecimalPoints**	uint8	all		0		M

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
3	**Currency**	uint16	max 999		0		O

9.2.7.8.1. CostType Field

This field SHALL indicate the type of cost being represented (see CostTypeEnum).

9.2.7.8.2. Value Field

This field SHALL indicate the value of the cost. This may be negative (indicating that it is not a cost, but a free benefit).

For example, if the Value was -302 and DecimalPoints was 2, then this would represent a benefit of 3.02.

9.2.7.8.3. DecimalPoints Field

This field SHALL indicate the number of digits to the right of the decimal point in the Value field. For example, if the Value was 102 and DecimalPoints was 2, then this would represent a cost of 1.02.

9.2.7.8.4. Currency Field

Indicates the currency for the value in the Value field. The value of the currency field SHALL match the values defined by [ISO 4217].

This is an optional field. It SHALL be included if CostType is Financial.

9.2.7.9. PowerAdjustStruct Type

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0	**MinPower**	power-mW	all		0		M
1	**MaxPower**	power-mW	all		0		M
2	**MinDuration**	elapsed-s	all		0		M
3	**MaxDuration**	elapsed-s	all				M

9.2.7.9.1. MinPower Field

This field SHALL indicate the minimum power that the ESA can have its power adjusted to.

Note that this is a signed value. Negative values indicate power flows away from loads (e.g. charging a battery inverter).

MinPower SHALL be less than MaxPower.

9.2.7.9.2. MaxPower Field

This field SHALL indicate the maximum power that the ESA can have its power adjusted to.

Note that this is a signed value. Negative values indicate power flows away from loads (e.g. charging a battery inverter).

MinPower SHALL be less than MaxPower.

For example, if the charging current of an EVSE can be adjusted within the range of 6A to 32A on a 230V supply, then the power adjustment range is between 1380W and 7360W. Here the MinPower would be 1380W, and MaxPower would be 7360W.

For example, if a battery storage inverter can discharge between 0 to 3000W towards a load, then its MinPower would be 0W and its MaxPower would be 3000W.

In another example, if a battery storage inverter can charge its internal battery, between 0W and 2000W. Here power is flowing away from the home loads, so the power values need to be negative. As such the MinPower becomes -2000W and MaxPower becomes 0W.

9.2.7.9.3. MinDuration field

This field SHALL indicate the minimum duration, in seconds, that a controller may invoke an ESA power adjustment. Manufacturers may use this to as an anti-cycling capability to avoid controllers from rapidly making power adjustments.

Note that MinDuration SHALL be less than MaxDuration.

9.2.7.9.4. MaxDuration field

This field SHALL indicate the maximum duration, in seconds, that a controller may invoke an ESA power adjustment. Manufacturers may use this to protect the user experience, to avoid over heating of the ESA, ensuring that there is sufficient headroom to use or store energy in the ESA or for any other reason.

Note that MinDuration SHALL be less than MaxDuration.

9.2.7.10. ForecastStruct Type

This indicates a list of 'slots' describing the overall timing of the ESA's planned energy and power use, with different power and energy demands per slot. For example, slots might be used to describe the distinct stages of a washing machine cycle.

Where an ESA does not know the actual power and energy use of the system, it may support the SFR feature and instead report its internal state.

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0	**ForecastId**	uint16	all		0		M

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
1	**ActiveSlotNumber**	uint16	all	X	0		M
2	**StartTime**	epoch-s	all				M
3	**EndTime**	epoch-s	all				M
4	**EarliestStartTime**	epoch-s	all	X			STA
5	**LatestEndTime**	epoch-s	all				STA
6	**IsPauseable**	bool	all				M
7	**Slots**	list[SlotStruct]	max 10				M
8	**ForecastUpdateReason**	ForecastUpdateReasonEnum	all				M

9.2.7.10.1. ForecastId Field

This field SHALL indicate the sequence number for the current forecast. If the ESA updates a forecast, it shall monotonically increase this value.

The ESA does not need to persist this value across reboots, since the EMS SHOULD be able to detect that any previous subscriptions are lost if a device reboots. The loss of a subscription and subsequent re-subscription allows the EMS to learn about any new forecasts.

The value of ForecastId is allowed to wrap.

9.2.7.10.2. ActiveSlotNumber Field

This field SHALL indicate which element of the Slots list is currently active in the Forecast sequence. A null value indicates that the sequence has not yet started.

9.2.7.10.3. StartTime Field

This field SHALL indicate the planned start time, in UTC, for the entire Forecast.

9.2.7.10.4. EndTime Field

This field SHALL indicate the planned end time, in UTC, for the entire Forecast.

9.2.7.10.5. EarliestStartTime Field

This field SHALL indicate the earliest start time, in UTC, that the entire Forecast can be shifted to.

A null value indicates that it can be started immediately.

9.2.7.10.6. LatestEndTime Field

This field SHALL indicate the latest end time, in UTC, for the entire Forecast.

e.g. for an EVSE charging session, this may indicate the departure time for the vehicle, by which time the charging session must end.

9.2.7.10.7. IsPauseable Field

This field SHALL indicate that some part of the Forecast can be paused. It aims to allow a client to read this flag and if it is false, then none of the slots contain SlotIsPausable set to true. This can save a client from having to check each slot in the list.

9.2.7.10.8. Slots Field

This field SHALL contain a list of SlotStructs.

It SHALL contain at least 1 entry, and a maximum of 10.

9.2.7.10.9. ForecastUpdateReason Field

This field SHALL contain the reason the current Forecast was generated.

9.2.7.11. SlotStruct

This indicates a specific stage of an ESA's operation.

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0	**MinDuration**	elapsed-s	all				M
1	**MaxDuration**	elapsed-s	all				M
2	**DefaultDuration**	elapsed-s	all				M
3	**ElapsedSlotTime**	elapsed-s	all				M
4	**RemainingSlotTime**	elapsed-s	all				M
5	**SlotIsPauseable**	bool	all				PAU
6	**MinPauseDuration**	elapsed-s	all				PAU

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
7	**MaxPauseDuration**	elapsed-s	all				PAU
8	**ManufacturerESAState**	uint16	all				SFR
9	**NominalPower**	power-mW	all				PFR
10	**MinPower**	power-mW	all				PFR
11	**MaxPower**	power-mW	all				PFR
12	**NominalEnergy**	energy-mWh	all				PFR
13	**Costs**	list[CostStruct]	max 5				O
14	**MinPowerAdjustment**	power-mW	all				FA & PFR
15	**MaxPowerAdjustment**	power-mW	all				FA & PFR
16	**MinDurationAdjustment**	elapsed-s	all				FA & (PFR|SFR)
17	**MaxDurationAdjustment**	elapsed-s	all				FA & (PFR|SFR)

9.2.7.11.1. MinDuration Field

This field SHALL indicate the minimum time (in seconds) that the appliance expects to be in this slot for.

9.2.7.11.2. MaxDuration Field

This field SHALL indicate the maximum time (in seconds) that the appliance expects to be in this slot for.

9.2.7.11.3. DefaultDuration Field

This field SHALL indicate the expected time (in seconds) that the appliance expects to be in this slot for.

9.2.7.11.4. ElapsedSlotTime Field

This field SHALL indicate the time (in seconds) that has already elapsed whilst in this slot. If the slot has not yet been started, then it SHALL be 0. Once the slot has been completed, then this reflects how much time was spent in this slot.

When subscribed to, a change in this field value SHALL NOT cause the Forecast attribute to be updated since this value may change every 1 second.

When the Forecast attribute is read, then this value SHALL be the most recent value.

9.2.7.11.5. RemainingSlotTime Field

This field SHALL indicate the time (in seconds) that is estimated to be remaining.

Note that it may not align to the DefaultDuration - ElapsedSlotTime since an appliance may have revised its planned operation based on conditions.

When subscribed to, a change in this field value SHALL NOT cause the Forecast attribute to be updated, since this value may change every 1 second.

Note that if the ESA is currently paused, then this value SHALL not change.

When the Forecast attribute is read, then this value SHALL be the most recent value.

9.2.7.11.6. SlotIsPausable Field

This field SHALL indicate whether this slot can be paused.

9.2.7.11.7. MinPauseDuration Field

This field SHALL indicate the shortest period that the slot can be paused for. This can be set to avoid controllers trying to pause ESAs for short periods and then resuming operation in a cyclic fashion which may damage or cause excess energy to be consumed with restarting of an operation.

9.2.7.11.8. MaxPauseDuration Field

This field SHALL indicate the longest period that the slot can be paused for.

9.2.7.11.9. ManufacturerESAState Field

This field SHALL indicate a manufacturer defined value indicating the state of the ESA.

This may be used by an observing EMS which also has access to the metering data to ascertain the typical power drawn when the ESA is in a manufacturer defined state.

Some appliances, such as smart thermostats, may not know how much power is being drawn by the HVAC system, but do know what they have asked the HVAC system to do.

Manufacturers can use this value to indicate a variety of states in an unspecified way. For example, they may choose to use values between 0-100 as a percentage of compressor modulation, or could use these values as Enum states meaning heating with fan, heating without fan etc.

NOTE An ESA SHALL always use the same value to represent the same operating state.

By providing this information a smart EMS may be able to learn the observed power draw when the ESA is put into a specific state. It can potentially then use the information in the PowerForecastReporting data to predict the power draw from the appliance and potentially ask it to modify its timing via one of the adjustment request commands, or adjust other ESAs power to compensate.

9.2.7.11.10. NominalPower Field

This field SHALL indicate the expected power that the appliance will use during this slot. It may be considered the average value over the slot, and some variation from this would be expected (for example, as it is ramping up).

9.2.7.11.11. MinPower Field

This field SHALL indicate the lowest power that the appliance expects to use during this slot. (e.g. during a ramp up it may be 0W).

Some appliances (e.g. battery inverters which can charge and discharge) may have a negative power.

9.2.7.11.12. MaxPower Field

This field SHALL indicate the maximum power that the appliance expects to use during this slot. (e.g. during a ramp up it may be 0W). This field ignores the effects of short-lived inrush currents.

Some appliances (e.g. battery inverters which can charge and discharge) may have a negative power.

9.2.7.11.13. NominalEnergy Field

This field SHALL indicate the expected energy that the appliance expects to use or produce during this slot.

Some appliances (e.g. battery inverters which can charge and discharge) may have a negative energy.

9.2.7.11.14. Costs Field

This field SHALL indicate the current estimated cost for operating.

For example, if the device has access to an Energy pricing server it may be able to use the tariff to estimate the cost of energy for this slot in the power forecast.

When an Energy Management System requests a change in the schedule, then the device MAY suggest a change in the cost as a result of shifting its energy. This can allow a demand side response service to be informed of the relative cost to use energy at a different time.

The Costs field is a list of CostStruct structures which allows multiple CostTypeEnum and Values to be shared by the energy appliance. These could be based on GHG emissions, comfort value for the consumer etc.

For example, comfort could be expressed in abstract units or in currency. A water heater that is heated earlier in the day is likely to lose some of its heat before it is needed, which could require a top-up heating event to occur later in the day (which may incur additional cost).

If the ESA cannot calculate its cost for any reason (such as losing its connection to a Price server) it may omit this field. This is treated as extra meta data that an EMS may use to optimize a system.

9.2.7.11.15. MinPowerAdjustment Field

This field SHALL indicate the minimum power that the appliance can be requested to use.

For example, some EVSEs cannot be switched on to charge below 6A which may equate to ~1.3kW in EU markets. If the slot indicates a NominalPower of 0W (indicating it is expecting to be off), this allows an ESA to indicate it could be switched on to charge, but this would be the minimum power limit it can be set to.

9.2.7.11.16. MaxPowerAdjustment Field

This field SHALL indicate the maximum power that the appliance can be requested to use.

For example, an EVSE may be limited by its electrical supply to 32A which would be ~7.6kW in EU markets. If the slot indicates a NominalPower of 0W (indicating it is expecting to be off), this allows an ESA to indicate it could be switched on to charge, but this would be the maximum power limit it can be set to.

9.2.7.11.17. MinDurationAdjustment Field

This field SHALL indicate the minimum time, in seconds, that the slot can be requested to shortened to.

For example, if the slot indicates a NominalPower of 0W (indicating it is expecting to be off), this would allow an ESA to specify the minimum time it could be switched on for. This is to help protect the appliance from being damaged by short cycling times.

For example, a heat pump compressor may have a minimum cycle time of order a few minutes.

9.2.7.11.18. MaxDurationAdjustment Field

This field SHALL indicate the maximum time, in seconds, that the slot can be requested to extended to.

For example, if the slot indicates a NominalPower of 0W (indicating it is expecting to be off), this allows an ESA to specify the maximum time it could be switched on for. This may allow a battery or water heater to indicate the maximum duration that it can charge for before becoming full. In the case of a battery inverter which can be discharged, it may equally indicate the maximum time the battery could be discharged for (at the MaxPowerAdjustment power level).

9.2.7.12. SlotAdjustmentStruct Type

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0	**SlotIndex**	uint8	desc				M
1	**NominalPower**	power-mW	desc				M
2	**Duration**	elapsed-s	desc				M

9.2.7.12.1. SlotIndex Field

This field SHALL indicate the index into the Slots list within the Forecast that is to be modified. It SHALL be less than the actual length of the Slots list (implicitly it must be in the range 0 to 9 based on the maximum length of the Slots list constraint).

9.2.7.12.2. NominalPower Field

This field SHALL indicate the new requested power that the ESA SHALL operate at. It MUST be between the MinPowerAdjustment and MaxPowerAdjustment for the slot as advertised by the ESA.

9.2.7.12.3. Duration Field

This field SHALL indicate the new requested duration, in seconds, that the ESA SHALL extend or shorten the slot duration to. It MUST be between the MinDurationAdjustment and MaxDurationAdjustment for the slot as advertised by the ESA.

9.2.7.13. ConstraintsStruct Type

The ConstraintsStruct allows a client to inform an ESA about a constraint period (such as a grid event, or perhaps excess solar PV). The format allows the client to suggest that the ESA can either turn up its energy consumption, or turn down its energy consumption during this period.

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0	**StartTime**	epoch-s	desc				M
1	**Duration**	elapsed-s	0 to 86400				M
2	**NominalPower**	power-mW	desc				PFR
3	**MaximumEnergy**	energy-mWh	all				PFR
4	**LoadControl**	int8	all				SFR

9.2.7.13.1. StartTime Field

This field SHALL indicate the start time of the constraint period that the client wishes the ESA to compute a new Forecast.

This value is in UTC and MUST be in the future.

9.2.7.13.2. Duration Field

This field SHALL indicate the duration of the constraint in seconds.

9.2.7.13.3. NominalPower Field

This field SHALL indicate the nominal power that client wishes the ESA to operate at during the constrained period. It MUST be between the AbsMinPower and AbsMaxPower attributes as advertised by the ESA if it supports PFR.

This is a signed value and can be used to indicate charging or discharging.

If the ESA does NOT support PFR this value SHALL be ignored by the ESA.

9.2.7.13.4. MaximumEnergy Field

This field SHALL indicate the maximum energy that can be transferred to or from the ESA during the constraint period.

This is a signed value and can be used to indicate charging or discharging.

If the ESA does NOT support PFR this value MAY be ignored by the ESA.

9.2.7.13.5. LoadControl Field

This field SHALL indicate the turn up or turn down nature that the grid wants as the outcome by the ESA during the constraint period.

This is expressed as a signed value between -100 to +100. A value of 0 would indicate no bias to using more or less energy. A negative value indicates a request to use less energy. A positive value indicates a request to use more energy.

Note that the mapping between values and operation is manufacturer specific.

9.2.8. Attributes

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0000	**ESAType**	ESATypeEnum	all	F	Other	R V	M
0x0001	**ESACanGenerate**	bool	all	F	false	R V	M
0x0002	**ESAState**	ESAStateEnum	desc		0	R V	M
0x0003	**AbsMinPower**	power-mW	all		0	R V	M

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0004	**AbsMaxPower**	power-mW	all		0	R V	M
0x0005	**PowerAdjustmentCapability**	list[PowerAdjustStruct]	max 8	X	null	R V	PA
0x0006	**Forecast**	ForecastStruct	all	X	null	R V	PFR | SFR
0x0007	**OptOutState**	OptOutStateEnum	desc		0	R V	PA | STA | PAU | FA | CON

9.2.8.1. ESAType Attribute

This attribute SHALL indicate the type of ESA.

This attribute enables an EMS to understand some of the basic properties about how the energy may be consumed, generated, and stored by the ESA.

For example, the heat energy converted by a heat pump will naturally be lost through the building to the outdoor environment relatively quickly, compared to storing heat in a well-insulated hot water tank. Similarly, battery storage and EVs can store electrical energy for much longer durations.

This attribute can also help the EMS display information to a user and to make basic assumptions about typical best use of energy. For example, an EVSE may not always have an EV plugged in, so knowing the type of ESA that is being controlled can allow advanced energy management strategies.

9.2.8.2. ESACanGenerate Attribute

This attribute SHALL indicate whether the ESA is classed as a generator or load. This allows an EMS to understand whether the power values reported by the ESA need to have their sign inverted when dealing with forecasts and adjustments.

For example, a solar PV inverter (being a generator) may produce positive values to indicate generation, however an EMS when predicting the total home load would need to subtract these positive values from the loads to compute a net import at the grid meter.

For example, a home battery storage system (BESS) which needs to charge the battery and then discharge to the home loads, would be classed as a generator. These types of devices SHALL have this field set to true. When generating its forecast or advertising its PowerAdjustmentCapability, the power values shall be positive to indicate discharging to the loads in the home, and negative to indicate when it is charging its battery.

```
GRID meter = Σ LoadPowers - Σ GeneratorPowers
```

Example:

Home has the following loads:	
Water Heater:	3000W
TV:	200W
FridgeFreezer:	200W
Total:	**3400W**

Home has the following generators (ESACanGenerate = true)	
Solar:	2500W
BESS:	900W
Total:	**3400W**

GRID Meter:	
Σ Loads:	3400W
Σ Generators:	-3400W
Total:	**0W**

9.2.8.3. ESAState Attribute

This attribute SHALL indicate the current state of the ESA.

If the ESA is in the Offline or Fault state it cannot be controlled by an EMS, and may not be able to report its Forecast information. An EMS may subscribe to the ESAState to get notified about changes in operational state.

The ESA may have a local user interface to allow a service technician to put the ESA into Offline mode, for example to avoid the EMS accidentally starting or stopping the appliance when it is being serviced or tested.

9.2.8.4. AbsMinPower Attribute

This attribute SHALL indicate the minimum electrical power that the ESA can consume when switched on. This does not include when in power save or standby modes.

Note that for Generator ESAs that can charge an internal battery (such as a battery storage inverter), the AbsMinPower will be a negative number representing the maximum power that the ESA can charge its internal battery.

9.2.8.5. AbsMaxPower Attribute

This attribute SHALL indicate the maximum electrical power that the ESA can consume when switched on.

The value of the AbsMaxPower attribute SHALL be limited such that:


```
AbsMaxPower >= AbsMinPower
```

Note that for Generator ESAs that can discharge a battery to loads in the home (such as a battery storage inverter), the AbsMaxPower will be a positive number representing the maximum power at which the ESA can discharge its internal battery.

For example, a battery storage inverter that can charge its battery at a maximum power of 2000W and can discharge the battery at a maximum power of 3000W, would have a AbsMinPower: -2000, AbsMaxPower: 3000W.

9.2.8.6. PowerAdjustmentCapability Attribute

This attribute SHALL indicate how the ESA can be adjusted at the current time. This attribute SHOULD be updated regularly by ESAs.

For example, a battery storage inverter may need to regulate its internal temperature, or the charging rate of the battery may be limited due to cold temperatures, or a change in the state of charge of the battery may mean that the maximum charging or discharging rate is limited.

An empty list SHALL indicate that no power adjustment is currently possible.

Multiple entries in the list allow to indicate that permutations of scenarios may be possible.

For example, a 10kWh battery could be at 80% state of charge. If charging at 2kW, then it would be full in 1 hour. However, it could be discharged at 2kW for 4 hours.

In this example the list of PowerAdjustStructs allows multiple scenarios to be offered as follows:

Entry [0] - Charging option:		
MinPower	-2000	Watts
MaxPower	0	Watts
MinDuration	60	seconds
MaxDuration	3600	seconds (1 hour)
Entry [1] - Discharging option:		
MinPower	0	Watts
MaxPower	2000	Watts
MinDuration	60	seconds
MaxDuration	14400	seconds (4 hours)

9.2.8.7. Forecast Attribute

This attribute allows an ESA to share its intended forecast with a client (such as an Energy Management System).

A null value indicates that there is no forecast currently available (for example, a program has not

yet been selected by the user).

A server MAY reset this value attribute to null on a reboot, and it does not need to persist any previous forecasts.

9.2.8.8. OptOutState Attribute

This attribute SHALL indicate the current Opt-Out state of the ESA. The ESA may have a local user interface to allow the user to control this OptOutState. An EMS may subscribe to the OptOutState to get notified about changes in operational state.

If the ESA is in the LocalOptOut or OptOut states, so it cannot be controlled by an EMS for local optimization reasons, it SHALL reject any commands which have the AdjustmentCauseEnum value LocalOptimization. If the ESA is in the GridOptOut or OptOut states, so it cannot be controlled by an EMS for grid optimization reasons, it SHALL reject any commands which have the AdjustmentCauseEnum value GridOptimization. If the ESA is in the LocalOptOut, GridOptOut, or NoOptOut states, the device is still permitted to optimize its own energy usage, for example, using tariff information it may obtain.

9.2.9. Commands

ID	Name	Direction	Response	Access	Conformance
0x00	**PowerAdjustRequest**	client ⇒ server	Y	O	PA
0x01	**CancelPowerAdjustRequest**	client ⇒ server	Y	O	PA
0x02	**StartTimeAdjustRequest**	client ⇒ server	Y	O	STA
0x03	**PauseRequest**	client ⇒ server	Y	O	PAU
0x04	**ResumeRequest**	client ⇒ server	Y	O	PAU
0x05	**ModifyForecastRequest**	client ⇒ server	Y	O	FA
0x06	**RequestConstraintBasedForecast**	client ⇒ server	Y	O	CON
0x07	**CancelRequest**	client ⇒ server	Y	O	STA|FA|CON

9.2.9.1. PowerAdjustRequest Command

Allows a client to request an adjustment in the power consumption of an ESA for a specified duration.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**Power**	power-mW	desc			M
1	**Duration**	elapsed-s	desc			M
2	**Cause**	AdjustmentCauseEnum	desc			M

9.2.9.1.1. Power Field

This field SHALL indicate the power that the ESA SHALL use during the adjustment period.

This value SHALL be between the MinPower and MaxPower fields of the PowerAdjustStruct in the PowerAdjustmentCapability attribute.

9.2.9.1.2. Duration Field

This field SHALL indicate the duration that the ESA SHALL maintain the requested power for.

This value SHALL be between the MinDuration and MaxDuration fields of the PowerAdjustStruct in the PowerAdjustmentCapability attribute.

9.2.9.1.3. Cause Field

This field SHALL indicate the cause of the request from the EMS.

9.2.9.1.4. Effect upon receipt

Upon receipt, the ESA SHALL validate that the Power and Duration specified in the command are within the limits of its current operation and advertised PowerAdjustmentCapability attribute, the OptOutState permits the specified AdjustmentCauseEnum (see OptOutState for details), and the ESAState is Online.

If the PowerAdjustRequest command is accepted, then the ESA SHALL change its ESAState to PowerAdjustActive. The command status returned SHALL be SUCCESS if the adjustment is accepted; otherwise the command SHALL be rejected with FAILURE. If the ESA supports the PFR or SFR features, the Forecast attribute is updated, including noting the Cause value in the ForecastUpdateReason.

Start of adjustment

The ESA SHALL generate a PowerAdjustStart Event when it begins to adjust its power.

The ESA then begins to adjust its power consumption or generation to the power level commanded in the Power field.

End of adjustment (normal completion)

After the elapsed duration, the ESA SHALL revert to normal (or idle) power levels.

The ESA SHALL also generate a PowerAdjustEnd Event with a cause code to indicate a 'Normal

completion' and the ESAState SHALL be restored to Online.

Failure or Opt-out during adjustment

If during the power adjustment session a failure or other condition occurs (such as the user deciding to opt-out by updating the OptOutState) then the ESA SHALL generate a PowerAdjustEnd Event to indicate the end of the session, with the appropriate cause code.

The ESAState SHALL be updated to reflect the new state.

Receiving a new PowerAdjustRequest command when adjustment is on-going

If an existing power adjustment is already happening, and a new PowerAdjustRequest command is received, then if the ESA allows it, the ESA SHALL return SUCCESS. If the ESA does not permit this new PowerAdjustmentRequest command to interrupt the adjustment that is in progress, it SHALL return BUSY.

Note that if the new command is accepted, then the ESA SHALL NOT generate a new PowerAdjustEnd Event until the new duration has elapsed. This is to avoid generating too many events in cases where a device is routinely commanded with overlapping power adjustments.

For example, a battery inverter ESA may be sent a new request every 5 seconds to adjust its discharge power based on real-time meter readings. Each command may have a 60 second duration, but this command is superseded after 5 seconds by a new request.

After the client has sent its last command and after this command duration expires then the last active power adjustment has been completed. This final command causes the PowerAdjustEnd Event to be generated when the ESAState is restored to Online.

9.2.9.2. CancelPowerAdjustRequest Command

Allows a client to cancel an ongoing PowerAdjustmentRequest operation.

9.2.9.2.1. Effect upon receipt

Upon receipt, the ESA SHALL end the active power adjustment session and return to normal (or idle) power levels.

If the ESAState is not PowerAdjustActive, then the command SHALL be rejected with FAILURE.

If the command is accepted, the ESA SHALL generate an PowerAdjustEnd Event and the ESAState SHALL be restored to Online and the command status returned SHALL be SUCCESS. If the ESA supports the `PFR` or `SFR` features, the Forecast attribute is updated, including updating the ForecastUpdateReason to `Internal Optimization`.

9.2.9.3. StartTimeAdjustRequest Command

Allows a client to adjust the start time of a Forecast sequence that has not yet started operation (i.e. where the current Forecast StartTime is in the future).

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**RequestedStartTime**	epoch-s	desc			M
1	**Cause**	AdjustmentCauseEnum	desc			M

9.2.9.3.1. RequestedStartTime Field

This field SHALL indicate the requested start time, in UTC, that the client would like the appliance to shift its Forecast to. This value MUST be in the future.

A client can estimate the entire Forecast sequence duration by computing the EndTime - StartTime fields from the Forecast attribute, and therefore avoid scheduling the start time too late.

This value SHALL be after the EarliestStartTime in the Forecast attribute. The new EndTime, that can be computed from the RequestedStartTime and the Forecast sequence duration, SHALL be before the LatestEndTime.

9.2.9.3.2. Cause Field

This field SHALL indicate the cause of the request from the EMS.

9.2.9.3.3. Effect upon receipt

Upon receipt, if the ESA supports STA, and the OptOutState permits the specified AdjustmentCauseEnum (see OptOutState for details), and the RequestedStartTime adjustment would be within constraints described in RequestedStartTime then ESA SHALL accept the request to modify the Start Time.

If the RequestedStartTime value resulted in a time shift which is outside the time constraints of EarliestStartTime and LatestEndTime, then the command SHALL be rejected with CONSTRAINT_ERROR; in other failure scenarios the command SHALL be rejected with FAILURE.

The command status returned SHALL be SUCCESS if the StartTime in the Forecast is updated. The ESA SHALL update its Forecast attribute using the new RequestedStartTime including a new ForecastId, StartTime, EndTime, and ForecastUpdateReason.

9.2.9.4. PauseRequest Command

Allows a client to temporarily pause an operation and reduce the ESAs energy demand.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**Duration**	elapsed-s	desc			M
1	**Cause**	AdjustmentCauseEnum	desc			M

9.2.9.4.1. Duration Field

This field SHALL indicate the duration that the ESA SHALL be paused for. This value SHALL be between the MinPauseDuration and MaxPauseDuration indicated in the ActiveSlotNumber index in the Slots list in the Forecast.

9.2.9.4.2. Cause Field

This field SHALL indicate the cause of the request from the EMS.

9.2.9.4.3. Effect upon receipt

Upon receipt, if the ESA supports PAU and the SlotIsPauseable field is true in the ActiveSlotNumber index in the Slots list, the OptOutState permits the specified AdjustmentCauseEnum (see OptOutState for details), and the OptOutState is Online or PowerAdjustActive, the ESA SHALL allow its current operation to be Paused.

If the ESA SlotIsPauseable field is false for the ActiveSlotNumber, then the command SHALL be rejected with FAILURE.

The ESA SHALL validate that the Duration field is within the range of MinPauseDuration and MaxPauseDuration. If it is outside of this range then the command SHALL be rejected with CONSTRAINT_ERROR.

The command status returned SHALL be SUCCESS if the ESA is paused.

Once the command has been accepted, the ESA SHALL also generate a Paused Event and the ESAState SHALL be set to Paused.

The ESA SHALL update its update its Forecast to account for the pause, including updating the ForecastUpdateReason from the Cause field.

Pause timer start The ESA SHALL start a Pause timer based on the value of the Duration field.

Paused State Whilst in the Paused state, the ESA SHALL NOT consume or produce significant power (other than required to keep its basic control system operational).

Pause timer expiry When the Pause timer expires the ESA SHALL automatically resume operation. When it does this, then it SHALL also generate a Resumed Event and the ESAState SHALL be updated accordingly to reflect its current state (for example, it may still have PowerAdjustActive now it has resumed from the Pause).

Receiving a further pause request command whilst paused If a further Pause Request is received in the same forecast slot whilst already in the paused state, the ESA SHALL again validate that the Duration value is within the range of the (possibly revised) MinPauseDuration and MaxPauseDuration. If it is outside of this range then the command SHALL be rejected with CONSTRAINT_ERROR.

If the command is accepted the pause timer SHALL be extended by the new Duration.

ResumeRequest A client MAY send a ResumeRequest command, before the Pause timer has expired, to request the ESA to return to its normal operating state. See ResumeRequest for details.

Failure or Opt-out during PauseRequest A consumer MAY decide to opt-out of the remotely commanded PauseRequest by updating the OptOutState, and the ESA SHALL also change the ESAState to the new state.

If the ESA develops a fault whilst Paused, the ESAState SHALL be set to Fault.

On change of ESAState (from Paused to another state), the ESA SHALL generate a Resumed Event.

9.2.9.5. ResumeRequest Command

Allows a client to cancel the PauseRequest command and enable earlier resumption of operation.

9.2.9.5.1. Effect upon receipt

Upon receipt, if the ESA supports PAU and it is currently Paused, the ESA MAY decide not to resume immediately if the MinPauseDuration has not yet elapsed. This behavior is manufacturer specific.

If accepted, the ESA SHALL resume its operation. The ESA SHALL also generate a Resumed Event and the ESAState SHALL be updated accordingly to reflect its current state.

The command status returned SHALL be SUCCESS if the operation is resumed, otherwise the command SHALL be rejected with FAILURE.

The ESA SHALL update its Forecast to account for the resumption after the pause.

9.2.9.6. ModifyForecastRequest Command

Allows a client to modify a Forecast within the limits allowed by the ESA.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**ForecastId**	uint32	all			M
1	**SlotAdjustments**	list[SlotAdjustmentStruct]	max 10			M
2	**Cause**	AdjustmentCauseEnum	desc			M

9.2.9.6.1. ForecastId Field

This field SHALL indicate the ForecastId that is to be modified.

9.2.9.6.2. SlotAdjustments Field

This field SHALL contain a list of SlotAdjustment parameters that should be modified in the corresponding Forecast with matching ForecastId.

9.2.9.6.3. Cause Field

This field SHALL indicate the cause of the request from the EMS.

9.2.9.6.4. Effect upon receipt

Upon receipt, if the ESA supports FA, and the OptOutState permits the specified AdjustmentCauseEnum (see OptOutState for details), it SHALL attempt to adjust its forecast.

The client may be an energy management system which has retrieved the forecasts from multiple ESA and then performed some optimization (for example, taking advantage of local solar PV generation).

The ESA SHALL inspect the requested forecast and ensure that it does not exceed the limits set in each of its slots. The ESA manufacturer may also reject the request if it could cause the user's preferences to be breached (for example, if it may cause the home to be too hot or too cold, or a battery to be insufficiently charged).

Note that the client may make an adjustment when the ESA's program has already been started. In this case the ActiveSlotNumber may have moved through the Forecast sequence. Any attempts to modify slots which have already been run SHALL result in the entire command being rejected.

The command allows a single modification to a particular slot, or to multiple slots in a single command by sending a list of modifications.

If the ESA accepts the requested Forecast then it SHALL update its Forecast attribute (incrementing its ForecastId, and updating its ForecastUpdateReason from the Cause value) and run the revised Forecast as its new intended operation.

Note that an ESA may adapt its Forecast based on a slot modification. For example, if an ESA was switched on earlier in the day to take account of available solar PV, then it may not need to use energy later in the day. It may add or remove slots as it chooses. This may result in several iterations for the Energy Management System to reprocess updated forecasts until it has stabilized on an agreed set of changes.

If for any reason the ESA cannot accept the entire requested forecast adjustments then it SHALL reject the entire command.

The command status returned SHALL be SUCCESS if the entire list of SlotAdjustmentStruct are accepted, otherwise the command SHALL be rejected with FAILURE.

9.2.9.7. RequestConstraintBasedForecast Command

Allows a client to ask the ESA to recompute its Forecast based on power and time constraints.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**Constraints**	list[ConstraintsStruct]	max 10			M
1	**Cause**	AdjustmentCauseEnum	desc			M

9.2.9.7.1. Constraints Field

This field SHALL indicate the series of turn up or turn down power requests that the ESA is being asked to constrain its operation within.

For example, a grid event which requires devices to reduce power (turn down) between 4pm and 6pm and due to excess power on the grid overnight, may request ESAs to increase their power demand (turn up) between midnight and 6am.

If this ESA supports PFR this would have 2 entries in the list as follows:

Entry [0] - Turn Down		
StartTime	2023-10-23T16:00:00	
Duration	7200	(2 hours)
NominalPower	20	W
MaximumEnergy	40	Wh

Entry [1] - Turn Up		
StartTime	2023-10-24T00:00:00	
Duration	21,600	(6 hours)
NominalPower	7000	W
MaximumEnergy	42000	Wh

If this ESA supports SFR where it does not know the actual power, but has an understanding of the functions that use more energy, it could be requested to use more or less energy using the LoadControl field as follows:

Entry [0] - Turn Down		
StartTime	2023-10-23T16:00:00	
Duration	7200	(2 hours)
LoadControl	-100	(Use less power)

Entry [1] - Turn Up		
StartTime	2023-10-24T00:00:00	
Duration	21,600	(6 hours)
LoadControl	+100	(Use more power)

9.2.9.7.2. Cause Field

This field SHALL indicate the cause of the request from the EMS.

9.2.9.7.3. Effect upon receipt

Upon receipt, if the ESA supports CON, and the OptOutState permits the specified Adjustment-

CauseEnum (see OptOutState for details), it may be requested to generate a new forecast by a client.

For example the client may be an energy management system which has determined that the peak load on the home should be reduced (for example a grid event) or that there is more local generation available and the solar export power needs to be consumed.

The EMS may not be best placed to make the forecast adjustment because the ESA knows more about its internal operation and system efficiencies.

For example the total energy used when charging a battery faster may result in increased heat losses in the inverter, so the total energy required to charge a battery with 10kWh of stored energy may vary based on charging power.

The ESA SHALL inspect the requested power limits to ensure that there are no overlapping elements. The ESA manufacturer may also reject the request if it could cause the user's preferences to be breached (e.g. may cause the home to be too hot or too cold, or a battery to be insufficiently charged).

If the ESA can meet the requested power limits, it SHALL regenerate a new Power Forecast with a new ForecastId and ForecastUpdateReason from the Cause value.

If an appliance has already begun running a program (and agreed to modify its schedule), it may continue to run the same program and meet the same user settings (e.g. ECO mode), but may take more or less time to complete the cycle. The new reported Forecast will start from the current time, i.e. the slots that have already been completed SHALL NOT be included in the new forecast.

The command status returned SHALL be SUCCESS, otherwise the command SHALL be rejected with FAILURE.

9.2.9.8. CancelRequest Command

Allows a client to request cancellation of a previous adjustment request in a StartTimeAdjustRequest, ModifyForecastRequest or RequestConstraintBasedForecast command.

9.2.9.8.1. Effect upon receipt

Upon receipt, the ESA SHALL attempt to cancel the effects of any previous adjustment request commands, and re-evaluate its forecast for intended operation ignoring those previous requests.

If the ESA ForecastUpdateReason was already `Internal Optimization`, then the command SHALL be rejected with FAILURE.

If the command is accepted, the ESA SHALL update its ESAState if required, and the command status returned SHALL be SUCCESS. The ESA SHALL update its Forecast attribute to match its new intended operation, and update the ForecastUpdateReason to `Internal Optimization`.

9.2.10. Events

ID	Name	Priority	Access	Conformance
0x00	**PowerAdjustStart**	INFO	V	PA

ID	Name	Priority	Access	Conformance
0x01	**PowerAdjustEnd**	INFO	V	PA
0x02	**Paused**	INFO	V	PAU
0x03	**Resumed**	INFO	V	PAU

9.2.10.1. PowerAdjustStart Event

This event SHALL be generated when the Power Adjustment session is started.

9.2.10.2. PowerAdjustEnd Event

This event SHALL be generated when the Power Adjustment session ends.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**Cause**	CauseEnum			NormalCompletion	M
1	**Duration**	elapsed-s				M
2	**EnergyUse**	energy-mWh				M

9.2.10.2.1. Cause Field

This field SHALL indicate the reason why the power adjustment session ended.

9.2.10.2.2. Duration Field

This field SHALL indicate the number of seconds that the power adjustment session lasted before ending.

9.2.10.2.3. EnergyUse Field

This field SHALL indicate the approximate energy used by the ESA during the session.

For example, if the ESA was on and was adjusted to be switched off, then this SHALL be 0W. If this was a battery inverter that was requested to charge it would have a negative energy use. If this was a normal load that was turned on, then it will have positive value.

9.2.10.3. Paused Event

This event SHALL be generated when the ESA enters the Paused state.

There is no data for this event.

9.2.10.4. Resumed Event

This event SHALL be generated when the ESA leaves the Paused state and resumes operation.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**Cause**	CauseEnum			NormalCompletion	M

9.2.10.4.1. Cause Field

This field SHALL indicate the reason why the pause ended.

9.3. Energy EVSE Cluster

Electric Vehicle Supply Equipment (EVSE) is equipment used to charge an Electric Vehicle (EV) or Plug-In Hybrid Electric Vehicle. This cluster provides an interface to the functionality of Electric Vehicle Supply Equipment (EVSE) management.

Devices targeted by this cluster include Electric Vehicle Supply Equipment (EVSE). The cluster generically assumes a signaling protocol (J1772 in NA and IEC61851 in Europe and Asia) between the EVSE and Electric Vehicle (EV) that utilizes a pilot signal to manage the states of the charging process. [SAE J2847/3_202311] version and IEC61841 define Pilot signal as a modulated DC voltage on a single wire.

Power Line Communication (PLC) is supported by some EVSEs (e.g. for support of ISO 15118 in Europe and SAE J2931/4 in NA) and may enable features such as Vehicle to Grid (V2G) or Vehicle to Home (V2H) that allows for bi-directional charging/discharging of electric vehicles.

More modern EVSE devices may optionally support ISO 15118-20 in Europe and SAE J2836/3 for NA to support bi-directional charging (Vehicle to Grid - V2G) and Plug and Charge capabilities.

This cluster definition assumes AC charging only. DC charging options may be added in future revisions of this cluster.

This cluster supports a safety mechanism that may lockout remote operation until the initial latching conditions have been met. Some of the fault conditions defined in SAE J1772, such as Ground-Fault Circuit Interrupter (GFCI) or Charging Circuit Interrupting Device (CCID), may require clearing by an operator by, for example, pressing a button on the equipment or breaker panel.

This EVSE cluster is written around support of a single EVSE. Having multiple EVSEs at home or a business is managed by backend system and outside scope of this cluster.

Note that in many deployments the EVSE may be outside the home and may suffer from intermittent network connections (e.g. a weak WiFi signal). It also allows for a charging profile to be preconfigured, in case there is a temporary communications loss during a charging session.

9.3.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Revision	Description
1	Initial Release
2	Updates after 0.7 Ballot review

9.3.2. Classification

Hierarchy	Role	Scope	PICS Code
Base	Application	Endpoint	EEVSE

9.3.3. Cluster ID

ID	Name
0x0099	Energy EVSE

9.3.4. Features

This cluster SHALL support the FeatureMap bitmap attribute as defined below.

Bit	Code	Feature	Conformance	Summary
0	**PREF**	ChargingPreferences	P	EVSE supports storing user charging preferences
1	**SOC**	SoCReporting	P	EVSE supports reporting of vehicle State of Charge (SoC)
2	**PNC**	PlugAndCharge	P	EVSE supports PLC to support Plug and Charge
3	**RFID**	RFID	O	EVSE is fitted with an RFID reader
4	**V2X**	V2X	P	EVSE supports bidirectional charging / discharging

9.3.4.1. ChargingPreferences Feature

Since some EVSEs cannot obtain the SoC from the vehicle, some EV charging solutions allow the consumer to specify a daily charging target (for adding energy to the EV's battery). This feature allows the consumer to specify how many miles or km of additional range they need for their typical daily commute. This range requirement can be converted into a daily energy demand with a target charging completion time.

The EVSE itself MAY use this information (or may allow a controller such as an EMS) to compute an

optimized charging schedule.

An EVSE device MAY implement the Device Energy Management cluster PFR (Power Forecast Reporting) and FA (Forecast Adjustment) features. This can help a controller (such as an EMS) to optimize the EVSE against other ESAs. For example, a solar PV ESA may share its Forecast and allow the EVSE to know the best time to charge so that any excess solar generation is used to charge the EV.

EVSE devices that support the Device Energy Management cluster's FA feature can have their charging profiles set by a controller device such as an EMS. For example, if the EVSE advertises a simple power forecast which allows the EMS to adjust over a wide range of power and time durations, then the EVSE may allow the EMS to propose a revised optimized forecast (which is the charging profile).

See the Device Energy Management Cluster for more details.

9.3.4.2. SoCReporting Feature

Vehicles and EVSEs which support ISO 15118 may allow the vehicle to report its battery size and state of charge. If the EVSE supports PLC it may have a vehicle connected which optionally supports reporting of its battery size and current State of Charge (SoC).

If the EVSE supports reporting of State of Charge this feature will only work if a compatible EV is connected.

Note some EVSEs may use other undefined mechanisms to obtain vehicle State of Charge outside the scope of this cluster.

9.3.4.3. PlugAndCharge Feature

If the EVSE supports PLC, it may be able to support the Plug and Charge feature. e.g. this may allow the vehicle ID to be obtained which may allow an energy management system to track energy usage per vehicle (e.g. to give the owner an indicative cost of charging, or for work place charging).

If the EVSE supports the Plug and Charge feature, it will only work if a compatible EV is connected.

9.3.4.4. RFID Feature

If the EVSE is fitted with an RFID reader, it may be possible to obtain the User or Vehicle ID from an RFID card. This may be used to record a charging session against a specific charging account, and may optionally be used to authorize a charging session.

An RFID event can be generated when a user taps an RFID card onto the RFID reader. The event must be subscribed to by the EVSE Management cluster client. This client may use this to enable the EV to charge or discharge. The lookup and authorization of RIFD UID is outside the scope of this cluster.

9.3.4.5. V2X Feature

If the EVSE can support bi-directional charging, it may be possible to request that the vehicle can discharge to the home or grid.

9.3.5. Dependencies

The server side of this cluster SHALL depend on setting time from another device or using a real-time clock.

This can either use a built-in real-time clock or non Matter time source, or can be derived using the Time Synchronization cluster.

9.3.5.1. Diagnostic Event logs

It is quite common that users may experience issues charging their vehicle, which, without logs to understand what happened, makes it very difficult to resolve the root cause.

Matter supports events (see Matter specification section 7.14) which includes a buffered log of previous events including a Timestamp (see Matter specification 7.14.2.2).

This Timestamp can be the System Time (since boot) or Epoch Time. It is recommended to use Epoch time (which can be set using the Time Synchronization cluster), which would allow remote support operatives to retrieve the event logs and analyze them against a user reported actual time.

9.3.6. Definitions

EVSE Session - An EVSE session starts when an EV is plugged in. It ends when it is unplugged.

9.3.7. Data Types

9.3.7.1. StateEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**NotPluggedIn**	The EV is not plugged in.	M
1	**PluggedInNoDemand**	The EV is plugged in, but not demanding current.	M
2	**PluggedInDemand**	The EV is plugged in and is demanding current, but EVSE is not allowing current to flow.	M
3	**PluggedInCharging**	The EV is plugged in, charging is in progress, and current is flowing	M

Value	Name	Summary	Conformance
4	**PluggedInDischarging**	The EV is plugged in, discharging is in progress, and current is flowing	V2X
5	**SessionEnding**	The EVSE is transitioning from any plugged-in state to NotPluggedIn	M
6	**Fault**	There is a fault (see FaultState attribute)	M

9.3.7.2. SupplyStateEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**Disabled**	The EV is not currently allowed to charge or discharge	M
1	**ChargingEnabled**	The EV is currently allowed to charge	M
2	**DischargingEnabled**	The EV is currently allowed to discharge	[V2X]
3	**DisabledError**	The EV is not currently allowed to charge or discharge due to an error. The error must be cleared before operation can continue.	M
4	**DisabledDiagnostics**	The EV is not currently allowed to charge or discharge due to Diagnostics Mode.	M

9.3.7.3. FaultStateEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**NoError**	The EVSE is not in an error state.	M
1	**MeterFailure**	The EVSE is unable to obtain electrical measurements.	M

Value	Name	Summary	Conformance
2	**OverVoltage**	The EVSE input voltage level is too high.	M
3	**UnderVoltage**	The EVSE input voltage level is too low.	M
4	**OverCurrent**	The EVSE detected charging current higher than allowed by charger.	M
5	**ContactWetFailure**	The EVSE detected voltage on charging pins when the contactor is open.	M
6	**ContactDryFailure**	The EVSE detected absence of voltage after enabling contactor.	M
7	**GroundFault**	The EVSE has an unbalanced current supply.	M
8	**PowerLoss**	The EVSE has detected a loss in power.	M
9	**PowerQuality**	The EVSE has detected another power quality issue (e.g. phase imbalance).	M
10	**PilotShortCircuit**	The EVSE pilot signal amplitude short circuited to ground.	M
11	**EmergencyStop**	The emergency stop button was pressed.	M
12	**EVDisconnected**	The EVSE detected that the cable has been disconnected.	M
13	**WrongPowerSupply**	The EVSE could not determine proper power supply level.	M
14	**LiveNeutralSwap**	The EVSE detected Live and Neutral are swapped.	M
15	**OverTemperature**	The EVSE internal temperature is too high.	M
255	**Other**	Any other reason.	M

9.3.7.4. EnergyTransferStoppedReasonEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**EVStopped**	The EV decided to stop	M
1	**EVSEStopped**	The EVSE decided to stop	M
2	**Other**	An other unknown reason	M

9.3.7.5. TargetDayOfWeekBitmap Type

This data type is derived from map8.

Bit	Name	Summary	Conformance
0	Sunday	Sunday	M
1	Monday	Monday	M
2	Tuesday	Tuesday	M
3	Wednesday	Wednesday	M
4	Thursday	Thursday	M
5	Friday	Friday	M
6	Saturday	Saturday	M

9.3.7.6. ChargingTargetStruct Type

This represents a single user specified charging target for an EV.

An EVSE or EMS system optimizer may use this information to take the Time of Use Tariff, grid carbon intensity, local generation (solar PV) into account to provide the cheapest and cleanest energy to the EV.

The optimization strategy is not defined here, however in simple terms, the AddedEnergy requirement can be fulfilled by knowing the charging Power (W) and the time needed to charge.

> To compute the Charging Time: Required Energy (Wh) = Power (W) x ChargingTime (s) / 3600
>
> Therefore: ChargingTime (s) = (3600 x RequiredEnergy (wH)) / Power (W)
>
> To compute the charging time: Charging StartTime = TargetTimeMinutesPastMidnight - ChargingTime

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0	**TargetTimeMinutesPastMidnight**	uint16	0 to 1439		0		M
1	**TargetSoC**	percent			0		SOC, O.a+
2	**AddedEnergy**	energy-mWh	min 0		0		O.a+

9.3.7.6.1. TargetTimeMinutesPastMidnight Field

This field SHALL indicate the desired charging completion time of the associated day. The time will be represented by a 16 bits unsigned integer to designate the minutes since midnight. For example, 6am will be represented by 360 minutes since midnight and 11:30pm will be represented by 1410 minutes since midnight.

This field is based on local wall clock time. In case of Daylight Savings Time transition which may result in an extra hour or one hour less in the day, the charging algorithm should take into account the shift appropriately.

Note that if the TargetTimeMinutesPastMidnight values are too close together (e.g. 2 per day) these may overlap. The EVSE may have to coalesce the charging targets into a single target. e.g. if the 1st charging target cannot be met in the time available, the EVSE may be forced to begin working towards the 2nd charging target and immediately continue until both targets have been satisfied (or the vehicle becomes full).

The EVSE itself cannot predict the behavior of the vehicle (i.e. if it cannot obtain the SoC from the vehicle), so should attempt to perform a sensible operation based on these targets. It is recommended that the charging schedule is pessimistic (i.e. starts earlier) since the vehicle may charge more slowly than the electrical supply may provide power (especially if it is cold).

If the user configures large charging targets (e.g. high values of AddedEnergy or SoC) then it is expected that the EVSE may need to begin charging immediately, and may not be able to guarantee that the vehicle will be able to reach the target.

9.3.7.6.2. TargetSoC Field

This field represents the target SoC that the vehicle should be charged to before the TargetTimeMinutesPastMidnight.

If the EVSE can obtain the SoC of the vehicle:

- the TargetSoC field SHALL take precedence over the AddedEnergy field.
- the EVSE SHOULD charge to the TargetSoC and then stop the charging automatically when it reaches that point.
- if the TargetSoC value is set to 100% then the EVSE SHOULD continue to charge the vehicle until the vehicle decides to stop charging.

If the EVSE cannot obtain the SoC of the vehicle:

- in this case, the AddedEnergy field SHALL take precedence over the TargetSoC field, and the TargetSoC field may only take the values null or 100%.
- if the AddedEnergy field has not been provided, the EVSE SHOULD assume the vehicle is empty and charge until the vehicle stops demanding a charge.

9.3.7.6.3. AddedEnergy Field

This field represents the amount of energy that the user would like to have added to the vehicle before the TargetTimeMinutesPastMidnight.

This represents a positive value in mWh that SHOULD be added during the session (i.e. if the vehicle charging is stopped and started several times, this equates to the total energy since the vehicle has been plugged in).

The maximum value (500kWh) is much larger than most EV batteries on the market today. If the client tries to set this value too high then the EVSE will need to start charging immediately and continue charging until the vehicle stops demanding charge (i.e. it is full). Therefore the maximum value should be set based on typical battery size of the vehicles on the market (e.g. 70000Wh), however this is up to the client to carefully choose a value.

NOTE	If the EVSE can obtain the Battery Capacity of the vehicle, it SHOULD not limit this AddedEnergy value to the Battery Capacity of the vehicle, since the EV may also require energy for heating and cooling of the battery during charging, or for heating or cooling the cabin.

9.3.7.7. ChargingTargetScheduleStruct Type

This represents a set of user specified charging targets for an EV for a set of specified days.

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0	DayOfWeekForSequence	TargetDayOfWeekBitmap	desc			M	
1	ChargingTargets	list[ChargingTargetStruct]	max 10			M	

9.3.7.8. DayOfWeekForSequence Field

This field SHALL indicate the days of the week that the charging targets SHOULD be associated to. This field is a bitmap and therefore the associated targets could be applied to multiple days.

9.3.7.9. ChargingTargets Field

This field SHALL indicate a list of up to 10 charging targets for each of the associated days of the week.

9.3.8. Attributes

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0000	**State**	StateEnum	all	X		R V	M
0x0001	**SupplyState**	SupplyStateEnum	all			R V	M
0x0002	**FaultState**	FaultStateEnum	all			R V	M
0x0003	**ChargingEnabledUntil**	epoch-s	all	X N	0	R V	M
0x0004	**DischargingEnabledUntil**	epoch-s	all	X N	0	R V	V2X
0x0005	**CircuitCapacity**	amperage-mA	min 0	N	0	R V	M
0x0006	**MinimumChargeCurrent**	amperage-mA	min 0	N	6000	R V	M
0x0007	**MaximumChargeCurrent**	amperage-mA	min 0	N	0	R V	M
0x0008	**MaximumDischargeCurrent**	amperage-mA	min 0	N	0	R V	V2X
0x0009	**UserMaximumChargeCurrent**	amperage-mA	desc	N	0	RW VM	O
0x000A	**RandomizationDelayWindow**	elapsed-s	0 to 86400	N	600	RW VM	O

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0023	**NextChargeStartTime**	epoch-s	all	X	null	R V	PREF
0x0024	**NextChargeTargetTime**	epoch-s	all	X	null	R V	PREF
0x0025	**NextChargeRequiredEnergy**	energy-mWh	min 0	X	null	R V	PREF
0x0026	**NextChargeTargetSoC**	percent		X	null	R V	PREF
0x0027	**ApproximateEVEfficiency**	uint16	desc	X N	null	RW VM	[PREF]
0x0030	**StateOfCharge**	percent		X	null	R V	SOC
0x0031	**BatteryCapacity**	energy-mWh	min 0	X	null	R V	SOC
0x0032	**VehicleID**	string	max 32	X	null	R V	PNC
0x0040	**SessionID**	uint32	all	X N	null	R V	M
0x0041	**SessionDuration**	elapsed-s	all	X N	null	R V	M
0x0042	**SessionEnergyCharged**	energy-mWh	min 0	X N	null	R V	M
0x0043	**SessionEnergyDischarged**	energy-mWh	min 0	X N	null	R V	V2X

9.3.8.1. State Attribute

This attribute SHALL indicate the current status of the EVSE. This higher-level status is partly derived from the signaling protocol as communicated between the EVSE and the vehicle through the pilot signal.

The State attribute SHALL change when the EVSE detects change of condition of the EV (plugged in or unplugged, whether the vehicle is asking for demand or not, and if it is charging or discharging).

NOTE	SessionEnding is not really a state but a transition. However, the transition period

may take a few seconds and is useful for some clean up purposes.

The Fault state is used to indicate that the FaultState attribute is not NoError.

9.3.8.2. SupplyState Attribute

This attribute SHALL indicate whether the EV is currently allowed to charge from or discharge to the EVSE.

9.3.8.3. FaultState Attribute

This attribute SHALL indicate the type of fault detected by the EVSE (internally or as detected in the pilot signal).

When the SupplyState attribute is DisabledError, the FaultState attribute will be one of the values listed in FaultStateEnum, except NoError. For all values of SupplyState other than DisabledError, the FaultState attribute SHALL be NoError.

9.3.8.4. ChargingEnabledUntil Attribute

This attribute SHALL indicate the time, in UTC, that the EVSE will automatically stop current flow to the EV.

A null value indicates the EVSE is always enabled for charging.

A value in the past or 0x0 indicates that EVSE charging SHALL be disabled. The attribute is only set via the payload of the EnableCharging command.

This attribute SHALL be persisted, for example a temporary power failure should not stop the vehicle from being charged.

9.3.8.5. DischargingEnabledUntil Attribute

This attribute SHALL indicate the time, in UTC, that the EVSE will automatically stop current flow from the EV.

A null value indicates the EVSE is always enabled for discharging.

A value in the past or 0x0 indicates that EVSE discharging SHALL be disabled. The attribute is only set via the payload of the EnableDischarging command.

This attribute SHALL be persisted, for example a temporary power failure should not stop the vehicle from being discharged.

9.3.8.6. CircuitCapacity Attribute

This attribute SHALL indicate the capacity that the circuit that the EVSE is connected to can provide. It is intended to allow implementation of a self-managed network of EVSEs. It is assumed that the device will allow the setting of such values by an installer.

9.3.8.7. MinimumChargeCurrent Attribute

This attribute SHALL indicate the minimum current that can be delivered by the EVSE to the EV.

The attribute can be set using the EnableCharging command.

9.3.8.8. MaximumChargeCurrent Attribute

This attribute SHALL indicate the maximum current that can be delivered by the EVSE to the EV.

This SHALL represent the actual maximum current offered to the EV at any time. Note that the EV can draw less current than this value. For example, the EV may be limiting its power draw based on the operating conditions of the battery, such as temperature and state of charge.

The attribute can be initially set using the EnableCharging command or by adjusting the UserMaximumChargeCurrent attribute.

This attribute value SHALL be the minimum of:

- CircuitCapacity - Electrician's installation setting
- CableAssemblyCurrentLimit (detected by the EVSE when the cable is plugged in)
- MaximumChargeCurrent field in the EnableCharging command
- UserMaximumChargeCurrent attribute

9.3.8.9. MaximumDischargeCurrent Attribute

This attribute SHALL indicate the maximum current that can be received by the EVSE from the EV.

This attribute can be set using the EnableDischarging command.

This attribute value SHALL be the minimum of:

- CircuitCapacity - Electrician's installation setting
- CableAssemblyCurrentLimit (detected by the EVSE when the cable is plugged in)
- MaximumDischargeCurrent field in the EnableDischarging command

9.3.8.10. UserMaximumChargeCurrent Attribute

This attribute SHALL indicate a maximum current that can set by the consumer (e.g. via an app) as a preference to further reduce the charging rate. This may be desirable if the home owner has a solar PV or battery storage system which may only be able to deliver a limited amount of power. The consumer can manually control how much they allow the EV to take.

This attribute value SHALL be limited by the EVSE to be in the range of:


```
MinimumChargeCurrent <= UserMaximumChargeCurrent <= MaximumChargeCurrent
```

where MinimumChargeCurrent and MaximumChargeCurrent are the values received in the EnableCharging command.

Its default value SHOULD be initialized to the same as the CircuitCapacity attribute. This value SHALL be persisted across reboots to ensure it does not cause charging issues during temporary power failures.

9.3.8.11. RandomizationDelayWindow Attribute

This attribute SHALL indicate the size of a random window over which the EVSE will randomize the start of a charging session. This value is in seconds.

This is a feature that is mandated in some markets (such as UK) where the EVSE should by default randomize its start time within the randomization window. By default in the UK this should be 600s.

For example, if the RandomizationDelayWindow is 600s (i.e. 10 minutes) and if there was a cheap rate energy starting at 00:30, then the EVSE must compute a random delay between 0-599s and add this to its initial planned start time.

9.3.8.12. NextChargeStartTime Attribute

This attribute SHALL indicate the time, in UTC, when the EVSE plans to start the next scheduled charge based on the charging preferences.

If this is null it indicates that there is no scheduled charging, or that the vehicle is not plugged in.

9.3.8.13. NextChargeTargetTime Attribute

This attribute SHALL indicate the time, in UTC, when the EVSE SHOULD complete the next scheduled charge based on the charging preferences.

If this is null it indicates that there is no scheduled charging, or that the vehicle is not plugged in.

9.3.8.14. NextChargeRequiredEnergy Attribute

This attribute SHALL indicate the amount of energy that the EVSE is going to attempt to add to the vehicle in the next charging target.

If this is null it indicates that there is no scheduled charging, or that the EVSE is using the TargetSoC method to charge the vehicle.

9.3.8.15. NextChargeTargetSoC Attribute

This attribute SHALL indicate the target SoC the EVSE is going to attempt to reach when the vehicle is next charged.

If this is null it indicates that there is no scheduled charging, or that the EVSE cannot obtain the cur-

rent State of Charge from the vehicle.

If the SOC feature is not supported, only the values null and 100% are supported.

9.3.8.16. ApproximateEVEfficiency Attribute

This attribute SHALL indicate the vehicle efficiency rating for a connected vehicle.

This can be used to help indicate to the user *approximately* how many miles or km of range will be added. It allows user interfaces to display to the user simpler terms that they can relate to compared to kWh.

This is value is stored in km per kWh multiplied by a scaling factor of 1000.

A null value indicates that the EV efficiency is unknown and the NextChargeRequiredEnergy attribute cannot be converted from Wh to miles or km.

To convert from Wh into Range:

```
AddedRange (km)    = AddedEnergy (Wh) x ApproxEVEfficiency (km/kWh x 1000)
AddedRange (Miles) = AddedEnergy (Wh) x ApproxEVEfficiency (km/kWh x 1000) x
0.6213
```

Example:

```
ApproxEVEfficiency (km/kWh x 1000): 4800   (i.e. 4.8km/kWh x 1000)
AddedEnergy (Wh): 10,000
```

```
AddedRange (km)    = 10,000 x 4800 / 1,000,000 = 48 km
AddedRange (Miles) = AddedEnergy (Wh) x ApproxEVEfficiency (km/kWh x 1000) x
0.6213
                     = 29.82 Miles
```

9.3.8.17. StateOfCharge Attribute

This attribute SHALL indicate the state of charge of the EV battery in steps of 1%. The values are in the 0-100%. This attribute is only available on EVSEs which can read the state of charge from the vehicle and that support the SOC feature. If the StateOfCharge cannot be read from the vehicle it SHALL be returned with a NULL value.

9.3.8.18. BatteryCapacity Attribute

This attribute SHALL indicate the capacity of the EV battery in mWh. This value is always positive.

9.3.8.19. VehicleID Attribute

This attribute SHALL indicate the vehicle ID read by the EVSE via ISO-15118 using the PNC feature, if the EVSE supports this capability.

The field may be based on the e-Mobility Account Identifier (EMAID).

A null value SHALL indicate that this is unknown.

9.3.8.20. Session Attributes

The following set of attributes provides information about a charging session, defined as the period from the EVSE detecting an EV plug-in to when it is unplugged.

Note that the attributes hold their values from the previous session and are reset when the next plug-in event happens.

Whenever the State leaves the NotPluggedIn state and moves to any 'Plugged' state (see the State attribute), the SessionID attribute SHALL be incremented. The SessionDuration, SessionEnergyCharged and SessionEnergyDischarged attributes SHALL be reset to zero.

9.3.8.20.1. SessionID Attribute

This attribute SHALL indicate a unique identifier for the current or last session. A default value of null indicates no sessions have occurred. The SessionID SHALL be incremented each time a plugin is detected. A session begins when the vehicle is plugged in and ends when the vehicle is unplugged.

SessionIDs are allowed to roll over, although the range of SessionID is a large number and it is unlikely that the EVSE will have this many sessions in its lifetime unless there is a electrical fault causes sessions to be detected at a rapid rate.

If there is no session in progress, the Session ID attribute will remain at the value for the last session.

The SessionID attribute SHALL be persisted across reboots.

9.3.8.20.2. SessionDuration Attribute

This attribute SHALL indicate the duration in seconds for the current or last charging session. A default value of null indicates no sessions have occurred. A charge session begins when the vehicle is plugged in and ends when the vehicle is unplugged.

The SessionDuration can be calculated from the start time of the session. Manufacturers SHALL ensure this can be correctly calculated if there has been an power failure or reboot since the start of the session.

Note that the SessionDuration should not update too frequently, in case the attribute is subscribed to, which would result in excessive subscription update events.

9.3.8.20.3. SessionEnergyCharged Attribute

This attribute SHALL indicate the energy, in mWh, delivered by the EVSE to the EV for the current or last charging session. A default value of null indicates no sessions have occurred.

The SessionEnergyCharged value can be calculated by knowing the initial value of the energy meter at the start time of the session. Manufacturers SHALL ensure this can be correctly calculated if there has been an power failure or reboot since the start of the session.

Note that the SessionEnergyCharged attribute should not update too frequently, in case the attribute is subscribed to, which would result in excessive subscription update events.

9.3.8.20.4. SessionEnergyDischarged Attribute

This attribute SHALL indicate the energy, in mWh, received by the EVSE from the EV for the current or last charging session. A default value of null indicates no sessions have occurred.

The SessionEnergyDischarged value can be calculated by knowing the initial value of the energy meter at the start time of the session. Manufacturers SHALL ensure this can be correctly calculated if there has been an power failure or reboot since the start of the session.

Note that the SessionEnergyDischarged attribute should not update too frequently, in case the attribute is subscribed to, which would result in excessive subscription update events.

9.3.9. Commands

ID	Name	Direction	Response	Access	Conformance
0x01	Disable	client ⇒ server	Y	O T	M
0x02	EnableCharging	client ⇒ server	Y	O T	M
0x03	EnableDischarging	client ⇒ server	Y	O T	V2X
0x04	StartDiagnostics	client ⇒ server	Y	O T	O
0x05	SetTargets	client ⇒ server	Y	O T	PREF
0x06	GetTargets	client ⇒ server	GetTargetsResponse	O T	PREF
0x07	ClearTargets	client ⇒ server	Y	O T	PREF
0x00	GetTargetsResponse	client ⇐ server	N		PREF

9.3.9.1. Disable Command

Allows a client to disable the EVSE from charging and discharging.

9.3.9.1.1. Effect on Receipt

Upon receipt of the Disable command the EVSE will stop power flow between the EV and EVSE.

If the command cannot be handled, a status of FAILURE SHALL be returned.

If successful, the ChargingEnabledUntil and DischargingEnabledUntil attributes SHALL be set to 0x0, the SupplyState attribute changed to Disabled, and a response with status of SUCCESS SHALL be returned.

If any energy was being transferred, then a corresponding EnergyTransferStopped event SHALL be generated.

9.3.9.2. EnableCharging Command

Allows a client to enable the EVSE to charge an EV.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**ChargingEnabledUntil**	epoch-s	all	X	null	M
1	**MinimumChargeCurrent**	amperage-mA	min 0			M
2	**MaximumChargeCurrent**	amperage-mA	min 0			M

9.3.9.2.1. ChargingEnabledUntil Field

This field SHALL indicate the expiry time, in UTC, when charging will be automatically disabled.

A value in the past in this field SHALL disable the EVSE charging whereas a null value SHALL enable it permanently.

9.3.9.2.2. MinimumChargeCurrent Field

This field SHALL indicate the minimum current that can be delivered by the EVSE to the EV in trickle mode. The EVSE current limit can be advertised to an EV in 0.6A steps.

The value of the MinimumChargeCurrent attribute SHALL be set to the value of this field (see MinimumChargeCurrent attribute for further details).

9.3.9.2.3. MaximumChargeCurrent Field

This field SHALL indicate the maximum current that can be delivered by the EVSE to the EV. The EVSE current limit can be advertised to an EV in 0.6A steps.

The value of the this field SHALL be stored by the EVSE to determine the value of MaximumChargeCurrent attribute. For example, if the UserMaximumChargeCurrent attribute is adjusted below then

this value, and then later adjusted above this value, the resulting MaximumChargeCurrent attribute will be limited to this value.

9.3.9.2.4. Effect on Receipt

Upon receipt, this SHALL allow the EVSE to charge the EV until the specified timestamp within the current limits specified in the fields of the command.

If there is currently an error present on the EVSE, or Diagnostics are currently active, then the command SHALL be ignored and a response with a status of FAILURE SHALL be returned.

If successful, the SupplyState attribute SHALL be set to ChargingEnabled, and a response with status of SUCCESS SHALL be returned.

The timestamp indicated in the ChargingEnabledUntil attribute SHALL be updated to the timestamp of the ChargingEnabledUntil field if the command is successful.

If the ChargingEnabledUntil is null (i.e. should be permanently enabled), then the EVSE should enable charging indefinitely.

If the ChargingEnabledUntil time is not null, then when this time expires then the EVSE SHALL stop charging and SHALL update the State attribute to indicate it is no longer charging, and SHALL update the SupplyState attribute to Disabled.

9.3.9.3. EnableDischarging Command

Allows a client to enable the EVSE to discharge an EV.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**DischargingEnabledUntil**	epoch-s	all	X	null	M
1	**MaximumDischargeCurrent**	amperage-mA	min 0			M

9.3.9.3.1. DischargingEnabledUntil Field

This field SHALL indicate the expiry time, in UTC, when discharging will be automatically disabled.

A value in the past in this field SHALL disable the EVSE whereas a null value SHALL enable it permanently.

9.3.9.3.2. MaximumDischargeCurrent Field

This field SHALL indicate the maximum current that can be received by the EVSE from the EV. The EVSE current limit can be advertised to an EV in 0.6A steps. The value of the MaximumDischargeCurrent attribute SHALL be stored and persisted across reboots by the EVSE to the value of this

field.

9.3.9.3.3. Effect on Receipt

Upon receipt, this SHALL allow the EVSE to discharge the EV until the specified timestamp within the current limits specified in the fields of the command.

If there is currently an error present on the EVSE, or Diagnostics are currently active, then the command SHALL be ignored and a response with a status of FAILURE SHALL be returned.

If successful, the SupplyState attribute SHALL be set to DischargingEnabled, and a response with status of SUCCESS SHALL be returned.

The timestamp indicated in the DischargingEnabledUntil attribute SHALL be updated to the timestamp of the DischargingEnabledUntil field if the command is successful.

If the DischargingEnabledUntil is null (i.e. should be permanently enabled), then the EVSE should enable discharging indefinitely.

If the DischargingEnabledUntil time is not null, then when this time expires then the EVSE SHALL stop discharging and SHALL update the State attribute to indicate it is no longer discharging, and SHALL update the SupplyState attribute to Disabled.

9.3.9.4. StartDiagnostics Command

Allows a client to put the EVSE into a self-diagnostics mode.

9.3.9.4.1. Effect on Receipt

Upon receipt of the StartDiagnostics command, the EVSE SHALL enter a Diagnostics state only if the SupplyState attribute is in the Disabled state.

If the EVSE cannot start diagnostics, a response SHALL be generated with a status of FAILURE.

If successful, the SupplyState attribute SHALL be set to DisabledDiagnostics, and a response with status of SUCCESS SHALL be returned.

The diagnostics are at the discretion of the manufacturer and usually include internal checks. Upon completion of the diagnostics, the EVSE SHALL restore SupplyState to the Disabled state (see SupplyState attribute for further details).

9.3.9.5. SetTargets Command

Allows a client to set the user specified charging targets.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**ChargingTargetSchedules**	list[ChargingTargetScheduleStruct]	max 7			M

9.3.9.5.1. ChargingTargetSchedules Field

This field SHALL indicate a list of up to 7 sets of daily charging targets together with their associated days of the week. Each of the days of the week may only be included in a single ChargingTargetSchedule within this list field.

9.3.9.5.2. Effect on Receipt

Upon receipt of the SetTargets command, the EVSE SHALL validate that all of the charging targets are within a valid range, and that each day of the week is included in at most one of the ChargingTargetSchedule, if they are not then the response SHALL be CONSTRAINT_ERROR.

When a command is received that requires a total number of charging targets greater than the device supports, the status of the response SHALL be RESOURCE_EXHAUSTED.

If the charging targets are accepted, then the charging targets SHALL be stored for all of days that are set in the DayOfWeekForSequence bitmap fields in all the ChargingTargetSchedules. If a ChargingTargetSchedule is defined with no ChargingTargets then the ChargingTargets are cleared for those days defined in the DayOfWeekForSequence

The SetTargets command is used to update the EVSE's weekly charging schedule. If the EVSE already has some stored weekly charging targets, then it SHALL replace each daily charging target as it receives the updates from the client. For example, if the EVSE has 2 charging targets for every day of the week and is sent a SetTargets command with one target for Saturday then the EVSE SHALL remove both charging targets for Saturday and replace those with the updated charging target but leave all other days unchanged.

NOTE	the EVSE may not be able to compute the schedules by itself, or may rely upon an EMS or other optimizer to do this.

In a standalone mode (e.g. without using a ToU tariff from the Pricing Cluster) the EVSE should be able to compute the latest NextChargeStartTime based on the preset MaximumChargeCurrent attribute and local grid voltage to determine the charging duration required. It should automatically begin charging and following the charging schedule if the EVSE is plugged in (and the EVSE is enabled for charging).

If the EVSE supports the PFR (PowerForecastReporting) feature in the Device Energy Management cluster, it SHALL auto update the Forecast with the indicative power forecast.

If the EVSE supports the FA (ForecastAdjustment) feature in the Device Energy Management cluster, it SHALL indicate its adjustment capability in each of the Forecast slots (see Device Energy Management Cluster for more details).

The EVSE SHALL be responsible for updating the NextChargeEndTime, NextChargeRequiredEnergy and/or NextChargeTargetSoC attributes as it runs through its internal schedule. For example, as a scheduled charging session is completed or as it transitions to the next day, it should compute the next time it expects to start charging, and then update the corresponding attributes accordingly.

9.3.9.6. GetTargets Command

Allows a client to retrieve the current set of charging targets.

9.3.9.6.1. Effect on Receipt

Upon receipt, the EVSE SHALL send the GetTargetsResponse command to the client.

9.3.9.7. GetTargetsResponse Command

The GetTargetsResponse is sent in response to the GetTargets Command.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**ChargingTargetSchedules**	list[ChargingTargetScheduleStruct]	max 7			M

9.3.9.7.1. ChargingTargetSchedules Field

This field SHALL indicate a list of up to 7 sets of daily charging targets together with their associated days of the week.

9.3.9.8. ClearTargets Command

Allows a client to clear all stored charging targets.

9.3.9.8.1. Effect on Receipt

Upon receipt, all weekly targets that are currently stored SHALL be cleared and a default response of SUCCESS SHALL be sent in response. There are no error responses to this command.

If the EVSE is currently charging based on being in automatic mode, then it SHALL stop the EVSE charging.

9.3.10. Events

ID	Name	Priority	Access	Conformance
0x00	**EVConnected**	INFO	V	M
0x01	**EVNotDetected**	INFO	V	M
0x02	**EnergyTransferStarted**	INFO	V	M
0x03	**EnergyTransferStopped**	INFO	V	M
0x04	**Fault**	CRITICAL	V	M
0x05	**RFID**	INFO	V	[RFID]

9.3.10.1. EVConnected Event

This event SHALL be generated when the EV is plugged in.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**SessionID**	uint32	all			M

9.3.10.1.1. SessionID Field

This is the new session ID created after the vehicle is plugged in.

9.3.10.2. EVNotDetected Event

This event SHALL be generated when the EV is unplugged or not detected (having been previously plugged in). When the vehicle is unplugged then the session is ended.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**SessionID**	uint32	all			M
1	**State**	StateEnum	all			M
2	**SessionDuration**	elapsed-s	all			M
3	**SessionEnergyCharged**	energy-mWh	min 0			M
4	**SessionEnergyDischarged**	energy-mWh	min 0			V2X

9.3.10.2.1. SessionID Field

This field SHALL indicate the current value of the SessionID attribute.

9.3.10.2.2. State Field

This field SHALL indicate the value of the State attribute prior to the EV not being detected.

9.3.10.2.3. SessionDuration Field

This field SHALL indicate the total duration of the session, from the start of the session when the EV was plugged in, until it was unplugged.

9.3.10.2.4. SessionEnergyCharged Field

This field SHALL indicate the total amount of energy transferred from the EVSE to the EV during the session. This value SHALL always be positive.

Note that if bi-directional charging occurs during the session, then this value SHALL only include the sum of energy transferred from the EVSE to the EV, and SHALL NOT be a net value of charging and discharging energy.

9.3.10.2.5. SessionEnergyDischarged Field

This field SHALL indicate the total amount of energy transferred from the EV to the EVSE during the session. This value SHALL always be positive.

Note that if bi-directional discharging occurs during the session, then this value SHALL only include the sum of energy transferred from the EV to the EVSE, and SHALL NOT be a net value of charging and discharging energy.

9.3.10.3. EnergyTransferStarted Event

This event SHALL be generated when the EV starts charging or discharging.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**SessionID**	uint32	all			M
1	**State**	StateEnum	all			M
2	**MaximumCurrent**	amperage-mA	min 0			M

9.3.10.3.1. SessionID Field

This field SHALL indicate the value of the SessionID attribute at the time the event was generated.

9.3.10.3.2. State Field

This field SHALL indicate the value of the State attribute at the time the event was generated.

9.3.10.3.3. MaximumCurrent Field

This field SHALL indicate the value of the maximum charging or discharging current at the time the event was generated.

This field is signed. A positive value indicates the EV has been enabled for charging and the value is taken directly from the MaximumChargeCurrent attribute.

A negative value indicates that the EV has been enabled for discharging and the value can be taken from the MaximumDischargeCurrent attribute with its sign inverted. i.e. if the MaximumDischargeCurrent was 32000mA, this would be represented here as -32000mA.

9.3.10.4. EnergyTransferStopped Event

This event SHALL be generated when the EV stops charging or discharging.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**SessionID**	uint32	all			M
1	**State**	StateEnum	all			M
2	**Reason**	Energy-Transfer-StoppedReasonEnum	all			M
4	**Energy-Transferred**	energy-mWh	min 0			M

9.3.10.4.1. SessionID Field

This field SHALL indicate the value of the SessionID attribute prior to the energy transfer stopping.

9.3.10.4.2. State Field

This field SHALL indicate the value of the State attribute prior to the energy transfer stopping.

9.3.10.4.3. Reason Field

This field SHALL indicate the reason why the energy transferred stopped.

9.3.10.4.4. Energy Transferred Field

This field SHALL indicate the amount of energy transferred between the EVSE and EV since the previous EnergyTransferStarted event. This is a signed value in mWh. Positive values indicate the amount of energy transferred to the vehicle. Negative values indicate energy transferred from the vehicle.

9.3.10.5. Fault Event

If the EVSE detects a fault it SHALL generate a Fault Event. The SupplyState attribute SHALL be set to DisabledError and the type of fault detected by the EVSE SHALL be stored in the FaultState attribute.

This event SHALL be generated when the FaultState changes from any error state. i.e. if it changes from NoError to any other state and if the error then clears, this would generate 2 events.

It is assumed that the fault will be cleared locally on the EVSE device. When all faults have been cleared, the EVSE device SHALL set the FaultState attribute to NoError and the SupplyState attribute SHALL be set back to its previous state.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**SessionID**	uint32	all	X		M
1	**State**	StateEnum	all			M

ID	Name	Type	Constraint	Quality	Default	Conformance
2	**FaultStatePreviousState**	FaultStateEnum	all			M
4	**FaultStateCurrentState**	FaultStateEnum	all			M

9.3.10.5.1. SessionID Field

This field SHALL indicate the value of the SessionID attribute prior to the Fault State being changed. A value of null indicates no sessions have occurred before the fault occurred.

9.3.10.5.2. State Field

This field SHALL indicate the value of the State attribute prior to the Fault State being changed.

9.3.10.5.3. FaultStatePreviousState Field

This field SHALL indicate the value of the FaultState attribute prior to the Fault State being changed.

9.3.10.5.4. FaultStateCurrentState Field

This field SHALL indicate the current value of the FaultState attribute.

9.3.10.6. RFID Event

This event SHALL be generated when a RFID card has been read. This allows a controller to register the card ID and use this to authenticate and start the charging session.

ID	Name	Type	Constraint	Quality	Default	Conformance
0	**UID**	octstr	max 10			M

9.3.10.6.1. UID Field

The UID field (ISO 14443A UID) is either 4, 7 or 10 bytes.

9.4. Energy EVSE Mode Cluster

This cluster is derived from the Mode Base cluster which also defines a namespace for the operation of EVSE devices.

9.4.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table

below.

Revision	Description
1	Initial Release

9.4.2. Classification

Hierarchy	Role	Context	PICS Code
Mode Base	Application	Endpoint	EEVSEM

9.4.3. Cluster ID

ID	Name
0x009D	Energy EVSE Mode

9.4.4. Derived Cluster Namespace

This namespace includes definitions for data associated exclusively with the derived cluster.

9.4.4.1. Mode Tags

The following table defines the derived cluster specific ModeTag values.

Mode Tag Value	Name
0x4000	**Manual**
0x4001	**TimeOfUse**
0x4002	**SolarCharging**

9.4.4.1.1. Manual Tag

While in this mode, the EVSE needs to be sent an EnableEvseCharging or EnableEvseDischarging command to make the EVSE start charging or discharging.

9.4.4.1.2. TimeOfUse Tag

While in this mode, the EVSE will attempt to automatically start charging based on the user's charging targets and a Time of Use tariff to charge at the cheapest times of the day.

9.4.4.1.3. SolarCharging Tag

While in this mode, the EVSE will attempt to automatically start charging based on available excess solar PV generation, limiting the charging power to avoid imported energy from the grid.

9.4.5. Mode Constraints

The table below lists the changes relative to the Mode Base cluster for the fields of the ModeOptionStruct type. A blank field indicates no change.

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
2	**ModeTags**		1 to 8				

At least one entry in the SupportedModes attribute SHALL include the Manual mode tag in the ModeTags field list.

9.4.6. Mode Examples

A few examples of EVSE modes and their mode tags are provided below.

- For the "Manual" mode, tags: 0x4000 (Manual)
- For the "Auto-scheduled" mode, tags: 0x4001 (TimeOfUse)
- For the "Solar" mode, tags: 0x4002 (SolarCharging)
- For the "Auto-scheduled with Solar charging" mode, tags: 0x4001 (TimeOfUse), 0x4002 (SolarCharging)

9.5. Energy Preference Cluster

This cluster provides an interface to specify preferences for how devices should consume energy.

NOTE	Support for Energy Preference cluster is provisional.

9.5.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Revision	Description
1	Initial Release

9.5.2. Classification

Hierarchy	Role	Scope	PICS Code
Base	Application	Endpoint	EPREF

9.5.3. Cluster ID

ID	Name	Conformance
0x009B	Energy Preference	P

9.5.4. Features

This cluster SHALL support the FeatureMap bitmap attribute as defined below.

Bit	Code	Feature	Conformance	Summary
0	**BALA**	EnergyBalance	O.a+	Device can balance energy consumption vs. another priority
1	**LPMS**	LowPowerModeSensitivity	O.a+	Device can adjust the conditions for entering a low power mode

9.5.4.1. EnergyBalance Feature

This feature allows a user to select from a list of energy balances with associated descriptions of which strategies a device will use to target the specified balance.

9.5.4.2. LowPowerModeSensitivity Feature

This feature allows the user to select a condition or set of conditions which will cause the device to switch to a mode using less power. For example, a device might provide a scale of durations that must elapse without user interaction before it goes to sleep.

9.5.5. Data Types

9.5.5.1. EnergyPriorityEnum Type

This data type is derived from enum8.

Value	Name	Summary	Conformance
0	**Comfort**		M
1	**Speed**		M
2	**Efficiency**		M
3	**WaterConsumption**		M

9.5.5.1.1. Comfort Value

This value SHALL emphasize user comfort; e.g. local temperature for a thermostat.

9.5.5.1.2. Speed Value

This value SHALL emphasize how quickly a device accomplishes its targeted use; e.g. how quickly a robot vacuum completes a cleaning cycle.

9.5.5.1.3. Efficiency Value

This value SHALL emphasize how much energy a device uses; e.g. electricity usage for a Pump.

9.5.5.1.4. Water consumption Value

This value SHALL emphasize how much water is consumed during device use; e.g. how much water a dishwasher uses during a cleaning cycle.

9.5.5.2. BalanceStruct Type

This represents a step along a scale of preferences.

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0	**Step**	percent	all	F	MS		M
1	**Label**	string	max 64	F			O

9.5.5.2.1. Step Field

This field SHALL indicate the relative value of this step.

9.5.5.2.2. Label Field

This field SHALL indicate an optional string explaining which actions a device might take at the given step value.

9.5.6. Attributes

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
0x0000	**EnergyBalances**	list[BalanceStruct]	2 to 10	F		R V	BALA
0x0001	**CurrentEnergyBalance**	uint8	all	N		RW VO	BALA
0x0002	**EnergyPriorities**	list[EnergyPriorityEnum]	2	F		R V	BALA
0x0003	**LowPowerModeSensitivities**	list[BalanceStruct]	2 to 10	F		R V	LPMS
0x0004	**CurrentLowPowerModeSensitivity**	uint8	all	N		RW VO	LPMS

9.5.6.1. EnergyBalances Attribute

This attribute SHALL indicate a list of BalanceStructs, each representing a step along a linear scale

of relative priorities. A Step field with a value of zero SHALL indicate that the device SHOULD entirely favor the priority specified by the first element in EnergyPriorities; whereas a Step field with a value of 100 SHALL indicate that the device SHOULD entirely favor the priority specified by the second element in EnergyPriorities. The midpoint value of 50 SHALL indicate an even split between the two priorities.

This SHALL contain at least two BalanceStructs.

Each BalanceStruct SHALL have a Step field larger than the Step field on the previous BalanceStruct in the list.

The first BalanceStruct SHALL have a Step value of zero, and the last BalanceStruct SHALL have a Step value of 100.

9.5.6.2. CurrentEnergyBalance Attribute

This attribute SHALL indicate the current preference of the user for balancing different priorities during device use. The value of this attribute is the index, 0-based, into the EnergyBalances attribute for the currently selected balance.

If an attempt is made to set this attribute to an index outside the maximum index for EnergyBalances, a response with the status code CONSTRAINT_ERROR SHALL be returned.

If the value of EnergyBalances changes after an update, the device SHALL migrate the value of the CurrentEnergyBalance attribute to the index which the manufacturer specifies most closely matches the previous value, while preserving extreme preferences as follows:

1. If the previous value of CurrentEnergyBalance was zero, indicating a total preference for the priority specified by the first element in EnergyPriorities, the new value of CurrentEnergyBalance SHALL also be zero.
2. If the previous value of CurrentEnergyBalance was the index of the last BalanceStruct in the previous value of EnergyBalances, indicating a total preference for the priority specified by the last element in EnergyPriorities, the new value of CurrentEnergyBalance SHALL be the index of the last element in the updated value of EnergyBalances.

9.5.6.3. EnergyPriorities Attribute

This attribute SHALL indicate two extremes for interpreting the values in the EnergyBalances attribute. These two priorities SHALL be in opposition to each other; e.g. Comfort vs. Efficiency or Speed vs. WaterConsumption.

If the value of EnergyPriorities changes after an update to represent a new balance between priorities, the value of the CurrentEnergyBalance attribute SHALL be set to its default.

9.5.6.4. LowPowerModeSensitivities Attribute

This attribute SHALL indicate a list of BalanceStructs, each representing a condition or set of conditions for the device to enter a low power mode.

This SHALL contain at least two BalanceStructs.

Each BalanceStruct SHALL have a Step field larger than the Step field on the previous BalanceStruct in the list.

9.5.6.5. CurrentLowPowerModeSensitivity Attribute

This attribute SHALL indicate the current preference of the user for determining when the device should enter a low power mode. The value of this attribute is the index, 0-based, into the LowPowerModeSensitivities attribute for the currently selected preference.

If an attempt is made to set this attribute to an index outside the maximum index for LowPowerModeSensitivities, a response with the status code CONSTRAINT_ERROR SHALL be returned.

If the value of LowPowerModeSensitivities changes after an update, the device SHALL migrate the value of the LowPowerModeSensitivity attribute to the index which the manufacturer specifies most closely matches the previous value.

9.6. Device Energy Management Mode Cluster

This cluster is derived from the Mode Base cluster, defining additional mode tags and namespaced enumerated values for Device Energy Management devices.

NOTE	Support for Device Energy Management Mode cluster is provisional.

9.6.1. Revision History

The global ClusterRevision attribute value SHALL be the highest revision number in the table below.

Revision	Description
1	Initial Release

9.6.2. Classification

Hierarchy	Role	Scope	PICS Code
Mode Base	Application	Endpoint	DEMM

9.6.3. Cluster ID

ID	Name	Conformance
0x009F	Device Energy Management Mode	P

9.6.4. Data Types

9.6.4.1. ModeOptionStruct Type

The table below lists the changes relative to the Mode Base cluster for the fields of the ModeOptionStruct type. A blank field indicates no change.

ID	Name	Type	Constraint	Quality	Default	Access	Conformance
2	**ModeTags**		1 to 8				M

9.6.5. Derived Cluster Namespace

This namespace includes definitions for data associated exclusively with the derived cluster.

9.6.5.1. Mode Tags

The following table defines the derived cluster specific ModeTag values.

Mode Tag Value	Name
0x4000	**NoOptimization**
0x4001	**DeviceOptimization**
0x4002	**LocalOptimization**
0x4003	**GridOptimization**

9.6.5.1.1. NoOptimization Tag

The device prohibits optimization of energy usage management: its energy usage is determined only by the user configuration and internal device needs. This tag cannot be included with any of the other tags defined below in a mode.

9.6.5.1.2. DeviceOptimization Tag

The device is permitted to manage its own energy usage. For example, using tariff information it may obtain.

9.6.5.1.3. LocalOptimization Tag

The device permits management of energy usage by an energy manager to optimize the local energy usage.

9.6.5.1.4. GridOptimization Tag

The device permits management of energy usage by an energy manager to optimize the grid energy usage.

9.6.6. Mode Examples

A few examples of Device Energy Management modes and their mode tags are provided below.

- For the "No Energy Management (Forecast reporting only)" mode, tags: 0x4000 (NoOptimiza-

tion).

- For the "Device Energy Management" mode, tags: 0x4001 (DeviceOptimization).
- For the "Home Energy Management" mode, tags: 0x4001 (DeviceOptimization), 0x4002 (LocalOptimization).
- For the "Grid Energy Management" mode, tags: 0x4003 (GridOptimization).
- For the "Full Energy Management" mode, tags: 0x4001 (DeviceOptimization), 0x4002 (LocalOptimization), 0x4003 (GridOptimization).