Software Design Specification

Z-Wave Command Class Specification, A-M

Document No.:	SDS12657
Version:	1.0
Description:	This is part I of the Z-Wave Command Class Specification. The document describes the Command Classes and associated Commands used by Z-Wave enabled products ensuring that compliant products will be interoperable.
Written By:	JFR;ABR;NOBRIOT
Date:	
Reviewed By:	ABR;BBR;JFR;NTJ;NOBRIOT
Restrictions:	Public

Approved by:

Date	CET	Initials	Name	Justification
2016-08-26	14:44:06	NTJ	Niels Thybo Johansen	

DOCUMENTATION DISCLAIMER

Copyright Notice

Trademark Notice

License Restrictions Warranty/Consequential Damages Disclaimer

Warranty Disclaimer

Restricted Rights Notice

Hazardous Applications Notice

REVISION RECORD

Doc. Rev	Date	By	Sections affected	Brief description of changes
12	20160823	JFR	All	Prepared for Public Z-Wave initiative

Table of Contents

Table of Figures

Table of Tables

1 ABBREVIATIONS

Abbreviation	Explanation
AMR	Automatic Meter Reading
API	Application Programming Interface
ASCII	American Standard Code for Information Interchange. An ASCII code is the numerical representation of a character.
AV	Audio/Video
DHCP	Dynamic Host Configuration Protocol.
DNS	Dynamic Host Service
DST	Daylight Savings Time
HRV	Heat Recovery Ventilation
ID	Identifier
IP	Internet Protocol
IPV4	Internet Protocol version 4
IPV6	Internet Protocol version 6
LF	Linefeed character.
LSB	Less significant bit
MSB	Most significant bit
NIF	Node Information Frame
PIR	Pyroelectric Infrared Motion Sensor
SUC	Static Update Controller
TZO	Time Zone Offset
Unicode	Unicode is a standard for encoding of characters. For more information please visit http://www.unicode.org/
UTC	Universal Time (sometimes also called “Zulu Time”) was called Greenwich Mean Time (GMT) before 1972
WMC	Windows Vista Media Center and Media Center 2005 remote controls
Z/IP	Z-Wave for IP

2 INTRODUCTION

This document describes the command classes and associated commands that must be used when designing and implementing Z-Wave™ products. A subset of command classes is typically mandatory for a given type of device. The application layer of the Z-Wave protocol handles all commands.

This document is part I of the former SDS11060-18 - Z-Wave Command Class Specification and covers command classes A-M. Refer to [2] for part II covering command classes N-Z.

Read also this document in conjunction with [1] for Z-Wave devices and [9] for Z-Wave Plus devices.

2.1 Purpose

The purpose of this document is to describe the command classes used by the application layer of the Z-Wave protocol.

2.2 Precedence of definitions

Device Class, Device Type and Command Class Specifications approved as final version (ver. 1.00) during the Device Class, Device Type and Command Class open review process have precedence over this document until integrated into this document.

2.3 Terms used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document MUST be interpreted as described in IETF RFC 2119 [5].

This document defines functionality as deprecated or obsoleted.

The term "obsolete" means that the functionality MUST NOT be supported in new implementations applying for certification.

A controller SHOULD provide controlling capabilities of the actual functionality for backwards compatibility with legacy devices.

The term "deprecated" also indicates an obsolete definition, but it permits new implementations applying for certification.

Thus, the term “deprecated” means that the functionality SHOULD NOT be supported in new implementations applying for certification. Often, another substitute functionality is REQUIRED if the deprecated functionality is implemented.

A controller SHOULD provide controlling capabilities of the actual functionality for backwards compatibility with legacy devices.

3 COMMAND CLASS OVERVIEW

Interoperability between devices relies on Command Classes. If one device controls a command class and another device support the same command class then these devices are able to communicate.

3.1 Overview

This document describes all the commands that are used by devices when communicating with other devices. These commands are divided into functionally related groups called command classes. Each command class contains the commands associated with a given functionality.

All commands MUST be implemented for a given Command Class when supported by the device itself. However, a device controlling a Command Class in another device can chose to implement control of selected commands within this Command Class depending on the requirements with respect to control.

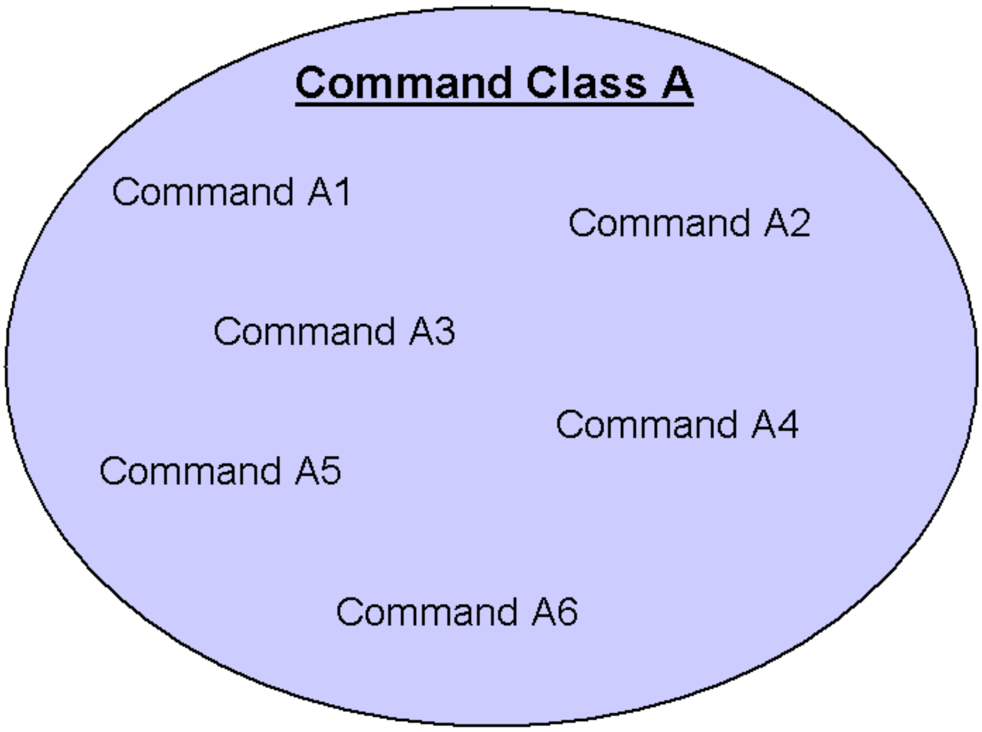


Figure 1, Command class and related commands

NOTICE: **It is mandatory to select and implement the device and command classes that accurately reflect the functionality available in device.**

E.g., a dimmable light module MUST be implemented as a "Multilevel Switch Generic Device Class" and "Multilevel Power Switch Specific Device Class".

3.2 Node Information Frame

The Node Information Frame (NIF) is used to inform other devices about node capabilities. The NIF contains a structure with a protocol specific part that is handled by the Z-Wave protocol and an application specific part that is filled in by the application. The protocol specific part consists of a bit telling if the node is a continuously listening device, the Basic Device Class the device is based on etc. The application specific part consists of the Generic and Specific Device Class and the Command Classes that are supported and/or controlled by the device.

A NIF will be sent to the controller when a node is to be included in the network, excluded from the network or upon request.

The figure below shows the parameters hosted by NIF.

Byte descriptor \ bit number	7	6	5	4	3	2	1	0
Capability	Liste-ning	Z-Wave Protocol Specific Part						
Security	Opt. Func.	Z-Wave Protocol Specific Part						
Reserved	Z-Wave Protocol Specific Part							
Basic *)	Basic Device Class (Z-Wave Protocol Specific Part)							
Generic	Generic Device Class							
Specific	Specific Device Class							
NodeInfo[0]	Command Class 1							
…	…							
NodeInfo[n-1]	Command Class n							

*) The “Basic” field is only included when the NIF is sent by a controller

Figure 2, Node Information Frame format

The Z-Wave Protocol in a controller saves all the Node Information except the supported and controlled Command Classes when a node is included in the network. The reserved field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

3.2.1 Z-Wave Protocol Specific Part

The protocol specific part of the NIF is handled by the Z-Wave protocol. This information is automatically inserted in the packet by the protocol layer when transferring data using the API.

Basic Device Class

The Basic Device Class field contains an identifier that identifies what Basic Device Class this node is based on and is set by the Z-Wave protocol. A detailed description of all available Basic Device Classes is given in [1] for Z-Wave devices and [9] for Z-Wave Plus devices. The Z-Wave Plus devices have an additional parameter Role Type defining device role in the network. The Role Type parameter is announced via the Z-Wave Plus Info Command Class.

Note that the “Basic” field is only included when the NIF frame is sent by a controller.

3.2.2 Application Specific Part

The application specific part of the NIF is handled by the application. The information must be in accordance with the defined classes to obtain interoperability.

Listening Flag

The Listening flag is used to indicate that the node is always listening if set. An always listening node must be powered continuously and reside on a fixed position in the installation. An always listening node is included in the routing table to assist as repeater in the network. The routing table is static during normal operation. In case the Listening flag is cleared the node is non-listening. This is typically used for battery operated nodes being asleep when the protocol is idle to prolong battery lifetime. A battery operated node is not included in the routing table and is not used as a router in the network. In some instances the node’s position in the network is still determined, and stored by the protocol.

Optional Functionality Flag

The Optional Functionality flag is used to indicate that this node supports other command classes than the mandatory for the selected generic/specific device class and that a controlling node needs to look at the supported command classes to fully control this device.

Generic Device Class

The Generic Device Class field contains an identifier that identifies what Generic Device Class this node is part of and must be set by the application. For a detailed description of all available Generic Device Classes, refer to [1] for Z-Wave devices and [9] for Z-Wave Plus devices.

Specific Device Class

The Specific Device Class field specifies what Specific Device Class this application is part of and must be set by the application. For a detailed description of all available Specific Device Classes, refer to [1] for Z-Wave devices and [9] for Z-Wave Plus devices.

Command Class

The Command Class field is used to advertise Command Classes implemented by the node. The field MUST NOT be longer than 35 bytes.

The field MUST advertise the list of Command Classes that the node supports.

The field MAY advertise the list of Command Classes that the node can control in other nodes. If present, the list of controlling Command Classes MUST be prepended by the COMMAND_CLASS_MARK Command Class identifier.

The Basic Command Class MUST NOT be advertised in the Node Information Frame.

If supported, the Z-Wave Plus Info Command Class MUST be the first Command Class in the list.

It has been found that legacy controllers MAY read as little as 6 lines from this list. For backwards compatibility, the list SHOULD advertise supported command classes in the order indicated in Table 1 :

Table 1, NIF :: Command Class advertising priorities

Priority	Command Class
1 (First line)	COMMAND_CLASS_ZWAVEPLUS_INFO Applies only to Z-Wave Plus products
2 (if supported)	COMMAND_CLASS_SWITCH_MULTILEVEL or COMMAND_CLASS_SWITCH_BINARY
3 (if supported)	COMMAND_CLASS_SWITCH_ALL
4 (if supported)	COMMAND_CLASS_ASSOCIATION
5	All other command classes

Figure 3 shows the structure of the Node Information frame:

Byte descriptor \ bit number	7	6	5	4	3	2	1	0
NodeInfo[0]	Command Class 1 (Support)							
…	…							
NodeInfo[x-1]	Command Class x (Support)							
NodeInfo[x]	Command Class x+1 (COMMAND_CLASS_MARK)							
NodeInfo[x+1]	Command Class x+2 (Control)							
…	…							
NodeInfo[n-1]	Command Class n (Control)							

Figure 3, Support/control command class list format

In case the device have no supported command classes and only can control other devices, the list start with the identifier COMMAND_CLASS_MARK.

3.2.3 NIF and Security Command Classes

The Command Class list advertised in the NIF MAY vary depending on the inclusion state and S0/S2 bootstrapping state of the node.

A node supporting security (S0 or S2) MUST advertise command classes in the NIF according to Table 2, Table 3 and Table 4.

Table 2, NIF Advertisements by security bootstrapped included node

Category	Advertising in the NIF
Command classes that the node supports via non-secure communication	MUST be advertised
Command classes that the node only supports via secure communication	MUST NOT be advertised
Command classes that the node controls via non-secure communication	MAY be advertised
Command classes that the node only controls via secure communication	MUST NOT be advertised
Support for Security Command Class	MUST be advertised
Support for Security 2 Command Class	MUST be advertised

Table 3, NIF Advertisements by non-security bootstrapped included node

Category	Advertising in the NIF
Command classes that the node supports via non-secure communication	MUST be advertised
Command classes that the node only supports via secure communication	MUST NOT be advertised
Command classes that the node controls via non-secure communication	MAY be advertised
Command classes that the node only controls via secure communication	MUST NOT be advertised
Support for Security Command Class	SHOULD NOT be advertised
Support for Security 2 Command Class	MUST NOT be advertised

Table 4, NIF Advertisements by non-included node

Category	Advertising in the NIF
Command classes that the node supports via non-secure communication	MUST be advertised
Command classes that the node only supports via secure communication	MUST NOT be advertised
Command classes that the node controls via non-secure communication	MAY be advertised
Command classes that the node only controls via secure communication	MUST NOT be advertised
Support for Security Command Class	MUST be advertised
Support for Security 2 Command Class	MUST be advertised

3.3 Command class frame format

All commands have a common header consisting of a Command Class identifier and a Command identifier. Further, the command can have from zero to n bytes of command data. The bit-numbering scheme is “LSB 0” because bit numbering starts at zero for the least significant bit (Notice that LSB is denoted as ‘Bit 0’… and MSB is denoted as ‘Bit 7 throughout the document). The figures below show the generic command frame for the two possible formats:

7	6	5	4	3	2	1	0
Command Class (0x20 – 0xEE)							
Command							
Command Data 1							
..							
Command Data n							

Figure 4, Generic command format

7	6	5	4	3	2	1	0
Command Class MSB (0xF1 – 0xFF)							
Command Class LSB (0x00 – 0xFF)							
Command							
Command Data 1							
..							
Command Data n							

Figure 5, Generic extended command format

3.3.1 Command class

The Command Class identifier range is as follows:

Table 5, Command Class identifier range

Command Class	Description
0x00	No Operation. Used by Z-Wave Protocol. MAY be used by the application.
0x01 – 0x1F	Reserved for the Z-Wave protocol
0x20 – 0xEE	Application Command Classes
0xEF	Support/Control Mark
0xF0	Non interoperable
0xF1 – 0xFF	Extended Application Command Classes

A Command Class can contain up to 255 different Commands. If the Command Class field is set to 0xF1 through 0xFF then there is another Command Class byte added. This allows for future extensions of the Command Classes. The strategy of having an Extended Command Class followed by the actual command identifier provides the possibility of having more than 4000 Command Classes.

3.3.2 Command

The command field contains the specific command that SHOULD be executed. The field has a length of 1 byte.

3.3.3 Command data (N bytes)

The command data field contains data related to the command. Simple commands, such as get commands, contain no command data. Other commands, such as set or report commands can contain several bytes of command data.

3.3.4 Command class versioning

All command classes have a version number. The following rules apply to avoid interoperability issues when introducing the same Command Class with different versions:

1. A node MUST NOT discard a frame based on the length field. Application MUST use command class identifier and the command identifier to interpret the application frame. This enables a device, which supports version 1 of a command class to interpret the version 1 part of a received version 2 of the command.
2. All implementations of a Command Class version higher than 1 MUST initialize all parameters associated with the version higher than 1. Thereby can a version 2 Command Class implementation in a device interpret a received version 1.
3. A device supporting a Command Class having a version higher than 1 MUST support the Version Command Class to be able to identify the supported version. In case the device does not support the Version Command Class then it can be assumed that all command classes are equal to version 1.
4. It is allowed to make devices only supporting an older version of the command class despite a newer version exists as long as the generic/specific device specification does not require a specific version implemented.

The number of data fields transmitted can be determined from a field usually called Length. The Length field is used in cases where the same command has a variable number of command data fields.

3.4 Command classes

The Command Class field defines a group of commands representing a given functionality. Table 6 lists general purpose Command Classes.

Table 6, General purpose Command Class identifiers

Command Class name	Command Class identifier	Value	Newest version
Alarm (see Notification)	COMMAND_CLASS_ALARM	0x71	8
Anti-theft	COMMAND_CLASS_ANTITHEFT	0x5D	2
Application Capability	COMMAND_CLASS_APPLICATION_CAPABILITY	0x57	1
Application Status	COMMAND_CLASS_APPLICATION_STATUS	0x22	1
Association	COMMAND_CLASS_ASSOCIATION	0x85	2
Association Command Configuration	COMMAND_CLASS_ASSOCIATION_COMMAND_CONFIGURATION	0x9B	1
Association Group Information	COMMAND_CLASS_ASSOCIATION_GRP_INFO	0x59	3
Battery	COMMAND_CLASS_BATTERY	0x80	1
Central Scene	COMMAND_CLASS_CENTRAL_SCENE	0x5B	3
Clock	COMMAND_CLASS_CLOCK	0x81	1
Configuration	COMMAND_CLASS_CONFIGURATION	0x70	4
Controller Replication	COMMAND_CLASS_CONTROLLER_REPLICATION	0x21	1
CRC-16 Encapsulation	COMMAND_CLASS_CRC_16_ENCAP	0x56	1
Device Reset Locally	COMMAND_CLASS_DEVICE_RESET_LOCALLY	0x5A	1
Firmware Update Meta Data	COMMAND_CLASS_FIRMWARE_UPDATE_MD	0x7A	5
Geographic Location	COMMAND_CLASS_GEOGRAPHIC_LOCATION	0x8C	1
Grouping Name	COMMAND_CLASS_GROUPING_NAME **[DEPRECATED]**	0x7B	1
Hail	COMMAND_CLASS_HAIL **[DEPRECATED]**	0x82	1
Indicator	COMMAND_CLASS_INDICATOR	0x87	2
IP Association	COMMAND_CLASS_IP_ASSOCIATION	0x5C	1
IP Configuration	COMMAND_CLASS_IP_CONFIGURATION **[OBSOLETED]**	0x9A	1
Language	COMMAND_CLASS_LANGUAGE	0x89	1
Mailbox	COMMAND_CLASS_MAILBOX	0x69	1
Manufacturer Proprietary	COMMAND_CLASS_MANUFACTURER_PROPRIETARY	0x91	1
Manufacturer Specific	COMMAND_CLASS_MANUFACTURER_SPECIFIC	0x72	2
Mark (Support/control mark)	COMMAND_CLASS_MARK	0xEF	N/A
Multi Channel	COMMAND_CLASS_MULTI_CHANNEL	0x60	4
Multi Channel Association	COMMAND_CLASS_MULTI_CHANNEL_ASSOCIATION	0x8E	3
Multi Command	COMMAND_CLASS_MULTI_COMMAND	0x8F	1
Network Management Basic Node	COMMAND_CLASS_NETWORK_MANAGEMENT_BASIC	0x4D	1
Network Management Inclusion	COMMAND_CLASS_NETWORK_MANAGEMENT_INCLUSION	0x34	1
Network Management Primary	COMMAND_CLASS_NETWORK_MANAGEMENT_PRIMARY	0x54	1
Network Management Proxy	COMMAND_CLASS_NETWORK_MANAGEMENT_PROXY	0x52	1
No Operation	COMMAND_CLASS_NO_OPERATION	0x00	1
Node Naming and Location	COMMAND_CLASS_NODE_NAMING	0x77	1
Non interoperable	COMMAND_CLASS_NON_INTEROPERABLE	0xF0	N/A
Notification (former Alarm)	COMMAND_CLASS_NOTIFICATION	0x71	8
Proprietary	COMMAND_CLASS_PROPRIETARY **[DEPRECATED]**	0x88	1
Remote Association Activate	COMMAND_CLASS_REMOTE_ASSOCIATION_ACTIVATE **[OBSOLETED]**	0x7C	1
Remote Association	COMMAND_CLASS_REMOTE_ASSOCIATION **[OBSOLETED]**	0x7D	1

Command Class name	Command Class identifier	Value	Newest version
Configuration			
Schedule	COMMAND_CLASS_SCHEDULE	0x53	3
Screen Attributes	COMMAND_CLASS_SCREEN_ATTRIBUTES	0x93	2
Screen Meta Data	COMMAND_CLASS_SCREEN_MD	0x92	2
Security	COMMAND_CLASS_SECURITY	0x98	1
Security Scheme 0 mark	COMMAND_CLASS_SECURITY_SCHEME0_MARK	0xF100	N/A
Supervision	COMMAND_CLASS_SUPERVISION	0x6C	1
Time	COMMAND_CLASS_TIME	0x8A	2
Time Parameters	COMMAND_CLASS_TIME_PARAMETERS	0x8B	1
Transport Service	COMMAND_CLASS_TRANSPORT_SERVICE	0x55	2
User Code	COMMAND_CLASS_USER_CODE	0x63	1
Version	COMMAND_CLASS_VERSION	0x86	2
Wake Up	COMMAND_CLASS_WAKE_UP	0x84	2
Z-Wave for IP	COMMAND_CLASS_ZIP	0x23	3
Z-Wave for IP, Naming and Location	COMMAND_CLASS_ZIP_NAMING	0x68	1
Z-Wave for IP,-ND	COMMAND_CLASS_ZIP_ND	0x58	1
Z-Wave for IP, 6LoWPAN	COMMAND_CLASS_ZIP_6LOWPAN (ITU-T G.9959 / IETF RFC 7428)	0x4F	8
Z-Wave for IP, Gateway	COMMAND_CLASS_ZIP_GATEWAY	0x5F	1
Z-Wave for IP, Portal	COMMAND_CLASS_ZIP_PORTAL	0x61	1
Z-Wave Plus Info	COMMAND_CLASS_ZWAVEPLUS_INFO	0x5E	2

Table 7 lists Command Classes targeting specific device types.

Table 7, Device related Command Class identifiers

Command Class name	Command Class identifier	Value	Newest version
Alarm Sensor	COMMAND_CLASS_SENSOR_ALARM **[DEPRECATED]**	0x9C	1
Alarm Silence	COMMAND_CLASS_SILENCE_ALARM	0x9D	1
All Switch	COMMAND_CLASS_SWITCH_ALL	0x27	1
Barrier Operator	COMMAND_CLASS_BARRIER_OPERATOR	0x66	1
Basic	COMMAND_CLASS_BASIC	0x20	2
Basic Tariff Information	COMMAND_CLASS_BASIC_TARIFF_INFO	0x36	1
Basic Window Covering	COMMAND_CLASS_BASIC_WINDOW_COVERING **[OBSOLETED]**	0x50	1
Binary Sensor	COMMAND_CLASS_SENSOR_BINARY **[DEPRECATED]**	0x30	2
Binary Switch	COMMAND_CLASS_SWITCH_BINARY	0x25	2
Binary Toggle Switch	COMMAND_CLASS_SWITCH_TOGGLE_BINARY **[DEPRECATED]**	0x28	1
Climate Control Schedule	COMMAND_CLASS_CLIMATE_CONTROL_SCHEDULE **[DEPRECATED]**	0x46	1
Color Switch	COMMAND_CLASS_SWITCH_COLOR	0x33	3
DCP List Configuration	COMMAND_CLASS_DCP_CONFIG	0x3A	1
DCP List Monitoring	COMMAND_CLASS_DCP_MONITOR	0x3B	1
Door Lock	COMMAND_CLASS_DOOR_LOCK	0x62	3
Door Lock Logging	COMMAND_CLASS_DOOR_LOCK_LOGGING	0x4C	1
Energy Production	COMMAND_CLASS_ENERGY_PRODUCTION	0x90	1
Entry Control	COMMAND_CLASS_ENTRY_CONTROL	0x6F	1
HRV Status	COMMAND_CLASS_HRV_STATUS	0x37	1
HRV Control	COMMAND_CLASS_HRV_CONTROL	0x39	1
Humidity Control Mode	COMMAND_CLASS_HUMIDITY_CONTROL_MODE	0x6D	1
Humidity Control Operating State	COMMAND_CLASS_HUMIDITY_CONTROL_OPERATING_STATE	0x6E	1
Humidity Control Setpoint	COMMAND_CLASS_HUMIDITY_CONTROL_SETPOINT	0x64	1
Irrigation	COMMAND_CLASS_IRRIGATION	0x6B	1
Lock	COMMAND_CLASS_LOCK	0x76	1

Command Class name	Command Class identifier	Value	Newest version
Meter	COMMAND_CLASS_METER	0x32	4
Meter Table Configuration	COMMAND_CLASS_METER_TBL_CONFIG	0x3C	1
Meter Table Monitor	COMMAND_CLASS_METER_TBL_MONITOR	0x3D	2
Meter Table Push Configuration	COMMAND_CLASS_METER_TBL_PUSH	0x3E	1
Move To Position Window Covering	COMMAND_CLASS_MTP_WINDOW_COVERING **[OBSOLETED]**	0x51	1
Multilevel Sensor	COMMAND_CLASS_SENSOR_MULTILEVEL	0x31	10
Multilevel Switch	COMMAND_CLASS_SWITCH_MULTILEVEL	0x26	4
Multilevel Toggle Switch	COMMAND_CLASS_SWITCH_TOGGLE_MULTILEVEL **[DEPRECATED]**	0x29	1
Powerlevel	COMMAND_CLASS_POWERLEVEL	0x73	1
Prepayment	COMMAND_CLASS_PREPAYMENT	0x3F	1
Prepayment Encapsulation	COMMAND_CLASS_PREPAYMENT_ENCAPSULATION	0x41	1
Protection	COMMAND_CLASS_PROTECTION	0x75	2
Pulse Meter	COMMAND_CLASS_METER_PULSE **[DEPRECATED]**	0x35	1
Rate Table Configuration	COMMAND_CLASS_RATE_TBL_CONFIG	0x48	1
Rate Table Monitoring	COMMAND_CLASS_RATE_TBL_MONITOR	0x49	1
Scene Activation	COMMAND_CLASS_SCENE_ACTIVATION	0x2B	1
Scene Actuator Configuration	COMMAND_CLASS_SCENE_ACTUATOR_CONF	0x2C	1
Scene Controller Configuration	COMMAND_CLASS_SCENE_CONTROLLER_CONF	0x2D	1
Schedule Entry Lock	COMMAND_CLASS_SCHEDULE_ENTRY_LOCK **[DEPRECATED]**	0x4E	3
Sensor Configuration	COMMAND_CLASS_SENSOR_CONFIGURATION **[OBSOLETED]**	0x9E	1
Simple AV Control	COMMAND_CLASS_SIMPLE_AV_CONTROL	0x94	4
Tariff Table Configuration	COMMAND_CLASS_TARIFF_TBL_CONFIG	0x4A	1
Tariff Table Monitor	COMMAND_CLASS_TARIFF_TBL_MONITOR	0x4B	1
Thermostat Fan Mode	COMMAND_CLASS_THERMOSTAT_FAN_MODE	0x44	4
Thermostat Fan State	COMMAND_CLASS_THERMOSTAT_FAN_STATE	0x45	2
Thermostat Mode	COMMAND_CLASS_THERMOSTAT_MODE	0x40	3
Thermostat Operating State	COMMAND_CLASS_THERMOSTAT_OPERATING_STATE	0x42	2
Thermostat Setback	COMMAND_CLASS_THERMOSTAT_SETBACK	0x47	1
Thermostat Setpoint	COMMAND_CLASS_THERMOSTAT_SETPOINT	0x43	3
Window Covering	COMMAND_CLASS_WINDOW_COVERING	0x6A	1

3.5 Common command class variables

This section defines of a number of variables that are common used across command classes.

3.5.1 Reserved values and reserved bits

Values of fields in commands that are marked as “reserved” MUST NOT be used by devices sending commands and MUST be ignored by devices receiving commands.

Bits in commands that are marked as “reserved” MUST be set to 0 by devices sending commands and MUST be ignored by devices receiving commands.

Undefined values for a given parameter MUST be treated as “reserved”. Reserved values MUST NOT be used by a sending node and MUST be ignored by a receiving node.

3.5.2 Meter Dataset (24 bits)

The Meter Dataset variable defines the parameters supported by a certain meter; this is used among others to report capabilities, and when requesting data from a meter. Meter Dataset currently used by the AEC framework.

Meter Type	Byte #	Bit #	Description
Single-E electric meter Twin-E electric meter 3P Single Direct electric meter 3P Single ECT electric meter 1 Phase Direct Electricity Meter	Byte 1	Bit 0	Total Primary Active Energy (kWh) Energy delivered to the unit before transformation.
	Byte 1	Bit 1	Total Primary Reactive Energy (kvarh) Energy delivered to the unit before transformation.
	Byte 1	Bit 2	Total Secondary Active Energy (kWh) Consumed energy after transformation.
	Byte 1	Bit 3	Total Secondary Reactive Energy (kvarh) Consumed energy after transformation.
	Byte 1	Bit 4	Instantaneous Primary Active Power (kW) Current power consumption.
	Byte 1	Bit 5	Instantaneous Primary Reactive Power (kvar) Current power consumption.
	Byte 1	Bit 6	Primary Active Maximum Demand (kW) All time maximum active power demand.
	Byte 1	Bit 7	Primary Reactive Maximum Demand (kvar) All time maximum reactive power demand.
	Byte 2	Bit 0	Cumulative Primary Active Maximum Demand (kW) Accumulated power over period.
	Byte 2	Bit 1	Cumulative Primary Reactive Maximum Demand (kvar) Accumulated power over period.
	Byte 2	Bit 2	Voltage Phase 1 (V) Voltage measured on phase 1.
	Byte 2	Bit 3	Voltage Phase 2 (V) Voltage measured on phase 2.
	Byte 2	Bit 4	Voltage Phase 3 (V) Voltage measured on phase 3.
	Byte 2	Bit 5	Ampere Phase 1 (A) Amperes measured on phase 1.
	Byte 2	Bit 6	Ampere Phase 2 (A) Amperes measured on phase 2.
	Byte 2	Bit 7	Ampere Phase 3 (A) Amperes measured on phase 3.
	Byte 3	Bit 0	Pulse Input (Pulse Count) Pulse count on pulse input port of meter.

	Byte 3	Bit 1	Current Transformation Ratio (ratio) Transformation ratio between primary and secondary energy.
	Byte 3	Bit 2	Power Factor (%) Power factor contains the ratio of real power to total power, and is expressed as a percentage (%).
	Byte 3	Bit 3-7	Reserved Value reserved for future expansion.
Gas meter Water meter	Byte 1	Bit 0	Accumulated Volume (m^3) Total accumulated volume.
	Byte 1	Bit 1	Current Flow (l/h, m^3/h) Current flow value.
	Byte 1	Bit 2	Current Pressure (kPa) Current pressure value.
	Byte 1	Bit 3	Peak Flow (l/h, m^3/h) Peak flow demand recorded.
	Byte 1	Bit 4	Hour Counter (Hours) Hours in operation.
	Byte 1	Bit 5	Input A (m^3/h, kW, m^3, kWh, MWh, Pulse Count) Value of input port A on the meter.
	Byte 1	Bit 6	Input B (m^3/h, kW, m^3, kWh, MWh, Pulse Count) Value of input port B on the meter.
	Byte 1 Byte 2 Byte 3	Bit 7 Bit 0-7 Bit 0-7	Reserved Value reserved for future expansion.
Heating meter Cooling meter	Byte 1	Bit 0	Heat Energy (kWh, MWh, GJ, GCal) Total accumulated energy used for heating.
	Byte 1	Bit 1	Cooling Energy (kWh, MWh, GJ, GCal) Total accumulated energy used for cooling.
	Byte 1	Bit 2	Volume 1 (m^3, ton) Volume counter 1 – forward flow.
	Byte 1	Bit 3	Volume2 (m^3, ton) Volume counter 2 – return flow.
	Byte 1	Bit 4	Temperature 1 (Forward) (C^o, ^oF) Forward temperature.
	Byte 1	Bit 5	Temperature 2 (Return) (C^o, ^oF) Return temperature.
	Byte 1	Bit 6	Temperature 3 (C^o, ^oF) Temperature sensor 3.
	Byte 1	Bit 7	Actual Flow (Volume 1) (l/h, m^3/h) Current flow on volume 1.

	Byte 2	Bit 0	Actual Flow (Volume 2) (l/h, m^3/h) Current flow on volume 2.
	Byte 2	Bit 1	Actual Power (kW, MW) Current power.
	Byte 2	Bit 2	Peak Flow (l/h, m^3/h) Peak flow recorded.
	Byte 2	Bit 3	Peak Power (kW, MW) Peak power recorded.
	Byte 2	Bit 4	Input A (m^3/h, kW, m^3, kWh, MWh, Pulse Count) Value of input port A on the meter.
	Byte 2	Bit 5	Input B (m^3/h, kW, m^3, kWh, MWh, Pulse Count) Value of input port B on the meter.
	Byte 2	Bit 6	Hour Counter (Hours) Hours in operation.
	Byte 2 Byte 3	Bit 7 Bit 0-7	Reserved Value reserved for future expansion.
Electric Sub-Meter	Byte 1	Bit 0	Accumulated (kWh)
	Byte 1	Bit 1	Accumulated (kVAh)
	Byte 1	Bit 2	Instant (W)
	Byte 1	Bit 3	Accumulated (Pulse Count)
	Byte 1	Bit 4	Instant (V)
	Byte 1	Bit 5	Instant (A)
	Byte 1	Bit 6	Power Factor (%)
	Byte 1 Byte 2 Byte 3	Bit 7 Bit 0-7 Bit 0-7	Reserved

3.5.3 Meter Rate Type (2 bits)

The Meter Rate Type variable to indicate which rate is reported/requested or supported. Meter Rate Type currently used by the AEC framework.

Rate Type	Value
Reserved	0x00
Import	0x01
Export	0x02
Reserved	0x03

3.5.4 Meter Scale (5 bits)

The Meter Scale variable used to report the scale (unit) of a reported value. Meter Scale currently used by the AEC framework.

Meter Type	Scale	Value
Electric meter	kWh	0x00
	kVARh	0x01
	%	0x02
	Pulse count	0x03
	kVAR	0x04
	Voltage (V)	0x05
	Amperes (A)	0x06
	kW	0x07
	Ratio	0x08
	Reserved	0x09- 0x1F
Gas and water meter	Cubic meter	0x00
	Cubic feet	0x01
	US gallon	0x02
	Pulse count	0x03
	IMP gallon	0x04
	Liter	0x05
	kPa	0x06
	Centum cubic feet	0x07
	Cubic meter per hour	0x08
	Liter per hour	0x09
	kWh	0x0A
	MWh	0x0B
	KW	0x0C
	Hours	0x0D
	Reserved	0x0E-0x1F

Heating and Cooling Meter	Cubic meter (m^3)	0x00
	Metric Ton (tonne) (t)	0x01
	Cubic meter per hour (m^3/h)	0x02
	Liter per hour (l/h)	0x03
	kW	0x04
	MW	0x05
	kWh	0x06
	MWh	0x07
	Giga Joule (GJ)	0x08
	Giga Calorie (Gcal)	0x09
	Celsius (C^o)	0x0A
	Fahrenheit (^oF)	0x0B
	Hours	0x0C
	Reserved	0x0D-0x1F
Electric Sub-Meter	kWh	0x00
	kVAh	0x01
	W	0x02
	Pulse Count	0x03
	V	0x04
	A	0x05
	Power Factor (%)	0x06
	Reserved	0x07-0x1F

3.5.5 Meter Type (6 bits)

The Meter Type variable describes the various meter types. Meter Type currently used by the AEC framework.

Meter Type	Value
Reserved	0x00
Single-E electric meter	0x01
Gas meter	0x02
Water meter	0x03
Twin-E electric meter	0x04
3P Single Direct electric meter	0x05
3P Single ECT electric meter	0x06
1 Phase Direct Electricity Meter	0x07
Heating meter	0x08
Cooling meter	0x09
Combined Heating and Cooling Meter	0x0A
Electric Sub-Meter	0x0B
Reserved	0x0C-0x3F

3.6 Multi Channel overview

The section presents the concept of Multi Channel End Points. The concept ties closely to command classes such as Multi Channel Command Class, Multi Channel Association Command Class, Association Group Information Command Class as well as the Z-Wave Plus Icon Type. Multi Channel functionality may be used for controlling as well as for supporting devices.

3.6.1 Terminology

A Z-Wave node is conceptually an application resource in a plastic box communicating via a Z-Wave radio. Composite devices pack multiple application resources in the same plastic box, thus sharing the same Z-Wave radio. Application resources can always be addressed individually.

Within a network, a Z-Wave node is identified by its NodeID. The NodeID represents the plastic box and the radio.
Multi-resource devices are organized as Multi Channel End Points. Each application resource is identified by its own unique End Point. The plastic box itself is referred to as the Root Device.

Multi Channel Encapsulation allows a sending node to specify a source End Point and a destination End Point. Further, the destination End Point may be structured as a multicast mask, targeting up to 7 End Points by one command. Multi Channel Encapsulation is used for transmission from one End Point to another, from an End Point to a Root Device as well as from a Root Device to an End Point.

Multi Channel Encapsulation is not used between Root Devices.

An Aggregated End Point implements a function which relates to multiple individual End Points. An Aggregated End Point is addressed just as an individual End Point.
One example of an Aggregated End Point is a common power meter of a power strip which measures the total power consumption of all End Points.
The aggregation of End Points should not be confused with multicast addressing. Sending a Meter Reset command via multi-End Point addressing to all individual End Points causes all individual End Points to reset their individual meters. Sending a Meter Reset command to the Aggregated End Point causes the common power meter to be reset.

Bridging devices may provide connectivity to other technologies than Z-Wave via dynamic End Points. Dynamic End Points may be created, changed or removed.

A controlling device MAY use Multi Channel encapsulation to communicate with Multi Channel End Points in other devices. If such a controlling device does not implement any End Points, the device MUST NOT advertise the Multi Channel Command Class in the Node Information Frame.

One may create a Multi Channel Association to allow an End Point to control another End Point. The End Point may also control a Root Device. Likewise, a Multi Channel Association may be created to allow a Root Device to control an End Point.

The Association Group Information advertises the association capabilities of each Association Group in each End Point as well as in the Root Device.

3.6.2 Backwards compatibility

The Multi Channel concept provides a toolbox for sub-addressing. Any controlling device should implement the functionality required to interact with supporting Multi Channel devices.

Legacy devices only understand the concept of the Root Device. Therefore, a Multi Channel device providing multiple application resources also provides a meaningful subset of the application functionality via the Root Device on behalf of one or more End Points.

One example is a composite temperature and humidity sensor, which exposes the humidity sensor functionality via the Root Device as if the device was a stand-alone temperature sensor.

Another example is a 5-output power strip implemented as 5 Multi Channel End Points. The Root Device exposes one Binary Switch resource but a command to the Root Device spawns internal control commands to all outputs, so the Root Device acts as a master switch.

It is seen that the mapping of application functionality to the Root Device depends heavily on the actual product feature set. However, a few principles apply:

1. The Root Device provides access to the primary functionality of the actual product.

2. The Root Device of a Multi Channel device does not implement any application functionality that cannot be reached via End Points.

3. The Root Device of a Multi Channel always presents functionality, which can also be reached via End Point 1. If it makes sense in the actual product, a Root Device command may affect more End Points than End Point 1.

As it is optional how to forward commands from the Root Device to multiple End Points, one cannot be sure that a command to the Root Device will target all End Points. The Multi Channel multicast feature may be used to send a Set command to the first 7 End Points. Alternatively, one may communicate to each individual End Point. This works for Set as well as Get commands.

3.6.3 GUI presentation

The Root Device always advertises application functionality, even when the application functionality is actually provided by forwarding commands to one or more End Points of the device.

Management tools may have a need to present expanded views of Multi Channel devices, e.g. in the floor plan view of a smart home deployment. By ignoring application-style Command Classes supported by one or more End Points, the Root Device can be presented the way it really works.

3.6.4 Applicability examples

- A gateway may target a destination End Point to control the individual output of a power strip
- A gateway may create a Multi Channel Association from the Root Device Lifeline association group of a two-channel indoor/outdoor temperature sensor to receive sensor reports. The Multi Channel encapsulation source End Point allows the gateway to distinguish indoor readings from outdoor readings.
- One End Point of a dual End Point indoor/outdoor temperature sensor may have a (non-Multi Channel) association created to control the Root Device of an air conditioner on basis of the measured indoor temperature.

3.7 Actuator Control

An actuator device may support one or more actuator command classes. This chapter presents terminology and conventions applying to all actuator command classes.

The actuator control group comprises the following command classes:

Table 8, Command classes of the actuator control group

Command Class
Basic
Basic Window Covering [OBSOLETED]
Binary Switch
Color Switch
Door Lock
Move To Position Window Covering [OBSOLETED]
Multilevel Switch
Window Covering

3.7.1 Terminology

A <u>controlling</u> device controls a <u>supporting</u> device.

Actuators may be controlled in two ways.

A position controlled actuator responds to a position control command, which specifies a target value and optionally a duration for the transition from the current value to the target value. The value may be binary or multilevel.
State control commands may be encoded as special multilevel position control values, e.g. 0xFF for "On" and 0x00 for "Off". The most recent (non-zero) target value is restored in response to the "On" state control command.

A motion controlled actuator responds to start/stop commands. A motion control command specifies a direction, optionally a start value, and a transition rate.

3.7.2 Reporting values

A position control command may be used to initiate a transition to a new target value. A position report advertises the current value of the device hardware, optionally the target value, and the duration of the remaining transition. If the transition is initiated by a motion control command, the reported target value is the min or max value and the duration is the time needed to reach the target value at the actual transition rate.

A controlling device should not assume that the value advertised in a position report is identical to a value previously issued with a position control command when a transition has ended.

3.7.3 Command values vs. hardware values

A device may implement fewer hardware levels than supported by a position control Command Class. The hardware levels should be distributed uniformly over the entire range. The mapping of command values to hardware levels must be monotonous, i.e. a higher value MUST be mapped to either the same or a higher hardware level. An example is found in Table 9.

Table 9, Mapping command values to hardware levels (example)

Multilevel Value	Hardware level	State
0	0	Off
1..33	33%	On
34..66	66%	On
67..99	100%	On

3.8 Encapsulation order overview

Commands may be encapsulated for a number of reasons.

- Multiple commands may be grouped by Multi Command encapsulation
- Multi Channel End Points may be addressed via Multi Channel encapsulation
- Transmission reliability may be improved via CRC-16 encapsulation
- Data may be protected by security encapsulation
- Long commands may be forwarded via Transport Service encapsulation

Encapsulation MUST be applied according to the following list, where 1 indicates the first encapsulation to be applied:

Table 10, Order of encapsulation

Order					
5					Transport Service
4	CRC-16	Security v1	Transport Service	Security v2	Security v2
2	Multi Channel				
1	Multi Command				
0	Payload				

The order of encapsulations MUST NOT be reordered.

The Security and Transport Service encapsulation formats provide the same level of integrity protection as the CRC-16 encapsulation format. Thus, CRC-16 encapsulation SHOULD NOT be used together with Security and Transport Service encapsulation.

The following sections outline examples of encapsulation.

3.8.1 Multi Command encapsulation

A device MAY implement the Multi Command Command Class.

The figure below shows one example of a Multi Command encapsulated command.

Multi Command Encapsulation Header	Encapsulated Command

Figure 6, Multi Command encapsulation

3.8.2 Multi Channel encapsulation

A device MAY implement the Multi Channel Command Class.

The figure below shows one example of a Multi Channel encapsulated command.

Multi Channel Encapsulation Header	Multi Command Encapsulation Header	Encapsulated Command

Figure 7, Multi Command in Multi Channel encapsulation

3.8.3 CRC-16 encapsulation

A device MAY implement the CRC-16 Command Class.

The figure below shows one example of a CRC-16 encapsulated command.

CRC-16 Encapsulation Header	Multi Channel Encapsulation Header	Multi Command Encapsulation Header	Encapsulated Command

Figure 8, Multi Command in Multi Channel in CRC-16 encapsulation

3.8.4 Security encapsulation

A device MAY implement the Security Command Class.

The figure below shows one example of a Security encapsulated command.

Security Encapsulation Header	Multi Channel Encapsulation Header	Multi Command Encapsulation Header	Encapsulated Command

Figure 9, Multi Command in Multi Channel in Security encapsulation

3.8.5 Transport Service encapsulation

A device MAY implement the Transport Service Command Class.

The figure below shows one example of a Transport Service encapsulated command.

Transport Service Encapsulation Header	Security v2 Encapsulation Header	Multi Channel Encapsulation Header	Multi Command Encapsulation Header	Encapsulated Command

Figure 10, Multi Command in Multi Channel in Security v2 in Transport Service encapsulation

4 COMMAND CLASS DEFINITIONS

The following subchapters contain definitions of Command Classes.

4.1 Terminology for Alarm and Notification Command Classes

Sensors may be designed for two purposes: A multilevel sensor advertises a measurement. A notification sensor advertises a specific event. Legacy notification sensors may implement the Alarm Command Class.

A notification sensor may be designed to operate in push or pull mode.

A push mode notification sensor sends unsolicited notification reports. The transmission of unsolicited notification reports may be disabled or enabled via the Notification Set message. Even if enabled, unsolicited notification reports can only be transmitted if an association destination is defined.
Push functionality such as event reporting via Notification CC and relay control via Basic CC are advertised via the Association Group Information (AGI) CC.

A pull mode notification sensor collects notification reports in a list of pending notification reports. A notification report is returned in response to a notification get message. Multiple notification reports may be retrieved from the list by repeated notification get messages.
A notification report may be returned persistently until it is actively cleared via the Notification Set message.

4.2 Alarm Command Class, version 1

The Alarm Command Class allows applications to report alarm or service conditions. Since these parameters are not standardized across devices the alarms/service parameters MUST be described in the user manual (or an installer manual).

4.2.1 Interoperability considerations

.

Alarm Command Class has been superseded by the Notification Command Class. Refer to most recent version of Notification Command Class.

4.2.2 Alarm Get Command

The Alarm Get Command is used to get the value of an alarm.

The Alarm Report Command MUST be returned in response to this command if the alarm type is supported.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ALARM							
Command = ALARM_GET							
Alarm Type							

Alarm Type (8 bits)

The Alarm Type field specifies which alarm is being requested. The alarm types are specific for each application.

4.2.3 Alarm Report Command

The Alarm Report Command used to report the type and level of an alarm.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ALARM							
Command = ALARM_REPORT							
Alarm Type							
Alarm Level							

Alarm Type (8 bits)

Refer to explanation under the Alarm Get Command.

Alarm Level (8 bits)

The alarm level is application specific.

4.3 Alarm Command Class, version 2

The Alarm Command Class is intended for Z-Wave enabled devices capable of reporting alarm reports.

Version 2 of the Alarm Command Class is improved with the following functionalities:

- Alarm Types defined by the Z-Wave Alliance
- Interview process of supported Alarm Types

The commands not mentioned here will remain the same as specified for Alarm Command Class, version 1.

4.3.1 Interoperability considerations

The interoperability considerations from version 1 also apply for this version. Refer to 4.2.1.

4.3.2 Alarm Set Command

The Alarm Set Command is used to set the activity of the Z-Wave Alarm Type and Status.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ALARM							
Command = ALARM_SET							
Z-Wave Alarm Type							
Z-Wave Alarm Status							

Z-Wave Alarm Type (8 bits)

See Alarm Report Command.

Z-Wave Alarm Status (8 bits)

This field is used to set the state of the Alarm Type. The value 0x00 will deactivate the alarm and 0xFF will activate the alarm i.e. unsolicited Alarm Report Command will be transmitted to the node(s) defined in the Node field(s) when triggered by an event. Any other value is reserved for future use.

Note: The factory default state MUST be described in the product manual. All Z-Wave enabled devices MUST be able to operate based on factory default settings i.e. an end-user MUST NOT be forced to set-up the states of the device in order to operate. The factory default state of the Z-Wave Alarm Status SHOULD be enabled.

Products that do not allow deactivation of a specific Alarm Type, MUST respond to a Alarm Configuration Set deactivating the Alarm Type in question by returning an Application Rejected Request Command of the Application Status Command Class.

4.3.3 Alarm Get Command

The Alarm Get Command is used to request the alarm state for a specific alarm type announced as supported through the Alarm Type Supported Report Command.

The Alarm Report Command MUST be returned in response to this command if the alarm type is supported.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ALARM							
Command = ALARM_GET							
Alarm Type							
Z-Wave Alarm Type							

Alarm Type (8 bits)

This field refers to the Alarm Type of Alarm Command Class (Version 1) i.e. the application specific Alarm Type which is not defined by the Z-Wave Alliance. If the 'V1 Alarm' field is set to '0' as reported via the Alarm Type Supported Report Command, this field MUST be set to '0' when requesting the report.

Z-Wave Alarm Type (8 bits)

The Z-Wave Alarm Type field MUST contain the Alarm Type identifier described in Alarm Report Command. This parameter refers to the Alarm Types defined by the Z-Wave Alliance.

A device receiving an Alarm Get Command containing a non-supported Z-Wave Alarm Type MUST ignore the command. A controlling node SHOULD interview the device for supported Alarm Types by means of Alarm Type Supported Get Command prior to Alarm Get.

4.3.4 Alarm Report Command

The Alarm Report Command is used by the application to report the alarm state.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ALARM							
Command = ALARM_REPORT							
Alarm Type							
Alarm Level							
Zensor Net Source Node ID							
Z-Wave Alarm Status							
Z-Wave Alarm Type							
Z-Wave Alarm Event							
Number of Event Parameters							
Event Parameter 1							
...							
Event Parameter N							

Alarm Type (8 bits)

Same as for Alarm Command Class (Version 1).

Alarm Level (8 bits)

Same as for Alarm Command Class (Version 1).

Zensor Net Source Node ID (8 bits)

Specify the Zensor Net Source Node ID, which detected the alarm condition. In Zensor Net it is not possible to determine the Source Node ID due to the broadcast forwarded frame is without this information on protocol level. If the device is not based on Zensor Net this field MUST be set to ‘0’.

Z-Wave Alarm Status (8 bits)

See Alarm Set Command.

Number of Event Parameters (8 bits)

Indicates the Number of Event Parameters fields used in bytes.

Z-Wave Alarm Type (8 bits), Z-Wave Alarm Event (8 bits) and Event Parameters (N Byte)

Table 11 specifies the Alarm Types and its subordinate parameters defined by the Z-Wave Alliance. The fields that do not contain any definition of the Z-Wave Alarm Type MUST be set to '0' in the Alarm Report Command.

The device MUST advertise support of the Command Class which is included for the specific Alarm Type. Example: for Smoke Alarm, Smoke Detected the Node Naming and Location Command Class MUST be advertised as supported in the Node Information Frame.

Table 11, Alarm Report :: Z-Wave Alarm Type & Event

Z-Wave Alarm Type (8 bits)		Z-Wave Alarm Event (8 bits)		Event Parameters (N Byte)
Reserved	0x00	Reserved	0x00	
Smoke Alarm	0x01	Reserved	0x00	
		Smoke detected	0x01	Node Location Report (Node Naming and Location Command Class, version 1)
		Smoke detected, Unknown Location	0x02	
		Reserved	0x03-0xFD	
		Unknown Event	0xFE	
		Reserved	0xFF	
CO Alarm	0x02	Reserved	0x00	
		Carbon monoxide detected	0x01	Node Location Report (Node Naming and Location Command Class, version 1)
		Carbon monoxide detected, Unknown Location	0x02	
		Reserved	0x03-0xFD	
		Unknown Event	0xFE	
		Reserved	0xFF	
CO_2 Alarm	0x03	Reserved	0x00	
		Carbon dioxide detected	0x01	Node Location Report (Node Naming and Location Command Class, version 1)
		Carbon dioxide detected, Unknown Location	0x02	
		Reserved	0x03-0xFD	

Z-Wave Alarm Type (8 bits)		Z-Wave Alarm Event (8 bits)		Event Parameters (N Byte)
		Unknown Event	0xFE	
		Reserved	0xFF	
Heat Alarm	0x04	Reserved	0x00	
		Overheat detected	0x01	Node Location Report (Node Naming and Location Command Class, version 1)
		Overheat detected, Unknown Location	0x02	
		Rapid Temperature Rise	0x03	Node Location Report (Node Naming and Location Command Class, version 1)
		Rapid Temperature Rise, Unknown Location	0x04	
		Underheat detected	0x05	Node Location Report (Node Naming and Location Command Class, version 1)
		Underheat detected, Unknown Location	0x06	
		Reserved	0x07-0xFD	
		Unknown Event	0xFE	
		Reserved	0xFF	
Water Alarm	0x05	Reserved	0x00	
		Water Leak detected	0x01	Node Location Report (Node Naming and Location Command Class, version 1)
		Water Leak detected, Unknown Location	0x02	
		Water Level Dropped	0x03	Node Location Report (Node Naming and Location Command Class, version 1)
		Water Level Dropped, Unknown Location	0x04	
		Reserved	0x05-0xFD	
		Unknown Event	0xFE	
		Reserved	0xFF	
Access Control	0x06	Reserved	0x00	

Z-Wave Alarm Type (8 bits)		Z-Wave Alarm Event (8 bits)		Event Parameters (N Byte)
Alarm		Manual Lock Operation	0x01	
		Manual Unlock Operation	0x02	
		RF Lock Operation	0x03	
		RF Unlock Operation	0x04	
		Keypad Lock Operation	0x05	User Identifier of User Code Report (User Code Command Class)
		Keypad Unlock Operation	0x06	User Identifier of User Code Report (User Code Command Class)
		Reserved	0x07-0xFD	
		Unknown Event	0xFE	
		Reserved	0xFF	
Burglar Alarm	0x07	Reserved	0x00	
		Intrusion	0x01	Node Location Report (Node Naming and Location Command Class, version 1)
		Intrusion, Unknown Location	0x02	
		Tampering, product covering removed	0x03	
		Tampering, Invalid Code	0x04	
		Glass Breakage	0x05	Node Location Report (Node Naming and Location Command Class, version 1)
		Glass Breakage, Unknown Location	0x06	
		Reserved	0x07-0xFD	
		Unknown Event	0xFE	
		Reserved	0xFF	
Power Management Alarm	0x08	Reserved	0x00	
		Power has been applied	0x01	
		AC mains disconnected	0x02	
		AC mains re-connected	0x03	

Z-Wave Alarm Type (8 bits)		Z-Wave Alarm Event (8 bits)		Event Parameters (N Byte)
		Surge Detection	0x04	
		Voltage Drop/Drift	0x05	
		Reserved	0x06-0xFD	
		Unknown Event	0xFE	
		Reserved	0xFF	
System Alarm	0x09	Reserved	0x00	
		System hardware failure	0x01	
		System software failure	0x02	
		Reserved	0x03-0xFD	
		Unknown Event	0xFE	
		Reserved	0xFF	
Emergency Alarm	0x0A	Reserved	0x00	
		Contact Police	0x01	
		Contact Fire Service	0x02	
		Contact Medical Service	0x03	
		Reserved	0x04-0xFD	
		Unknown Event	0xFE	
		Reserved	0xFF	
Alarm Clock	0x0B	Reserved	0x00	
		Wake Up	0x01	
Reserved	0x0C-0xFE			
Return first Alarm on supported list	0xFF			

Alarm Type = 0xFF is used by the Alarm Get Command to retrieve the first alarm detection from the list of pending alarms.

Example: a device supports the Z-Wave Alarm Types: Smoke, CO_2 and Heat. The Heat Alarm is active e.g. overheat has been detected. When the device receives Alarm Get, Z-Wave Alarm Type (0xFF), it must return Alarm Report, Z-Wave Alarm Type (0x04), Z-Wave Alarm Event (0x01/0x02) and the accompanied parameters.

4.3.5 Alarm Type Supported Get Command

The Alarm Type Supported Get Command is used to request the supported alarm types.

The Alarm Type Supported Report Command MUST be returned in response to an Alarm Type Supported Get command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_ALARM</td></tr>
<tr><td colspan="8">Command = ALARM_TYPE_SUPPORTED_GET</td></tr>
</table>

4.3.6 Alarm Type Supported Report Command

The Alarm Type Supported Report Command used to advertise the supported alarm types in the application.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_ALARM</td></tr>
<tr><td colspan="8">Command = ALARM_TYPE_SUPPORTED_REPORT</td></tr>
<tr><td>V1 Alarm</td><td colspan="2">Reserved</td><td colspan="5">Number of Bit Masks</td></tr>
<tr><td colspan="8">Bit Mask 1</td></tr>
<tr><td colspan="8">...</td></tr>
<tr><td colspan="8">Bit Mask N</td></tr>
</table>

V1 Alarm (1 bit)

0 = the device implements only Notification CC V2 (or newer) Notification Type(s).

1 = the device implements Notification CC V2 Notification Types as well as proprietary Alarm CC V1 Alarm Types and Alarm Levels.

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Number of Bit Masks (5 bits)

Indicates the Number of Bit Masks fields used in bytes.

Bit Mask (N Bytes)

The Bit Mask fields describe the supported Z-Wave Alarm Types by the device.

- Bit 0 in Bit Mask 1 is not allocated to any Z-Wave Alarm Type and MUST therefore be set to zero.
- Bit 1 in Bit Mask 1 indicates if Z-Wave Alarm Type = 1 (Smoke) is supported.
- Bit 2 in Bit Mask 1 indicates if Z-Wave Alarm Type = 2 (CO) is supported.
- Bit 3 in Bit Mask 1 indicates if Z-Wave Alarm Type = 3 (CO_2) is supported
- …

If the Z-Wave Alarm Type is supported the corresponding bit MUST be set to 1. If the Z-Wave Alarm Type is not supported the corresponding bit MUST be set to 0.

Z-Wave Alarm Type = 0xFF (Return first Alarm on supported list) MUST NOT be advertised in the Bit Masks.

The number of Bit Mask fields MUST match the value advertised in the Number of Bit Masks field.

Note that the mapping of bit 1 to Alarm Type =1 differs from the support mapping used by the Multilevel Sensor Command Class. The Multilevel Sensor Command Class maps bit 0 to Sensor Type = 1.

4.4 Alarm Sensor Command Class, version 1 [DEPRECATED]

THIS COMMAND CLASS HAS BEEN DEPRECATED

A device MAY implement this command class, but it is RECOMMENDED that new implementations use the Notification Command Class.
If implementing this Command Class, it is RECOMMENDED that the Notification Command Class is also implemented.

The Alarm Sensor Command Class is used to realize Sensor Alarms.

4.4.1 Alarm Sensor Get Command

The Alarm Sensor Get Command is used to request the status of a sensor.

The Alarm Sensor Report Command MUST be returned in response to this command if the sensor type is supported.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing.
The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SENSOR_ALARM							
Command = SENSOR_ALARM_GET							
Sensor Type							

Sensor Type (8 bits)

Sensor Type specifies what type of sensor this command originates from. Refer to the table below with respect to sensors types defined by the Z-Wave Alliance. The sensor type value 0xFF returns the first found supported sensor type in the bit mask (starting from bit 0 in Bit Mask 1) by the Alarm Sensor Supported Report.

Sensor Type	Value
General Purpose Alarm	0x00
Smoke Alarm	0x01
CO Alarm	0x02
CO_2 Alarm	0x03
Heat Alarm	0x04
Water Leak Alarm	0x05
Return first Alarm on supported list	0xFF

4.4.2 Alarm Sensor Report Command

This command is used to advertise the alarm state.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SENSOR_ALARM							
Command = SENSOR_ALARM_REPORT							
Source Node ID							
Sensor Type							
Sensor State							
Seconds 1 (MSB)							
Seconds 2 (LSB)							

Source Node ID (8 bits)

Specify the source node ID, which detected the alarm condition. In a Zensor Net is it not possible to determine the source node ID because the frame is broadcast forwarded without this information on protocol level.

Sensor Type (8 bits)

See description under Alarm Sensor Get. The Sensor Type equal to 0xFF cannot be return by the report.

Sensor State (8 bits)

The Sensor State parameter returns the current alarm state. The value 0x00 indicates no alarm and 0xFF indicates alarm. Furthermore it can return values from 0x01 to 0x64 to indicate severity of the alarm in percentage.

The values 101…254 (0x65…0xFE) are reserved and MUST be ignored by receiving devices.

Seconds (16 bits)

The field Seconds indicates time the remote alarm must be active since last received report. The value 0x0000 indicates that the time field MUST be ignored.

4.4.3 Alarm Sensor Supported Get Command

The Alarm Sensor Supported Get Command is used to request the supported sensor types from the device.

The Alarm Sensor Supported Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SENSOR_ALARM							
Command = SENSOR_ALARM_SUPPORTED_GET							

4.4.4 Alarm Sensor Supported Report Command

The Alarm Sensor Supported Report Command used to report the supported sensor types from the device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SENSOR_ALARM							
Command = SENSOR_ALARM_SUPPORTED_REPORT							
Number of Bit Masks							
Bit Mask 1							
…							
Bit Mask N							

Number of Bit Masks (8 bits)

Indicates the Number of Bit Masks fields used in bytes.

Bit Mask (N Bytes)

The Bit Mask fields describe the supported sensor types by the device.

- Bit 0 in Bit Mask 1 indicates if Sensor Type = 0 (General Alarm) is supported.
- Bit 1 in Bit Mask 1 indicates if Sensor Type = 1 (Smoke Alarm) is supported.
- …

The sensor type is supported if the bit is 1 and the opposite if 0. It is only necessary to send the Bit Mask fields from 1 and up to the one indicating the last supported sensor type. The number of Bit Mask fields transmitted MUST be determined from the length field in the frame.

Note that the mapping of bit 1 to Sensor Type =1 differs from the support mapping used by the Multilevel Sensor Command Class. The Multilevel Sensor Command Class maps bit 0 to Sensor Type = 1.

4.5 Alarm Silence Command Class, version 1

The Alarm Silence Command Class may be used to temporarily disable the sounding of the alarm but still keep the alarm operating.

4.5.1 Alarm Silence Set Command

The Alarm Silence Set Command may be used to remotely silence the sensor alarm.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SILENCE_ALARM							
Command = SENSOR_ALARM_SET							
Mode							
Seconds 1 (MSB)							
Seconds 2 (LSB)							
Number of Bit Masks							
Bit Mask 1							
…							
Bit Mask N							

Mode (8 bits)

Mode specifies the different options to silence sensor alarms. Modes are defined by the Z-Wave Alliance.

Mode	Value
Disable sounding of all sensor alarms independent of bit mask	0x00
Disable sounding of all sensor alarms independent of bit mask which have received the alarm via the Sensor Alarm Report command	0x01
Disable sounding of all sensor alarms according to bit mask	0x02
Disable sounding of all sensor alarms according to bit mask which have received the alarm via the Alarm Sensor Report Command	0x03

Seconds (16 bits)

The field Seconds indicates the duration sounding of the alarm must be disable but still keep the alarm operating. If silence is engaged, the alarm will come back on when the duration expires unless the originating sensor clears the alarm. The value 0x0000 indicates that the time field MUST be ignored.

Number of Bit Masks (8 bits)

Indicates the Number of Bit Masks fields used in bytes.

Bit Mask (N Bytes)

The Bit Mask fields describe the sensor types to disable sounding from.

- Bit 0 in Bit Mask 1 indicates if Sensor Type = 0 (General Alarm) is disabled.
- Bit 1 in Bit Mask 1 indicates if Sensor Type = 1 (Smoke Alarm)) is disabled.
- …

If the sensor type is disabled the bit MUST be set to 1. If the sensor type is not disabled the bit MUST be set to 0. It is only necessary to send the Bit Mask fields from 1 and up to the one indicating the last sensor type to be disabled.

4.6 All Switch Command Class, version 1

The All Switch Command Class is used to switch all devices on or off. Devices may be excluded/included from the all on/all off functionality. The application determines which devices there are included in the all on/all off functionality as default.

4.6.1 All Switch Set Command

The All Switch Set Command used to instruct a device if it is included or excluded from the all on/all off functionality.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_ALL							
Command = SWITCH_ALL_SET							
Mode							

Mode (8 bits)

The mode field used to set the all on/all off functionality of the device.

Mode	Description
0x00	Indicate that the switch is excluded from the all on/all off functionality.
0x01	Indicate that the switch is excluded from the all on functionality but not all off.
0x02	Indicate that the switch is excluded from the all off functionality but not all on.
0xFF	Indicates that the switch is included in the all on/all off functionality.

4.6.2 All Switch Get Command

The All Switch Get Command may be used to ask a device if it is included or excluded from the all on/all off functionality.

The All Switch Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_ALL							
Command = SWITCH_ALL_GET							

4.6.3 All Switch Report Command

The All Switch Report Command is used to report if the device is included or excluded from the all on/all off functionality.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_ALL							
Command = SWITCH_ALL_REPORT							
Mode							

Mode (8 bits)

Refer to explanation under the All Switch Command.

4.6.4 All Switch On Command

The All Switch On Command is used to inform a switch that it SHOULD be turned on. A receiving device MUST NOT react to this command if the actual operation has been prohibited via the Switch All Set command. Like the Basic Set On command, this command MUST cause the device to restore the most recent (non-zero) level.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_ALL							
Command = SWITCH_ALL_ON							

4.6.5 All Switch Off Command

The All Switch Off Command is used to inform a switch that it SHOULD be turned off. A receiving device MUST NOT react to this command if the actual operation has been prohibited via the Switch All Set command.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_ALL							
Command = SWITCH_ALL_OFF							

4.7 Anti-theft Command Class, version 1 [OBSOLETED]

<u>THIS COMMAND CLASS VERSION HAS BEEN OBSOLETED</u>

New implementations MUST use the Anti-theft Command Class Version 2.

The Anti-theft Command Class is used to disable a subset of supported/controlled command classes in a device if the device is being excluded and re-included into a Z-Wave network again. This command class is typically used when installing a Z-Wave device in a public location such as a hotel room or conference center. The command class allows the user to lock the device to the actual Z-Wave network and to render it useless if it is removed from the local network without being unlocked. Another application would be to protect service provider owned products from leaving the service providers network before they are paid for.

Version 2 limits the Magic Code and Anti-theft Hint maximum bytes to 10. This makes it possible to embed Anti-theft Command Class Version 2 in one Security Command Class and thereby avoid splitting it.

4.8 Anti-theft Command Class, version 2

The Anti-theft Command Class MUST NOT be supported unless the Device Class or Device Type implemented by the device explicitly allows for support of the Anti-theft Command Class.

The Anti-theft Command Class is intended for devices operating in public locations such as hotel rooms or a conference center. The purpose of the Anti-theft Command Class is to render a device useless if it is removed from its actual network without being unlocked by the owner or a service provider.

The Anti-theft Command Class is used to disable all command classes related to the actual application functionality of a device if is it excluded and later included in another network. It does not matter if the device implements a single resource addressed via the Root Device or a collection of resources addressed via individual Multi Channel End Points Enabling anti-theft protection in a device MUST NOT change any operation with respect to supported/controlled command classes as long as the device stays in the actual network.

If a locked device is excluded, it MUST enter the protected state. When in the protected state, the node information frame (NIF) MUST NOT advertise support of the protected command classes.The NIF MUST however continue advertising support of the Anti-theft Command Class and all other non-application specific command classes; just as when the device operates in its home network.
The device MUST NOT respond to application commands while in the protected state. A device in protected state MUST NOT leave its protected state if it is re-included into its home network. Another Anti-Theft Set command MUST be used to either disable locking or to clear the protected state.

The following non-device specific command classes must not be protected by the anti-theft functionality (i.e. will always be available in the device regardless of protection state):

- Manufacturer Specific Command Class
- Version Command Class
- Anti-theft Command Class
- Security Command Class (optional)
  Security encapsulated command classes are allowed to be protected. In that case they must be removed from the *Security Commands Supported Report Command* when in the protected state in a foreign network.

The protection state may be updated by sending the set command with the correct magic code to the device at any time and in any network. When the protection state is updated the device must return to normal operation, regardless of whether the update is to disable or re-enable protection. However, it is not possible to update protection state when device is excluded because it must be able to receive a command.

The Anti-theft protection state must be preserved in the following situations:

- Exclusion of a network
- Reset to factory default
- OTA update of firmware

If secure device, supports Anti-theft Command Class, Security Command Class needs to be supported regardless of anti-theft protection state. A security enabled device MUST be able to join any secure Z-Wave network regardless of its anti-theft protection state.

It is RECOMMENDED for a device that supports anti-theft protection to have physical mark that indicates that this device is capable of being locked. It is further RECOMMENDED that the device is capable of signaling via a LED or other means if the device device refuses inclusion in a network because the device is locked to another network. Finally, the user guide and installation manual MUST advertise support of anti-theft protection.

4.8.1 Anti-theft Set Command

The Anti-theft Set Command is used to enable/disable anti-theft protection in a device already included into a Z-Wave Network by sending a magic code to device in question. The same magic code MUST be used to disable anti-theft protection again. A new magic code may be used the next time to enable anti-theft protection in the device, but only if protection is disabled at the time. A new device MUST have anti-theft protection disabled. Enabling anti-theft protection in an already-enabled device restores it to normal operation if it is in reduced functionality mode, but otherwise has no effect.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_ANTITHEFT</td></tr>
<tr><td colspan="8">Command = ANTITHEFT_SET</td></tr>
<tr><td>Enable</td><td colspan="7">Number of Magic Code bytes</td></tr>
<tr><td colspan="8">Magic Code 1</td></tr>
<tr><td colspan="8">…</td></tr>
<tr><td colspan="8">Magic Code N</td></tr>
<tr><td colspan="8">Manufacturer ID MSB</td></tr>
<tr><td colspan="8">Manufacturer ID LSB</td></tr>
<tr><td colspan="8">Anti-theft Hint Number Bytes</td></tr>
<tr><td colspan="8">Anti-theft Hint Byte 1</td></tr>
<tr><td colspan="8">…</td></tr>
<tr><td colspan="8">Anti-theft Hint Byte N</td></tr>
</table>

Enable (1 bit)

The value MAY be 0 (Attempt to disable anti-theft protection in device) or 1 (Attempt to enable or re-enable anti-theft protection in device). It is not necessary to first disable an exclude device having protection enable; it can be re-enabled directly in a new network by using correct magic code again.

Number of Magic Code bytes (7 bits)

Indicates the Number of Magic Code fields N used in bytes. Maximum number of Magic Code fields MUST NOT exceed 10 bytes.

Magic Code (N bytes)

The Magic Code fields hold the code to enable/disable the Z-Wave device in question.

Manufacturer ID (2 bytes)

The Manufacturer ID of the company's product having a central role in the application requiring anti-theft protection enabled. Device should report 0xFFFF if anti-theft protection is disabled.

Manufacturer identifiers can be found in [11].

Number of Anti-theft Hint bytes (8 bits)

Indicates the Number of Anti-theft Hint fields N used in bytes. If length is 0 no Hint provided. Maximum number of Anti-theft Hint fields MUST NOT exceed 10 bytes.

Anti-theft Hint Byte (N bytes)

Anti-theft Hint Bytes that may be used as an identifier or key value for retrieving the Magic Code. The exact format and meaning of these Bytes is specific to the product or service that enabled anti-theft protection on the device, as identified by the Manufacturer ID above. If it is necessary to render the Hint Bytes for display, each byte should be interpreted as an unsigned integer value and represented in hexadecimal.

4.8.2 Anti-theft Get Command

The Anti-theft Get Command is used to get an Anti-theft Report Command showing status of the Z-Wave device in question.

The Anti-theft Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ANTITHEFT							
Command = ANTITHEFT_GET							

4.8.3 Anti-theft Report Command

The Anti-theft Report Command is used to report status of the Z-Wave device in question.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ANTITHEFT							
Command = ANTITHEFT_REPORT							
Anti-theft Protection Status							
Manufacturer ID MSB							
Manufacturer ID LSB							
Anti-theft Hint Number Bytes							
Anti-theft Hint Byte 1							
…							
Anti-theft Hint Byte N							

Anti-theft Protection Status (8 bits)

Anti-theft Protection Status specifies the actual status of Z-Wave device in question. Refer to the table below with respect to defined status values.

Anti-theft Protection Status	Value
Reserved.	0x00
Anti-Theft Protection is currently DISABLED, and the Z-Wave Device is fully functional.	0x01
Anti-Theft Protection is currently ENABLED, and the Z-Wave Device is fully functional.	0x02
Anti-Theft Protection is currently ENABLED, and the Z-Wave Device is NOT fully functional (i.e., the Device was excluded from a network without disabling protection, and an ANTITHEFT_SET command with the correct Magic Code has not yet been received in the current network).	0x03
Reserved	0x04-0xFF

Manufacturer ID (2 bytes)

The Manufacturer ID of the company’s product having a central role in the application requiring anti-theft protection enabled. Device should report 0xFFFF if anti-theft protection is disabled.

Number of Anti-theft Hint bytes (8 bits)

Indicates the Number of Anti-theft Hint fields N used in bytes. If length is 0 no Hint provided. Maximum number of Anti-theft Hint fields MUST NOT exceed 10 bytes.

Anti-theft Hint Byte (N bytes)

Anti-theft Hint Bytes. See the Anti-Theft Set Command for more details.

4.8.4 Examples

Following examples are for reference only.

4.8.4.1 Example of a non-secure Thermostat

Below is shown an example of the Node Information Frame (NIF) content for a non-secure thermostat. The first NIF shows a device having anti-theft protection disabled. The device may never been included into a network or reside in a network or excluded from a network:

Disabled anti-theft protection
Manufacturer Specific Command Class
Version Command Class
Anti-theft Command Class
Thermostat Operating State Command Class
Thermostat Mode Command Class
Association Command Class
Battery Command Class

The second NIF shows a device having anti-theft protection enabled. The device may be excluded from network in which it was originally anti-theft protection enabled or re-included into a network. This also applies in case device is re-included into the network, which device originally was anti-theft protection enabled:

Enabled anti-theft protection
Manufacturer Specific Command Class
Version Command Class
Anti-theft Command Class

4.8.4.2 Example of a security enabled Thermostat

Below is shown an example of the Node Information Frame (NIF) content for a security enabled thermostat. The first NIF shows a device having anti-theft protection disabled. The device may never been included into a network or reside in a network or excluded from a network:

Disabled anti-theft protection
Manufacturer Specific Command Class
Version Command Class
Security Command Class

Finally, the *Security Commands Supported Report Command* reports support of the following command classes:

Anti-theft Command Class
Thermostat Operating State Command Class
Thermostat Mode Command Class
Association Command Class
Battery Command Class

The second NIF shows a device having anti-theft protection enabled. The device may be excluded from network in which it was originally anti-theft protection enabled or re-included into a network. This also applies in case device is re-included into the network, which device originally was anti-theft protection enabled. The NIF is unchanged because all application oriented command classes are security encapsulated except the default command classes:

Enabled anti-theft protection
Manufacturer Specific Command Class
Version Command Class
Security Command Class

Finally, the *Security Commands Supported Report Command* reports support of at least the Anti-theft Command Class to be able to disable anti-theft protection:

Anti-theft Command Class

The Anti-theft Command Class is supported securely making malicious attempts to enable anti-theft protection very difficult.

4.8.4.3 Example of a security enabled Door Lock

Below is shown an example of the Node Information Frame (NIF) content for a security enabled door lock. The first NIF shows a device having anti-theft protection disabled. The device may never been included into a network or reside in a network or excluded from a network:

Disabled anti-theft protection
Security Command Class
Anti-theft Command Class
Version Command Class
Manufacturer Specific Command Class

The *Security Commands Supported Report Command* reports support of the following command classes:

Door Lock Command Class
User Code Command Class
Battery Command Class
Association Command Class

The second NIF shows a device having anti-theft protection enabled. The device may be excluded from network in which it was originally anti-theft protection enabled or re-included into a network. This also applies in case device is re-included into the network, which device originally was anti-theft protection enabled. The NIF is unchanged because all application oriented command classes are security encapsulated except the default command classes:

Enabled anti-theft protection
Security Command Class
Anti-theft Command Class
Version Command Class
Manufacturer Specific Command Class

Finally, no command classes are returned in the *Security Commands Supported Report Command* due to enabled anti-theft protection.

Even though the device is security enabled then the Anti-theft Command Class is supported non-securely making malicious attempts to enable anti-theft protection a lot easier.

4.9 Application Capability Command Class, version 1

The Application Capability Command Class comprises commands for handling issues related to dynamic support for command classes.
Examples include nodes, which only support certain command classes when included securely, and controllers that only support command classes for primary controllers when they are actually operating as primary controllers.

The intention of the Application Capability Command Class is to provide tools for handling exceptions. Controlling nodes SHOULD maintain a local representation of supported command classes by destination nodes rather than relying on destination nodes returning Application Capability messages.

Destination nodes capable of returning Application Capability messages MUST list Application Capability Command Class as supported only.

4.9.1 Not Supported Command Class Command

The Not Supported Command Class Command is used by a node to indicate to a requesting node that the requested command class is not supported. The offending command and command class are encapsulated.

A node SHOULD use this command to indicate to other nodes that a command is not supported.

A node receiving this command SHOULD use the information to clean up internal states by requesting up-to-date information on command class support from the sending node using Node Information frame. This includes adjusting user interface details accordingly.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_APPLICATION_CAPABILITY							
Command = COMMAND_COMMAND_CLASS_NOT_SUPPORTED							
Dyna-mic	*Reserved*						
Offending Command Class (0x20 – 0xEE)							
Offending Command							

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_APPLICATION_CAPABILITY							
Command = COMMAND_COMMAND_CLASS_NOT_SUPPORTED							
Dyna-mic	*Reserved*						
Offending Command Class MSB (0xF1 – 0xFF)							
Offending Command Class LSB (0x00 – 0xFF)							
Offending Command							

Payload data following the offending command is omitted.

Dynamic (1 bit)

If the “Dynamic” flag is set (‘1’), the sending node has been designed to support this command class but the current operational conditions prevent the node from supporting this command class.
One example is the AddNode command used for network management. This command is only supported when a controller is operating as primary controller.

Value	Meaning
‘0’	Command not supported (Permanently)
‘1’	Dynamic support. (Currently no support)

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Offending Command Class (1 or 2 bytes)

This field indicates the offending command class received by the destination node. The size of the command class field depends on the value of the first byte.

Offending Command (1 byte)

This field indicates the offending command received by the destination node.

4.10 Application Status Command Class, version 1

This command class contains commands that are not directly related to a specific functionality in the application, but are useful for maintaining an optimal Z-Wave system.

4.10.1 Application Busy Command

The Application Busy Command used to instruct a node that the node that it is trying to communicate with is busy and is unable to service the request right now.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_APPLICATION_STATUS							
Command = APPLICATION_BUSY							
Status							
Wait Time							

Status (8 bits)

The status field can have the following values

Status	Description
0	Try again later
1	Try again in Wait Time seconds
2	Request queued, executed later

Wait Time (8 bits)

The wait time field indicates the number of seconds a node should wait before retrying the request.

4.10.2 Application Rejected Request Command

All supported commands are typically executed unconditionally and the only handshake is acknowledgement on the protocol level. Some applications can however be in a state where the application rejects to execute a supported command. The Application Rejected Request Command used to instruct a node that the command was rejected by the application in the receiving node.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8" align="center">Command Class = COMMAND_CLASS_APPLICATION_STATUS</td></tr>
<tr><td colspan="8" align="center">Command = APPLICATION_REJECTED_REQUEST</td></tr>
<tr><td colspan="8" align="center">Status</td></tr>
</table>

Status (8 bits)

The status field can have the following values

Status	Description
0	Supported command rejected by the application in the receiving node

4.11 Association Command Class, version 1

The Association Command Class is used to manage associations to NodeID destinations. A NodeID destination may be a simple device or the Root Device of a Multi Channel device.

An association group sends an unsolicited command to the configured destinations when triggered by an event. The parameters of the command may be dynamic, e.g. the temperature of a sensor reading or the light level for a dimmer.

4.11.1 Compatibility considerations

The Association Group Information (AGI) Command Class SHOULD be supported to enable automated discovery of association group properties.

It is RECOMMENDED that a device which implements the Association Command Class also implements the Multi Channel Association Command Class for compatibility with End Point destinations. For instance, a wall switch may be configured to control one specific outlet of a power strip if the wall switch supports Multi Channel Association.

4.11.2 Z-Wave Plus considerations

The Z-Wave Plus certification program mandates that association group 1 is reserved for the Lifeline association group. Association group 1 MUST NOT be assigned to any other use than the Lifeline group. The actual Device Type of a given product specifies mandatory commands which the device must be able to send to a lifeline group destination. A manufacturer MAY add additional commands to the Lifeline group.

The Z-Wave Plus certification program mandates support for the Association Group Information (AGI) Command Class if a device supports the Association Command Class.

The Z-Wave Plus certification program recommends that a composite device is designed as a Multi Channel device.

4.11.3 Security considerations

A device which is included securely MUST NOT accept Association commands unless the commands are received via the highest security key assigned to the device.

4.11.4 Association Set Command

This command is used to add destinations to a given association group.

The receiving node SHOULD add the specified NodeID destinations to the specified association group. This command MAY be ignored if the association group is already full.

Routing slaves MUST have return routes assigned to all association destinations.

Unless the association destination is a gateway, a controlling node SHOULD NOT create an association if the association destination node does not support the commands that the actual association group will be sending. The AGI Command Class SHOULD be used to probe the commands that a given association group will be sending.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ASSOCIATION							
Command = ASSOCIATION_SET							
Grouping Identifier							
NodeID 1							
..							
NodeID N							

Grouping Identifier (8 bits)

This field is used to specify the actual association group. Grouping Identifiers MUST be assigned in a consecutive range starting from 1.

A receiving node MUST ignore an unsupported Grouping Identifier.

NodeID (N bytes)

This field specifies a list of NodeIDs that are to be added to the specified association group.

4.11.5 Association Get Command

This command is used to request the current destinations of a given association group.

The Association Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing.
The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ASSOCIATION							
Command = ASSOCIATION_GET							
Grouping Identifier							

Grouping Identifier (8 bits)

This field is used to specify the actual association group. Grouping Identifiers MUST be assigned in a consecutive range starting from 1.

A node that receives an unsupported Grouping Identifier SHOULD return information relating to Grouping Identifier 1.

4.11.6 Association Report Command

This command is used to advertise the current destinations of a given association group.

If the node supports the Multi Channel Association Command Class,

- the node MUST advertise the same Node ID destinations in this report as in the Multi Channel Association Report Command
- the node MUST NOT advertise the NodeID of End Point destinations in this report.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ASSOCIATION							
Command = ASSOCIATION_REPORT							
Grouping Identifier							
Max Nodes Supported							
Reports to Follow							
NodeID 1							
…							
NodeID N							

Grouping Identifier (8 bits)

This field is used to advertise the actual association group. Grouping Identifiers MUST be assigned in a consecutive range starting from 1.

Max Nodes Supported (8 bits)

The maximum number of destinations supported by the advertised association group. Each destination MAY be a NodeID destination or an End Point destination (if the node supports the Multi Channel Association Command Class).

Reports to Follow (8 bits)

The entire list destinations of the advertised association group may be too long for one command. This field MUST advertise how many report frames will follow this report.

NodeID (N bytes)

This field advertises a list of NodeID destinations of the advertised association group.
The list of NodeIDs MUST be empty if there are no NodeID destinations configured for the advertised association group.

4.11.7 Association Remove Command

This command is used to remove destinations from a given association group.

If the node supports the Multi Channel Association Command Class the node MUST NOT remove End Point destinations in response to this command.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ASSOCIATION							
Command = ASSOCIATION_REMOVE							
Grouping Identifier							
NodeID 1							
..							
NodeID N							

Grouping Identifier (8 bits)

This field is used to specify from which association group the specified NodeID destinations should be removed.

Grouping Identifiers MUST be assigned in a consecutive range starting from 1.

This field MUST be interpreted in combination with the NodeID field.

A node that receives an unsupported Grouping Identifier MUST ignore this command.

NodeID (N bytes)

This field specifies a list of NodeID destinations that are to be removed from the specified association group.

The Grouping Identifier and NodeID fields MUST be interpreted as indicated in Table 12.

Table 12, Association Remove, V1 :: Parameter interpretation

Grouping identifier	Number of NodeIDs	
> 0	> 0	Remove specified NodeIDs from the specified association group (MANDATORY V1)
> 0	= 0	Remove all NodeIDs from the specified association group (RECOMMENDED V1)
= 0	≥ 0	(*Reserved* V1)

A sending node MUST NOT send reserved parameter combinations.
A receiving node MUST ignore reserved parameter combinations.

4.11.8 Association Supported Groupings Get Command

This command is used to request the number of association groups that this node supports.

The Association Supported Groupings Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ASSOCIATION							
Command = ASSOCIATION_GROUPINGS_GET							

4.11.9 Association Supported Groupings Report Command

This command is used to advertise the maximum number of association groups implemented by this node.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ASSOCIATION							
Command = ASSOCIATION_GROUPINGS_REPORT							
Supported Groupings							

Supported Groupings (8 bits)

This field is used to advertise the number of association groups that this node supports.

Grouping Identifiers MUST be assigned in a consecutive range starting from 1.

4.12 Association Command Class, version 2

The Association Command Class is used to manage associations to NodeID destinations. A NodeID destination may be a simple device or the Root Device of a Multi Channel device.

The following sections specify commands which were extended or added in version 2.

4.12.1 Compatibility considerations

A device supporting this command class version MUST also support the Association Command Class, version 1.

This version introduces

- New methods for the removal of associations via the Association Remove Command
- New commands
    - Association Specific Group Get Command
    - Association Specific Group Report Command

The considerations of section 4.11.1 also apply to this version.

4.12.2 Z-Wave Plus considerations

The considerations of section 4.11.2 also apply to this version.

4.12.3 Security considerations

The considerations of section 4.11.3 also apply to this version.

4.12.4 Association Remove Command

The Association Remove Command is used to remove NodeID destinations from a given association group.

If the node supports the Multi Channel Association Command Class the node MUST NOT remove End Point destinations in response to this command.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ASSOCIATION							
Command = ASSOCIATION_REMOVE							
Grouping Identifier							
NodeID 1							
..							
NodeID N							

Grouping identifier (8 bits)

This field is used to specify from which association group the specified NodeID destinations are to be removed.

This field MUST be interpreted in combination with the NodeID field.

A node that receives an unsupported Grouping Identifier MUST ignore this command; with the exception of Grouping Identifier 0, which MUST be accepted. Refer to Table 13.

NodeID (N bytes)

This field specifies a list of NodeID destinations that are to be removed. The Grouping Identifier and NodeID fields MUST be interpreted as indicated in Table 13:

Table 13, Association Remove, V2 :: Parameter interpretation

Grouping identifier	Number of NodeIDs	
> 0	> 0	Remove destination NodeIDs from association group (MANDATORY V1, MANDATORY V2)
> 0	= 0	Remove all destination NodeIDs from association group (RECOMMENDED V1, MANDATORY V2)
= 0	> 0	Remove destination NodeIDs from all association groups (*Reserved* V1, MANDATORY V2)
= 0	= 0	Remove all destination NodeIDs from all association groups (*Reserved* V1, MANDATORY V2)

4.12.5 Association Specific Group Get Command

This command allows a portable controller to interactively create associations from a multi-button device to a destination that is out of direct range.

It is OPTIONAL for a device to support this functionality. However, a receiving device MUST always return the Association Specific Group Report Command in response to this command.
If the device does not support this functionality, a Group parameter value of 0 SHOULD be advertised in the Association Specific Group Report Command that is returned in response to this command.

This functionality allows a supporting multi-button device to detect a key press and subsequently advertise the identity of the key. The following sequence of events takes place:

- The user activates a special identification sequence and pushes the button to be identified
- The device issues a Node Information frame (NIF)
- The NIF allows the portable controller to determine the NodeID of the multi-button device
- The portable controller issues an Association Specific Group Get Command to the multi-button device
- The multi-button device returns an Association Specific Group Report Command that advertises the association group that represents the most recently detected button

The Association Group Information (AGI) Command Class provides a centralised alternative for the discovery of available association groups and the capabilities of these groups. A device supporting this functionality SHOULD also support the Association Group Information (AGI) Command Class.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing.
The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ASSOCIATION							
Command = ASSOCIATION_SPECIFIC_GROUP_GET							

Clarification: Previous text revisions of this specification presented conflicting guidelines for the Association Specific Group Get Command when received by non-supporting devices.
The values 0 and 1 were both suggested for the Group field of the Association Specific Group Report Command. It is RECOMMENDED that the value 0 is returned by non-supporting devices.

4.12.6 Association Specific Group Report Command

This command is used to advertise the association group that represents the most recently detected button.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ASSOCIATION							
Command = ASSOCIATION_SPECIFIC_GROUP_REPORT							

Group

Group (8 bits)

This field is used to advertise the association group that represents the most recently detected button.

The value of this field MUST be in the range 0..255.

If a supporting device implements a multi-button device, the Group field MUST advertise an association group which represents the most recently activated button. The actual association group SHOULD be able to control an actuator device, e.g. via the Basic Command Class.

This field SHOULD be set to 0 if the functionality is not supported or if the most recent button event does not map to an association group.

Clarification: Previous text revisions of this specification presented conflicting guidelines for the Association Specific Group Get Command when received by non-supporting devices.
The values 0 and 1 were both suggested for the Group field of the Association Specific Group Report Command. It is RECOMMENDED that the value 0 is returned by non-supporting devices.

4.13 Association Command Configuration Command Class, version 1

The Association Command Configuration Command Class defines the commands necessary for a 2nd node to add and delete commands to Node IDs in a group as defined in the Association Command Class in a 1st node.

The device MUST implement the Association Command Class as 'supported'

The Association Command Class and the Command Configuration Command Class MUST be linked through the following dependencies:

- Nodes added to an association through an Association Set or Association Composite Set MUST be reported in Command Configuration Reports with the Command Class and command identifiers transmitted to the nodes.

- Nodes added to an association through a Command Configuration Set MUST also be reported in an Association Report and Association Composite Report

- All commands associated to a grouping identifier//NodeID pair will be removed as a result of an Association Remove. The related command records will be released

- Command(s) associated to a grouping identifier/NodeID/endpoint pair will be removed as a result of an Association Composite remove. The related command(s) record will be released.

The memory consumption of supporting full command sizes in all combinations of groupings identifiers and Node IDs is very extensive; hence the command class supports a memory flexible implementation.

The Command Class allows a device to support a number of command records. A command record consists of the grouping identifier, the Node ID and the command. The size of the command MAY be restricted by the device through the Max command length field. The command MUST be the complete command needed (I.e. All relevant encapsulations MUST be included in the command).

When no command records are free (all has been used), new commands MUST NOT be allocated to Node IDs before one or more command records have been freed up (through the Association Remove Command)

If the 2nd node runs out of free command records before it has finalized its command configurations, it MUST accept that the application on the device has full control of the remaining Node IDs. Alternatively the 2nd node can abort the command configuration process. In this case it is RECOMMENDED that the 2nd node free up the used command records in the aborted command configuration attempt.

A device can report the maximum number of command records, the number of free command records, and the max command length supported through the Command Records Supported Report.

In order to support sharing knowledge of how a device controls nodes without using extensive memory resources a Configurable Cmd field is supported. When configurable Cmd= 0x0 then a 2nd node can only monitor the commands. It cannot control them.

The V/C field allows a device to decrease the memory utilization to a minimum. When a device reports V/C=0x01, then the Command Configuration Set and Command Configuration Report MUST always use command class identifier and command identifier equal to a Basic Set Command Records Supported Get. This allows the device to only store the value field and thereby save memory resources.

4.13.1 Command Records Supported Get Command

The Command Records Supported Get Command is used to request the number of free command records available (grouping Identifier, Node ID, command), the maximum command records supported in the device and information regarding the maximum command length supported in the device.

The maximum number of groupings the given node supports is available through the Association Command Class.

The Command Records Supported Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing.
The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ASSOCIATION_COMMAND_CONFIGURATION							
Command = COMMAND_RECORDS_SUPPORTED_GET							

4.13.2 Command Records Supported Report Command

The Command Records Supported Report Command used to report information regarding the Command records.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ASSOCIATION_COMMAND_CONFIGURATION							
Command = COMMAND_RECORDS_SUPPORTED_REPORT							
Max command length						V/C	Conf. Cmd
Free Command records 1							
Free Command records 2							
Max Command records 1							
Max Command records 2							

Configurable Cmd (1 bit)

Configurable Cmd	Functionality
0	The local application has full control of the commands associated with the grouping. The commands can be monitored from a 2^{nd} network node.
1	The specific commands associated with the grouping can be controlled and monitored from a 2^{nd} network node. This option includes also the level field used by the command *Transfer Scene* in the *Controller Replication Command* Class. In this case the level value is transferred via the command *Basic Set* in the *Basic Command Class*.

V/C (1 bit)

V/C	Functionality
0	Command type. A Z-Wave command can be added to the node
1	Value type. A Value field is specified for a Node ID using the command *Basic Set* in the *Basic Command Class*. Level field originates from the command *Transfer Scene* in the *Controller Replication Command* Class.

Max command length (6 bits)

If Configurable Cmd is equal to 0x0, the field SHOULD return 0x0.

If Configurable Cmd is equal to 0x1 the field SHOULD return the maximum length of a command which can be associated to a node in a grouping. The minimum max command length allowed is 0x03.

Example:

A product reports a Max command length = 0x03. In this case the product can be programmed with a Multilevel Switch Set, but not a Switch Multilevel Start Level Change.

Multilevel Switch Set has a command length of 3

Multilevel Switch Start Level Change has a command length of 4

Free Command Records (16 bits)

The field specifies the current number of free Command Records which can be configured in the device through the Command Configuration Set Command.

Max Command records (16 bits)

The field specifies the maximum number of Command Records which can be configured in the device through the Command Configuration Set Command.

4.13.3 Command Configuration Set Command

The Command Configuration Set Command used to specify which commands SHOULD be sent to nodes within a given Grouping identifier.

Every Command Configuration Set will utilize one Command record from the pool of free Command records.

If the device has no free command records when receiving the Command Configuration Set, the command will be ignored.

The Application on the node may alter the commands when needed.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ASSOCIATION_COMMAND_CONFIGURATION							
Command = COMMAND_CONFIGURATION_SET							
Grouping identifier							
Node ID							
Command length							
Command Class identifier							
Command identifier							
Command byte 1							
..							
Command byte n							

Grouping identifier (8 bits)

This grouping identifier used to specify how nodes are grouped together. The grouping identifier values MUST be a sequence starting from 1. This field MAY be ignored in case the node only supports one group.

Node ID (8 bits)

This field contains the node ID within the grouping specified, that should receive the command.

Command length (8 bits)

This field specifies the complete command length (including command class and command identifiers).

Example:

Switch Multilevel Set 0x20. The command length field MUST be equal to 0x03.

Command Class Identifier (8 bits)

This field contains the identifier of the command class which should be sent to the Node ID. In case the Configurable Cmd field is equal to 0x0 is this field and the following ignored and can therefore be omitted.

In case the device has reported V/C=0x1, the Command Class and Command Identifiers MUST be equal to Basic Set command.

Command identifier (8 bits)

This field contains the identifier of the Command, which should be sent to the specified Node ID. In case the Configurable Cmd field is equal to 0x0 is this field ignored and can therefore be omitted.

In case the device has reported V/C=0x1, the Command Class and Command Identifiers MUST be equal to Basic Set Command.

Command byte (N bytes)

These fields contain the command parameters, which should be sent to the Node ID. In case the Configurable Cmd field is equal to 0x0 is these fields ignored and can therefore be omitted.

4.13.4 Command Configuration Get Command

The Command Configuration Get Command is used to request the commands specified for to a Node ID within a given Grouping identifier.

The Command Configuration Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_ASSOCIATION_COMMAND_CONFIGURATION</td></tr>
<tr><td colspan="8">Command = COMMAND_CONFIGURATION_GET</td></tr>
<tr><td colspan="8">Grouping identifier</td></tr>
<tr><td colspan="8">Node ID</td></tr>
</table>

Grouping identifier (8 bits)

This group identifier used to specify how nodes are grouped together. The group identifier values MUST be a sequence starting from 1. This field MUST be ignored in case the node only supports one group.

Node ID (8 bits)

This field specifies the node ID within the grouping.

4.13.5 Command Configuration Report Command

The Command Configuration Report Command used to report the commands specified for a Node ID within a given Grouping identifier.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ASSOCIATION_COMMAND_CONFIGURATION							
Command = COMMAND_CONFIGURATION_REPORT							
Grouping identifier							
Node ID							
First	Reserved			Reports to follow			
Command length							
Command Class identifier							
Command identifier							
Command byte 1							
..							
Command byte n							

Grouping identifier (8 bits)

This group identifier identifies the group. The group identifier values MUST be a sequence starting from 1.

Node ID (8 bits)

This field contains the node ID as requested in the Association Command Get.

Reports to follow (4 bits)

This value indicates how many report frames left before transferring the entire list of commands.

First (1 bit)

This field indicates that this report is the first report relating to a Grouping identifier/Node ID pair

Command length (8 bits)

This field specifies the complete command length (including command class and command identifiers).

Example:

Multilevel Switch Set 0x20. The command length field MUST be equal to 0x03.

Command Class Identifier (8 bits)

This field contains the identifier of the command class which is sent to the Node ID.

In case the device has reported V/C=0x1, the Command Class and Command Identifiers MUST be equal to Basic Set command

Command identifier (8 bits)

This field contains the identifier of the Command which is sent to the specified Node ID.

In case the device has reported V/C=0x1, the Command Class and Command Identifiers MUST be equal to Basic Set Command

Command byte (N bytes)

These fields contain the command parameter which is sent to the Node ID.

4.14 Association Group Information (AGI) command class, version 1

The Association Group Information (AGI) Command Class allows a device to report the capabilities of each association group supported by a given application resource.

Controllers and installer tools SHOULD use AGI information to support controller-assisted button-to-button association and GUI-based drag-and-drop association in a plug-and-play fashion.

Centralized gateway-based deployments may create a single association from the lifeline association group to a central management application..

The Association Group Information (AGI) Command Class also applies to Multi Channel devices. Refer to 3.6 for an introduction to the Multi Channel concept.

4.14.1 Association Principles

A light control transmitter may have one or more keys. A key is mapped to one or more association groups.

Figure 11, Light control transmitter (example)

Figure 12, Lamp module (example)

A light control transmitter key can be configured to control a lamp by adding the NodeID of the lamp to an association group that represents the key.

4.14.2 The Association Group Information Table

A device MAY implement one or more association groups. If the device implements association groups, the device SHOULD provide an Association Group Information (AGI) table. A Z-Wave Plus certified device MUST provide an Association Group Information (AGI) table. If the device implementsMulti Channel End Points, the device MUST provide an AGI table per End Point.

Table 14, Layout of one line of the Association Group Information table

Group	Profile	Command Class & Command (list) N bytes	Group Name (UTF-8)

An AGI table entry carries a number of fields. The information is typically static and may be defined at compile time. Thus, the AGI table does not need to occupy any RAM or non-volatile storage.

4.14.2.1 Group

The Group number indicates which association group the actual table line relates to. As association groups are always numbered from one and upwards, an actual implementation does not have to store the group number in physical memory. The number of association groups may be requested via the Association Groupings Get or Multi Channel Association Groupings Get commands.

Association group 1 is reserved for the Lifeline association group. Group 1 MUST NOT be assigned to any other use than the Lifeline group. The actual Device Type specifies a mandatory list of commands which the device must be able to send to all targets associated to the Lifeline group. A manufacturer MAY add additional commands to the lifeline group.

The Lifeline group (association group 1) MUST be advertised for the Root Device of a Multi Channel device. A Multi Channel End Point SHOULD advertise commands for association group 1 which the End Point can send via the Root Device Lifeline group if a Multi Channel Association is created from the Root Device Lifeline group.

4.14.2.2 Profile

A device MAY implement a number of AGI profiles. The profile defines the scope of events which triggers the transmission of commands to members of the actual association group. As an example a temperature sensor may issue different Basic Set command parameters when the temperature exceeds a threshold and when the temperature drops below a threshold. The actual behavior is application dependent and out of scope of the AGI command class.

The profile identifiers are referenced in Table 21.

4.14.2.2.1 General profiles

The “General” category comprises the “Not Applicable” and “Lifeline” Profiles. The “Not Applicable” profile identifier MUST be advertised if the actual association group does not provide any functionality.

The “Lifeline” profile identifier MUST be advertised for association group 1 of the Root Device.

4.14.2.2.2 Control profiles

“Control” profiles allow an implementation to provide information that can guide an installer in choosing between apparently identical association groups. As an example, a light control transmitter may comprise two control keys that can each control a group of lamps.

A control key may control up and down dimming as well as on/off state. Thus, a control key may send more than one command with one specific parameter. A control key may be implemented as a physical key, a group of icons, touch screen gestures or other means. The actual implementation is out of scope of this command class.

In case of multiple logical functions in a device, the device MAY implement one Multi Channel End Point for each logical function; e.g. each push button.
If an End Point is defined for each logical function, an association table and corresponding AGI table MUST be presented for each Multi Channel End Point.

Functions such as local reset, battery alarm and tamper alarm which do not relate to the actual application functionality of the product MUST reside in the Root Device if the product is implemented as a Multi Channel device.

The following example outlines how a battery-powered two-button light control transmitter is represented as a Multi Channel device:

Figure 13, Two-button light control transmitter

Root Device: 

Group	Profile 2 bytes	Command Class & Command (list) N bytes	Group Name (UTF-8)
1	General:Lifeline	Central Scene Notification, Notification Report, Battery Report, Device Reset Locally Notification	Lifeline
2	Control:Key1	Basic Set	On/Off control (Button 1)
3	Control:Key1	Multilevel Switch Set	Dimmer control (Button 1)
4	Control:Key2	Basic Set	On/Off control (Button 2)
5	Control:Key2	Multilevel Switch Set	Dimmer control (Button 2)

End Point 1: 

Group	Profile 2 bytes	Command Class & Command (list) N bytes	Group Name (UTF-8)
1	Control:Key1	Central Scene Notification	Button 1 via Lifeline
2	Control:Key1	Basic Set	On/Off control (Button 1)
3	Control:Key1	Multilevel Switch Set	Dimmer control (Button 1)

End Point 2: 

Group	Profile 2 bytes	Command Class & Command (list) N bytes	Group Name (UTF-8)
1	Control:Key2	Central Scene Notification	Button 2 via Lifeline
2	Control:Key2	Basic Set	On/Off control (Button 2)
3	Control:Key2	Multilevel Switch Set	Dimmer control (Button 2)

Table 15, Example: AGI tables for two-button light control transmitter

The Profile identifiers Control_Key 1 and Control_Key 2 allows an installer tool to determine which Multi Channel End Point and which association group to use in order to configure buttons to control a light dimmer.The Group Name allows a human user to make a qualified decision even if the installer tools is not, e.g. because it is running an older version not updated with newer profile identifiers.

The Root device in the above example advertises the Central Scene Notification command in the Lifeline group. Residing in a Multi Channel device, the Root Device does not actually implement any application functionality. The Central Scene Notification command is a feature of End Point 1 which is advertised for backwards compatibility with legacy devices not supporting Multi Channel End Points.
Likewise, the Root device advertises the Basic Set command in association group 2. This is also a feature of End Point 1. Refer to the Multi Channel Command Class for details on backwards compatibility.

End Point 1 in the above example advertises the Central Scene Notification command in association group 1 (the Lifeline group). By advertising that zero NodeIDs are supported for the Lifeline group, End Point 1 indicates that this command is sent via the Root Device Lifeline group if a Multi Channel association is created from the Root Device Lifeline group. The same principle applies to End Point 2. The number of NodeIDs supported by a specific Association Group may be requested via the Association Get or Multi Channel Association Get commands.

A controlling node SHOULD NOT create associations from other Root Device association groups than the Lifeline group if it supports the Multi Channel Command Class. The purpose of the Root Device Association Groups (with the exception of the Lifeline group) is only to provide backwards compatibility for legacy devices without support for the Multi Channel Command Class.

4.14.2.2.3 Sensor profiles

Sensor profile identifiers are constructed by prepending the command class identifier AGI_PROFILE_SENSOR to a multilevel sensor type. The values in Table 21 only serve as examples to illustrate the construction of AGI Sensor Profile Identifiers. For the full list of available sensor types, refer to “Multilevel Sensor Report Command”.

“Sensor” profiles allow an implementation to provide information that can guide an installer in choosing between apparently identical association groups that can issue sensor reports. As an example, a sensor product could comprise two temperature sensors.

In case of multiple logical functions in a device, the device SHOULD implement one Multi Channel End Point for each logical function; e.g. each sensor instance.
An association table and corresponding AGI table MUST be presented for each End Point.

Functions such as local reset, battery alarm and tamper alarm which do not relate to the actual application functionality of the product MUST reside in the Root Device if the product is implemented as a Multi Channel device.

The following example outlines how a battery-powered sensor device with two multilevel sensor functions is represented as a Multi Channel device:

Figure 14, Multi-function Sensor

Root Device: 

Group	Profile 2 bytes	Command Class & Command (list) N bytes	Group Name (UTF-8)
1	General:Lifeline	Multilevel Sensor Report, Notification Report, Battery Report, Device Reset Locally Notification	Lifeline
2	Sensor:Temperature	Basic Set	On/Off control (Indoor temperature)
3	Sensor:Temperature	Basic Set	On/Off control (Outdoor temperature)

End Point 1: 

Group	Profile 2 bytes	Command Class & Command (list) N bytes	Group Name (UTF-8)
1	Sensor:Temperature	Multilevel Sensor Report	Indoor temperature via Lifeline
2	Sensor:Temperature	Basic Set	On/Off control (Indoor temperature)

End Point 2: 

Group	Profile 2 bytes	Command Class & Command (list) N bytes	Group Name (UTF-8)
1	Sensor:Temperature	Multilevel Sensor Report	Outdoor temperature via Lifeline
2	Sensor:Temperature	Basic Set	On/Off control (Outdoor temperature)

Table 16, Example: AGI tables for two-function sensor

The Profile identifier is Sensor:Temperature for both End Points but the Group Name allows an installer tool to determine which End Point and which association group to use in order to configure the indoor temperature sensor to control a heating element.

The Root Device in the above example advertises the Multilevel Sensor Report command in the Lifeline group. Residing in a Multi Channel device, the Root Device does not actually implement any application functionality. The Multilevel Sensor Report command is a feature of the End Points which is advertised for backwards compatibility with legacy devices not supporting the Multi Channel Command Class. Likewise, the Root device advertises the Basic Set command in association groups 2 and 3. This is also a feature of the End Point. Refer to the Multi Channel Command Class for details on backwards compatibility.

End Point 1 in the above example advertises the Multilevel Sensor Report command in association group 1. By advertising that zero NodeIDs are supported for association group 1, End Point 1 indicates that this command is sent via the Root Device Lifeline group if a Multi Channel association is created for the Root Device Lifeline group. The same principle applies to End Point 2.

4.14.2.2.4 Notification profiles

Notification profile identifiers are constructed by prepending the command class identifier AGI_PROFILE_NOTIFICATION to a notification type. The values below only serve as an example to illustrate the construction of AGI Notification Profile identifiers. For the full list of available notification types, refer to “Notification Types & Events” of [2].

“Notification” profiles allow an implementation to provide information that can guide an installer in choosing between apparently identical association groups. As an example, a detector product could comprise one smoke sensor and one CO2 detektor.

In case of multiple logical functions in a device, the device SHOULD implement one Multi Channel End Point for each logical function.
If implementingMulti Channel End Points, an association table and corresponding AGI table MUST be presented for each End Point.

Functions such as local reset, battery alarm and tamper alarm which do not relate to the actual application functionality of the product MUST reside in the Root Device if the product is implemented as a Multi Channel device.

The following example outlines how a stand-alone smoke alarm is represented by an AGI table:

Root Device: 

Group #	Profile 2 bytes	Command Class & Command (list) N bytes	Group Name (UTF-8)
1	General:Lifeline	Notification Report, Battery Report, Device Reset Locally Notification	Lifeline
2	Notification:SmokeAlarm	Basic Set	On/Off control (Smoke)

Table 17, Example: AGI table for one-function alarm device

The Profile identifier Notification:SmokeAlarm indicates to an installer that the the Basic Set command of association group 2 is issued in response to smoke alarm events.

4.14.2.3 Command Class and Command Identifier

This field contains the command class and the command that is sent to targets associated with this association group if an event occurs.
This MAY be a list of commands. The list MAY contain a combination of extended and normal command classes. Refer to 4.14.8.

In case Multi Channel End Points are implemented, the Lifeline association group (group 1) of End Points SHOULD be used to advertise commands that will be sent via the Root Device Lifeline group if a Multi Channel association is created from the Root Device Lifeline group.

4.14.2.4 Association group name

A vendor SHOULD assign an association group name which reflects the purpose of the group, e.g. “On/Off control (Smoke)”.
As the only exception from this rule, the Root Device Lifeline group MUST be named “Lifeline”.

42 bytes are available for text. Text MUST be encoded using UTF-8. The available capacity for characters depends on the actual characters encoded with UTF-8. Plain ASCII characters only occupy one byte while special characters may need two or more bytes for representation.

4.14.3 Association Group Name Get

The Association Group Name Get command is used to query the name of an association group.

The Association Group Name Report command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ASSOCIATION_GRP_INFO							
Command = ASSOCIATION_GROUP_NAME_GET							
Grouping Identifier							

Grouping Identifier (1 byte)

The number of the actual association group.

A node that receives an unsupported Grouping Identifier SHOULD return information relating to Grouping Identifier 1.

4.14.4 Association Group Name Report

The Association Group Name Report command is used to advertise the name of an association group.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ASSOCIATION_GRP_INFO							
Command = ASSOCIATION_GROUP_NAME_REPORT							
Grouping Identifier							
Name Length							
Name 1							
...							
Name N							

Grouping Identifier (1 byte)

The number of the actual association group.

Name Length (1 byte)

The length of the Name field. The value MUST be in the range 0..42.

Name (N bytes)

The name of the requested group. The length MUST be defined by the "Name Length" field.
The field MUST be formatted as a byte array with no zero termination.
The characters MUST be encoded in UTF-8 format.

4.14.5 Association Group Info Get

The Association Group Info Get command is used to request the properties of an association group.

The Association Group Info Report MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing.
The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ASSOCIATION_GRP_INFO							
Command = ASSOCIATION_GROUP_INFO_GET							
Refresh cache	List Mode	*Reserved*					
Grouping Identifier							

List Mode (1 bit)

If List Mode is 1, a responding node SHOULD advertise the properties of all association groups in one report. A responding node MAY return many reports, e.g. due to memory constraints or a high number of association groups.

If List Mode is 0, a responding node MUST advertise the properties of the association group identified by the Grouping Identifier.

The Association Group Info Report returned in response to this command MUST advertise the same List Mode value as specified by this field.

Refresh Cache (1 bit)

If AGI information is transferred via a gateway, the gateway MUST cache information for all nodes; also listening nodes.
The AGI Get command flag "Refresh Cache" is used by a management application to instruct a gateway to update its cache on the first chance given. In case of a sleeping device, this may require a user operation to wake up the device. In case of a WakeUp device, the gateway may wait for the next WakeUp Notification from the actual device.

The "Refresh Cache" value 0 MUST specify that the gateway is to return its cached information.
The "Refresh Cache" value 1 MUST specify that the gateway is to return its cached information AND update its cache on the first chance given.

The "Refresh Cache" SHOULD be set to 0 by a sending node.
A receiving node may have re-configurable AGI properties, e.g. if it is a configurable multi-purpose remote control. If the "Dynamic Info" flag of the AGI Report command is set to 1, a sending node MAY set the "Refresh Cache" flag in the AGI Get command to force the gateway to update its cache, e.g. after the sending node has re-configured the AGI properties of the receiving node.

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Grouping Identifier (1 byte)

The number of the actual association group. This value MAY be ignored if the List Mode field is set to 1.

A node that receives an unsupported Grouping Identifier SHOULD return information relating to Grouping Identifier 1.

4.14.6 Association Group Info Report

The Association Group Info Report command is used to advertise the properties of one or more association groups.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ASSOCIATION_GRP_INFO							
Command = ASSOCIATION_GROUP_INFO_REPORT							
List mode	Dynamic Info	Group Count					
Grouping Identifier							
Mode = 0							
Profile MSB							
Profile LSB							
Reserved = 0							
Event Code MSB = 0							
Event Code LSB = 0							
...							
Grouping Identifier							
Mode = 0							
Profile MSB							
Profile LSB							
Reserved = 0							
Event Code MSB = 0							
Event Code LSB = 0							

List Mode (1 bit)

The List Mode field MUST advertise the same value as specified in the Association Group Info Get command which caused this command to be returned.

If List Mode is 1, a responding node SHOULD advertise the properties of all association groups in one report. Each block MUST start with the Grouping Identifier advertising the Association Group relating to the actual block of data. A responding node MAY return many reports.

If List Mode is 0 a responding node MUST advertise the properties of one association group identified by the Grouping Identifier.

Dynamic Info (1 bit)

If the Dynamic Info field is set to 1, the information MAY change and the Z-Wave Gateway device SHOULD perform periodic cache refresh for this node. Nodes MUST set this bit if they are able to change information.

If this field is set to 0, a Gateway device MUST NOT request this information more than one time.

Group Count (6 bits)

Number of groups in the frame.

If List Mode is 0, the Group Count field MUST be set to 1.

If List Mode is 1, the Group Count field MUST advertise the number of association group property blocks following this field.

A node receiving this report may determine the total number of association groups via the Association Supported Groupings Get or Multi Channel Association Supported Groupings Get commands.

Grouping Identifier (1 byte)

This field MUST advertise the association group identifier of the actual association group property block.

Mode (1 byte)

Reserved field. This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Profile (2 bytes)

Refer to section 4.14.2.2.

Reserved

Reserved field. This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Event Code (2 bytes)

Reserved field. This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

4.14.7 Association Group Command List Get

Command used to request the commands that may be sent to targets of this the association group.

The Association Group Command List Report MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_ASSOCIATION_GRP_INFO</td></tr>
<tr><td colspan="8">Command = ASSOCIATION_GROUP_COMMAND_LIST_GET</td></tr>
<tr><td>Allow cache</td><td colspan="7">Reserved</td></tr>
<tr><td colspan="8">Grouping Identifier</td></tr>
</table>

Allow Cache (1 bit)

This parameter indicates that a Z-Wave Gateway device is allowed to intercept the request and return a cached response on behalf of the specified target.
If this bit is not set, a Z-Wave Gateway device MUST forward the request to the specified target.

A requesting node SHOULD allow caching to save network bandwidth. A Z-Wave Gateway device MUST cache information for all nodes; also listening nodes. This will save network bandwidth.

A Z-Wave Gateway device MUST forward the request if it does not hold cached information for the specified target. The Z-Wave Gateway device MUST cache the data returned.

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Grouping Identifier (1 byte)

The number of the actual association group.

A node that receives an unsupported Grouping Identifier SHOULD return information relating to Grouping Identifier 1.

4.14.8 Association Group Command List Report

Command reporting the commands that MAY be sent from the association group.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ASSOCIATION_GRP_INFO							
Command = ASSOCIATION_GROUP_COMMAND_LIST_REPORT							
Grouping Identifier							
List Length							
Command Class A (1 or 2 bytes)							
Command A							
Command Class B (1 or 2 bytes)							
Command B							
Command Class C (1 or 2 bytes)							
Command C							
…							

Grouping Identifier (1 byte)

The number of the actual association group.

List Length (1 byte)

The length (in bytes) of the command list.

Command (N bytes)

Commands MAY be normal or extended commands.
Normal command classes are represented as one byte while extended command classes are represented as two bytes. Thus, a command list entry (command class + command) is either 2 or 3 bytes long.

The receiving node must parse individual command list entries to determine if the individual command class is a normal or an extended command class.
The first byte of normal command classes is in the range 0x20 – 0xEE, while the first byte of extended command classes is in the range 0xF1 – 0xFF.

No command payload bytes are included in the command list.

Notice that command classes already containing destination node ID such as Wake Up Command Class MUST NOT be listed in the Association Group Command List Report. This also means that the Wake Up Command Class MUST NOT be covered by the Lifeline association group.

4.15 Association Group Information (AGI) command class, version 2

The Association Group Information (AGI) Command Class allows a device to report the capabilities of each association group supported by a given application resource.

4.15.1 Compatibility considerations

The Association Group Information (AGI) Command Class, version 2 introduces the Profile category "Meter".

The Association Group Information (AGI) Command Class, version 2 defines no new command fields or changes to the interpretation of Association Group Information (AGI) Command Class, version 1.

The version 2 section only specifies issues relating to Profile category "Meter".

4.15.2 The Association Group Information Table

A device MAY implement one or more association groups. If the device implements association groups, the device SHOULD provide an Association Group Information (AGI) table. A Z-Wave Plus certified device MUST provide an Association Group Information (AGI) table. If the device implements Multi Channel End Points, the device MUST provide an AGI table per End Point.

Table 18, Layout of one line of the Association Group Information table

Group	Profile	Command Class & Command (list) N bytes	Group Name (UTF-8)

An AGI table entry carries a number of fields. The information is typically static and may be defined at compile time. Thus, the AGI table does not need to occupy any RAM or non-volatile storage.

4.15.2.1 Group

Refer to version 1.

4.15.2.2 Profile

A device MAY implement a number of AGI profiles. The profile defines the scope of events which triggers the transmission of commands to members of the actual association group. As an example a temperature sensor may issue different Basic Set command parameters when the temperature exceeds a threshold and when the temperature drops below a threshold. The actual behavior is application dependent and is out of scope of the AGI command class.

The profile identifiers are referenced in Table 21.

4.15.2.2.1 General profiles

Refer to version 1.

4.15.2.2.2 Control profiles

Refer to version 1.

4.15.2.2.3 Sensor profiles

Refer to version 1.

4.15.2.2.4 Notification profiles

Refer to version 1.

4.15.2.2.5 Meter profiles

Meter profile identifiers are constructed by prepending the command class identifier AGI_PROFILE_METER to a meter type. The values in Table 21 only serve as examples to illustrate the construction of AGI Meter Profile identifiers. For the full list of available meter types, refer to “Meter Report Command”.

“Meter” profiles allow an implementation to provide information that can guide an installer in choosing between association groups that can be triggered by meter events. As an example, a two-phase meter product could comprise three electric meters: one for each phase and one for the total consumption.

In case of multiple logical functions in a device, the device SHOULD implement one Multi Channel End Point for each logical function; e.g. each meter instance.
An association table and corresponding AGI table MUST be presented for each End Point.

Functions such as local reset, battery alarm and tamper alarm which do not relate to the actual application functionality of the product MUST reside in the Root Device if the product is implemented as a Multi Channel device.

The following example outlines how a meter device with three meter functions is represented as a Multi Channel device:

Figure 15, Multi-function Meter

Root Device: 

Group	Profile 2 bytes	Command Class & Command (list) N bytes	Group Name (UTF-8)
1	General:Lifeline	Meter Report, Device Reset Locally Notification	Lifeline
2	Meter:Electric	Basic Set	On/Off control (Phase 1)
3	Meter:Electric	Basic Set	On/Off control (Phase 2)
4	Meter:Electric	Basic Set	On/Off control (Total consumption)

End Point 1: 

Group	Profile 2 bytes	Command Class & Command (list) N bytes	Group Name (UTF-8)
1	Meter:Electric	Meter Report	Phase 1 via Lifeline
2	Meter:Electric	Basic Set	On/Off control (Phase 1)

End Point 2: 

Group	Profile 2 bytes	Command Class & Command (list) N bytes	Group Name (UTF-8)
1	Meter:Electric	Meter Report	Phase 2 via Lifeline
2	Meter:Electric	Basic Set	On/Off control (Phase 2)

End Point 3: 

Group	Profile 2 bytes	Command Class & Command (list) N bytes	Group Name (UTF-8)
1	Meter:Electric	Meter Report	Total consumption via Lifeline
2	Meter:Electric	Basic Set	On/Off control (Total consumption)

Table 19, Example: AGI tables for three-function meter

The Profile identifier is Meter:Electric for all End Points but the Group Name allows an installer tool to determine which End Point and which association group to use in order to configure the phase 2 meter to control an alarm lamp.

The Root Device in the above example advertises the Meter Report command in the Lifeline group. Residing in a Multi Channel device, the Root Device does not actually implement any application functionality. The Meter Report command is a feature of the End Points which is advertised for backwards compatibility with legacy devices not supporting the Multi Channel Command Class. Likewise, the Root device advertises the Basic Set command in association groups 2-4. This is also a feature of the End Points. Refer to the Multi Channel Command Class for details on backwards compatibility.

End Points 1-3 in the above example advertise the Meter Report command in association group 1. By advertising that zero NodeIDs are supported for association group 1, End Point 1 indicates that this command is sent via the Root Device Lifeline group if a Multi Channel association is created for the Root Device Lifeline group. The same principle applies to End Points 2 and 3.

4.16 Association Group Information (AGI) command class, version 3

The Association Group Information (AGI) Command Class allows a device to report the capabilities of each association group supported by a given application resource.

4.16.1 Compatibility considerations

The Association Group Information (AGI) Command Class, version 3 introduces the Profile category “Irrigation”.

The Association Group Information (AGI) Command Class, version 3 defines no new command fields or changes to the interpretation of Association Group Information (AGI) Command Class, version 1 and version 2.

The version 3 section only specifies issues relating to Profile category “Irrigation”.

4.16.2 The Association Group Information Table

A device MAY implement one or more association groups. If the device implements association groups, the device SHOULD provide an Association Group Information (AGI) table. A Z-Wave Plus certified device MUST provide an Association Group Information (AGI) table. If the device implements Multi Channel End Points, the device MUST provide an AGI table per End Point.

Table 20, Layout of one line of the Association Group Information table

Group	Profile	Command Class & Command (list) N bytes	Group Name (UTF-8)

An AGI table entry carries a number of fields. The information is typically static and may be defined at compile time. Thus, the AGI table does not need to occupy any RAM or non-volatile storage.

4.16.2.1 Group

Refer to version 1.

4.16.2.2 Profile

A device MAY implement a number of AGI profiles. The profile defines the scope of events which triggers the transmission of commands to members of the actual association group. As an example a temperature sensor may issue different Basic Set command parameters when the temperature exceeds a threshold and when the temperature drops below a threshold. The actual behavior is application dependent and is out of scope of the AGI command class.

The profile categories are referenced in Table 21.

Table 21, AGI Profiles

Profile	Explanation	Profile identifier MS	 LS
General:NA (v1)	"Not Applicable" There is no specific class of events for this association group	AGI_PROFILE_ GENERAL = 0x00	AGI_ GENERAL_NA = 0x00
General:Lifeline (v1)	"Lifeline" This association group is intended for all events relevant for theLifeline group.	AGI_PROFILE_ GENERAL = 0x00	AGI_ GENERAL_LIFELINE = 0x01
Control:Key01 (v1)	"Control Key 1" Members of this association group are controlled in response to user input for key 1	AGI_PROFILE_ CONTROL = 0x20	AGI_ CONTROL_KEY01 = 0x01
Control:Key xx (v1)	…		
Control:Key32 (v1)	"Control Key 32" Members of this association group are controlled in response to user input for key 32	AGI_PROFILE_ CONTROL = 0x20	AGI_ CONTROL_KEY32 = 0x20
Sensor: Air temperature (v1)	"Sensor, Air Temperature" Members of this association group are controlled when the sensor value changes or receives a sensor report of the given sensor type	AGI_PROFILE_ SENSOR = 0x31	MULTILEVEL_SENSOR_TYPE_ TEMPERATURE = 0x01
Sensor: Humidity (v1)	"Sensor, Humidity" Members of this association group are controlled when the sensor value changes or receives a sensor report of the given sensor type	AGI_PROFILE_ SENSOR = 0x31	MULTILEVEL_SENSOR_TYPE_ HUMIDITY = 0x05
…	(Only examples above. For all sensor types, refer to the table of defined Multilevel Sensor types)	…	…
Notification: Smoke Alarm (v1)	"Notification, Smoke Alarm" Members of this association group are controlled when an event is detected or receives a notification of the given notification type	AGI_PROFILE_ NOTIFICATION = 0x71	NOTIFICATION_TYPE_ SMOKE = 0x01
Notification: CO_2 Alarm (v1)	"Notification, CO_2 Alarm" Members of this association group are controlled when an event is detected or receives a notification of the given notification type	AGI_PROFILE_ NOTIFICATION = 0x71	NOTIFICATION_TYPE_ CO2 = 0x03
…	(Only examples above. For all notification types, refer to the table of defined Notification Types & Events)	…	…

Profile	Explanation	Profile identifier MS	 LS
Meter: Electric, kWh (v2)	"Meter, Electric" Members of this association group receive meter reports or they are controlled when a metering event is detected or	AGI_PROFILE_ METER = 0x32	METER_TYPE_ELECTRIC = 0x01
Meter: Gas (v2)	"Meter, Gas" Members of this association group receive meter reports or they are controlled when a metering event is detected or	AGI_PROFILE_ METER = 0x32	METER_TYPE_GAS = 0x02
Meter: Water (v2)	"Meter, Water" Members of this association group receive meter reports or they are controlled when a metering event is detected or	AGI_PROFILE_ METER = 0x32	METER_TYPE_WATER = 0x03
…	(Only examples above. For all meter profiles, refer to the table of defined Meter Types)	…	…
Irrigation: Channel 01 (v3)	"Irrigation Channel 01" Member(s) of this association group are controlled by channel 1 of an irrigation control device	AGI_PROFILE_ IRRIGATION = 0x6B	AGI_ IRRIGATION_CHANNEL_01 = 0x01
Irrigation: Channel xx (v3)	…		
Irrigation: Channel 32 (v3)	"Irrigation Channel 32" Member(s) of this association group are controlled by channel 32 of an irrigation control device	AGI_PROFILE_ IRRIGATION = 0x6B	AGI_ IRRIGATION_CHANNEL_32 = 0x20

4.16.2.2.1 General profiles

Refer to version 1.

4.16.2.2.2 Control profiles

Refer to version 1.

4.16.2.2.3 Sensor profiles

Refer to version 1.

4.16.2.2.4 Notification profiles

Refer to version 1.

4.16.2.2.5 Meter profiles

Refer to version 2.

4.16.2.2.6 Irrigation Profiles

“Irrigation” profiles allow an implementation to provide information that can guide an installer in choosing between apparently identical association groups. As an example, an irrigation control device may provide 8 channels that can each control an external resource such as a valve or a pump.

4.17 Barrier Operator Command Class, version 1

The Barrier Operator Command Class is used to control and query the status of motorized barriers.

A device MAY ignore a Barrier Operator Command if the requested operation violates operational limitations or safety regulations.

The Barrier Command Class MUST use secure communication.

4.17.1 Barrier Operator Set Command

The Barrier Operator Set Command is used to initiate an unattended change in state of the barrier.

A device MAY ignore the Barrier Set Command if the requested operation violates operational limitations or safety regulations.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_BARRIER_OPERATOR							
Command = BARRIER_OPERATOR_SET							
Target Value							

Target Value

The Target Value field MUST specify the intended state of the device.

The encoding of the Target Value field MUST be according to Table 22.

Table 22, Barrier Set::Target Value

Target Value		Description	CC Version
0x00	CLOSE	Initiate unattended close	1
0xFF	OPEN	Initiate unattended open	1

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

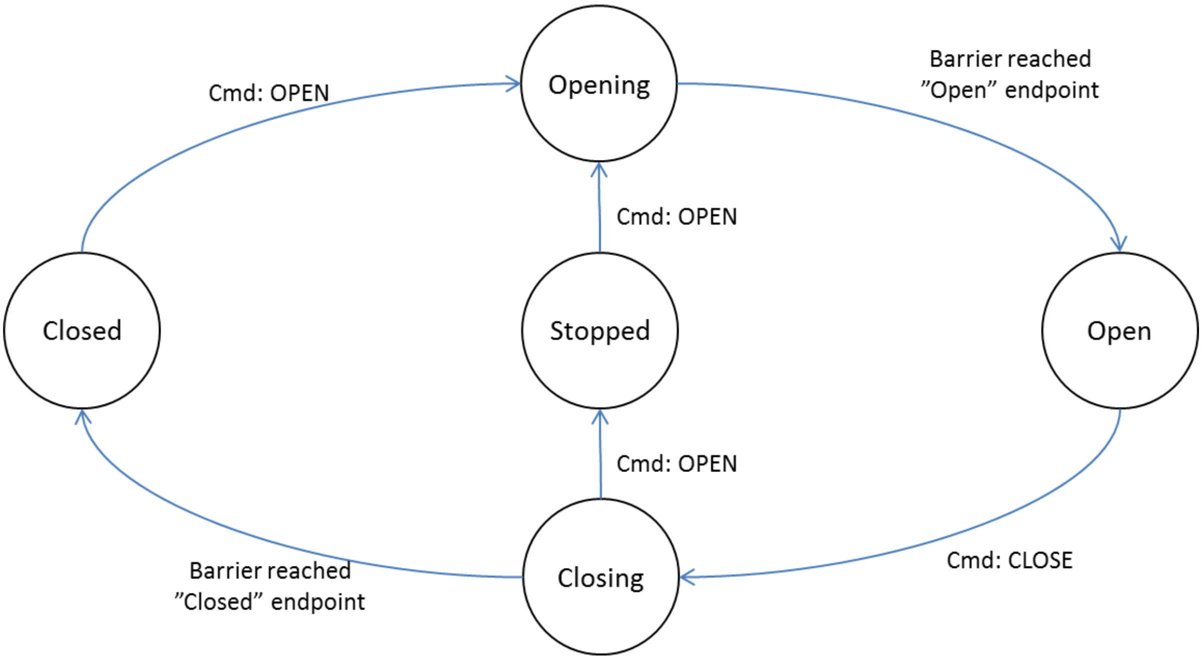


Figure 16, Barrier Set, state transitions

A receiving device MUST comply with the allowed state changes and triggering commands specified in Figure 16. Any state/command pair not specified in Figure 16 MUST NOT cause any action.

The receiving device SHOULD respond to OPEN and CLOSE commands. However,
the device MUST NOT respond to OPEN or CLOSE commands if any safety issue is detected.
The receiving device MUST NOT respond to the CLOSE command unless it is in the "Open" state.

4.17.1.1 Error Handling

The device MUST stop if any safety issue is detected.

If the requested operation is overruled by a safety mechanism, the device MUST notify the requester via a Notification Report (Notification Type: “Access Control”, Event: “Barrier unable to perform requested operation due to safety requirements”). The Notification Report command is a member of the Notification Command Class.

If the requested operation is prohibited by the specific device class, e.g. SPECIFIC_TYPE_SECURE_BARRIER_OPEN_ONLY or SPECIFIC_TYPE_SECURE_BARRIER_CLOSE_ONLY, the device MUST notify the requester by issuing an Application Rejected Request command. The Application Rejected Request command is a member of the Application Status.

4.17.2 Barrier Operator Get Command

The Barrier Operator Get Command is used to request the state of the device.

The Barrier Report command MUST be returned in response to a Barrier Get command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_BARRIER_OPERATOR							
Command = BARRIER_OPERATOR_GET							

4.17.3 Barrier Operator Report Command

Barrier Operator Report Command is used to advertise the status of the device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_BARRIER_OPERATOR							
Command = BARRIER_OPERATOR_REPORT							
State							

State

The State field MUST advertise the current state of the device.

The encoding of the State field MUST be according to Table 23.

Table 23, Barrier Report :: State

State		Requirement level	Description	CC Version
0x00	Closed	REQUIRED	The barrier is in the Closed position	1
0x01 .. 0x63	Stopped at exact Position	RECOMMENDED	0x01 = 1% (Near Closed) 0x63 = 99% (Near Open)	1
0xFC	Closing	REQUIRED	The barrier is closing. The current position is unknown.	1
0xFD	Stopped	REQUIRED	The barrier is stopped. The current position is unknown.	1
0XFE	Opening	REQUIRED	The barrier is opening. The current position is unknown.	1
0xFF	Open	REQUIRED	The barrier is in the Open position	1

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

4.17.4 Barrier Operator Get Signaling Capabilities Supported Command

The Barrier Operator Get Event Signaling Capabilities Supported Command is used to query a device for available subsystems which may be controlled via Z-Wave.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_BARRIER_OPERATOR							
Command = BARRIER_OPERATOR_SIGNAL_SUPPORTED_GET							

4.17.5 Barrier Operator Report Event Signaling Capabilities Supported Command

The Barrier Operator Report Event Signaling Capabilities Supported command returns a bit mask of signaling subsystem(s) supported by the device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_BARRIER_OPERATOR							
Command = BARRIER_OPERATOR_SIGNAL_SUPPORTED_REPORT							
Bit Mask 1							
….							
Bit Mask N							

Bit Mask (N bytes)

The Bit Mask field describes the event signaling capabilities supported by the barrier device. The first bit (Bit 0) in Bit Mask 1 indicates support for subsystem = 0x01 (Audible Notification). The full list of supported subsystems defined by the command class may be found in the Table: “Subsystem Type”.

The length of the Bit Mask is dependent upon the number of subsystems supported by the device, with a maximum size of N =32, and minimum size of N = 1. A device MUST limit the size of the Bit Mask to represent only those subsystems supported, and MUST NOT extend the length of the Bit Mask to pad the packet with 0x00.

4.17.6 Barrier Operator Event Signal Set Command

The Barrier Operator Event Signaling Set Command is used to turn ON or OFF and an event signaling subsystem that is supported by the device. UL requirements take precedent over any ON / OFF commands sent to a subsystem as such it is up to the discretion of the device accept or request the requested action. If the requested action is rejected then the device must notify the requester via the Notification Command Class Report, Notification Type: Access Control, Event: Barrier unable to perform requested operation due to UL requirements.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_BARRIER_OPERATOR							
Command = BARRIER_OPERATOR_SIGNAL_SET							
Subsystem Type							
Subsystem State							

Subsystem Type

Subsystem Type	Description	CC Version
0x00	NOT SUPPORTED	1
0x01	The Barrier Device has an Audible Notification subsystem controllable via Z-Wave (For example: Siren)	1
0x02	The Barrier Device has an Visual Notification subsystem controllable via Z-Wave (For example: Flashing Light)	1
0x03 – 0xFF	RESERVED	1

Subsystem State

Subsystem State	Description	CC Version
0x00	Subsystem OFF	1
0x01 – 0xFE	RESERVED	1
0xFF	Subsystem ON	1

4.17.7 Barrier Operator Event Signaling Get Command

The Barrier Operator Event Signaling Get command is used to request the state of a notification subsystem of a Barrier Device.

The Barrier Operator Event Signaling Report Command MUST be returned in response to this command if the specified type is supported.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_BARRIER_OPERATOR							
Command = BARRIER_OPERATOR_SIGNAL_GET							
Subsystem Type							

4.17.8 Barrier Operator Event Signaling Report

The Barrier Operator Event Signaling Report command is used to indicate the state of a notification subsystem of a Barrier Device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_BARRIER_OPERATOR							
Command = BARRIER_OPERATOR_SIGNAL_REPORT							
Subsystem Type							
Subsystem State							

4.18 Basic Command Class, version 1

The Basic Command Class allows a controlling device to operate the primary functionality of a supporting device without any further knowledge.

The Basic Command Class ensures a basic interoperability if no other command class is shared by two devices.

The Basic Command Class is an actuator control command class. Refer to 3.7.

4.18.1 Compatibility considerations

A specific device may not be able to support all Basic CC commands or parameter levels. For instance, a relay can only open fully in response to any non zero value.

Any device SHOULD support the Basic Command Class. If mandated by a generic or specific device class, a device MUST support the Basic Command Class.

A device MUST implement mappings from the Basic Command Class to specific commands according to the advertised generic and specific device class of the Node Info frame. Mappings defined by a specific device class have precedence over the generic device class.

For Z-Wave Plus devices, the Basic Command Class MUST be mapped according to the actual Z-Wave Plus Device Type. For further information, refer to [9].

The Basic Command Class MUST NOT be advertised in the Node Information Frame.
The Basic Command Class MUST NOT be advertised in the Security Commands Supported Report (S0 as well as S2)

A securely included node MAY support the Basic Command Class at the highest security level but it MUST NOT support the Basic Command Class at any lower security level or non-securely.

The following sections only present frame formats. For details on the mapping to other command classes, refer to [1] and [9].

4.18.2 Basic Set Command

The Basic Set Command, version 1 is used to set a value in a supporting device.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_BASIC</td></tr>
<tr><td colspan="8">Command = BASIC_SET</td></tr>
<tr><td colspan="8">Value</td></tr>
</table>

Value (8 bits)

A supporting device SHOULD support all parameter values in the range {0x00..0x63, 0xFF}.

A controlling device MUST NOT assume that a supporting device will always react to a Basic Set Command.

A receiving device MUST interpret Basic Set parameter values according to the requirements of the Device Class implemented by the device. Refer to [1] and [9].

4.18.3 Basic Get Command

The Basic Get Command, version 1 is used to request the status of a supporting device.

The Basic Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_BASIC							
Command = BASIC_GET							

4.18.4 Basic Report Command

The Basic Report command, version 1 is used to advertise the status of the primary functionality of the device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_BASIC							
Command = BASIC_REPORT							
Value							

Value (8 bits)

The Value field SHOULD advertise the current value of the device hardware; also while in transition to a new target value.

For details on the mapping of values, refer to [1], [9].

A controlling device MUST NOT assume that the Value is identical to a value previously specified in a Set command; not even when a transition has ended.

A receiving device MUST interpret the Value field according to Table 24.

Table 24, Basic Report :: Value

Value		Level	State
0	(0x00)	0%	Off
1..99	(0x01..0x63)	1..100%	On
…		reserved	reserved
254	(0xFE)	Unknown	Unknown
255	(0xFF)	100%	On

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

4.19 Basic Command Class, version 2

The Basic Command Class allows a controlling device to operate the primary functionality of another device without any further knowledge.

The Basic Command Class ensures a basic interoperability if no other command class is shared by two devices.

The Basic Command Class is an actuator control command class. Refer to chapter 3.7.

Commands not described in Version 2 stays unchanged from Version 1.

4.19.1 Compatibility considerations

A device supporting Basic CC, Version 2 MUST support Basic CC, version 1.

Version 2 adds the distinction between the Current Value and the Target State of the device.
Version 2 elevates the requirement level so that the Current Value field of the Basic Report command MUST advertise the current value of the device hardware; also while in transition to a new target value.

Unchanged commands have been omitted.

A specific device may not be able to support all commands or parameter levels. For instance, a relay can only open fully in response to any non-zero value.

All devices SHOULD support the Basic Command Class. If mandated by a generic or specific device class, a device MUST support the Basic Command Class.

A device MUST implement mappings from the Basic Command Class to specific commands according to the advertised generic and specific device class of the Node Info frame. Mappings defined by a specific device class have precedence over the Generic Device Class.

For Z-Wave Plus devices, the Basic Command Class MUST be mapped according to the actual Z-Wave Plus Device Type. For further information, refer to [9].

The Basic Command Class MUST NOT be advertised in the Node Information Frame.
The Basic Command Class MUST NOT be advertised in the Security Commands Supported Report (S0 as well as S2)

A securely included node MAY support the Basic Command Class at the highest security level
A securely included node MUST NOT support the Basic Command Class at any lower security level or non-securely.

The following sections only present frame formats. For details on the mapping to other command classes, refer to [1] and [9].

4.19.2 Basic Report Command

The Basic Report command, version 2 is used to advertise the status of the primary functionality of the device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_BASIC							
Command = BASIC_REPORT							
Current Value							
Target Value							
Duration							

Current Value (8 bits)

The Current Value field MUST advertise the current value of the device hardware; also while in transition to a new target value.

The advertised values may vary depending on the device class implemented by the supporting device. A controlling device MUST interpret the value according to Table 24.For details on the mapping of values, refer to [1] and [9].

The Current Value SHOULD be identical to the Target Value when a transition has ended.

Target Value (8 bits)

The Target Value field MUST advertise the target value of an ongoing transition or the most recent transition.

The advertised values may vary depending on the device class implemented by the supporting device. A controlling device MUST interpret the value according to Table 24.

For details on the mapping of values, refer to [1] and [9].

If queried after receiving a Set command, the Target Value field MUST advertise the target value specified in the Set command. The Target Value MAY change at a later time due to local control or a “Stop” motion control command.

If the device is in a motion controlled transition, the Target Value field MUST advertise the min or max value (depending on the direction) according to the command class mappings defined in [1] and [9].

Duration (8 bits)

The Duration field SHOULD advertise the time needed to reach the Target Value at the actual transition rate. The encoding of the Duration field MUST be according to Table 25.

Table 25, Basic Report :: Duration

Duration	Description
0x00	0 seconds. Already at the Target Value.
0x01-0x7F	1 second (0x01) to 127 seconds (0x7F) in 1 second resolution.
0x80-0xFD	1 minute (0x80) to 126 minutes (0xFD) in 1 minute resolution.
0xFE	Unknown duration
0xFF	Reserved

All other values are reserved. Reserved values MUST NOT be used by a sending node and MUST be ignored by a receiving node.

4.20 Basic Tariff Information Command Class, version 1

This Basic Tariff Information Command Class for use with a single element or dual element meter, and for use with import (electricity received from grid) rates only. The command class is kept as simple as possible without any pricing information.

This command class supports a GET and REPORT.

No Set command is supported, as it is not appropriate to set any of the parameters through Z-Wave.

4.20.1 Basic Tariff Information Get Command

The Basic Tariff Information Get command is used to request current tariff information from the meter.

The Basic Tariff Information Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_BASIC_TARIFF_INFO							
Command = BASIC_TARIFF_INFO_GET							

The response is the Basic Tariff Information Report.

4.20.2 Basic Tariff Information Report Command

The Basic Tariff Information Report command returns information on the number of import rates supported, and current import rate information. Application can send unsolicited Basic Tariff Report commands or requested by the Basic Tariff Get Information command.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_BASIC_TARIFF_INFO</td></tr>
<tr><td colspan="8">Command = BASIC_TARIFF_INFO_REPORT</td></tr>
<tr><td>Dual</td><td colspan="3">Reserved</td><td colspan="4">Total No. Import Rates</td></tr>
<tr><td colspan="4">Reserved</td><td colspan="4">E1 – Current Rate in Use</td></tr>
<tr><td colspan="8">E1 - Rate Consumption Register – MSB</td></tr>
<tr><td colspan="8">E1 - Rate Consumption Register</td></tr>
<tr><td colspan="8">E1 - Rate Consumption Register</td></tr>
<tr><td colspan="8">E1 - Rate Consumption Register – LSB</td></tr>
<tr><td colspan="8">E1 – Time for Next Rate – Hours</td></tr>
<tr><td colspan="8">E1 – Time for Next Rate – Minutes</td></tr>
<tr><td colspan="8">E1 – Time for Next Rate – Seconds</td></tr>
<tr><td colspan="4">Reserved</td><td colspan="4">E2 – Current Rate in Use</td></tr>
<tr><td colspan="8">E2 - Rate Consumption Register – MSB</td></tr>
<tr><td colspan="8">E2 - Rate Consumption Register</td></tr>
<tr><td colspan="8">E2 - Rate Consumption Register</td></tr>
<tr><td colspan="8">E2 - Rate Consumption Register – LSB</td></tr>
</table>

Dual (1 bit)

Single Element = 0, Two Elements = 1.

If single element the E2 fields are skipped in the frame. E1 – Time for Next Rate – Seconds will be the last byte of the message and the number of data bytes will be 9.

If two elements the E2 fields are present and the number of data bytes will be 14.

Total Number of Import Rates Supported (7 bits)

Field specifies the number of import rates E1 (and E2) supported by the meter. Range of legal decimal values are 1…8. No units used. The decimal values 0 and 9…15 are reserved and MUST be ignored by receiving devices.

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

E1 – Current Rate in Use (8 bits)

Field specifies the current rate in use. Range of legal decimal values are 1…8. No units used. The decimal values 0 and 9…15 are reserved and MUST be ignored by receiving devices.

E1 – Rate Consumption Register (32 bits)

The meter has a 32-bit consumption register, for the energy used in each rate. This register is the rate consumption register for the current rate in use now on element 1. Units are in Wh.

E1 – Time for Next Rate – Hours (8 bits)

Field specifies the hour value of the time that the rate is due to change on element 1. Range of legal decimal values are 0…23, or 255. The values 24…254 are reserved and MUST be ignored by receiving devices.

E1 – Time for Next Rate – Minutes (8 bits)

Field specifies the minute value of the time that the rate is due to change on element 1. Range of legal decimal values are 0…59, or 255. The decimal values 60…254 are reserved and MUST be ignored by receiving devices.

E1 – Time for Next Rate – Seconds (8 bits)

Field specifies the second value of the time that the rate is due to change on element 1. Range of legal decimal values are 0…59, or 255. The decimal values 60…254 are reserved and MUST be ignored by receiving devices.

NOTE: 255 in each field of the Time to Next Rate specifies no switching time is in use, which is appropriate for single rate meters. 255 is only a legal value if used in all three Time to Next Rate fields.

E2 – Current Rate in Use (8 bits)

Field specifies the current rate in use on element 2. Range of legal decimal values are 1…8. No units used. The decimal values 0 and 9…15 are reserved and MUST be ignored by receiving devices.

E2 – Rate Consumption Register (32 bits)

The meter has a 32-bit consumption register, for the energy used in each rate. This register is the rate consumption register for the current rate in use now on element 2. Units are in Wh.

4.21 Basic Window Covering Command Class, version 1 [OBSOLETED]

THIS COMMAND CLASS HAS BEEN OBSOLETED

New implementations MUST NOT support this command class.

New window covering device implementations SHOULD support the Window Covering Command Class.

This section contains Commands that may be used to control a Basic Window Covering Command Class.

4.21.1 Basic Window Covering Start Level Change Command

The Basic Window Covering Start Level Change Command used to start moving drapes, shades, blinds in a given direction. The speed of the movement is implementation specific.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_BASIC_WINDOW_COVERING							
Command = BASIC_WINDOW_COVERING_START_LEVEL_CHANGE							
Reserved	Open/ Close	Reserved					

Open/Close (1 bit)

If the Open/Close bit is set to 0 the window covering SHOULD open. If field is set to 1 the window covering SHOULD close.

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

4.21.2 Basic Window Covering Stop Level Change Command

The Basic Window Covering Stop Level Change Command used to stop moving drapes, shades, blinds in a given direction.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_BASIC_WINDOW_COVERING							
Command = BASIC_WINDOW_COVERING_STOP_LEVEL_CHANGE							

4.22 Battery Command Class, version 1

The Battery Command Class is used to request and report battery levels for a given device.

4.22.1 Battery Level Get Command

The Battery Level Get Command is used to request the level of a battery.

The Battery Level Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_BATTERY							
Command = BATTERY_GET							

4.22.2 Battery Level Report Command

The Battery Level Report Command used to report the battery level of a battery operated device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_BATTERY							
Command = BATTERY_REPORT							
Battery Level							

Battery Level (8 bits)

The battery level reports percentage of the full battery. The field can take values from 0 to 100% (0x00 – 0x64). The value 0xFF indicates a battery low warning.

4.23 Binary Sensor Command Class, version 1 [DEPRECATED]

THIS COMMAND CLASS HAS BEEN DEPRECATED

A device MAY implement this command class, but it is RECOMMENDED that new implementations use the Alarm Command Class.
If implementing this command class, it is RECOMMENDED that the Alarm Command Class is also implemented.

The Binary Sensor Command Class is used to realize binary sensors, such as movement sensors.

4.23.1 Binary Sensor Get Command

The Binary Sensor Get Command may be used to request the status of a sensor.

The Binary Sensor Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SENSOR_BINARY							
Command = SENSOR_BINARY_GET							

4.23.2 Binary Sensor Report Command

This command is used to advertise a sensor value.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SENSOR_BINARY							
Command = SENSOR_BINARY_REPORT							
Sensor Value							

Sensor Value (8 bits)

If the Sensor Value is 0x00 indicates that the sensor is idle and 0xFF indicates that the sensor has detected an event.

4.24 Binary Sensor Command Class, version 2 [DEPRECATED]

THIS COMMAND CLASS HAS BEEN DEPRECATED

A device MAY implement this command class, but it is RECOMMENDED that new implementations use the Alarm Command Class.
If implementing this command class, it is RECOMMENDED that the Alarm Command Class is also implemented.

The Binary Sensor Command Class is used to realize binary sensors, such as movement sensors and door/window sensors. Version 2 of this command class is extended with the following functionalities:

- A “get-supported” mechanism for the controlling device to interview the binary sensor for its supported sensor types
- A list of defined sensor types capable of reporting a binary value

NOTE: A binary sensor is defined as a sensor unit capable of providing a binary value in the report i.e. that an event was “triggered” or “not triggered” and not other intermediate values. For sensor units providing multiple values the Multilevel Sensor Command Class is more suitable. The Binary Sensor Command Class can further advance through implementation of the Notification Command Class to communicate details about the event.

4.24.1 Binary Sensor Get Command

The Binary Sensor Get Command may be used to request the status of the specific sensor device.

The Binary Sensor Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SENSOR_BINARY							
Command = SENSOR_BINARY_GET							
Sensor Type							

Sensor Type (8 bits)

Sensor Type specifies what type of sensor this command originates from. Refer to the table below with respect to defined sensors. The sensor type value 0xFF returns the first found supported sensor type in the bit mask (starting from bit 0 in Bit Mask 1) by the Binary Sensor Supported Report. Sensor types are defined by the Z-Wave Alliance.

Sensor Type	Value
Reserved	0x00
General purpose	0x01
Smoke	0x02
CO	0x03
CO_2	0x04
Heat	0x05
Water	0x06
Freeze	0x07
Tamper	0x08
Aux	0x09
Door/Window	0x0A
Tilt	0x0B
Motion	0x0C
Glass Break	0x0D
Reserved	0x0E-0xFE
Return 1st Sensor Type on supported list	0xFF

4.24.2 Binary Sensor Report Command

This command is used to advertise a sensor value.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SENSOR_BINARY							
Command = SENSOR_BINARY_REPORT							
Sensor Value							
Sensor Type							

Sensor Value (8 bits)

Sensor Value = 0x00 indicates that the sensor is idle and 0xFF indicates that the sensor has detected an event.

4.24.3 Binary Sensor Get Supported Sensor Command

The Binary Sensor Get Supported Sensor Command is used to request the supported sensor types from the binary sensor device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SENSOR_BINARY							
Command = SENSOR_BINARY_SUPPORTED_GET_SENSOR							

4.24.4 Binary Sensor Supported Sensor Report Command

This command must be sent as requested by a received Binary Sensor Get Supported Sensor Command. This command indicates the supported sensor types of the binary sensor device in a bit mask format and MUST NOT be transmitted unsolicited.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SENSOR_BINARY							
Command = SENSOR_BINARY_SUPPORTED_SENSOR_REPORT							
Bit Mask 1							
…							
Bit Mask N							

Bit Mask (N bytes)

The Bit Mask fields describe the supported sensor types by the binary sensor device and refer to the Sensor Type table of Binary Sensor Report command.

- Bit 0 in Bit Mask 1 is not allocated to any Sensor Type and must therefore be set to zero.
- Bit 1 in Bit Mask 1 indicates if Sensor Type = 1 (General Purpose) is supported.
- Bit 2 in Bit Mask 1 indicates if Sensor Type = 2 (Smoke) is supported.
- Bit 3 in Bit Mask 1 indicates if Sensor Type = 3 (CO) is supported
- …

If the Sensor Type is supported the bit MUST be set to 1. If the Sensor Type is not supported the bit MUST be set to 0. Sensor Type = 0xFF (Return 1st Sensor Type on supported list) cannot be indicated by the Bit Masks.

Note: It is only necessary to transmit Bit Mask 1 and up to the Bit Mask N indicating the last supported sensor type. The number of Bit Mask fields transmitted MUST be determined from the length field in the frame.

4.25 Binary Switch Command Class, version 1

The Binary Switch Command Class is used to control devices with On/Off or Enable/Disable capability.

The Binary Switch Command Class is an actuator control command class. Refer to 3.7.

4.25.1 Binary Switch Set Command

The Binary Switch Set command, version 1 is used to set a binary value.
The device MAY apply a non-zero duration to the transition from one value to a new value.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_SWITCH_BINARY</td></tr>
<tr><td colspan="8">Command = SWITCH_BINARY_SET</td></tr>
<tr><td colspan="8">Value</td></tr>
</table>

Value (8 bits)

The encoding of the Value field MUST be according to Table 26.

Table 26, Binary Switch Set :: Value

Value	Level	State
0 (0x00)	0%	Off
1..99 (0x01..0x63)	100%	On
…	*reserved*	*reserved*
255 (0xFF)	100%	On

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

The above mapping of the Binary Switch Command Class Value allows a controlling device to control a mixed group of Binary Switch and Multilevel Switch devices via Basic Set commands. Devices implementing the Binary Switch CC turn On or Off while devices implementing the Multilevel Switch CC sets the specified level.

4.25.2 Binary Switch Get Command

The Binary Switch Get command, version 1 is used to request the status of a device with On/Off or Enable/Disable capability.

The Binary Switch Report command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing.
The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_BINARY							
Command = SWITCH_BINARY_GET							

4.25.3 Binary Switch Report Command

The Binary Switch Report command, version 1 is used to advertise the status of a device with On/Off or Enable/Disable capability.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_BINARY							
Command = SWITCH_BINARY_REPORT							
Value							

Value (8 bits)

The encoding of the Value field MUST be according to Table 27

Table 27, Binary Switch Report :: Value

Value	Level	State
0 (0x00)	0%	Off
…	*Reserved*	*Reserved*
254 (0xFE)	Unknown	Unknown
255 (0xFF)	100%	On

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

The Value field SHOULD advertise the current value of the device hardware; also while in transition to a new target value.

A controlling device MUST NOT assume that the Value is identical to a value previously issued with a Set command when a transition has ended.

4.26 Binary Switch Command Class, version 2

The Binary Switch Command Class is used to control devices with On/Off or Enable/Disable capability.

The Binary Switch Command Class is an actuator control command class. Refer to chapter 3.7.

Commands not described in Version 2 stays unchanged from Version 1.

4.26.1 Compatibility considerations

A device supporting Binary Switch CC, Version 2 MUST support Binary Switch CC, Version 1.

Version 2 adds duration and target value control and reporting.

4.26.2 Binary Switch Set Command

The Binary Switch Set command, version 2 is used to set a binary value in a supporting device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_BINARY							
Command = SWITCH_BINARY_SET							
Target Value							
Duration							

Target Value (8 bits)

The encoding of the Target Value field MUST be according to Table 26.

Table 28, Binary Switch Set :: Value

Value		Level	State
0	(0x00)	0%	Off
1..99	(0x01..0x63)	100%	On
…		*reserved*	*reserved*
255	(0xFF)	100%	On

All other values are reserved. Reserved values MUST NOT be used by a sending node and MUST be ignored by a receiving node.

The above mapping of the Binary Switch Command Class Value allows a controlling device to control a mixed group of Binary Switch and Multilevel Switch devices via Basic Set commands. Devices implementing the Binary Switch CC turn On or Off while devices implementing the Multilevel Switch CC sets the specified level.

Duration (8 bits)

The duration field MUST specify the duration of a transition from the current value to the Target Value. A supporting device SHOULD respect the specified Duration value.

The encoding of the Duration value MUST be according to Table 29.

Table 29, Binary Switch Set :: Duration

Duration	Description
0x00	Instantly
0x01-0x7F	1 second (0x01) to 127 seconds (0x7F) in 1 second resolution.
0x80-0xFE	1 minute (0x80) to 127 minutes (0xFE) in 1 minute resolution.
0xFF	Factory default duration.

4.26.3 Binary Switch Report Command

The Binary Switch Report command, version 2 is used to advertise the status of a device with On/Off or Enable/Disable capability.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_BINARY							
Command = SWITCH_BINARY_REPORT							
Current Value							
Target Value							
Duration							

Current Value (8 bits)

The encoding of the Current Value field MUST be according to Table 27

Table 30, Binary Switch Report :: Value

Value	Level	State
0 (0x00)	0%	Off
255 (0xFF)	100%	On

All other values are reserved. Reserved values MUST NOT be used by a sending node and MUST be ignored by a receiving node.

The advertised Current Value MUST NOT be updated to the Target Value before the hardware actuator has actually reached the Target Value.

Target Value (8 bits)

The Target Value field MUST advertise the target value of an ongoing transition or the most recent transition.

The encoding of the Target Value field MUST be according to Table 27.

All other values are reserved. Reserved values MUST NOT be used by a sending node and MUST be ignored by a receiving node.

Duration (8 bits)

The duration field SHOULD advertise the duration of a transition from the Current Value to the Target Value.

The encoding of the Duration field MUST be according toTable 31.

Table 31, Binary Switch Report :: Duration

Duration	Description
0x00	0 seconds. Already at the Target Value.
0x01-0x7F	1 second (0x01) to 127 seconds (0x7F) in 1 second resolution.
0x80-0xFD	1 minute (0x80) to 126 minutes (0xFD) in 1 minute resolution.
0xFE	Unknown duration
0xFF	Reserved

All other values are reserved. Reserved values MUST NOT be used by a sending node and MUST be ignored by a receiving node.

4.27 Binary Toggle Switch Command Class, version 1 [DEPRECATED]

THIS COMMAND CLASS HAS BEEN DEPRECATED

A device MAY implement this command class, but it is RECOMMENDED that new implementations use the Binary Switch Command Class.
If implementing this command class, it is RECOMMENDED that the Binary Switch Command Class is also implemented.

The Binary Toggle Switch Command Class is used for toggle-style control of binary actuator devices.

4.27.1 Binary Toggle Switch Set Command

The Binary Toggle Switch Set Command may be used to toggle a device e.g. from on to off and from off to on.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_TOGGLE_BINARY							
Command = SWITCH_TOGGLE_BINARY_SET							

4.27.2 Binary Toggle Switch Get Command

The Binary Switch Get Command may be used to request the state of the load controlled by the device.

The Binary Toggle Switch Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing.
The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_TOGGLE_BINARY							
Command = SWITCH_TOGGLE_BINARY_GET							

4.27.3 Binary Toggle Switch Report Command

This command is used to advertise the value of a toggle switch.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_TOGGLE_BINARY							
Command = SWITCH_TOGGLE_BINARY_REPORT							
Value							

Value (8 bits)

The value MAY be 0x00 (off) or 0xFF (on).

4.28 Climate Control Schedule Command Class, version 1 [DEPRECATED]

THIS COMMAND CLASS HAS BEEN DEPRECATED

A device MAY implement this command class, but it is RECOMMENDED that new implementations use the Schedule Command Class.
If implementing this command class, it is RECOMMENDED that the Schedule Command Class is also implemented.

The Climate Control Schedule Command Class allows devices to exchange schedules and overrides, which specify when to perform a setback on the setpoint.

Note: The setpoint is the temperature a device will try to maintain. The setback is a deviation from the setpoint. When a setback is in use the device will apply the setback to the setpoint, resulting in a different temperature. When using schedules and overrides it is possible to define several setbacks occurring at specific times.

Schedules are exchanged using the Schedule Commands.
Overrides of schedules are exchanged using the Schedule Override Commands.
Detection of updated schedules is done using the Schedule Changed Commands.

The Climate Control Schedule uses the Schedule State type to define each setback. The Schedule State type has the following format:

7	6	5	4	3	2	1	0
Schedule State							

Schedule State (8 bits)

The values are as follows:

Schedule State		Description
Hexadecimal	**Decimal**	
0x80 … 0xFF 0x00 0x01 … 0x78	-128 … -1 0 1 … 120	The setback in 1/10 degrees (Kelvin) Example: 0 = 0 degrees setback 1 = 0.1 degrees is added to the setpoint 2 = 0.2 degrees is added to the setpoint -1 = 0.1 degrees is subtracted from the setpoint -2 = 0.2 degrees is subtracted from the setpoint
0x79	121	Frost Protection
0x7A	122	Energy Saving Mode
0x7B – 0x7E	123 – 126	Reserved
0x7F	127	Unused State

When converting between Celsius and Fahrenheit proper rounding MUST be applied with at least two decimals in the internal calculations of a device to avoid rounding errors.

When displaying converted Fahrenheit values it is RECOMMENDED that the displayed value is rounded to nearest quarter of a degree.

4.28.1 Schedule Set Command

The Schedule Set Command used to set the climate control schedule in a device for a specific weekday. A climate control schedule defines when to use a setback on the setpoint in a device. A schedule can hold a maximum of 9 switchpoints. A switchpoint defines one setback from the current setpoint.

The entire list of switchpoints in the Command MUST be ordered by time, ascending from 00:00 towards 23:59. Switchpoints which have a Schedule State set to “Unused” MUST be placed last. Duplicates MUST NOT be allowed for Switchpoints which have a Schedule State different from “Unused”.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CLIMATE_CONTROL_SCHEDULE							
Command = SCHEDULE_SET							
Reserved					Weekday		
Switchpoint 0 Byte 1							
Switchpoint 0 Byte 2							
Switchpoint 0 Byte 3							
Switchpoint 1 Byte 1							
Switchpoint 1 Byte 2							
Switchpoint 1 Byte 3							
…							
Switchpoint 8 Byte 1							
Switchpoint 8 Byte 2							
Switchpoint 8 Byte 3							

Weekday (3 bits)

The possible values are:

Binary	Decimal	Description
0b000	0	Reserved
0b001	1	Monday
0b010	2	Tuesday
0b011	3	Wednesday
0b100	4	Thursday
0b101	5	Friday
0b110	6	Saturday
0b111	7	Sunday

Switchpoint (24 bits)

7	6	5	4	3	2	1	0	
Reserved			Hour					Byte 1
Reserved		Minute						Byte 2
Schedule State								Byte 3

Hour (5 bits):

Specifies the hour of the of the actual switchpoint. Possible values are:

Binary	Decimal	Description
0b00000	0	0 hour
0b00001	1	1st hour
0b00010	2	2nd hour
…	…	…
0b10111	23	23rd hour
0b11000	24	Reserved
…	…	Reserved
0b11111	31	Reserved

Minute (6 bits):

Specifies the minute of the actual switchpoint. Possible values are:

Binary	Decimal	Description
0b000000	0	0 minute
0b000001	1	1st minute
0b000010	2	2nd minute
…	…	…
0b111011	59	59th minute
0b111100	60	Reserved
…	…	Reserved
0b111111	63	Reserved

Schedule State (8 bits):

Schedule State uses the Schedule State type format, see section 4.27.3. If Schedule State has the value of “Unused”, then the Hour and Minute field MUST be ignored. Once a Schedule State of “Unused” is encountered, the parsing of switchpoints must stop.

4.28.2 Schedule Get Command

The Schedule Get Command is used to request the climate control schedule in a device for a specific weekday.

The Schedule Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_CLIMATE_CONTROL_SCHEDULE</td></tr>
<tr><td colspan="8">Command = SCHEDULE_GET</td></tr>
<tr><td colspan="5">Reserved</td><td colspan="3">Weekday</td></tr>
</table>

Weekday (3 bits)

The possible values are:

Binary	Decimal	Description
0b000	0	Reserved
0b001	1	Monday
0b010	2	Tuesday
0b011	3	Wednesday
0b100	4	Thursday
0b101	5	Friday
0b110	6	Saturday
0b111	7	Sunday

4.28.3 Schedule Report Command

The Schedule Report Command used to report the climate control schedule in a device for a specific weekday.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_CLIMATE_CONTROL_SCHEDULE</td></tr>
<tr><td colspan="8">Command = SCHEDULE_REPORT</td></tr>
<tr><td colspan="5">Reserved</td><td colspan="3">Weekday</td></tr>
<tr><td colspan="8">Switchpoint 0 Byte 1</td></tr>
<tr><td colspan="8">Switchpoint 0 Byte 2</td></tr>
<tr><td colspan="8">Switchpoint 0 Byte 3</td></tr>
<tr><td colspan="8">Switchpoint 1 Byte 1</td></tr>
<tr><td colspan="8">Switchpoint 1 Byte 2</td></tr>
<tr><td colspan="8">Switchpoint 1 Byte 3</td></tr>
<tr><td colspan="8">…</td></tr>
<tr><td colspan="8">Switchpoint 8 Byte 1</td></tr>
<tr><td colspan="8">Switchpoint 8 Byte 2</td></tr>
<tr><td colspan="8">Switchpoint 8 Byte 3</td></tr>
</table>

Weekday (3 bits)

The possible values are:

Binary	Decimal	Description
0b000	0	Reserved
0b001	1	Monday
0b010	2	Tuesday
0b011	3	Wednesday
0b100	4	Thursday
0b101	5	Friday
0b110	6	Saturday
0b111	7	Sunday

Switchpoint (24 bits)

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th><th></th></tr>
<tr><td colspan="3">Reserved</td><td colspan="5">Hour</td><td>Byte 1</td></tr>
<tr><td colspan="2">Reserved</td><td colspan="6">Minute</td><td>Byte 2</td></tr>
<tr><td colspan="8">Schedule State</td><td>Byte 3</td></tr>
</table>

Hour (5 bits):

Specifies the hour of the actual switchpoint. Possible values are:

Binary	Decimal	Description
0b00000 0b00001 0b00010 … 0b10111 0b11000 … 0b11111	0 1 2 … 23 24 … 31	0 hour 1st hour 2nd hour … 23rd hour Reserved Reserved Reserved

Minute (6 bits):

Specifies the minute of the actual switchpoint. Possible values are:

Binary	Decimal	Description
0b000000 0b000001 0b000010 … 0b111011 0b111100 … 0b111111	0 1 2 … 59 60 … 63	0 minute 1st minute 2nd minute … 59th minute Reserved Reserved Reserved

Schedule State (8 bits):

Schedule State uses the Schedule State type format, see section 4.27.3.
If Schedule State has the value of "Unused", then the Hour and Minute field MUST be ignored. Once a Schedule State of "Unused" is encountered, the parsing of switchpoints must stop.

4.28.4 Schedule Changed Get Command

The Schedule Changed Get Command used to check if the climate control schedule has changed.

The Schedule Changed Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_CLIMATE_CONTROL_SCHEDULE</td></tr>
<tr><td colspan="8">Command = SCHEDULE_CHANGED_GET</td></tr>
</table>

4.28.5 Schedule Changed Report Command

The Schedule Changed Report Command used to report if the climate control schedule has changed.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CLIMATE_CONTROL_SCHEDULE							
Command = SCHEDULE_CHANGED_REPORT							
ChangeCounter							

ChangeCounter (8 bits)

The ChangeCounter is a timestamp for a climate control schedule and it is kept in devices which exchange climate control schedules.

One device holds a climate control schedule and other devices uses this climate control schedule. Whenever the climate control schedule changes, the device MUST update its ChangeCounter, and the other devices MUST regularly use the Schedule Changed Get Command on the device which holds the Climate Control Schedule to see if the ChangeCounter is different from last time – indicating a change in a climate control schedule.

The possible values are:

Hexadecimal	Decimal	Description
0x00	0	The climate control schedule change mechanism is temporarily disabled by the override function.
0x01 … 0xFF	1 - 255	The climate control schedule change mechanism is enabled. ChangeCounter is incremented by one every time climate control schedule changes. When ChangeCounter eventually reach 0xFF, then the next increment, will rollover to 0x01.

When a device is fresh and has no climate control schedule it MUST retrieve a climate control schedule using the Schedule Get Command and it MUST also use the Schedule Changed Get to get the first copy of the current ChangeCounter, thus avoiding getting the climate control schedule initially twice.

When a device is awake after sleep mode it SHOULD use this Command to detect if the schedule has been changed.

4.28.6 Schedule Override Set Command

The Schedule Override Set Command used to set the override in a device.

The purpose of an override is to inform a device to ignore its current climate control schedule and assume the setting provided by the Override Type and Override State fields.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CLIMATE_CONTROL_SCHEDULE							
Command = SCHEDULE_OVERRIDE_SET							
Reserved						Override Type	
Override State							

Override Type (2 bits)

The override type field can assume the following values:

Binary	Decimal	Description
0b00	0	No override
0b01	1	Temporary override
0b10	2	Permanent override
0b11	3	Reserved

Note: The difference between a temporary and a permanent override is that a temporary override only overrides the current switchpoint in the climate control schedule.

Both temporary and permanent overrides MAY be cancelled in the device, which receives the SCHEDULE_OVERRIDE_SET. This cancellation MUST be notified in an unsolicited SCHEDULE_OVERRIDE_REPORT as specified in the following sequence diagram:

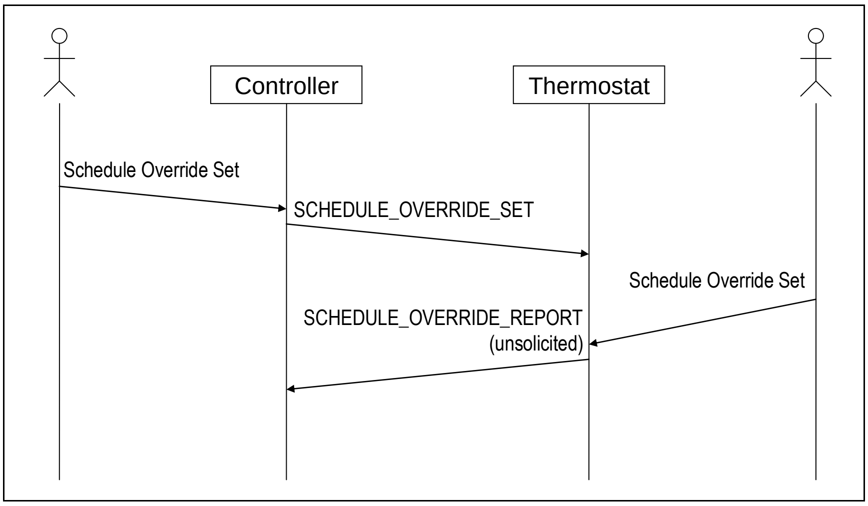


Figure 17, Sequence diagram for cancellation of a Schedule Override Set

Override State (8 bits)

The Override State uses the Schedule State type format, see section 4.27.30

4.28.7 Schedule Override Get Command

The Schedule Override Get Command is used to request the override, currently in use in a device.

The Schedule Override Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing.
The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CLIMATE_CONTROL_SCHEDULE							
Command = SCHEDULE_OVERRIDE_GET							

4.28.8 Schedule Override Report Command

The Schedule Override Report Command used to report the override, currently in use in a device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CLIMATE_CONTROL_SCHEDULE							
Command = SCHEDULE_OVERRIDE_REPORT							
Reserved						Override Type	
Override State							

Override Type (2 bits)

The override type field can assume the following values:

Binary	Decimal	Description
0b00	0	No override
0b01	1	Temporary override
0b10	2	Permanent override
0b11	3	Reserved

Note: The difference between a temporary and a permanent override is that a temporary override only overrides the current switchpoint in the climate control schedule.

Both temporary and permanent overrides MAY be cancelled in the device, which receives the SCHEDULE_OVERRIDE_SET. This cancellation MUST be notified in an unsolicited SCHEDULE_OVERRIDE_REPORT as shown on figure in section 4.28.6

Override State (8 bits)

The Override State uses the Schedule State type format, see section 4.27.3

4.29 Central Scene Command Class, version 1

The Central Scene Command Class is used to communicate central scene activations to a central controller using the lifeline concept. The central scene controller only need to configure lifeline association in relevant nodes before the home control application can take action in the central scene controller. The typical application contains up to 10-15 selected nodes creating a nice “out of the box” experience with minimum effort by the customer. A scene is typically activated via a push button on the device in question.

The Central Scene Command Class can instruct the central scene controller to perform the relevant actions as shown on Figure 18.

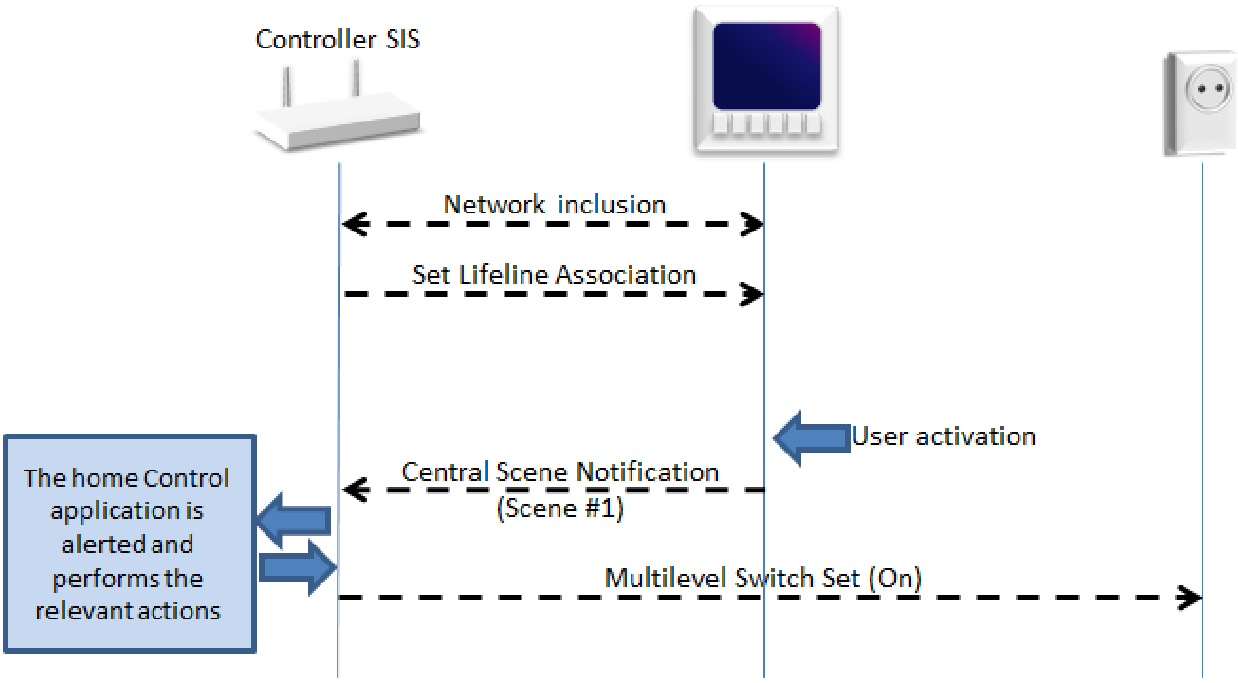


Figure 18, Centralized feedback and control

Multiple sending nodes MAY advertise the same Scene number.
Therefore, a receiving node MUST interpret the advertised Scene number in combination with the source NodeID of the sending node.

For instance, {Node A, Scene 1} may trigger changes to another scene than {Node B, Scene 1} in the receiving node

Notice that this configuration has a single point of failure in case the central scene controller is broken. The likelihood to experience the “popcorn effect” is increased because the central scene controller may be located far from the push button activated and therefore out of direct range with respect to devices to control.

To compensate for single point of failure in case the central scene controller is broken the devices could do fallback to alternative association groups to control devices directly. These association groups are activated only if the device does not receive acknowledge from configured lifeline. This will ensure some basic level of light control in the house until the central scene controller is restored. Alternatively, a backup central scene controller could be added to the system to take over in case the first one fails.

4.29.1 Central Scene Supported Get Command

The Central Scene Supported Get Command is used to request the maximum number of scenes that this device supports.

The Central Scene Supported Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing.
The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CENTRAL_SCENE							
Command = CENTRAL_SCENE_SUPPORTED_GET							

4.29.2 Central Scene Supported Report Command

The Central Scene Supported Report Command is used to report the maximum number of scenes that the requested device supports.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CENTRAL_SCENE							
Command = CENTRAL_SCENE_SUPPORTED_REPORT							
Supported Scenes							

Supported Scenes (8 bits)

This field indicates the maximum number of scenes supported by the requested device.

This field MUST advertise the number of available scenes.
Scenes MUST be numbered in the range 1..[Supported Scenes].

A device MAY implement virtual buttons via two-button presses, touch screen swipes or other means.
Virtual buttons MUST be numbered after physical push buttons or graphical button objects.

The advertised number of Supported Scenes MUST cover physical push buttons, graphical button objects as well as virtual buttons. Therefore, the advertised number of Supported Scenes MAY be larger than the number of physical push buttons or graphical button objects.

4.29.3 Central Scene Notification Command

The Central Scene Notification Command ,if enabled, is used to report activated scene on device in question including how notification must be interpreted.

In case a lifeline is configured in group #1 using Association Command Class then device must use this lifeline for transmitting the Central Scene Notification Command. It is allowed to define other association groups handling Central Scene Notification Command and setup rules between the groups. However, lifeline has precedence over other groups handling Central Scene Notification Command in case lifeline is defined.

NOTICE: The Central Scene Notification Command is by default enabled when device supporting it is included into a network.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_CENTRAL_SCENE</td></tr>
<tr><td colspan="8">Command = CENTRAL_SCENE_NOTIFICATION</td></tr>
<tr><td colspan="8">Sequence Number</td></tr>
<tr><td colspan="5">Reserved</td><td colspan="3">Key Attributes</td></tr>
<tr><td colspan="8">Scene Number</td></tr>
</table>

Sequence Number (8 bits)

The sequence number is incremented each time a Central Scene Notification Command is issued. The receiving device uses the sequence number to ignore duplicates.

Key Attributes (3 bits)

The key Attributes field specifies the state of the key. The field MUST be encoded according to Table 32

Table 32, Central Scene Notification :: Key Attributes

Key Attribute	Description
0x00	Key Pressed
0x01	Key Released
0x02	Key Held Down

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Notifications carrying these Key Attributes MUST comply with Figure 19.

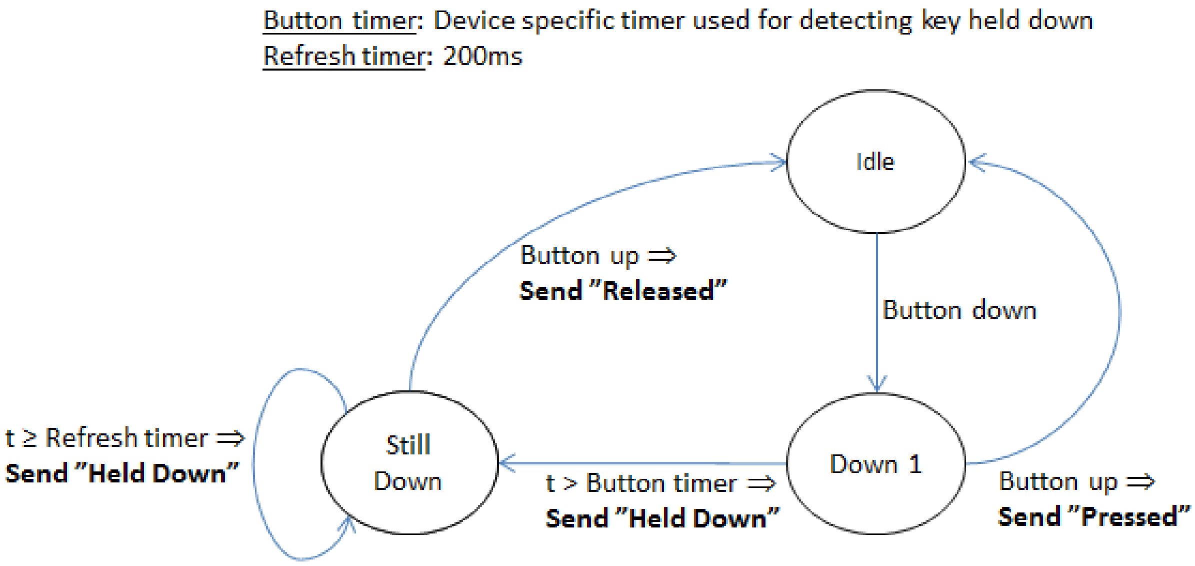


Figure 19, Central Scene version 1 button press decoding and timer management

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Scene Number (8 bits)

The advertised number of Supported Scenes MUST cover physical push buttons, graphical button objects as well as virtual buttons. Therefore, the advertised number of Supported Scenes MAY be larger than the number of physical push buttons or graphical button objects.

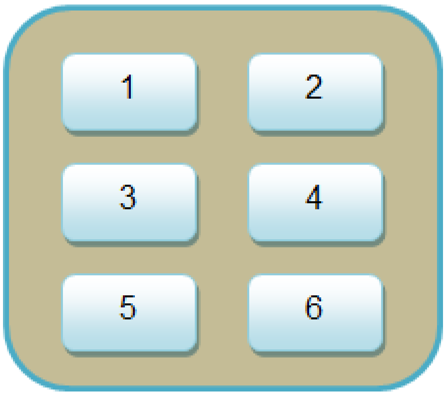


Figure 20, Scene number mapping to button layout

4.30 Central Scene Command Class, version 2

The Central Scene Command Class is used to communicate central scene activations to a central controller using the lifeline concept.

4.30.1 Compatibility considerations

Central Scene Command Class version 2 is extended on the following areas:

- The Central Scene Supported Report Command is updated to advertise Key Attributes support for each scene.
- Additional Key Attributes are defined.

Commands and paragraphs not mentioned in this version remain unchanged from version 1.

4.30.2 Central Scene Supported Report Command

The Central Scene Supported Report Command is used to report the maximum number of supported scenes and the Key Attributes supported for each scene.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_CENTRAL_SCENE</td></tr>
<tr><td colspan="8">Command = CENTRAL_SCENE_SUPPORTED_REPORT</td></tr>
<tr><td colspan="8">Supported Scenes</td></tr>
<tr><td colspan="5">Reserved</td><td colspan="2">Number of Bit Mask Bytes</td><td>Identical</td></tr>
<tr><td colspan="8">Supported Key Attributes for Scene 1 – Byte 1</td></tr>
<tr><td colspan="8">…..</td></tr>
<tr><td colspan="8">Supported Key Attributes for Scene 1 – Byte N</td></tr>
<tr><td colspan="8">Supported Key Attributes for Scene M – Byte 1</td></tr>
<tr><td colspan="8">…..</td></tr>
<tr><td colspan="8">Supported Key Attributes for Scene M – Byte N</td></tr>
</table>

Fields not described below remain unchanged from version 1.

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Number of Bit Mask Bytes (2 bits)

This field advertises the size of each “Supported Key Attributes” field measured in bytes.
The value MUST be in the range 1..3.

Identical (1 bit)

This field indicates if all scenes are supporting the same Key Attributes:

The value 1 MUST indicate that all scenes support the same set of Key Attributes. In this case, the field "Supported Key Attributes for Scene 1" MUST advertise the supported Key Attributes for all scenes

The value 0 MUST indicate that scenes support different Key Attributes. In this case, Supported Key Attributes MUST be advertised for each individual scene.

Supported Key Attributes (N bytes)

This Bit Mask field advertises the attributes supported by the corresponding scene. The field MUST be encoded according to Table 33.

Table 33: Central Scene Supported Report::Supported Key Attributes

Byte:Bit	Supported Key Attributes
1:0	Key Pressed 1 time
1:1	Key Released.
1:2	Key Held Down.
1:3	Key Pressed 2 times
1:4	Key Pressed 3 times
1:5	Key Pressed 4 times
1:6	Key Pressed 5 times
1:7	Reserved

If the Key Released attribute is supported, the Key Held Down attribute MUST also be supported.
If the Key Held Down attribute is supported, the Key Released attribute MUST also be supported.

4.30.3 Central Scene Notification Command

The Central Scene Notification Command is used to advertise one or more key events.

If the device implements Z-Wave Plus Lifeline support is implemented, the device MUST send the Central Scene Notification Command to the actual Lifeline targets.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_CENTRAL_SCENE</td></tr>
<tr><td colspan="8">Command = CENTRAL_SCENE_NOTIFICATION</td></tr>
<tr><td colspan="8">Sequence Number</td></tr>
<tr><td colspan="5">Reserved</td><td colspan="3">Key Attributes</td></tr>
<tr><td colspan="8">Scene Number</td></tr>
</table>

Fields not described below remain unchanged from version 1.

Key Attributes (3 bits)

This field advertises one or more events detected by the key. The field MUST be encoded according to Table 34

Table 34: Central Scene Notification::Key Attributes

Key Attribute	Description
0x00	Key Pressed 1 time
0x01	Key Released
0x02	Key Held Down
0x03	Key Pressed 2 times
0x04	Key Pressed 3 times
0x05	Key Pressed 4 times
0x06	Key Pressed 5 times

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Notification carrying these Key Attributes MUST comply with Figure 21.

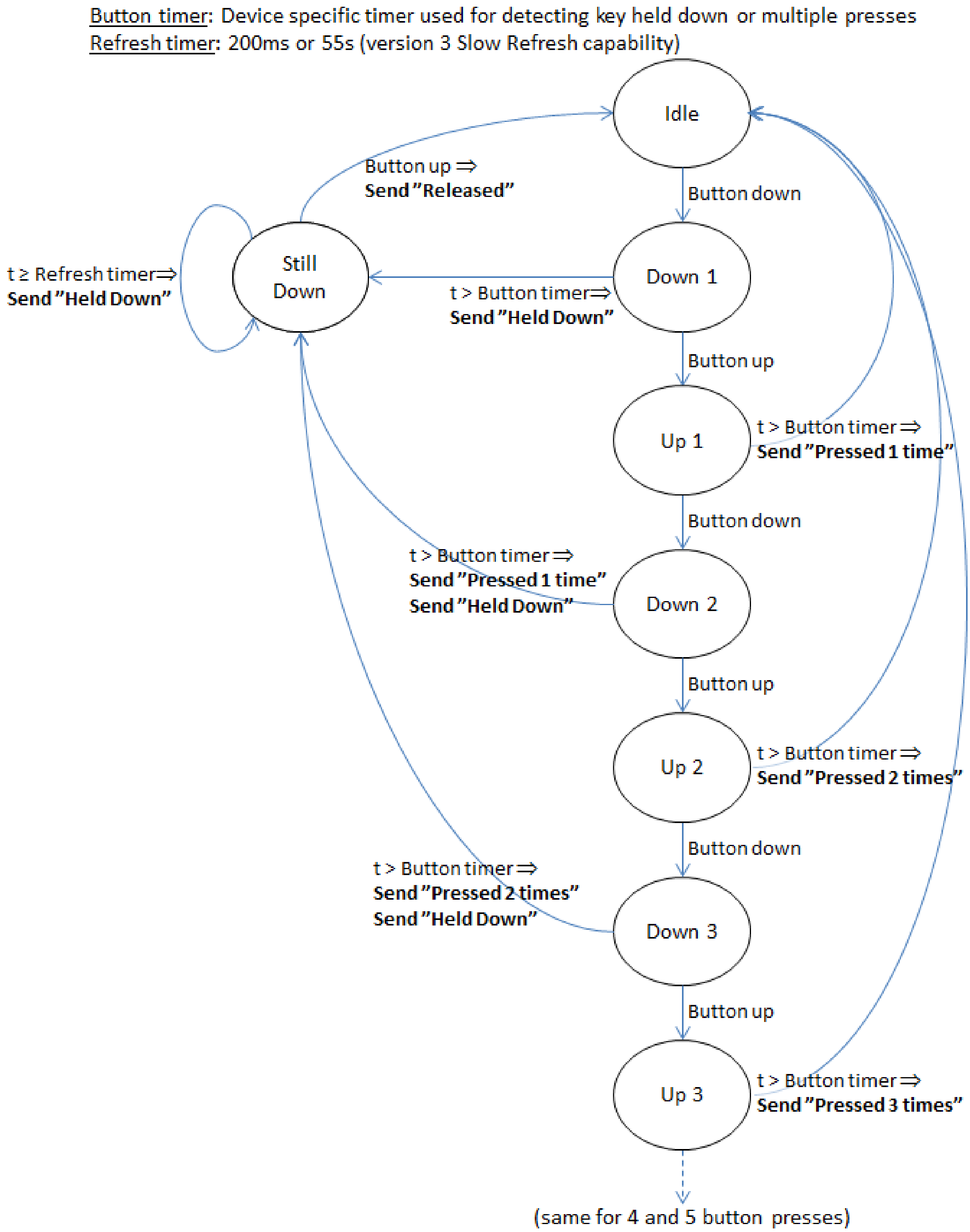


Figure 21, Central Scene button press decoding and timer management

4.31 Central Scene Command Class, version 3

The Central Scene Command Class, version 3 extends version 2 by adding a new mode for the transmission of Held Down Key notifications. This allows a node to send Key Held Down refreshes at a low rate for the duration of the key held down.

The purpose of this new feature is to provide improved compatibility with FLiRS beaming.

Central Scene Command Class, version 3 introduces three new commands to allow a controller to query and configure node capabilities.

- Configuration Set Command
- Configuration Get Command
- Configuration Report Command

Central Scene Command Class, version 3 extends the following commands for a scene launching node to advertise its optional capabilities.

- Supported Report Command
- Notification Command

Commands not mentioned in this version remain unchanged from version 2.

4.31.1 Compatibility considerations

A device implementing Central Scene Command Class, version 3 MUST also implement Central Scene Command Class, version 2.

Central Scene Command Class, version 3 is backwards compatible with Central Scene Command Class, version 2.

A scene controller SHOULD NOT try to detect multiple key presses from the reception of successive key press notifications in any single or multiple key attribute combination. Beaming, collisions or routing may add delays between messages and render the timing between notifications unreliable.

For backwards compatibility with previous versions, the Slow Refresh capability MUST be disabled by default after inclusion. For details about the Slow Refresh capability, refer to 4.31.6.

When creating associations from a version 3 node, the node creating the association MUST enable the Slow Refresh capability in the node if the association destination does also support Central Scene Command Class, version 3.

When using the version 3 Slow Refresh capability, the controller relies on the reception of a Key Up notification instead of the Key Held Down refreshes. The new mode still sends refreshes at a low rate, for the controller to detect if a node failed after sending a Held Down Key notification.

4.31.2 Central Scene Configuration Set Command

The Central Scene Configuration Set Command is used to configure the use of optional node capabilities for scene notifications.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CENTRAL_SCENE							
Command = CENTRAL_SCENE_CONFIGURATION_SET							
Slow Refresh	Reserved						

Slow Refresh (1 bit)

This flag is used to configure the use of the Slow Refresh capability.

The value 1 MUST indicate that the scene launching node MUST use Slow Refresh.

The value 0 MUST indicate that the scene launching node MUST NOT use Slow Refresh.

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

4.31.3 Central Scene Configuration Get Command

The Central Scene Configuration Get Command is used to query the configuration of optional node capabilities for scene notifications.

The Central Scene Configuration Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CENTRAL_SCENE							
Command = CENTRAL_SCENE_CONFIGURATION_GET							

4.31.4 Central Scene Configuration Report Command

The Central Scene Configuration Report Command is used to advertise the configuration of optional node capabilities for scene notifications.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_CENTRAL_SCENE</td></tr>
<tr><td colspan="8">Command = CENTRAL_SCENE_CONFIGURATION_REPORT</td></tr>
<tr><td>Slow Refresh</td><td colspan="7">Reserved</td></tr>
</table>

Slow Refresh (1 bit)

This flag is used to advertise the use of the Slow Refresh capability.

The value 1 MUST indicate that the scene launching node MUST use Slow Refresh.

The value 0 MUST indicate that the scene launching node MUST NOT use Slow Refresh capability.

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

4.31.5 Central Scene Supported Report Command

The Central Scene Supported Report Command is used to report the maximum number of supported scenes and the Key Attributes supported for each scene.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_CENTRAL_SCENE</td></tr>
<tr><td colspan="8">Command = CENTRAL_SCENE_SUPPORTED_REPORT</td></tr>
<tr><td colspan="8">Supported Scenes</td></tr>
<tr><td>Slow Refresh Support</td><td colspan="4">Reserved</td><td colspan="2">Number of Bit Mask Bytes</td><td>Identical</td></tr>
<tr><td colspan="8">Supported Key Attributes for Scene 1 – Byte 1</td></tr>
<tr><td colspan="8">…..</td></tr>
<tr><td colspan="8">Supported Key Attributes for Scene 1 – Byte N</td></tr>
<tr><td colspan="8">Supported Key Attributes for Scene M – Byte 1</td></tr>
<tr><td colspan="8">…..</td></tr>
<tr><td colspan="8">Supported Key Attributes for Scene M – Byte N</td></tr>
</table>

All fields not described below are the same as in version 2.

Slow Refresh Support (1 bit)

This field indicates whether the node supports the Slow Refresh capability.

The value 1 MUST indicate that the node supports the Slow Refresh capability.

The value 0 MUST indicate that the node does not support the Slow Refresh capability.

4.31.6 Central Scene Notification Command

The Central Scene Notification Command is used to advertise a key event.

If the device implements Z-Wave Plus Lifeline support, the device MUST send the Central Scene Notification Command to the actual Lifeline targets.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_CENTRAL_SCENE</td></tr>
<tr><td colspan="8">Command = CENTRAL_SCENE_NOTIFICATION</td></tr>
<tr><td colspan="8">Sequence Number</td></tr>
<tr><td>Slow Refresh</td><td colspan="4">Reserved</td><td colspan="3">Key Attribute</td></tr>
<tr><td colspan="8">Scene Number</td></tr>
</table>

All fields not described below are the same as in version 2.

Slow Refresh (1 bit)

This flag is used to advertise if the node is sending Key Held Down notifications at a slow rate. A sending node MUST always set this field according to the configured mode.
A receiving node MUST ignore this field if the command is not carrying the Key Held Down key attribute.

If the Slow Refresh field is 0:

- A new Key Held Down notification MUST be sent every 200ms until the key is released.
- The Sequence Number field MUST be updated at each notification transmission.
- If not receiving a new Key Held Down notification within 400ms after the most recent Key Held Down notification, a receiving node MUST respond as if it received a Key Release notification.

If the Slow Refresh field is 1:

- A new Key Held Down notification MUST be sent every 55 seconds until the key is released.
- The Sequence Number field MUST be updated at each notification refresh.
- If not receiving a new Key Held Down notification within 60 seconds after the most recent Key Held Down notification, a receiving node MUST respond as if it received a Key Release notification.

4.32 Clock Command Class, version 1

The Clock Command Class is used to implement a simple clock functionality.

4.32.1 Clock Set Command

The Clock Set Command may be used to set the clock in a device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CLOCK							
Command = CLOCK_SET							
Weekday			Hour				
Minute							

Weekday (3 bits)

The weekday field can take the following values 0 = Unused (24 hour clock), 1 = Monday, 2 = Tuesday, 3 = Wednesday, 4 = Thursday, 5 = Friday, 6 = Saturday and 7 = Sunday.

Hour (5 bits)

The hour field can take values from 0 to 23.

Minute (8 bits)

The minute field can take values from 0 to 59.

4.32.2 Clock Get Command

The Clock Get Command is used to request the clock report from a device.

The Clock Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CLOCK							
Command = CLOCK_GET							

4.32.3 Clock Report Command

The Clock Report Command used to report the actual weekday and clock in a device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CLOCK							
Command = CLOCK_REPORT							
Weekday			Hour				
Minute							

Weekday (3 bits)

The weekday field can take the following values 0 = Unused (24 hour clock), 1 = Monday, 2 = Tuesday, 3 = Wednesday, 4 = Thursday, 5 = Friday, 6 = Saturday and 7 = Sunday.

Hour (5 bits)

The hour field can take values from 0 to 23.

Minute (8 bits)

The minute field can take values from 0 to 59.

4.33 Color Switch Command Class, version 1

The Color Switch Command Class is used to control color capable devices.

The Color Switch Command Class manipulates the color components of a device. Each color component is scaled by the brightness level previously set by a Multilevel Switch Set or Basic Set command.

The Color Switch Command Class is an actuator control command class. Refer to 3.7.

4.33.1 Compatibility considerations

The Color Switch Command Class, version 1 was previously named the Color Control Command Class, version 1. The Color Switch Command Class, version 1 is binary compatible with the Color Control Command Class, version 1.

A device supporting the Color Switch Command Class, Version 1 MUST support the Multilevel Switch Command Class, version 2.

4.33.2 Interoperability requirements

The Color Switch Command Class MUST be treated as a separate command class.
Basic and Multilevel Switch Command Class commands MUST NOT affect color component levels controlled by the Color Switch Command Class.
The Color Switch Command Class commands MUST NOT affect the brightness level controlled by Basic and Multilevel Switch Command Classes.

If the brightness level is requested via a Basic Get or Multilevel Switch Get command, the reported value MUST reflect the brightness level previously set via Basic or Multilevel Switch commands.
If a color component is requested via the Color Switch Get command, the reported value MUST reflect the color component level previously set via Color Switch commands.

An implementation MUST scale color component levels by the brightness level . Thus, to achieve a resulting light level of 100% for a given color component, the color component level must be set to 100% (via the Color Switch Command Class) and the brightness level must be set to 100% (via Basic or Multilevel Switch Command Class).

A controlling device SHOULD specify the highest possible color component levels of a given color tone to achieve the highest light yield when scaled by the brightness level. The controlling device MAY however limit color component levels to achieve the same brightness for blended color tones as for pure colors. For this reason, a supporting device MUST NOT normalize color components manipulated by the Color Switch Command Class.
Likewise, a supporting device implementing indexed colors should reduce its internal color component levels for blended colors to achieve the same brightness for all color tones.

4.33.3 Color Switch Supported Get Command

The Color Switch Supported Get Command is used to request the supported color components of a device.

The Color Switch Supported Report command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_COLOR							
Command = SWITCH_COLOR_SUPPORTED_GET							

4.33.4 Color Switch Supported Report Command

The Color Switch Supported Report Command is used to report the supported color components of a device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_COLOR							
Command = SWITCH_COLOR_SUPPORTED_REPORT							
Color Component Mask 1							
Color Component Mask 2							

Color Component Mask (variable length)

The Color Component Mask field MUST advertise the color components supported by the device.

- Bit 0 in Bit Mask 1 indicates if color component 0 is supported
- Bit 1 in Bit Mask 1 indicates if color component 1 is supported
- …

For the definition of Color Component IDs, refer to section 4.33.7.

4.33.5 Color Switch Get Command

The Color Switch Get command, version 1 is used to request the status of a specified color component.

The Color Switch Report command MUST be returned in response to this command.

The Color Switch Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_COLOR							
Command = SWITCH_COLOR_GET							
Color Component ID							

Color Component ID (8 bits)

This field MUST specify the color component for which the status is requested.

For the definition of Color Component IDs, refer to section 4.33.7.

4.33.6 Color Switch Report Command

The Color Switch Report Command, version 1 is used to advertise the status of a color component.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_COLOR							
Command = SWITCH_COLOR_REPORT							
Color Component ID							
Value							

Color Component ID (8 bits)

This field MUST advertise the color component covered by this report.

For the definition of Color Component IDs, refer to section 4.33.7.

Value (8 bits)

The Value field MUST advertise the value of the color component identified by the Color Component ID.

The Value field SHOULD advertise the current value of the device hardware; also while in transition to a new target value.

A controlling device MUST NOT assume that the Value is identical to a value previously issued with a Set command; not even when a transition has ended.

4.33.7 Color Switch Set Command

The Color Switch Set command, version 1 is used to control one or more color components in a device.

Color component levels MUST be scaled by the brightness level. Refer to 4.33.2.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_COLOR							
Command = SWITCH_COLOR_SET							
Reserved			Color Component Count				
Color Component ID 1							
Value 1							
…….							
Color Component ID 2							
Value 2							

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Color Component Count (5 bits)

This field MUST specify the number of (Color Component ID, Value) datasets contained in the Color Switch Set command.

Color Component ID (8 bits)

This field MUST specify the color component to receive a new value. Refer to Table 35.

Table 35, Color Switch Component IDs

Component ID	Label	Value
0	Warm White	0x00 – 0xFF: 0 – 100%
1	Cold White	0x00: - 0xFF: 0 – 100%
2	Red	0x00 – 0xFF: 0 – 100%
3	Green	0x00 – 0xFF: 0 – 100%
4	Blue	0x00 – 0xFF: 0 – 100%
5	Amber (for 6ch Color mixing)	0x00 – 0xFF: 0 – 100%
6	Cyan (for 6ch Color mixing)	0x00 – 0xFF: 0 – 100%
7	Purple (for 6ch Color mixing)	0x00 – 0xFF: 0 – 100%
8	Indexed Color	0x00 – 0xFF: Color Index 0-255

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Value (8 bits)

This field MUST specify the value of the color component identified by the Color Component ID field.

Unlike other actuator control command classes, the Value used by the Color Switch Command Class spans the entire range from 0x00 to 0xFF. Thus, except for the Indexed Color, the value 0x00 MUST represent 0% and 0xFF MUST represent 100% of the actual color component.

4.33.8 Color Switch Start Level Change Command

The Color Switch Start Level Change command, version 1 is used to initiate a transition of one color component to a new level.

A receiving device MUST initiate the transition to a new value for the specified Color Component ID.

The device MAY apply a non-zero duration to the transition from one value to a new value.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_SWITCH_COLOR</td></tr>
<tr><td colspan="8">Command = SWITCH_COLOR_START_LEVEL_CHANGE</td></tr>
<tr><td>Reserved</td><td>Up/ Down</td><td>Ignore Start Level</td><td colspan="5">Reserved</td></tr>
<tr><td colspan="8">Color Component ID</td></tr>
<tr><td colspan="8">Start Level</td></tr>
</table>

Up/Down (1 bit)

This field MUST specify the direction of the level change.

If the Up/Down bit is set to 0 the level change MUST be increasing.
If the Up/Down bit is set to 1 the level change MUST be decreasing.

Ignore Start Level (1 bit)

A receiving device SHOULD respect the Start Level if the Ignore Start Level bit is 0.
A receiving device MUST ignore the Start Level if the Ignore Start Level bit is 1.

A controlling device SHOULD set the Ignore Start Level bit to 1.

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Color Component ID (8 bits)

This field MUST specify the color component to start a transition.

For the definition of Color Component IDs, refer to section 4.33.7.

Start Level (8 bits)

The Start Level field MUST specify the initial value of the level change.

4.33.9 Color Switch Stop Level Change Command

The Color Switch Stop Level Change command, version 1 is used to stop an ongoing transition initiated by a Color Switch CC command.

A receiving device MUST stop the transition if the specified Color Component ID is currently in transition to a new value.
A receiving device MUST NOT stop ongoing transitions for other Color Component IDs than the one specified.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_COLOR							
Command = SWITCH_COLOR_STOP_LEVEL_CHANGE							
Color Component ID							

Color Component ID (8 bits)

This field MUST specify the color component to stop a transition.

For the definition of Color Component IDs, refer to section 4.33.7.

4.34 Color Switch Command Class, version 2

The Color Switch Command Class, version 2 is used to control color capable devices.

The Color Switch Command Class is an actuator control command class. Refer to 3.7.

4.34.1 Compatibility considerations

A device supporting Color Switch CC, Version 2 MUST support Color Switch CC, Version 1.
A device supporting Color Switch CC, Version 2 MUST support Multilevel Switch CC, version 2.

Version 2 adds a Duration parameter to the Color Switch Set command.

Commands not described in Version 2 stays unchanged from Version 1.

4.34.2 Interoperability considerations

Refer to section 4.33.2 for details on interoperability with Basic and Multilevel Switch Command Classes.

4.34.3 Color Switch Set Command

The Color Switch Set command, version 2 is used to control one or more color components in a device.

Color component levels MUST be scaled by the brightness level. Refer to 4.33.2.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_COLOR							
Command = SWITCH_COLOR_SET							
Reserved			Color Component Count				
Color Component ID 1							
Value 1							
…….							
Color Component ID 2							
Value 2							
Duration							

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Color Component Count (5 bits)

This field MUST specify the number of (Color Component ID, Value) datasets contained in the Color Switch Set command.

Color Component ID (8 bits)

This field MUST specify the color component to receive a new value.

For the definition of Color Component IDs, refer to section 4.33.7.

Value (8 bits)

This field MUST specify the value of the color component identified by the Component ID field.

Refer to section Table 35.

Duration (8 bits)

The Duration field MUST specify the time that the transition should take from the current value to the new target value.
A supporting device SHOULD respect the specified Duration value.

The encoding of the Duration field MUST be according to Table 36.

Table 36, Color Switch Duration

Duration	Description
0x00	Instantly
0x01-0x7F	1 second (0x01) to 127 seconds (0x7F) in 1-second resolution.
0x80-0xFE	1 minute (0x80) to 127 minutes (0xFE) in 1-minute resolution.
0xFF	Factory default duration.

The factory default duration SHOULD be the same as the duration used for the Color Switch Set command, version 1.

4.35 Color Switch Command Class, version 3

The Color Switch Command Class is used to control color capable devices.

The Color Switch Command Class manipulates the color components of a device. Each color component is scaled by the brightness level previously set by a Multilevel Switch Set or Basic Set command.

The Color Switch Command Class is an actuator control command class. Refer to chapter 3.7.

Commands not described in Version 3 stays unchanged from Version 2.

4.35.1 Compatibility considerations

A device supporting Color Switch CC, Version 3 MUST support Color Switch CC, Version 2.

A device supporting Color Switch CC, Version 3 MUST support Multilevel Switch CC, version 4.

Version 3 adds duration and target value control and reporting.

A device receiving a Version 1 Color Switch Set command MAY apply a factory default duration to the transition.

4.35.2 Interoperability requirements

The Color Switch Command Class MUST be treated as a separate command class.
Basic and Multilevel Switch Command Class commands MUST NOT affect color component levels controlled by the Color Switch Command Class.
The Color Switch Command Class commands MUST NOT affect the brightness level controlled by Basic and Multilevel Switch Command Classes.

If the brightness level is requested via a Basic Get or Multilevel Switch Get command, the reported value MUST reflect the brightness level previously set via Basic or Multilevel Switch commands.
If a color component is requested via the Color Switch Get command, the reported value MUST reflect the color component level previously set via Color Switch commands.

An implementation MUST scale color component levels by the brightness level . Thus, to achieve a resulting light level of 100% for a given color component, the color component level must be set to 100% (via the Color Switch Command Class) and the brightness level must be set to 100% (via Basic or Multilevel Switch Command Class).

A controlling device SHOULD specify the highest possible color component levels of a given color tone to achieve the highest light yield when scaled by the brightness level. The controlling device MAY however limit color component levels to achieve the same brightness for blended color tones as for pure colors. For this reason, a supporting device MUST NOT normalize color components manipulated by the Color Switch Command Class.
Likewise, a supporting device implementing indexed colors should reduce its internal color component levels for blended colors to achieve the same brightness for all color tones.

4.35.3 Color Switch Report Command

The Color Switch Report Command, version 3 is used to advertise the status of a color component.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_COLOR							
Command = SWITCH_COLOR_REPORT							
Color Component ID							
Current Value							
Target Value							
Duration							

Color Component ID (8 bits)

This field MUST advertise the color component covered by this report.

Current Value (8 bits)

The Current Value field MUST advertise the current value of the color component identified by the Color Component ID.

Refer to the Color Switch Set command for valid values.

The Current Value SHOULD be identical to the Target Value when a transition has ended.

Target Value (8 bits)

The Target Value field MUST advertise the target value of an ongoing transition or the most recent transition for the advertised Color Component ID.

Refer to the Color Switch Set command for valid values.

If a transition is initiated in an interactive fashion via a local user interface or via a Start Level Change command, the advertised Target Value MUST be 0x00 or 0xFF, depending on the direction.

The Current Value SHOULD be identical to the Target Value when a transition has ended.

Duration (8 bits)

The Duration field SHOULD advertise the time needed to reach the Target Value at the actual transition rate. The encoding of the Duration field MUST be according to Table 37.

Table 37, Color Switch Report :: Duration

Duration	Description
0x00	0 seconds. Already at the Target Value.
0x01-0x7F	1 second (0x01) to 127 seconds (0x7F) in 1 second resolution.
0x80-0xFD	1 minute (0x80) to 126 minutes (0xFD) in 1 minute resolution.
0xFE	Unknown duration
0xFF	Reserved

All other values are reserved. Reserved values MUST NOT be used by a sending node and MUST be ignored by a receiving node.

4.35.4 Color Switch Start Level Change Command

The Color Switch Start Level Change command, version 3 is used to initiate a transition of one color component to a new value.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_SWITCH_COLOR</td></tr>
<tr><td colspan="8">Command = SWITCH_COLOR_START_LEVEL_CHANGE</td></tr>
<tr><td>Res</td><td>Up/ Down</td><td>Ignore Start Level</td><td colspan="5">Res</td></tr>
<tr><td colspan="8">Color Component ID</td></tr>
<tr><td colspan="8">Start Level</td></tr>
<tr><td colspan="8">Duration</td></tr>
</table>

Up/Down (1 bit)

This field MUST specify the direction of the level change.

If the Up/Down bit is set to 0 the level change MUST be increasing.
If the Up/Down bit is set to 1 the level change MUST be decreasing.

Ignore Start Level (1 bit)

A receiving device SHOULD respect the Start Level if the Ignore Start Level bit is 0.
A receiving device MUST ignore the Start Level if the Ignore Start Level bit is 1.

A controlling device SHOULD set the Ignore Start Level bit to 1.

Res

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Color Component ID (8 bits)

Refer to Color Switch Set command.

Start Level (8 bits)

The Start Level field MUST specify the initial value of the level change.

Duration (8 bits)

The level change rate MUST be calculated to match a level change from 0x00 to 0xFF during the time specified by the Duration field.
A supporting device SHOULD respect the specified Duration value.

For encoding of the Duration value, refer to the Color Switch Set command.

4.36 Configuration Command Class, version 1

The Configuration Command Class allows product specific configuration parameters to be changed. One example could be the default dimming rate of a light dimmer.

Configuration parameters MUST be specified in the product documentation. Configuration parameters accessed via this command class MUST NOT replace similar commands provided by other existing Command Classes.

A device MUST be able to operate with default factory configuration parameter values.

It is RECOMMENDED that configuration parameters can be manipulated via a local user interface.
It is RECOMMENDED that default factory configuration parameter values can be restored via a local user interface.
It is RECOMMENDED that an including controller resets all configuration parameters to factory default values during inclusion by issuing the Configuration Set Command for each individual parameter with the Default bit set to 1.

4.36.1 Compatibility considerations

4.36.1.1 "Default" flag

Earlier text revisions of the Configuration Command Class, versions 1-3 presented conflicting interpretations of the default field. Controllers should be aware that nodes with version 3 or less MAY reset all configuration parameters to default when receiving a Configuration Set or Configuration Bulk Set Command with the Default bit set to 1.

A controller SHOULD probe a node in order to discover what behavior it implements. The following steps are RECOMMENDED:

1. If the device is version 1- 2, scan the node for available configuration parameters. Else discover the available parameters thanks to the Configuration Properties Get Command.
2. Determine how the default bit works if the node implements 2 or more parameters

Step 1:

- One at a time, issue a Configuration Set for a parameter number, and check whether the value can be read back. Try each parameter for 1, 2 and 4 bytes sizes.
- Store the discovered parameter list for future use.

Step 2:

- Choose 2 parameters and issue a Configuration Set with the default bit set to 1 for each parameter.
- Read back both parameters value with a Configuration Get.
- Try to modify both parameters to non-default values (it is recommended to try the default value ± 1)
- Send a Configuration Set for one of parameters with the default bit set to 1.
- Read back both parameters and check whether one or both returned to the default value.

4.36.2 Configuration Set Command

The Configuration Set Command is used to set the value of a configuration parameter.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_CONFIGURATION</td></tr>
<tr><td colspan="8">Command = CONFIGURATION_SET</td></tr>
<tr><td colspan="8">Parameter Number</td></tr>
<tr><td>Default</td><td colspan="4">Reserved</td><td colspan="3">Size</td></tr>
<tr><td colspan="8">Configuration Value 1</td></tr>
<tr><td colspan="8">…</td></tr>
<tr><td colspan="8">Configuration Value N</td></tr>
</table>

Parameter Number (8 bits)

This field is used to specify the actual configuration parameter. Parameter Numbers MAY be product specific. Parameter Numbers SHOULD be assigned in a sequence starting from 1.

Default (1 bit)

This field is used to specify if the default value is to be restored for the specified configuration parameter.

The value 1 MUST indicate that default factory settings must be restored for the specified Parameter Number. The Configuration Value field MUST be ignored. The receiving node MUST reset the specified parameter and SHOULD NOT reset any other parameters.

The value 0 MUST indicate that the specified Parameter Number must assume the value specified by the Configuration Value field.

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Size (3 bits)

This field specifies the number of bytes used for the Value field. The value of this field MUST comply with Table 38.

Table 38, Configuration Set :: Size encoding

Size	Size of Value field
1	8 bit
2	16 bit
4	32 bit

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Configuration Value (N bytes)

This field carries the value to be assigned. The size of the field MUST comply with the size advertised by the Size field.

The field MUST carry a signed value. The binary encoding of the signed value MUST comply with Table 39. The field Value 1 MUST be the most significant byte.

Table 39, Configuration Set :: Value field encoding

Raw value (hex)	Signed 8 bit representation (decimal)	Raw value (hex)	Signed 16 bit representation (decimal)	Raw value (hex)	Signed 32 bit representation (decimal)
0x7F	127	0x7FFF	32767	0x7FFFFFFF	2147483647
0x02	2	0x0002	2	0x00000002	2
0x01	1	0x0001	1	0x00000001	1
0x00	0	0x0000	0	0x00000000	0
0xFF	-1	0xFFFF	-1	0xFFFFFFFF	-1
0xFE	-2	0xFFFE	-2	0xFFFFFFFE	-2
0x80	-128	0x8000	-32768	0x80000000	-2147483648

4.36.3 Configuration Get Command

The Configuration Get Command is used to query the value of a configuration parameter.

The Configuration Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CONFIGURATION							
Command = CONFIGURATION_GET							
Parameter Number							

Parameter Number (8 bits)

This field is used to specify the requested configuration parameter. Parameter Numbers MAY be product specific. Parameter Numbers SHOULD be assigned in a sequence starting from 1.

4.36.4 Configuration Report Command

This Configuration Report Command is used to advertise the actual value of the advertised parameter.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_CONFIGURATION</td></tr>
<tr><td colspan="8">Command = CONFIGURATION_REPORT</td></tr>
<tr><td colspan="8">Parameter Number</td></tr>
<tr><td colspan="5">Reserved</td><td colspan="3">Size</td></tr>
<tr><td colspan="8">Configuration Value 1</td></tr>
<tr><td colspan="8">..</td></tr>
<tr><td colspan="8">Configuration Value N</td></tr>
</table>

Refer to the Configuration Set Command for parameter details.

4.37 Configuration Command Class, version 2

The Configuration Command Class allows product specific configuration parameters to be changed. One example could be the default dimming rate of a light dimmer.

Configuration parameters MUST be specified in the product documentation. Configuration parameters accessed via this command class MUST NOT replace similar commands provided by other existing Command Classes.

A device MUST be able to operate with default factory configuration parameter values.

It is RECOMMENDED that configuration parameters can be manipulated via a local user interface.
It is RECOMMENDED that default factory configuration parameter values can be restored via a local user interface.
It is RECOMMENDED that an including controller resets all configuration parameters to factory default values during inclusion by issuing the Configuration Set Command or the Configuration Bulk Set Command for each individual parameter(s) with the Default bit set to 1.

4.37.1 Compatibility considerations

Configuration Command Class, version 2 extends the addressing space to 65.535 product specific configuration parameters. Further, Configuration Command Class, version 2 makes it possible to set multiple configuration parameters with one Command.

A device supporting Configuration Command Class, version 2 MUST support Configuration Command Class, version 1.

4.37.1.1 “Default” flag

Refer to 4.36.1.1

4.37.2 Interoperability considerations

Configuration Command Class, version 2 is backwards compatible with Configuration Command Class, version 1.

Configuration Command Class, version 2 does not extend any existing commands of Configuration Command Class, version 1. The 255 parameters that can be addressed by the Configuration Set Command MUST be identical to the first 255 parameters that can be addressed by the (version 2) Configuration Bulk Set Command.

4.37.3 Configuration Set Command

The Configuration Set Command is used to set the value of a configuration parameter.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_CONFIGURATION</td></tr>
<tr><td colspan="8">Command = CONFIGURATION_SET</td></tr>
<tr><td colspan="8">Parameter Number</td></tr>
<tr><td>Default</td><td colspan="4">Reserved</td><td colspan="3">Size</td></tr>
<tr><td colspan="8">Configuration Value 1</td></tr>
<tr><td colspan="8">…</td></tr>
<tr><td colspan="8">Configuration Value N</td></tr>
</table>

Parameter Number (8 bits)

This field is used to specify the actual configuration parameter. Parameter Numbers MAY be product specific. Parameter Numbers SHOULD be assigned in a sequence starting from 1.

Default (1 bit)

This field is used to specify if the default value is to be restored for the specified configuration parameter.

The value 1 MUST indicate that default factory settings must be restored for the specified Parameter Number. The Configuration Value field MUST be ignored. The receiving node MUST reset the specified parameter and SHOULD NOT reset any other parameters.

The value 0 MUST indicate that the specified Parameter Number must assume the value specified by the Configuration Value field.

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Size (3 bits)

This field specifies the number of bytes used for the Value field. The value of this field MUST comply with Table 38.

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Configuration Value (N bytes)

This field carries the value to be assigned. The size of the field MUST comply with the size advertised by the Size field.

The field MUST carry a signed value. The binary encoding of the signed value MUST comply with Table 40. The field Value 1 MUST be the most significant byte.

Table 40, Configuration Set :: Value field encoding

Raw value (hex)	Signed 8 bit representation (decimal)	Raw value (hex)	Signed 16 bit representation (decimal)	Raw value (hex)	Signed 32 bit representation (decimal)
0x7F	127	0x7FFF	32767	0x7FFFFFFF	2147483647
0x02	2	0x0002	2	0x00000002	2
0x01	1	0x0001	1	0x00000001	1
0x00	0	0x0000	0	0x00000000	0
0xFF	-1	0xFFFF	-1	0xFFFFFFFF	-1
0xFE	-2	0xFFFE	-2	0xFFFFFFFE	-2
0x80	-128	0x8000	-32768	0x80000000	-2147483648

4.37.4 Configuration Bulk Set Command

The Configuration Bulk Set Command is used to set the value of one or more configuration parameters.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_CONFIGURATION</td></tr>
<tr><td colspan="8">Command = CONFIGURATION_BULK_SET</td></tr>
<tr><td colspan="8">Parameter Offset 1 (MSB)</td></tr>
<tr><td colspan="8">Parameter Offset 2 (LSB)</td></tr>
<tr><td colspan="8">Number of Parameters</td></tr>
<tr><td>Default</td><td>Hand-shake</td><td colspan="3">Reserved</td><td colspan="3">Size</td></tr>
<tr><td colspan="8">Parameter 1 – Configuration Value 1 (MSB)</td></tr>
<tr><td colspan="8">…</td></tr>
<tr><td colspan="8">Parameter 1 – Configuration Value N (LSB)</td></tr>
<tr><td colspan="8">…</td></tr>
<tr><td colspan="8">Parameter M – Configuration Value 1 (MSB)</td></tr>
<tr><td colspan="8">…</td></tr>
<tr><td colspan="8">Parameter M – Configuration Value N (LSB)</td></tr>
</table>

Parameter Offset (16 bits)

This field is used to specify the first parameter in a range of one or more parameters. The first byte MUST carry the most significant byte of the 16 bit value.

Number of Parameters (8 bits)

This field is used to specify the number (M) of configuration parameters contained in this command.

The value of this field MUST be in the range 1..255.

Default (1 bit)

This field is used to specify if the default value is to be restored for the specified configuration parameters.

The value 1 MUST indicate that default factory settings must be restored for the Parameter Numbers defined by the Parameter Offset and Number of Parameters fields. The Configuration Value field MUST be ignored. The receiving node MUST reset the specified parameters and SHOULD NOT reset any other parameters.

The value 0 MUST indicate that the specified Parameter Numbers must assume the value specified by the Configuration Value field.

Handshake (1 bit)

This field is used to indicate if a Configuration Bulk Report Command is to be returned when the specified configuration parameters have been stored in non-volatile memory.

If the Handshake bit is set to 1, a Configuration Bulk Report Command MUST be returned. The command MUST echo all parameters found in this Configuration Bulk Set Command. A node returning a Configuration Bulk Report Command with the Handshake bit set MUST be ready to receive another Configuration Bulk Set Command. If the Handshake bit is set, an originating node MUST wait for the Configuration Bulk Report Command for at least one second. If not receiving a report, the originating node SHOULD assume that the operation failed and resend the same Configuration Bulk Set Command one more time.

If the Handshake bit is set to 0, a receiving node MUST NOT return a Configuration Bulk Report in response.

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Size (3 bits)

This field specifies the number of bytes used for the Value field. The value of this field MUST comply with Table 38.

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Parameter – Configuration Value (M*N bytes)

These fields carry the values to be assigned. Each field MUST have the same size. The size of each field MUST comply with the size advertised by the Size field.

The field MUST carry a signed value. The binary encoding of the signed value MUST comply with Table 41. The Value 1 field MUST be the most significant byte.

Table 41, Configuration Bulk Set :: Value field encoding

Raw value (hex)	Signed 8 bit representation (decimal)	Raw value (hex)	Signed 16 bit representation (decimal)	Raw value (hex)	Signed 32 bit representation (decimal)
0x7F	127	0x7FFF	32767	0x7FFFFFFF	2147483647
0x02	2	0x0002	2	0x00000002	2
0x01	1	0x0001	1	0x00000001	1
0x00	0	0x0000	0	0x00000000	0
0xFF	-1	0xFFFF	-1	0xFFFFFFFF	-1
0xFE	-2	0xFFFE	-2	0xFFFFFFFE	-2
0x80	-128	0x8000	-32768	0x80000000	-2147483648

4.37.5 Configuration Bulk Get Command

The Configuration Bulk Get Command is used to query the value of one or more configuration parameters.

The Configuration Bulk Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CONFIGURATION							
Command = CONFIGURATION_BULK_GET							
Parameter Offset 1 (MSB)							
Parameter Offset 2 (LSB)							
Number of Parameters							

Parameter Offset (16 bits)

This field is used to specify the first parameter in a range of one or more parameters. The first byte MUST carry the most significant byte of the 16 bit value.

Parameter Numbers SHOULD be assigned in a sequence starting from 1.

Number of Parameters (8 bits)

This field is used to specify the number of requested configuration parameters.

The value of this field MUST be in the range 1..255.

4.37.6 Configuration Bulk Report Command

The Configuration Bulk Report Command is used to advertise the actual value of one or more advertised parameters.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_CONFIGURATION</td></tr>
<tr><td colspan="8">Command = CONFIGURATION_BULK_REPORT</td></tr>
<tr><td colspan="8">Parameter Offset 1 (MSB)</td></tr>
<tr><td colspan="8">Parameter Offset 2 (LSB)</td></tr>
<tr><td colspan="8">Number of Parameters</td></tr>
<tr><td colspan="8">Reports to follow</td></tr>
<tr><td>Default</td><td>Hand-shake</td><td colspan="3">Reserved</td><td colspan="3">Size</td></tr>
<tr><td colspan="8">Parameter 1 – Configuration Value 1 (MSB)</td></tr>
<tr><td colspan="8">…</td></tr>
<tr><td colspan="8">Parameter 1 – Configuration Value N (LSB)</td></tr>
<tr><td colspan="8">…</td></tr>
<tr><td colspan="8">Parameter M – Configuration Value 1 (MSB)</td></tr>
<tr><td colspan="8">…</td></tr>
<tr><td colspan="8">Parameter M – Configuration Value N (LSB)</td></tr>
</table>

Parameter Offset (16 bits)

This field is used to advertise the first parameter in a range of one or more parameters. The first byte MUST carry the most significant byte of the 16 bit value.

Number of Parameters (8 bits)

This field is used to advertise the number (M) of configuration parameters contained in this command.

The value of this field MUST be in the range 1..255.

Reports to follow (8 bits)

This field MUST be used to advertise the number of reports left before all requested configuration parameters values have been transferred.

The value 0 MUST indicate that this is the last report.

If the Handshake field is 1 , this field MUST be set to 0.

The value of this field MUST be in the range 0..255.

Default (1 bit)

This field MUST be used to advertise if all advertised configuration parameters have the factory default value.

The value 1 MUST indicate that all advertised configuration parameters have the factory default value.

The value 0 MUST indicate that one or more of the advertised configuration parameters do not have the factory default value.

Handshake (1 bit)

This field is used to indicate if this report is returned in response to a Configuration Bulk Set Command.

The value 1 MUST indicate that all configuration parameters have been stored in non-volatile memory and that the sending node is ready to receive another Configuration Bulk Set Command. Except for the Reports to Follow field, all other fields MUST echo the values received in the Configuration Bulk Set Command. The Reports to Follow field MUST be 0.

The value 0 MUST indicate that this report is returned in response to a Configuration Bulk Get Command.

Size (3 bits)

This field specifies the number of bytes used for the Value field. The value of this field MUST comply with Table 38.

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Parameter – Configuration Value (M*N bytes)

These fields carry the values advertised by the Parameter Offset and Number of Parameters fields. Each field MUST have the same size. The size of each field MUST comply with the size advertised by the Size field.

The field MUST carry a signed value. The binary encoding of the signed value MUST comply with Table 42. The Value 1 field MUST be the most significant byte.

Table 42, Configuration Bulk Report :: Value field encoding

Raw value (hex)	Signed 8 bit representation (decimal)	Raw value (hex)	Signed 16 bit representation (decimal)	Raw value (hex)	Signed 32 bit representation (decimal)
0x7F	127	0x7FFF	32767	0x7FFFFFFF	2147483647
0x02	2	0x0002	2	0x00000002	2
0x01	1	0x0001	1	0x00000001	1
0x00	0	0x0000	0	0x00000000	0
0xFF	-1	0xFFFF	-1	0xFFFFFFFF	-1
0xFE	-2	0xFFFE	-2	0xFFFFFFFE	-2
0x80	-128	0x8000	-32768	0x80000000	-2147483648

4.38 Configuration Command Class, version 3

The Configuration Command Class, version 3 allows product specific configuration parameters to be changed. Examples include the default dimming rate of a light dimmer and the endpoints of a window covering device.

Configuration parameters accessed via this command class MUST NOT replace similar commands provided by other existing application Command Classes.
Configuration parameter documentation MUST be provided in the product documentation.
Configuration parameter documentation SHOULD be provided via Z-Wave via this command class version.

A device MUST be able to operate with default factory configuration parameter values.

It is RECOMMENDED that configuration parameters can be manipulated via a local user interface.

It is RECOMMENDED that default factory configuration parameter values can be restored via a local user interface.

It is RECOMMENDED that an including controller resets all configuration parameters to factory default values during inclusion by issuing the Set Command or the Bulk Set Command for each individual parameter(s) with the Default bit set to 1.

4.38.1 Compatibility considerations

The following additions are made to the Configuration Command Class, version 3:

- Configuration Name Get Command
- Configuration Name Report Command
- Configuration Info Get Command
- Configuration Info Report Command
- Configuration Properties Get Command
- Configuration Properties Report Command

The purpose of the above commands is to allow a manufacturer to advertise information relating to the use of a specific configuration parameter directly from the configuration interface. The advertised information should be the same as can be found in the printed documentation.

It should be emphasized that, just like versions 1 and 2, the Configuration Command Class, version 3 is intended only for access to product specific configuration parameters. The access to parameter information is only meant to facilitate access via Z-Wave to information which can also be found in the printed documentation.

Configuration Command Class versions 1 and 2 explicitly require that the binary value of all configuration parameters are represented as signed values encoded as binary 2's complement.

Configuration Command Class v3 allows for bit field, signed integer or unsigned integer representations of each individual parameter.

A device supporting the Configuration Command Class v3 MUST respond to a Configuration Properties Get command. If a Configuration Properties Report command is not returned in response to a Configuration Properties Get command, a controlling device MUST treat the parameter value as a signed integer.

Configuration Command Class version 2 extended the possible parameter range from 1..255 to 1..65535. The first 255 parameters of the two ranges are identical. Parameters in the range 1..255 can be addressed using either the Bulk Set/Get Commands or Set/Get Commands while parameters in the range 256..65535 can only be addressed using the Bulk Set/Get Commands.

4.38.1.1 “Default” flag

Refer to 4.36.1.1

4.38.2 Internationalization considerations

It is RECOMMENDED that the name and information stored in a supporting device are in English.

It is RECOMMENDED that the last line of the information field presents an Internet URL which points to updated information for the actual configuration parameter as well as other language variants.

4.38.3 Configuration Name Get Command

This command is used to request the name of a configuration parameter.

The Configuration Name Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CONFIGURATION							
Command = CONFIGURATION_NAME_GET							
Parameter Number 1 (MSB)							
Parameter Number 2 (LSB)							

Parameter Number (16 bits)

This field is used to specify the requested configuration parameter.

The first byte MUST carry the most significant byte of the 16 bit value.

4.38.4 Configuration Name Report Command

This command is used to advertise the name of a parameter.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CONFIGURATION							
Command = CONFIGURATION_NAME_REPORT							
Parameter Number 1 (MSB)							
Parameter Number 2 (LSB)							
Reports to follow							
Name 1							
..							
Name N							

Parameter Number (16 bits)

This field is used to advertise the actual configuration parameter.

The first byte MUST carry the most significant byte of the 16 bit value.

If a non-existing parameter is specified in a Configuration Name Get Command, a receiving node MAY return a zero-length Name field in the Configuration Name Report Command. The receiving node SHOULD however return an error message e.g. “Unassigned parameter”.

Parameter numbers above 255 can only be addressed using the Bulk Set/Get Commands. Parameter numbers in the range 1..255 can be addressed using either the Bulk Set/Get Commands or the Set/Get Commands.

Reports to follow (8 bits)

This field is used to advertise the number of reports left before all parts of the command have been transferred.

The value 0 MUST indicate that this is the last report.

Name (N bytes)

This field is used to advertise the name of the parameter. It is RECOMMENDED that the Name field advertises the name found in the documentation.

The field MUST carry a byte string with no zero termination. The Name field MAY span multiple reports. The number of Name bytes transmitted MUST be determined from the length field in the frame.

Bytes MUST be encoded in UTF-8 format.

It is RECOMMENDED that the name stored in a supporting device is in English.

4.38.5 Configuration Info Get Command

This command is used to request usage information for a configuration parameter.

The Configuration Info Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CONFIGURATION							
Command = CONFIGURATION_INFO_GET							
Parameter Number 1 (MSB)							
Parameter Number 2 (LSB)							

Parameter Number (16 bits)

This field is used to specify the requested configuration parameter.

The first byte MUST carry the most significant byte of the 16 bit value.

4.38.6 Configuration Info Report Command

This command is used to advertise usage information for a configuration parameter.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CONFIGURATION							
Command = CONFIGURATION_INFO_REPORT							
Parameter Number 1 (MSB)							
Parameter Number 2 (LSB)							
Reports to follow							
Info 1							
..							
Info N							

Parameter Number (16 bits)

This field is used to advertise the actual configuration parameter.

The first byte MUST carry the most significant byte of the 16 bit value.

If a non-existing parameter is specified in a Configuration Info Get Command, a receiving node MAY return a zero-length Info field in the Configuration Info Report Command. The receiving node SHOULD however return an error message, e.g. “Unassigned parameter”.

Parameter numbers above 255 can only be addressed using the Bulk Set/Get Commands. Parameter numbers in the range 1..255 can be addressed using either the Bulk Set/Get Commands or the Set/Get Commands

Reports to follow (8 bits)

This field is used to advertise the number of reports left before all parts of the command have been transferred.

The value 0 MUST indicate that this is the last report.

Info (N bytes)

This field is used to advertise the detailed information available for the parameter. It is RECOMMENDED that the Info field carries all information found in the documentation.

The field MUST carry a byte string with no zero termination. The Info field MAY span multiple reports. The number of Info bytes transmitted MUST be determined from the length field in the frame. The number of Info bytes MAY be zero.

Bytes MUST be encoded in UTF-8 format.

It is RECOMMENDED that the information stored in a supporting device is in English.

It is RECOMMENDED that the last line of the information field presents an Internet URL which points to updated information for the actual configuration parameter as well as other language variants.

4.38.7 Configuration Properties Get Command

This command is used to request the properties of a configuration parameter.

The Configuration Properties Report Command MUST be returned in response to this command.

If a Configuration Properties Report command is not returned in response to a Configuration Properties Get command, a controlling device MUST treat the parameter value as a signed integer.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing.
The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CONFIGURATION							
Command = CONFIGURATION_PROPERTIES_GET							
Parameter Number 1 (MSB)							
Parameter Number 2 (LSB)							

Parameter Number (16 bits)

This field is used to specify the requested configuration parameter.

The first byte MUST carry the most significant byte of the 16 bit value.

4.38.8 Configuration Properties Report Command

This command is used to advertise the properties of a configuration parameter.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_CONFIGURATION</td></tr>
<tr><td colspan="8">Command = CONFIGURATION_PROPERTIES_REPORT</td></tr>
<tr><td colspan="8">Parameter Number 1 (MSB)</td></tr>
<tr><td colspan="8">Parameter Number 2 (LSB)</td></tr>
<tr><td colspan="2">Reserved</td><td colspan="3">Format</td><td colspan="3">Size</td></tr>
<tr><td colspan="8">Min Value 1 (MSB)</td></tr>
<tr><td colspan="8">..</td></tr>
<tr><td colspan="8">Min Value N (LSB)</td></tr>
<tr><td colspan="8">Max Value 1 (MSB)</td></tr>
<tr><td colspan="8">..</td></tr>
<tr><td colspan="8">Max Value N (LSB)</td></tr>
<tr><td colspan="8">Default Value 1 (MSB)</td></tr>
<tr><td colspan="8">..</td></tr>
<tr><td colspan="8">Default Value N (LSB)</td></tr>
<tr><td colspan="8">Next Parameter Number 1 (MSB)</td></tr>
<tr><td colspan="8">Next Parameter Number 2 (LSB)</td></tr>
</table>

Parameter Number (16 bits)

This field is used to advertise the actual configuration parameter.

The first byte MUST carry the most significant byte of the 16 bit value.

If a non-existing parameter is specified in a Configuration Properties Get Command, a receiving node MUST advertise zero values in all fields, except for the Next Parameter Number field which MUST advertise the next available configuration parameter.

Parameter numbers above 255 can only be addressed using the Bulk Set/Get Commands. Parameter numbers in the range 1..255 can be addressed using either the Bulk Set/Get Commands or the Set/Get Commands.

It is RECOMMENDED that a controlling device initiates probing of supported configuration parameters by issuing a Configuration Properties Get Command for parameter number 0.
If a Size field value of zero is returned, a controlling device SHOULD issue a Configuration Properties Get Command for the parameter advertised in the Next Parameter Number field.

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Format (3 bits)

This field is used to advertise the format of the parameter.

The value of this field MUST comply with Table 43.

Table 43, Configuration Properties Report :: Format encoding

Format	Parameter format and presentation
0	Signed Integer
1	Unsigned Integer
2	Enumerated (Radio buttons)
3	Bit field (Checkboxes)

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

If the parameter format is “Unsigned integer”, normal binary integer encoding MUST be used.

If the parameter format is “Signed integer”, the binary encoding MUST comply with Table 44.

Table 44, Signed parameter and value encoding

Raw value (hex)	Signed 8 bit representation (decimal)	Raw value (hex)	Signed 16 bit representation (decimal)	Raw value (hex)	Signed 32 bit representation (decimal)
0x7F	127	0x7FFF	32767	0x7FFFFFFF	2147483647
…	…	…	…	…	…
0x02	2	0x0002	2	0x00000002	2
0x01	1	0x0001	1	0x00000001	1
0x00	0	0x0000	0	0x00000000	0
0xFF	-1	0xFFFF	-1	0xFFFFFFFF	-1
0xFE	-2	0xFFFE	-2	0xFFFFFFFE	-2
…	…	…	…	…	…
0x80	-128	0x8000	-32768	0x80000000	-2147483648

If the parameter format is “Enumerated”, the parameter MUST be treated as an unsigned integer.
A graphical configuration tool SHOULD present this parameter as a series of radio buttons [10].

If the parameter format is “Bit field” the parameter MUST be treated as a bit field where each individual bit can be set or reset.
A graphical configuration tool SHOULD present this parameter as a series of checkboxes [10].

Size (3 bits)

This field is used to advertise the size of the actual parameter.

The advertised size MUST also apply to the fields “Min Value”, “Max Value”, “Default Value” carried in this command.

The value of this field MUST comply with Table 45.

Table 45, Configuration Properties Report :: Size encoding

Size	Size of parameter
0	(Unassigned parameter)
1	8 bit
2	16 bit
4	32 bit

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Min Value (N bytes)

This field MUST advertise the minimum value that the actual parameter can assume.

If the parameter is “Bit field”, this field MUST be set to 0.

The size of the field MUST comply with the size advertised by the Size field.

The sign encoding MUST comply with the mode advertised by the Signed field.

Max Value (N bytes)

This field MUST advertise the maximum value that the actual parameter can assume.

If the parameter is “Bit field”, each individual supported bit MUST be set to ‘1’, while each un-supported bit of MUST be set to ‘0’.
A graphical configuration tool SHOULD NOT present checkboxes for un-supported bits.

The size of the field MUST comply with the size advertised by the Size field.

The sign encoding MUST comply with the mode advertised by the Signed field.

Default Value (N bytes)

This field MUST advertise the default value of the actual parameter.

The size of the field MUST comply with the size advertised by the Size field.

The sign encoding MUST comply with the mode advertised by the Signed field.

Next Parameter Number (16 bits)

Configuration parameter identifiers may be assigned in a non-sequential order.

This field MUST advertise the next available configuration parameter.
The value 0x0000 MUST indicate that this is the last available configuration parameter.

The first byte MUST carry the most significant byte of the 16 bit value.

4.39 Configuration Command Class, version 4

The Configuration Command Class, version 4 introduces the support of read-only parameters, parameters requiring network re-inclusion and advanced parameters as well as allows optional support for Bulk Commands.

A read-only parameter can be accessed with Get commands but cannot be modified with Set Commands. A parameter requiring re-inclusion is typically a parameter altering the node capabilities, which requires performing the inclusion again in order to have the controller to learn about the node capabilities once more. A parameter may be advertised as "advanced" to indicate to the controlling application that the actual parameter is intended only for advanced use, e.g. calibration.

The following commands are modified in version 4:

- Configuration Set Command
- Configuration Bulk Set Command
- Configuration Properties Report Command

The following new command is introduced:

- Configuration Default Reset Command

All commands not mentioned in this version remain unchanged from previous versions.

4.39.1 Compatibility considerations

Configuration Command Class, version 4 is backwards compatible with the previous versions of Configuration Command Class.

A device supporting Configuration Command Class, version 4 MUST support Configuration Command Class, version 3.

4.39.1.1 “Default” flag

Earlier text revisions of the Configuration Command Class, versions 1-3 have presented conflicting interpretations of the default field. Controllers should be aware that nodes with version 3 or less MAY reset all configuration parameters to default when receiving a Set or Bulk Set Command with the Default bit set to 1.

From version 4 and onwards, a node MUST NOT reset all Configuration Parameters when receiving a Set or Bulk Set Command with the default flag set to 1.

A controller may probe a version 1-3 node in order to discover what behaviour it implements. The following steps are suggested:

3. If the device is version 1-2, scan the node for available configuration parameters. Else discover the available parameters thanks to the Configuration Properties Get Command.
4. Determine how the default bit works if the node implements 2 or more parameters.

Step 1:

- One at a time, issue a Configuration Get for a parameter number and check if a Configuration Report with the same parameter number is returned.

- Store the discovered parameter list for future use.

Step 2:

- Choose 2 parameters and issue a Configuration Set with the default bit set to 1 for each parameter.
- Read back both parameters value with a Configuration Get.
- Try to modify both parameters to non-default values (it is recommended to try the default value ± 1)
- Send a Configuration Set for one of parameters with the default bit set to 1.
- Read back both parameters and check whether one or both returned to the default value.

4.39.1.2 "Re-inclusion required" flag

Configuration parameters modifying a node's NIF and/or Multi Channel End Point capabilities MUST be advertised as "Re-inclusion required".

When such a parameter is modified, the node MUST NOT advertise the new NIF and/or Multi Channel End Point capabilities before being excluded from its current network.

4.39.1.3 "Advanced" flag

The only difference between normal and advanced parameters lies on the controlling node side.

A parameter MAY be advertised as "advanced" in order to simplify the configuration of a node by normally not showing such a parameter to the end user.
Advanced parameters SHOULD be presented to the user only when an [Advanced] option is selected in the controller user interface.

4.39.1.4 Parameters value and network inclusion/exclusion

From version 4 and onwards, a node MUST NOT modify or reset any configuration parameter when being included or excluded of a Z-Wave network.

A node MUST reset all its configuration parameters if either:

- It is manually reset to factory default
- It receives a Configuration Default Reset Command.

A node MUST NOT reset all its configuration parameters in any other case.

4.39.1.5 Bulk commands support

A node supporting Configuration Command Class, version 4 MAY elect to ignore the following Commands:

- Configuration Bulk Set Command
- Configuration Bulk Get Command
- Configuration Bulk Report Command.

If it is the case, then the node MUST:

- Return an Application Rejected Request Command when receiving one of the ignored commands.
- Advertise that it ignores the Bulk Commands in the Configuration Properties Report.

If Bulk Commands are supported, they MUST be supported for all parameters. If Bulk Commands are ignored, they MUST be ignored for all parameters.

4.39.2 Configuration Set Command

The Configuration Set Command is used to set the value of a configuration parameter.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_CONFIGURATION</td></tr>
<tr><td colspan="8">Command = CONFIGURATION_SET</td></tr>
<tr><td colspan="8">Parameter Number</td></tr>
<tr><td>Default</td><td colspan="4">Reserved</td><td colspan="3">Size</td></tr>
<tr><td colspan="8">Configuration Value 1</td></tr>
<tr><td colspan="8">…</td></tr>
<tr><td colspan="8">Configuration Value N</td></tr>
</table>

A receiving node MUST ignore this command if the specified parameter is advertised as a Read-only parameter.

All fields not specified below are unchanged from version 2 Configuration Set Command.

Default (1 bit)

This field is used to specify if the default value is to be restored for the specified configuration parameter.

The value 1 MUST indicate that default factory settings must be restored for the specified Parameter Number. The Configuration value field MUST be ignored. Any other parameters MUST NOT be reset by a receiving node.

The value 0 MUST indicate that the specified Parameter Number MUST assume the value specified by the Configuration Value field.

Configuration Value (N bytes)

This field carries the value to be assigned. The size of the field MUST comply with the size advertised by the Size field.

The configuration value MUST be encoded according to the Format field advertised in the Configuration Properties Report Command for the parameter number.

4.39.3 Configuration Bulk Set Command

The Configuration Bulk Set Command is used to set the value of one or more configuration parameters.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CONFIGURATION							
Command = CONFIGURATION_BULK_SET							
Parameter Offset 1 (MSB)							
Parameter Offset 2 (LSB)							
Number of Parameters							
Default	Hand-shake	Reserved			Size		
Parameter 1 – Configuration Value 1 (MSB)							
…							
Parameter 1 – Configuration Value N (LSB)							
…							
Parameter M – Configuration Value 1 (MSB)							
…							
Parameter M – Configuration Value N (LSB)							

A receiving node MUST ignore this command for parameters that are advertised as Read-only parameters.

All fields not specified below are unchanged from version 2 Configuration Bulk Set Command.

Default (1 bit)

This field is used to specify if the default value is to be restored for the specified configuration parameters.

The value 1 MUST indicate that default factory settings must be restored for all the Parameter Numbers defined by the Parameter Offset and Number of Parameters fields. The Configuration Value field MUST be ignored. Any other parameters MUST NOT be reset by a receiving node.

The value 0 MUST indicate that the specified Parameter Numbers MUST assume the value specified by the Configuration Value field.

Parameter –Configuration Value (M*N bytes)

These fields carry the parameter values to be assigned. Each parameter value field MUST have the same size. The size of each field MUST comply with the size advertised by the Size field.

The values MUST be encoded according to the Format field advertised in the Configuration Properties Report Command for each parameter number.

4.39.4 Configuration Properties Report Command

This command is used to advertise the properties of a configuration parameter.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_CONFIGURATION</td></tr>
<tr><td colspan="8">Command = CONFIGURATION_PROPERTIES_REPORT</td></tr>
<tr><td colspan="8">Parameter Number 1 (MSB)</td></tr>
<tr><td colspan="8">Parameter Number 2 (LSB)</td></tr>
<tr><td>Re-inclusion required</td><td>Read-only</td><td colspan="3">Format</td><td colspan="3">Size</td></tr>
<tr><td colspan="8">Min Value 1 (MSB)</td></tr>
<tr><td colspan="8">..</td></tr>
<tr><td colspan="8">Min Value N (LSB)</td></tr>
<tr><td colspan="8">Max Value 1 (MSB)</td></tr>
<tr><td colspan="8">..</td></tr>
<tr><td colspan="8">Max Value N (LSB)</td></tr>
<tr><td colspan="8">Default Value 1 (MSB)</td></tr>
<tr><td colspan="8">..</td></tr>
<tr><td colspan="8">Default Value N (LSB)</td></tr>
<tr><td colspan="8">Next Parameter Number 1 (MSB)</td></tr>
<tr><td colspan="8">Next Parameter Number 2 (LSB)</td></tr>
<tr><td colspan="6">Reserved</td><td>No Bulk support</td><td>Advanced</td></tr>
</table>

All fields not specified below are unchanged from version 3 Configuration Properties Report Command.

Read-only (1 bit)

This field is used to indicate if the parameter is read-only.

The value 1 MUST indicate that the advertised parameter is read-only.

The value 0 MUST indicate that the advertised parameter is not read-only.

Re-inclusion required (1 bit)

This field is used to indicate if the advertised parameter requires the node to be re-included in the network before the parameter value change takes effect.

The value 1 MUST indicate that the advertised parameter requires re-inclusion after being changed.

The value 0 MUST indicate that the advertised parameter does not require re-inclusion after being changed.

Advanced (1 bit)

This field is used to indicate if the advertised parameter is to be presented in the “Advanced” parameter section in the controller GUI.

The value 1 MUST indicate that the advertised parameter is an Advanced parameter.
The value 0 MUST indicate that the advertised parameter is not an Advanced parameter.

No Bulk support (1 bit)

This field is used to advertise if the sending node supports Bulk Commands.

The value 1 MUST indicate that the Bulk Commands will be ignored by the sending node.
The value 0 MUST indicate that the Bulk Commands are supported by the sending node

A sending node MUST always advertise the same value in this field, regardless of the parameter number.

4.39.5 Configuration Default Reset Command

The Configuration Default Reset Command is used to reset all configuration parameters to their default value.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CONFIGURATION							
Command = CONFIGURATION_DEFAULT_RESET							

A node receiving this command MUST reset all its Configuration Parameters to their default value.

4.40 Controller Replication Command Class, version 1

The Controller Replication Command Class is used to copy scene and group data to another controller. The Command Class may be used in conjunction with a controller shift or when including a new controller to the network. It is OPTIONAL to use this command class during a controller shift or when including a new controller to the network.

4.40.1 Transfer Group Command

The Transfer Group Command is used to replicate mappings between Group ID and Node ID.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CONTROLLER_REPLICATION							
Command = CTRL_REPLICATION_TRANSFER_GROUP							
Sequence Number							
Group ID							
Node ID							

Sequence Number (8 bits)

Sequence Number of this particular command.

Group ID (8 bits)

Group ID of the group that the node is member of.

Node ID (8 bits)

Node ID of slave device.

4.40.2 Transfer Group Name Command

The Transfer Group Name Command is used to replicate group names.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CONTROLLER_REPLICATION							
Command = CTRL_REPLICATION_TRANSFER_GROUP_NAME							
Sequence Number							
Group ID							
Group Name 1							
…							
Group Name N							

Sequence Number (8 bits)

Sequence Number of this particular command.

Group ID (8 bits)

Group ID associated with a specific group.

Group Name (N bytes)

The Group Name fields contain the assign group name in ASCII characters.

4.40.3 Transfer Scene Command

The Transfer Scene Command is used to replicate mappings between Scene ID and Node ID.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CONTROLLER_REPLICATION							
Command = CTRL_REPLICATION_TRANSFER_SCENE							
Sequence Number							
Scene ID							
Node ID							
Level							

Sequence Number (8 bits)

Sequence Number of this particular command.

Scene ID (8 bits)

The scene ID is the parameter used to link together the different devices that takes part of a scene.

Node ID (8 bits)

The Node ID for a device that is part of the scene.

Level (8 bits)

The level is the parameter used for the specified scene.

4.40.4 Transfer Scene Name Command

The Transfer Scene Name Command is used to replicate scene names.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CONTROLLER_REPLICATION							
Command = CTRL_REPLICATION_TRANSFER_SCENE_NAME							
Sequence Number							
Scene ID							
Scene Name 1							
…							
Scene Name N							

Sequence Number (8 bits)

Sequence Number of this particular command.

Scene ID (8 bits)

Scene ID associated with a specific scene.

Scene Name (N bytes)

The Scene Name fields contain the assign scene name in ASCII characters.

4.41 CRC-16 Encapsulation Command Class, version 1

The CRC-16 Encapsulation Command Class is used to encapsulate a command with an additional CRC-16 checksum to ensure integrity of the payload. The purpose for this command class is to ensure a higher integrity level of payloads carrying important data using 9.6/40kbps communication, in case the LRC checksum (8 bits) provided on protocol level is not sufficient to ensure integrity.

The CRC-16 Encapsulation Command Class MUST NOT be encapsulated by Multi Command Command Class, Security Command Class, and CRC-16 Encapsulation Command Class.

Alternatives to using CRC-16 Encapsulation Command Class could be:

- Z-Wave Security solution to ensure privacy and integrity of data.
- A pure 100kbps solution that already provides a CRC-16 checksum solution on protocol level.

Devices supporting CRC-16 Encapsulation Command Class

A device that supports the CRC-16 Encapsulation Command Class MAY receive a combination of encapsulated and normal non-encapsulated requests and the response MUST be as follows:

a) If the request is sent encapsulated, the response MUST be returned encapsulated.

b) If the request is sent non-encapsulated, the response MUST be sent non-encapsulated.

A device that supports the CRC-16 Encapsulation Command Class MUST be able to receive and interpret the encapsulated version of all the command classes that it list in the NIF.

Devices controlling devices supporting CRC-16 Encapsulation Command Class

Before sending an encapsulated command to another node, the sending node MUST ensure, that the destination supports the CRC-16 Encapsulation Command Class.

4.41.1 CRC-16 Encapsulated Command

The CRC-16 Encapsulation Command is used to encapsulate a command with an additional checksum to ensure integrity of the payload. Be aware of the payload limitations with respect to a routed single cast frame.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_CRC_16_ENCAP							
Command = CRC_16_ENCAP							
Command Class							
Command							
Data 1							
…							
Data N							
Checksum 1							
Checksum 2							

Command Class (8 bits)

The Command Class field indicates the command class identifier of the encapsulated Command.

Command (8 bits)

The Command field indicates the Command identifier of the encapsulated Command.

Data (N bytes)

The Data fields in the encapsulated Command of the respective command class and Command identifiers.

Checksum (16 bits)

The checksum field used to ensure consistency of the command class identifier, command identifier, report number and firmware data downloaded. The checksum is derived from the bytes starting with the command class identifier = COMMAND_CLASS_CRC_16_ENCAP and until the last data field (Data N). The first byte Checksum 1 is the most significant byte. The checksum algorithm implements a CRC-CCITT using initialization value equal to 0x1D0F and 0x1021 (normal representation) as the poly. Refer to appendices in [2] with respect to CRC_CCITT source code.

4.41.2 Example

The BASIC GET command encapsulated in CRC-16 Encapsulation Command has the following format:

```
CRC-16 Encapsulation Command Class = 0x56
CRC-16 Encapsulation Command = 0x01
Basic Command Class = 0x20
Basic Get Command = 0x02
CRC1 = 0x4D
CRC2 = 0x26
```

4.42 Demand Control Plan Configuration Command Class, version 1

The Demand Control Plan Configuration Command Class allows Utility Suppliers to issue and manage a list of Demand Control Plan (DCP) events to the end consumer.

The DCP Configuration commands are separated for the DCP monitoring commands in the Demand Control Plan Monitor Command Class, allowing the classes to be optionally supported at different security levels. (E.g. DCP monitoring commands could be supported using non-secure communication, while enabling strict secure-only communication for the DCP Configuration Command Class). Refer to the Security and Security 2 command classes for more details.

A DCP event contains information regarding criticality, products involved, requested reduction, time duration and if a certain rate (identified by a Demand Control Plan Rate ID - please refer to the Rate Table Configuration Command Class of [2]) is associated with the event. When a DCP event is outdated, it is removed from the list. It is the utility supplier responsibility to prevent overflow of the list by query the number of free positions in the list before submitting a new DCP event to the list. Each DCP event is uniquely identified by a timestamp issued the Utility Supplier.

A DCP event may also include information, which enables devices not supporting this class to use in the Demand Control Plan through the Start & Stop Association Group functionality. The installer, the end user or Utility Supplier (remote management) configures the Association groups. During the configuration process the devices are selected and the association entries are created. These associations can additionally be configured with the specific Z-Wave commands (through the Association Command Configuration Command Class). If no Z-Wave commands are specified in the Associations groups, it is the responsibility of the device to issue the relevant commands based on Utility Supplier specific algorithms.

4.42.1 DCP List Supported Get Command

The DCP List Supported Get Command is used to request the total size of the DCP list along with the number of free entries in the list.

The DCP List Supported Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_DCP_CONFIG							
Command = DCP_LIST_SUPPORTED_GET							

4.42.2 DCP List Supported Report Command

The DCP List Supported Report Command used to provide the total size of the DCP list along with the number of free entries in the list.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_DCP_CONFIG							
Command = DCP_LIST_SUPPORTED_REPORT							
DCP List Size							
Free DCP List entries							

DCP List Size (8 bits)

This value specifies the DCP list size. 0x00 is reserved

Free DCP List entries (8 bits)

This value specifics the number of free entries for new DCP events. The value 0x00 specifies a full list.

4.42.3 DCP List Set Command

The DCP List Set Command used to place a new DCP event in the DCP list. Each DCP event is time stamped for future reference.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_DCP_CONFIG</td></tr>
<tr><td colspan="8">Command = DCP_LIST_SET</td></tr>
<tr><td colspan="8">Timestamp -Year 1</td></tr>
<tr><td colspan="8">Timestamp -Year 2</td></tr>
<tr><td colspan="8">Timestamp -Month</td></tr>
<tr><td colspan="8">Timestamp -Day</td></tr>
<tr><td colspan="8">Timestamp -Hour Local Time</td></tr>
<tr><td colspan="8">Timestamp -Minute Local Time</td></tr>
<tr><td colspan="8">Timestamp -Second Local Time</td></tr>
<tr><td colspan="8">DCP Rate ID</td></tr>
<tr><td colspan="6">Reserved</td><td colspan="2">Number of DC</td></tr>
<tr><td colspan="8">Generic Device Class 1</td></tr>
<tr><td colspan="8">Specific Device Class 1</td></tr>
<tr><td colspan="8">….</td></tr>
<tr><td colspan="8">Generic Device Class N</td></tr>
<tr><td colspan="8">Specific Device Class N</td></tr>
<tr><td colspan="8">Start Year 1</td></tr>
<tr><td colspan="8">Start Year 2</td></tr>
<tr><td colspan="8">Start Month</td></tr>
<tr><td colspan="8">Start Day</td></tr>
<tr><td colspan="8">Start Hour Local Time</td></tr>
<tr><td colspan="8">Start Minute Local Time</td></tr>
<tr><td colspan="8">Start Second Local Time</td></tr>
<tr><td colspan="8">Duration Hour Time</td></tr>
<tr><td colspan="8">Duration Minute Time</td></tr>
<tr><td colspan="8">Duration Second Time</td></tr>
<tr><td colspan="8">Event Priority</td></tr>
<tr><td colspan="8">Load shedding</td></tr>
<tr><td colspan="8">Start Association Group</td></tr>
<tr><td colspan="8">Stop Association Group</td></tr>
<tr><td colspan="8">Randomization interval</td></tr>
</table>

Timestamp -Year (16 bits)

Specify the year in the usual Gregorian calendar. The first byte (Year 1) is the most significant byte.

Timestamp -Month (8 bits)

Specify the month of the year between 01 (January) and 12 (December).

Timestamp -Day (8 bits)

Specify the day of the month between 01 and 31.

Timestamp -Hour Local Time (8 bits)

Specify the number of complete hours that have passed since midnight (00-23) in local time.

Timestamp -Minute Local Time (8 bits)

Specify the number of complete minutes that have passed since the start of the hour (00-59) in local time.

Timestamp -Second Local Time (8 bits)

Specify the number of complete seconds since the start of the minute (00-59) in local time. The value 60 used to keep UTC from wandering away is not supported.

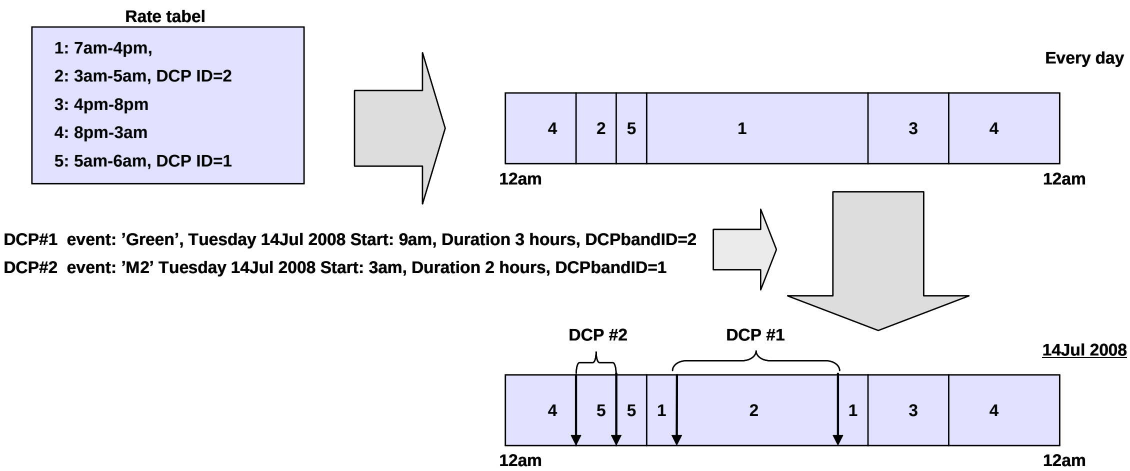
DCP Rate ID (8 bits)

Specify if a specific rate is applicable when participating in the DCP event. If an entry in the Rata table contains a matching DCP Rate ID this rate will be active for the duration of the event regardless of other parameters in the rate is not met.

<u>DCP Rate ID usage Example</u>

Prior to the DCP events a given rate table is configured defining when rates are active during a day. The Table contains two entries with DCP Rate IDs allowing the Utility Supplier to activate the rates outside of the time defined in the Table when a DCP event is placed in the DCP list with the corresponding DCP rate ID.

Prior to Jul14 2008 two DCP events are placed in the List by the Utility Supplier, which changes the Rate profile of Jul14 compared to an ‘standard’ day.



Randomization interval (8 bits)

Specify the randomization interval in units of seconds, which must be applied as an offset to the start and stopping of the events as requested in the DCP event.

E.g. A value of 0x10 specifies that every device should randomly select a start and stop time offset between 0 and 16 seconds. This offset SHOULD be applied to the start and duration fields in the DCP event.

Number of DC (2 bits)

Specify the number of Generic/Specific Device Classes, which are requested to participate in the DCP event

Generic Device Class (8bits)

Specify the Generic Device Class identifier. Please refer to [1].

Specific Device Class (8 bits)

Specify the Specific Device Class identifier. Please refer to [1].

Start Year (16 bits)

Specify the year in the usual Gregorian calendar for the start of the event. The first byte (Year 1) is the most significant byte.

Start Month (8 bits)

Specify the month of the year between 01 (January) and 12 (December) for the start of the event.

Start Day (8 bits)

Specify the day of the month for the start of the event between 01 and 31.

Start Hour Local Time (8 bits)

Specify the number of complete hours that have passed since midnight (00-23) in local time for the start of the event.

Start Minute Local Time (8 bits)

Specify the number of complete minutes that have passed since the start of the hour (00-59) in local time for the start of the event.

Start Second Local Time (8 bits)

Specify the number of complete seconds since the start of the minute (00-59) in local time. The value 60 used to keep UTC from wandering away is not supported.

Duration Hour Time (8 bits)

Specify the number of complete hours of the event.

Duration Minute Time (8 bits)

Specify the number of complete minutes of the event.

Duration Second Time (8 bits)

Specify the number of complete seconds of the event.

Event Priority (8 bits)

The parameter specifies the priority of the DCP event. The High priority is used by the utility Supplier to mandate device participation and lower priorities are used by devices to voluntary to participate in the event and to which degree.

Event Priority	Description	Device participation
0x00	Reserved	Voluntary
0x01	V1 - Green Energy	Voluntary
0x02	V2	Voluntary
0x03	V3	Voluntary
0x04	V4	Voluntary
0x05	V5	Voluntary
0x06	V6	Voluntary
0x07	V7	Voluntary
0x08	M1 - Emergency	Mandatory
0x09	M2	Mandatory
0x0A	M3	Mandatory
0x0B	M4	Mandatory
0x0C	Utility defined	Utility defined
0x0D	Utility defined	Utility defined
0x0E	Utility defined	Utility defined
0x0F	Utility defined	Utility defined

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Load shedding (8 bits)

Specify the load shedding in percentage of average consumption requested for the event. This load shedding will be applied to the devices with the device classes specified in the command.

This field MUST be in the range 0x01..0x64, which represents values in the range 1%..100%. All other values are reserved. Reserved values MUST NOT be used by a sending node and MUST be ignored by a receiving node.

Start Association group (8 bits)

Specify which association group (see the Association Command Class) should be activated when the event is started. The value 0x00 specifies no group should be activated.

If the device also supports the Association Command Configuration Command Class it possible to dynamically and remotely (from the Energy Supplier or others) to specify precisely which current and future Z-Wave commands should be send to which Z-Wave nodes.

The configuration of the Start Association group is done either manually by the installer or automatically when the device detect other relevant devices

Stop Association group (8 bits)

Specify which association group (see the Association Command Class) should be activated when the event is stopped. The value 0x00 specifies no group should be activated.

The configuration of the Stop Association group is done either manually by the installer or automatically when the device detect other relevant devices

<u>Start Associating group and Stop Associating Group usage example</u>

A small energy control system consisting of a device supporting the DCP command class and 3 z-wave devices which can participate in the application. NodeId 1: Setback Thermostat device, NodeId 8: Simple Thermostat device, NodeId 5: Multilevel Power Switch device.

Through the use of the Association Command Class and the Association Command Configuration Command Class the following Associations has been established in the device.

Group 1
Node1, Thermostat_Setback_SetPpermanent override,energy saving mode)
Node8, Thermostat_Setpoint_Set(Heating setpoint#1, 19,5^{o}C)
Node5, Multilevel_Switch:Set (Dimlevel =0x20)

Group 2
Node1, Thermostat_Setback_Set(No override,Setback = 0^{o}C)
Node8, Thermostat_Setpoint_Set(Heating setpoint#1, 21,5^{o}C)
Node5, Multilevel_Switch:Set (Dimlevel =0x40)

A DCP event can now specifically invoke Group1 when starting the event and invoking group2 when stopping the event by specifying Start Association Group=0x01 and Stop Association Group=0x02.

4.42.4 DCP List Remove

The DCP List Remove Command used to remove a DCP event from the DCP list.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_DCP_CONFIG							
Command = DCP_LIST_REMOVE							
Timestamp -Year 1							
Timestamp -Year 2							
Timestamp -Month							
Timestamp -Day							
Timestamp -Hour Local Time							
Timestamp -Minute Local Time							
Timestamp -Second Local Time							

Please refer to DCP List report command class (section 4.43.2) for description of fields

4.43 Demand Control Plan Monitor Command Class, version 1

The Demand Control Plan Monitor Command Class allows devices to monitor the list of Demand Control Plan (DCP). A DCP event contains information regarding criticality, products involved, requested reduction, time duration and if a certain rate (identified by a Demand Control Plan Rate ID - please refer to the Rate Table Configuration Command Class of [2]) is associated with the event. When a DCP event is outdated, it is removed from the list. Each DCP event is uniquely identified by a timestamp issued the Utility Supplier.

4.43.1 DCP List Get Command

The DCP List Get Command is used to request the pending DCP event in a device.

The DCP List Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_DCP_MONITOR							
Command = DCP_LIST_GET							

4.43.2 DCP List Report Command

The DCP List Report Command reports the pending DCP event in a device. If more than one DCP event is pending – the reports will be submitted in chronically order as to when the events was placed on the list. Newest entry will be reported first.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_DCP_MONITOR							
Command = DCP_LIST_REPORT							
Reports to Follow							
Timestamp -Year 1							
Timestamp -Year 2							
Timestamp -Month							
Timestamp -Day							
Timestamp -Hour Local Time							
Timestamp -Minute Local Time							
Timestamp -Second Local Time							
DCP ID							
Reserved						Number of DC	
Generic Device Class 1							
Specific Device Class 1							
….							
Generic Device Class N							
Specific Device Class N							
Start Year 1							
Start Year 2							
Start Month							
Start Day							
Start Hour Local Time							
Start Minute Local Time							
Start Second Local Time							
Duration Hour Time							
Duration Minute Time							
Duration Second Time							
Event Priority							
Load shedding							
Start Association Group							
Stop Association Group							
Randomization interval							

Reports to Follow (8 bits)

This value indicates how many report frames there are left, the value 0xFF means that the number of reports have not been calculated yet or that there is more than 255 reports to follow.

Please refer to DCP List Set command class (section 4.42.3) for detailed description of the fields.

4.43.3 DCP Event Status Get

The DCP Event Status Get Command used query the status of a specific DCP event in the DCP list.

The DCP Event Status Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing.
The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_DCP_MONITOR							
Command = DCP_EVENT_STATUS_GET							
Timestamp -Year 1							
Timestamp -Year 2							
Timestamp -Month							
Timestamp -Day							
Timestamp -Hour Local Time							
Timestamp -Minute Local Time							
Timestamp -Second Local Time							

Please refer to DCP List Set command class (section 4.42.3) for detailed description of the fields.

4.43.4 DCP Event Status Report

The DCP Event Status Report Command used to provide the status of a specific DCP event in the DCP list.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_DCP_MONITOR							
Command = DCP_EVENT_STATUS_GET							
Timestamp -Year 1							
Timestamp -Year 2							
Timestamp -Month							
Timestamp -Day							
Timestamp -Hour Local Time							
Timestamp -Minute Local Time							
Timestamp -Second Local Time							
Event status							

Please refer to DCP List report command class (section 4.43.2) for description of fields

Event Status (8 bits)

The field contains the status of the event.

Event status	Description
0x00	Reserved
0x01	Event Started
0x02	Event Completed
0x03	Event Rejected by the user
0x04	Event not Applicable
0x05-0xFF	Reserved

4.44 Device Reset Locally Command Class, version 1

The Device Reset Locally Command Class is used to notify central controllers that a Z-Wave device is resetting its network specific parameters.

Any device SHOULD implement this command class.

Z-Wave Plus compliant devices MUST implement this command class.

4.44.1 Device Reset Locally Notification Command

The Device Reset Locally Notification Command is used to advertise that the device will be reset to defaultThe reset operation MUST reset protocol data (HomeID, NodeID, etc.) as well as all application specific data to factory default values. In case a Lifeline destination is configured in Association Group #1, the device MUST send a Device Reset Locally Notification Command to the Lifeline destination.

A Multi Channel device MUST send the Device Reset Locally Notification Command without Multi Channel Encapsulation.

A controller device receiving the Device Reset Locally Notification Command SHOULD consider the sending node to be a failing node and accordingly perform relevant maintenance operations like removing failing nodes, removing associations to failing nodes, etc.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_DEVICE_RESET_LOCALLY							
Command = DEVICE_RESET_LOCALLY_NOTIFICATION							

4.45 Door Lock Command Class, version 1-2

The Door Lock Command Class is used to operate and configure a door lock device.

The Door Lock Command Class is an actuator control command class. Refer to 3.7.

4.45.1 Compatibility considerations

A device supporting Door Lock CC, Version 2 MUST support Door Lock CC, version 1.

The Door Lock CC, Version 2 adds the "unknown" state to the Door Lock Operation Report command.

A door lock device MAY implement a subset of the features represented by the Door Lock Mode, Door Handles Mode and Door Condition fields which are provided by the commands of this command class.

4.45.2 Door Lock Operation Set Command

The Door Lock Operation Set command, version 1 is used to set the operation mode of a supporting door lock device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_DOOR_LOCK							
Command = DOOR_LOCK_OPERATION_SET							
Door Lock Mode							

Door Lock Mode (8 bits)

The Door Lock Mode field is used to specify the operation mode of the door lock device.

The encoding of the Door Lock Mode field MUST be according to Table 46.

Table 46, Door Lock Operation Set :: Mode

Mode	Description
0x00	Door Unsecured 1)
0x01	Door Unsecured with timeout 2)
0x10	Door Unsecured for inside Door Handles 1)
0x11	Door Unsecured for inside Door Handles with timeout 2)
0x20	Door Unsecured for outside Door Handles 1)
0x21	Door Unsecured for outside Door Handles with timeout 2)
0xFF	Door Secured

1) Constant mode. Door will be unsecured until set to secured mode by another command.
2) Timeout mode. Fallback to secured mode after timeout has expired.

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

A controlling device MUST NOT specify modes using timeout-based fall back if the Operation Type of the supporting device is set to Constant operation (set by Door Lock Configuration Set).
A supporting device MUST ignore modes using timeout-based fall back if Operation Type is set to Constant operation.

A receiving device MAY implement a subset of the Door Lock Modes defined by Table 46.
A receiving device MUST accept the Door Lock Mode values 0x00 and 0xFF.

4.45.3 Door Lock Operation Get Command

The Door Lock Operation Get command, version 1 is used to request the status of a door lock device.

The Door Lock Operation Report command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_DOOR_LOCK							
Command = DOOR_LOCK_OPERATION_GET							

4.45.4 Door Lock Operation Report Command

The Door Lock Operation Report command, version 1 is used to advertise the status of a door lock device.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_DOOR_LOCK</td></tr>
<tr><td colspan="8">Command = DOOR_LOCK_OPERATION_REPORT</td></tr>
<tr><td colspan="8">Door Lock Mode</td></tr>
<tr><td colspan="4">Outside Door Handles Mode</td><td colspan="4">Inside Door Handles Mode</td></tr>
<tr><td colspan="8">Door Condition</td></tr>
<tr><td colspan="8">Lock Timeout Minutes</td></tr>
<tr><td colspan="8">Lock Timeout Seconds</td></tr>
</table>

Door Lock Mode (8 bits)

The Door Lock Mode field MUST advertise the mode of the door lock device.

The encoding of the Door Lock Mode field MUST be according to Table 47.

Table 47, Door Lock Operation Report :: Door Lock Mode

Mode	Description
0x00	Door Unsecured 1) (Version 1)
0x01	Door Unsecured with timeout 2) (Version 1)
0x10	Door Unsecured for inside Door Handles 1) (Version 1)
0x11	Door Unsecured for inside Door Handles with timeout 2) (Version 1)
0x20	Door Unsecured for outside Door Handles 1) (Version 1)
0x21	Door Unsecured for outside Door Handles with timeout 2) (Version 1)
0xFE	Door/Lock State Unknown 3) (Version 2)
0xFF	Door Secured (Version 1)

1) Constant mode. Door will be unsecured until set back to secured mode by command
2) Timeout mode. Fallback to secured mode after timeout has expired
3) Bolt is not fully retracted/engaged

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

A sending device MAY advertise a subset of the available Door Lock Mode values.
A receiving device MUST accept all Door Lock Mode values.

The Door Lock Mode field SHOULD advertise the current value of the device hardware; also while in transition to a new target value.

Outside Door Handles Mode (4 bits)

This field MUST advertise the status of each individual outside door handle.

The encoding of the Outside Door Handles Mode bitmask field MUST be according to Table 48 and Table 49.

Table 48, Door Handles Mode bitmask

Bit 3	Bit 2	Bit 1	Bit 0
Handle 4	Handle 3	Handle 2	Handle 1

Table 49, Door Handles Mode bit encoding

Bit value	Description
‘0’	Disabled
‘1’	Enabled

The value ‘0’ MUST signify that the actual handle cannot open the door locally.
The value ‘1’ MUST signify that the actual handle can open the door locally.

A sending device MAY advertise just a subset of the available Door Handles Mode values.
A receiving device MUST accept all Door Handles Mode values.

Inside Door Handles Mode (4 bits)

This field MUST advertise the status of each individual inside door handle.

The encoding of the Inside Door Handles Mode bitmask field MUST be according to Table 48 and Table 49.

A sending device MAY advertise just a subset of the available Door Handles Mode values.
A receiving device MUST accept all Door Handles Mode values.

Door Condition (8 bits)

The Door Condition field MUST advertise the status of the door lock components.

The encoding of the Door Condition bitmask field MUST be according to Table 50.

Table 50, Door Condition bitmask

	Bit 7..3	Bit 2	Bit 1	Bit 0
bit = ‘0’	*Reserved*	Latch Open	Bolt Locked	Door Open
bit = ‘1’	*Reserved*	Latch Closed	Bolt Unlocked	Door Closed

All other bits are reserved and MUST be set to zero by a sending node. Reserved bits MUST be ignored by a receiving node.

A sending device MAY advertise just a subset of the available Door Condition values.
A receiving device MUST accept all Door Condition values.

Lock Timeout Minutes (8 bits)

This field MUST advertise the remaining time before the door lock will automatically be locked again.

The encoding of the Lock Timeout Minutes field MUST be according to Table 51. The time the supporting device stays unlocked MUST be determined by combining the Lock Timeout Minutes and Lock Timeout Seconds fields.

Table 51, Door Lock Operation Report :: Lock Timeout Minutes

Value	Operation
0..253 (0x01..0xFC)	Unlocked 0 .. 253 minutes (Operation Type = Timed Operation)
254 (0xFE)	No unlocked period (Operation Type = Constant Operation)

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

A sending device MAY advertise just a subset of the available Lock Timeout values.
A receiving device MUST accept all Lock Timeout values.

Lock Timeout Seconds (8 bits)

This field MUST advertise the remaining time before the door lock will automatically be locked again.

The encoding of the Lock Timeout Seconds field MUST be according to Table 52. The time to stay unlocked MUST be determined by combining the Lock Timeout Minutes and Lock Timeout Seconds fields.

Table 52, Door Lock Operation Report :: Lock Timeout Seconds

Value	Operation
0..59 (0x00..0x3B)	Unlocked 0 .. 59 seconds (Operation Type = Timed Operation)
254 (0xFE)	No unlocked period (Operation Type = Constant Operation)

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

A sending device MAY advertise just a subset of the available Lock Timeout values.
A receiving device MUST accept all Lock Timeout values.

4.45.5 Door Lock Configuration Set Command

The Door Lock Configuration Set command, version 1 is used to set the configuration of a supporting door lock device.

A door lock device MUST be able to operate with the factory default settings.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_DOOR_LOCK							
Command = DOOR_LOCK_CONFIGURATION_SET							
Operation Type							
Outside Door Handles Mode				Inside Door Handles Mode			
Lock Timeout Minutes							
Lock Timeout Seconds							

Operation Type (1 byte)

The Operation Type field MUST be set according to Table 53. When timed operation is specified, the Lock Timeout Minutes and Lock Timeout Seconds fields MUST be set to valid values.

Table 53, Door Lock Operation Type

Operation Type	Description	Valid Lock Timeout values
0x01	Constant operation	Minutes = 0xFE Seconds= 0xFE
0x02	Timed operation	Minutes = 0x00..0xFD Seconds= 0x00..0x3B

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

A receicing device MAY accept just a subset of the available Operation Type values.
A receiving device MUST accept the Operation Type value 0x01.

Outside Door Handles Mode (4 bits)

This field is used to advertise the status of each individual outside door handle.

The encoding of the Outside Door Handles Mode bitmask field MUST be according to Table 48 and Table 49.

A receiving device MAY ignore the Door Handles Mode value.

Inside Door Handles Mode (4 bits)

This field MUST advertise the status of each individual inside door handle.

The encoding of the Inside Door Handles Mode bitmask field MUST be according to Table 48 and Table 49.

A receiving device MAY ignore the Door Handles Mode value.

Lock Timeout Minutes (1 byte)

This field MUST specify the time that a door lock must wait before automatically being locked again.

The encoding of the Lock Timeout Minutes field MUST be according to Table 51. The time to stay unlocked MUST be determined by combining the Lock Timeout Minutes and Lock Timeout Seconds fields.

A receiving device MAY ignore the Lock Timeout values if it implements only the Operation Type = 0x01 (Constant operation).

Lock Timeout Seconds (1 byte)

This field MUST specify the time that a door lock must wait before automatically being locked again.

The encoding of the Lock Timeout Seconds field MUST be according to Table 52. The time to stay unlocked MUST be determined by combining the Lock Timeout Minutes and Lock Timeout Seconds fields.

A receiving device MAY ignore the Lock Timeout values if it implements only the Operation Type = 0x01 (Constant operation).

4.45.6 Door Lock Configuration Get Command

The Door Lock Configuration Get command, version 1 is used to request the configuration parameters of a door lock device.

The Door Lock Configuration Report command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing.
The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_DOOR_LOCK							
Command = DOOR_LOCK_CONFIGURATION_GET							

4.45.7 Door Lock Configuration Report Command

The Door Lock Configuration Report command, version 1 is used to advertise the configuration parameters of a door lock device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_DOOR_LOCK							
Command = DOOR_LOCK_CONFIGURATION_REPORT							
Operation Type							
Outside Door Handles Mode				Inside Door Handles Mode			
Lock Timeout Minutes							
Lock Timeout Seconds							

Refer to Door Lock Configuration Set for details on fields.

4.46 Door Lock Command Class, version 3

The Door Lock Command Class is used to operate and configure a door lock device.

The Door Lock Command Class is an actuator control command class. Refer to chapter 3.7.

Commands not described in Version 3 stays unchanged from Version 2.

4.46.1 Compatibility considerations

A device supporting Door Lock CC, Version 3 MUST support Door Lock CC, version 2.

Version 3 adds duration and target value reporting to the Door Lock Operation Report command.

4.46.2 Door Lock Operation Report Command

The Door Lock Operation Report command, version 3 is used to advertise the status of a door lock device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_DOOR_LOCK							
Command = DOOR_LOCK_OPERATION_REPORT							
Current Door Lock Mode							
Outside Door Handles Mode				Inside Door Handles Mode			
Door Condition							
Lock Timeout Minutes							
Lock Timeout Seconds							
Target Door Lock Mode							
Duration							

Current Door Lock Mode (8 bits)

The Current Door Lock Mode field MUST advertise the current mode of the door lock device.

The encoding of the Current Door Lock Mode field MUST be according to Table 47.

Table 54, Door Lock Operation Report :: Door Lock Mode

Mode	Description
0x00	Door Unsecured 1) (Version 1)
0x01	Door Unsecured with timeout 2) (Version 1)
0x10	Door Unsecured for inside Door Handles 1) (Version 1)
0x11	Door Unsecured for inside Door Handles with timeout 2) (Version 1)
0x20	Door Unsecured for outside Door Handles 1) (Version 1)
0x21	Door Unsecured for outside Door Handles with timeout 2) (Version 1)
0xFE	Door/Lock Mode Unknown 3) (Version 2)
0xFF	Door Secured (Version 1)

1) Constant mode. Door will be unsecured until set back to secured mode by command

2) Timeout mode. Fallback to secured mode after timeout has expired

3) Bolt is not fully retracted/engaged

All other values are reserved. Reserved values MUST NOT be used by a sending node and MUST be ignored by a receiving node.

An Operation Set command may be used to initiate a transition to a new target mode. The device may be queried for its current mode while in transition to a new mode. The response to an Operation Get SHOULD be the current mode of the device hardware, e.g. Door/Lock State Unknown. (applies to Version 2 and later).

Outside Door Handles Mode (4 bits)

This field MUST advertise the status of each individual outside door handle.

The encoding of the Outside Door Handles Mode bitmask field MUST be according to Table 48 and Table 49.

Table 55, Door Handles Mode bitmask

Bit 3	Bit 2	Bit 1	Bit 0
Handle 4	Handle 3	Handle 2	Handle 1

Table 56, Door Handles Mode bit encoding

Bit value	Description
‘0’	Disabled
‘1’	Enabled

The value ‘0’ MUST signify that the actual handle cannot open the door locally.
The value ‘1’ MUST signify that the actual handle can open the door locally.

Inside Door Handles Mode (4 bits)

This field MUST advertise the status of each individual inside door handle.

The encoding of the Inside Door Handles Mode bitmask field MUST be according to Table 48 and Table 49.

Door Condition (8 bits)

The Door Condition field MUST advertise the status of the door lock components.

The encoding of the Door Condition bitmask field MUST be according to Table 57.

Table 57, Door Condition bitmask

	Bit 7..3	Bit 2	Bit 1	Bit 0
bit = ‘0’	*Reserved*	Latch Open	Bolt Locked	Door Open
bit = ‘1’	*Reserved*	Latch Closed	Bolt Unlocked	Door Closed

Reserved bits MUST be set to zero by a sending node. Reserved bits MUST be ignored by a receiving node.

Lock Timeout Minutes (8 bits)

This field MUST advertise the remaining time before the door lock will automatically be locked again.

The encoding of the Lock Timeout Minutes field MUST be according to Table 51. The time the supporting device stays unlocked MUST be determined by combining the Lock Timeout Minutes and Lock Timeout Seconds fields.

Table 58, Door Lock Operation Report :: Lock Timeout Minutes

Value		Operation
Decimal	hex	
0	(0x00)	Stay unlocked 0 minutes (Operation Type = Timed Operation)
..	..	
253	(0xFD)	Stay unlocked 253 minutes (Operation Type = Timed Operation)
254	(0xFE)	No unlocked period (Operation Type = Constant Operation)

All other values are reserved. Reserved values MUST NOT be used by a sending node and MUST be ignored by a receiving node.

Lock Timeout Seconds (8 bits)

The encoding of the Lock Timeout Seconds field MUST be according to Table 52. The time the supporting device stays unlocked MUST be determined by combining the Lock Timeout Minutes and Lock Timeout Seconds fields.

Table 59, Door Lock Operation Report :: Lock Timeout Seconds

Value		Operation
Decimal	hex	
0	(0x00)	Stay unlocked 0 seconds (Operation Type = Timed Operation)
..	..	
59	(0x3B)	Stay unlocked 59 seconds (Operation Type = Timed Operation)
254	(0xFE)	No unlocked period (Operation Type = Constant Operation)

All other values are reserved. Reserved values MUST NOT be used by a sending node and MUST be ignored by a receiving node.

Target Door Lock Mode (8 bits)

The Target Door Lock Mode field MUST advertise the target mode of an ongoing transition or the most recent transition.

The encoding of the Target Door Lock Mode field MUST be according to Table 47.

Duration (8 bits)

The Duration field SHOULD advertise the remaining time before the target mode is reached. The encoding of the Duration field MUST be according to Table 60.

Table 60, Door Lock Operation Report :: Duration

Duration	Description
0x00	0 seconds. Already at the Target Value.
0x01-0x7F	1 second (0x01) to 127 seconds (0x7F) in 1 second resolution.
0x80-0xFD	1 minute (0x80) to 126 minutes (0xFD) in 1 minute resolution.
0xFE	Unknown duration
0xFF	Reserved

All other values are reserved. Reserved values MUST NOT be used by a sending node and MUST be ignored by a receiving node.

4.47 Door Lock Logging Command Class, version 1

This Door Lock Logging Command Class provides an audit trail in an access control application. Each time an event takes place at the door lock, the system logs the user's ID, date, time etc.

4.47.1 Door Lock Logging Records Supported Get Command

The Door Lock Logging Records Supported Get Command is used to request the number of records that the audit trail supports.

The Door Lock Logging Records Supported Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_DOOR_LOCK_LOGGING							
Command = DOOR_LOCK_LOGGING_RECORDS_SUPPORTED_GET							

4.47.2 Door Lock Logging Records Supported Report Command

The Door Lock Logging Records Supported Report Command may be used to report the maximum number of reports the audit trail supports.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_DOOR_LOCK_LOGGING							
Command = DOOR_LOCK_LOGGING_RECORDS_SUPPORTED_REPORT							
Max records stored							

Max records stored (8 bits)

The number of records the audit trail supports stored in a queue.

4.47.3 Door Lock Logging Record Get Command

The Door Lock Logging Record Get Command is used to request the audit trail.

The Door Lock Logging Record Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_DOOR_LOCK_LOGGING							
Command = RECORD_GET							
Record number							

Record number (8 bits)

The Record number field indicates the record to be requested.

A value 0 – *Max records stored* is acceptable with a value of 0 being the most recent entry. When requesting with a value of 0, the report will contain the record number so the latest record is known.

4.47.4 Door Lock Logging Record Report Command

The Door Lock Logging Record Report Command returns records from the audit trail.

To provide flexibility the user associated with the record may be identified by one of two methods: the user identifier or the user code.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_DOOR_LOCK_LOGGING</td></tr>
<tr><td colspan="8">Command = RECORD_REPORT</td></tr>
<tr><td colspan="8">Record number</td></tr>
<tr><td colspan="8">Timestamp - Year 1</td></tr>
<tr><td colspan="8">Timestamp - Year 2</td></tr>
<tr><td colspan="8">Timestamp – Month</td></tr>
<tr><td colspan="8">Timestamp – Day</td></tr>
<tr><td colspan="3">Record status</td><td colspan="5">Timestamp - Hour Local Time</td></tr>
<tr><td colspan="8">Timestamp - Minute Local Time</td></tr>
<tr><td colspan="8">Timestamp - Second Local Time</td></tr>
<tr><td colspan="8">Event Type</td></tr>
<tr><td colspan="8">User Identifier</td></tr>
<tr><td colspan="8">User Code Length</td></tr>
<tr><td colspan="8">USER_CODE 1</td></tr>
<tr><td colspan="8">..</td></tr>
<tr><td colspan="8">USER_CODE N</td></tr>
</table>

Record number (8 bits)

Record number requested (1- 255).

Timestamp - Year (16 bits)

Specify the year in the usual Gregorian calendar. The first byte (Year 1) is the most significant byte.

Timestamp - Month (8 bits)

Specify the month of the year between 01 (January) and 12 (December).

Timestamp - Day (8 bits)

Specify the day of the month between 01 and 31.

Record status (3-bits)

The Record Status field is used to indicate if legal data is stored in the record.

Record State	Description
0	Requested record is empty.
1	Requested record holds legal data.

Timestamp - Hour Local Time (8 bits)

Specify the number of complete hours that have passed since midnight (00-23) in local time.

Timestamp - Minute Local Time (8 bits)

Specify the number of complete minutes that have passed since the start of the hour (00-59) in local time.

Timestamp - Second Local Time (8 bits)

Specify the number of complete seconds since the start of the minute (00-59) in local time. The value 60 used to keep UTC from wandering away is not supported.

Note: If RTC (Real Time Clock) all values in Time Stamp record SHOULD be set to 0. Most recent Record MUST be stored under Record Number 1.

Event Type (8 bits)

Following Events are supported:

Event type	Description
1	Lock Command: Keypad access code verified lock command
2	Unlock Command: Keypad access code verified unlock command
3	Lock Command: Keypad lock button pressed
4	Unlock command: Keypad unlock button pressed
5	Lock Command: Keypad access code out of schedule
6	Unlock Command: Keypad access code out of schedule
7	Keypad illegal access code entered
8	Key or latch operation locked (manual)
9	Key or latch operation unlocked (manual)
10	Auto lock operation
11	Auto unlock operation
12	Lock Command: Z-Wave access code verified
13	Unlock Command: Z-Wave access code verified
14	Lock Command: Z-Wave (no code)
15	Unlock Command: Z-Wave (no code)
16	Lock Command: Z-Wave access code out of schedule
17	Unlock Command Z-Wave access code out of schedule
18	Z-Wave illegal access code entered
19	Key or latch operation locked (manual)
20	Key or latch operation unlocked (manual)
21	Lock secured
22	Lock unsecured
23	User code added
24	User code deleted
25	All user codes deleted
26	Master code changed
27	User code changed
28	Lock reset
29	Configuration changed
30	Low battery
31	New Battery installed

Not all events types are supported and it is up to manufacturer to decide which one are to be supported.

User Identifier (8 bits)
The User Identifier is used to recognize the user identity. A valid User Identifier MUST be a value starting from 1 to the maximum number of users supported by the device; refer to the User Code Command Class. A User Identifier of 0 is acceptable when the record does not need to identify a user or if the User Code is provided in this report.

User Code Length (8 bits)

The User Code Length field indicates the number of bytes used to hold the User Code. A length of 0 is acceptable when the record does not need to identify a User Code or when the User Identifier field is non-zero.

USER_CODE (N bytes)

These fields contain the user code. Minimum code length is 4 and maximum 10 ASCII digits. For further details about the user code, refer to the User Code Command Class.

4.48 Energy Production Command Class, version 1

The Energy Production Command Class is used to retrieve various production data from the device.

4.48.1 Energy Production Get Command

The Energy Production Get Command is used to request various production data from the device.

The Energy Production Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ENERGY_PRODUCTION							
Command = ENERGY_PRODUCTION_GET							
Parameter Number							

Parameter Number (8 bits)

The parameter number specifies the kind of production data to retrieve. Currently the following parameter numbers are defined:

Parameter Number	Description
0x00	Instant energy production
0x01	Total energy production
0x02	Energy production today
0x03	Total production time

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

4.48.2 Energy Production Report Command

The Energy Production Report Command used to retrieve various production data from the device.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_ENERGY_PRODUCTION</td></tr>
<tr><td colspan="8">Command = ENERGY_PRODUCTION_REPORT</td></tr>
<tr><td colspan="8">Parameter Number</td></tr>
<tr><td colspan="3">Precision</td><td colspan="2">Scale</td><td colspan="3">Size</td></tr>
<tr><td colspan="8">Value 1</td></tr>
<tr><td colspan="8">…</td></tr>
<tr><td colspan="8">Value N</td></tr>
</table>

Parameter Number (8 bits)

Refer to description under the Energy Production Get Command.

Precision (3 bits)

The precision field describes what the precision of the value is. The number indicates the number of decimals. The decimal value 1025 with precision 2 is therefore equal to 10.25.

Scale (2 bits)

The Scale field indicates the scale used for the specified parameter number:

Parameter Number	Scale	Description
0x00	0x00	W
0x01	0x00	Wh
0x02	0x00	Wh
0x03	0x00	Seconds
	0x01	Hours

Size (3 bits)

The size field indicates the number of bytes used for the value. This field can take values from 1 (001b), 2 (010b) or 4 (100b).

Value (N bytes)

The value is a signed field. The field MAY be 1, 2 or 4 bytes in size. The first byte is the most significant byte. The table below shows signed decimal values together with their hexadecimal equivalents.

Signed 1 byte decimal value	Hexadecimal		Signed 2 bytes decimal value	Hexadecimal
127	0x7F		32767	0x7FFF
25	0x19		1025	0x0401
2	0x02		2	0x0002
1	0x01		1	0x0001
0	0x00		0	0x0000
-1	0xFF		-1	0xFFFF
-2	0xFE		-2	0xFFFE
-25	0xE7		-1025	0xFBFF
-128	0x80		-32768	0x8000

4.49 Entry Control Command Class, version 1

The Entry Control Command Class defines a method for advertising user input to a central Entry Control application and for the discovery of capabilities. User input may be button presses, RFID tags or other means.

It is RECOMMENDED that this command class is only supported via secure communication. This recommendation may be raised to a stronger requirement at the Device Class or Device Type level.

The Entry Control Command Class provides the following commands:

- Key Supported Get/Report – Used by the controller to request which ASCII keys are present on the Entry Control device.
- Event Supported Get/Report – Used by the controller to request which Event Types are supported by the Entry Control device.
- Configuration Set/Get/Report – Used by the controller to configure the Entry Control device
- Notification Command – Used by the Entry Control device to notify the controller of inputs.

4.49.1 Interoperability Considerations

It is RECOMMENDED that an Entry Control input device that implements the Entry Control Command Class also implements the Indicator Command Class, version 2.

If implementing the Indicator Command Class, version 2, it is RECOMMENDED that all indicator resources are addressable via the Indicator Command Class, version 2.
The input device MAY implement local control of indicator resources. If the Indicator Command Class, version 2 is supported, the first received Indicator Set command MUST disable all local control of indicator resources – except for interface feedback functionality like a short beep on each button press.

Transmissions may fail due to central control application fault or due to RF jamming. It is RECOMMENDED that an Entry Control input device provides local user feedback if the transmission of notifications to the central control application fails.

If the device only features light indicators, it is RECOMMENDED that all light flashes at 2Hz for at least 5 seconds to indicate transmission error.

If the device features a buzzer, it is RECOMMENDED that the buzzer generates sound pulses at a rate of 2Hz for at least 5 seconds to indicate transmission error.

If implementing the indicator Command Class, version 2, in devices based on Role Type RSS (using Wakeup Command Class), the control of indicators must be synchronized with the Wakeup Notification. The controller must therefore wait for the Wakeup Notification, before it can control the indictors. The device implementing the indicator Command Class must therefore send the Wakeup Notification after sending the Entry Control Notification Command, in case the device expects to be controlled.

4.49.2 Security Considerations

An attacker could theoretically determine the length of manually entered user credentials even if they are transmitted via encryption. It is therefore MANDATORY to add padding bytes to ASCII strings so that all transferred ASCII strings are structured as one or more blocks of 16 characters. The ASCII code 0xFF MUST be reserved for padding purposes and it MUST NOT be used for any other purposes.

4.49.3 Handling user supplied data

The Entry Control device must cache the user input before sending the full entry in one Notification Frame. A user input MUST therefore have a termination, which MAY be determined by:

- The Key Cache Size is exceeded
- The Key Cache Timeout is exceeded
- A Command Button is pressed
- User data is received by other means, e.g. from an RFID tag

Key Cached Size:

The Key Cached Size is configured to specify the number of user inputs before the Notification Frame is sent. After sending the Notification Frame the cache must be cleared and subsequent user inputs MUST be considered a new entry.

The Key Cached Size May be configured to 1, in which case a Notification Frame is send for each user input.

It is RECOMMENDED to have a default Key Cached Size of 4, in which case the Notification Frame will be send after 4 entries.

Key Timeout:

The Key Timeout is configured to specify the maximum time between user inputs. If the time between user inputs exceeds the Timeout, the cached user inputs will be send in a Notification Frame. After sending the Notification Frame the cache must be cleared and subsequent user inputs MUST be considered a new entry.

Based on Command Button:

A user input may be terminated by the user pressing on of the Command Buttons like ENTER or ARM_ALL. The cached entry will be sent in a Notification Frame immediately after the user presses the Command Button. After sending the Notification Frame the cache must be cleared and subsequent user inputs MUST be considered a new entry.

Based on RFID:

When presenting an RFID tag, the ID will be read from the tag, and this terminates the user input. So the cached ID ‘s from the RFID tag data can immediately be sent in a Notification Frame. After sending the Notification Frame the cache must be cleared and subsequent user inputs MUST be considered a new entry.

Presenting an RFID tag may also terminate an ongoing user input. For instance, a user may enter four characters and then present an RFID tag. In that case, the four entries must first be sent in one Notification Frame, and subsequently the ID ‘s from the RFID tag data is sent in a second Notification Frame.

4.49.4 Handling Incorrect Entry

A user may do an incorrect entry e.g. the credential is 1234 but the user enters 1734. In this case the terminal may provide a “delete” option, to allow the user to fix or re-enter the credential. If a “delete” option is not provided, the invalid code must first be send, followed by a new (correct) entry.

The terminal must not assume that the receiving controller does error handling like 1+7+Delete+2+3+4 = 1234.

4.49.5 Entry Control Notification Command

This command is used to advertise user input.

Depending on the Event Type, this command MAY carry manually entered user credentials.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ENTRY_CONTROL							
Command = ENTRY_CONTROL_NOTIFICATION							
Sequence Number							
Reserved						Data Type	
Event Type							
Event Data Length							
Event Data 1 (optional)							
...							
Event Data N (optional)							

Sequence Number (8 bit)

The sequence number MUST be incremented each time a new command of this type is issued.

The value MUST be in the range 0..255. The initial value MAY be any value in the range 0..255.
A receiving device MUST use the Sequence Number to detect and ignore duplicates.

Reserved

This field MUST be set to 0 by a sending device and MUST be ignored by a receiving device.

Data Type (2 bits)

This field is used to advertise the type of data (if any) that is appended to this command.

Table 61, Entry Control Notification :: Event Data Type

Data Type		Description
0x00	NA	No data included
0x01	RAW	1 to 32 bytes of arbitrary binary data
0x02	ASCII	1 to 32 ASCII encoded characters. ASCII codes MUST be in the value range 0x00-0xF7. The string MUST be padded with the value 0xFF to fit 16 byte blocks when sent in a notification.
0x03	MD5	16 byte binary data encoded as a MD5 hash value.

All other values are reserved and MUST NOT be used by a sending device. Reserved values MUST be ignored by a receiving device.

Event Type (4 bit)

This field is used to advertise the actual Event Type.

The field MUST be encoded according to section 4.49.13

Event Data Length (8 bit)

This field MUST advertise the length of the Event Data field in bytes.

The value MUST be in the range 0..32.

If no data bytes are included, this field MUST be set to 0.

Event Data (n bytes)

This field is used to carry data related to the event, e.g. received from an RFID tag.

The length of this field MUST comply with the length advertised by the Event Data Length field.

The format of this field MUST comply with the data format advertised by the Data Type field.

4.49.6 Entry Control Key Supported Get Command

This command is used to query the keys that a device implements for entry of user credentials.

The Entry Control Key Supported Report command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing.
The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ENTRY_CONTROL							
Command = ENTRY_CONTROL_KEY_SUPPORTED_GET							

4.49.7 Entry Control Key Supported Report Command

This command is used to advertise the keys that a device implements for entry of user credentials.

A management interface may determine the available keys for credential entry from this command.
The range of available Command Keys may be determined via the Entry Control Event Supported Report Command.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ENTRY_CONTROL							
Command = ENTRY_CONTROL_KEY_SUPPORTED_REPORT							
Key Supported Bit Mask Length							
Key Supported Bit Mask 1							
…							
Key Supported Bit Mask N							

Key Supported Bit Mask Length

This field is used to advertise the number of Key Supported Bit Mask Bytes to follow.

Only the Key Supported Bit Mask bytes until the last supported ASCII key SHOULD be included.

Key Supported Bit Mask (Variable length)

This field is used to advertise the keys that a device implements for entry of user credentials.

The Key Supported Bit Mask field MUST advertise ASCII codes that represent the supported keys.

- Bit 0 in Bit Mask 1 indicates that the supporting device may issue ASCII code 0
- Bit 1 in Bit Mask 1 indicates that the supporting device may issue ASCII code 1
- …
- Bit 7 in Bit Mask 16 indicates that the supporting device may issue ASCII code 127

4.49.8 Entry Control Event Supported Get Command

This command is used to request the supported Events of a device.

The Entry Control Event Supported Report command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing.
The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ENTRY_CONTROL							
Command = ENTRY_CONTROL_EVENT_SUPPORTED_GET							

4.49.9 Entry Control Event Supported Report Command

This command is used to advertise the supported Event Types.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_ENTRY_CONTROL</td></tr>
<tr><td colspan="8">Command = ENTRY_CONTROL_EVENT_SUPPORTED_REPORT</td></tr>
<tr><td colspan="6">Reserved</td><td colspan="2">Data Type Supported Bit Mask Length</td></tr>
<tr><td colspan="8">Data Type Supported Bit Mask 1</td></tr>
<tr><td colspan="8">…</td></tr>
<tr><td colspan="8">Data Type Supported Bit Mask M</td></tr>
<tr><td colspan="3">Reserved</td><td colspan="5">Event Supported Bit Mask Length</td></tr>
<tr><td colspan="8">Event Type Supported Bit Mask 1</td></tr>
<tr><td colspan="8">…</td></tr>
<tr><td colspan="8">Event Type Supported Bit Mask N</td></tr>
<tr><td colspan="8">Key Cached Size supported Minimum</td></tr>
<tr><td colspan="8">Key Cached Size supported Maximum</td></tr>
<tr><td colspan="8">Key Cached Timeout supported Minimum</td></tr>
<tr><td colspan="8">Key Cached Timeout supported Maximum</td></tr>
</table>

Data Type Supported Bit Mask Length (2 bits)

This field is used to advertise the number of Data Type Supported Bit Mask Bytes to follow.

Reserved

This field MUST be set to 0 by a sending device and MUST be ignored by a receiving device.

Data Type Supported Bit Mask (Variable length)

This field is used to advertise the supported Data Types.

- Bit 0 in Bit Mask 1 indicates if Data Type 0 is supported
- Bit 1 in Bit Mask 1 indicates if Data Type 1 is supported
- …

For the definition of Data Type IDs, refer to Table 61.

Event Type Supported Bit Mask Length (1 byte)

This field is used to advertise the number of Event Type Supported Bit Mask Bytes to follow.

Event Type Supported Bit Mask (variable length)

This field is used to advertise the supported Event Type.

- Bit 0 in Bit Mask 1 indicates if Event Type 0 is supported
- Bit 1 in Bit Mask 1 indicates if Event Type 1 is supported
- …

For the definition of Event Type IDs, refer to Table 62.

Key Cached Size Supported Minimum

The minimum configurable number of key entries before the CACHED_KEYS notification is sent.
Refer to 4.49.10

Key Cached Size Supported Maximum

The maximum configurable number of key entries before the CACHED_KEYS notification is sent.
Refer to 4.49.10

Key Cached Timeout Supported Minimum

The minimum configurable timeout before the CACHED_KEYS notification is sent.
Refer to 4.49.10

Key Cached Timeout Supported Maximum

The maximum configurable timeout before the CACHED_KEYS notification is sent.
Refer to 4.49.10

4.49.10 Entry Control Configuration Set Command

This command is used to configure Event Type specific parameters.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ENTRY_CONTROL							
Command = ENTRY_CONTROL_CONFIGURATION_SET							
Key Cache Size							
Key Cache Timeout							

Key Cache Size (1 byte)

This field specifies the number of characters the key cache MUST store before sending data to the central control application. Data MUST be sent when the number of characters in the cache matches or exceeds the value of this field.

The value MUST be in the range 1..32. The default value of this field SHOULD be 4.

In deployments where characters are entered first and a command button is pressed subsequently, it is RECOMMENDED that this field is set to 32 and that the Key Cache Timeout is set to 2 seconds.

Key Cache Timeout (1 byte)

This field specifies the number of seconds the key cache MUST wait for additional characters before sending data to the central control application. Data MUST be sent if a Key Cache Timeout occurs.

The Key Cache Timeout MUST be measured from the most recent reception of a character.

The value SHOULD be in the range 1..10 seconds. The default value SHOULD be 2 seconds.

4.49.11 Entry Control Configuration Get Command

This command is used to request the operational mode of a device.

The Entry Control Configuration Report command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing.
The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ENTRY_CONTROL							
Command = ENTRY_CONTROL_CONFIGURATION_GET							

Reserved

This field MUST be set to 0 by a sending device and MUST be ignored by a receiving device.

4.49.12 Entry Control Configuration Report Command

This command is used to advertise the current operational mode of a device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ENTRY_CONTROL							
Command = ENTRY_CONTROL_CONFIGURATION_REPORT							
Key Cache Size							
Key Cache Timeout							

Key Cache Size (1 byte)

Refer to 4.49.10.

Key Cache Timeout (1 byte)

Refer to 4.49.10.

4.49.13 Event Types

The Event Types MUST be encoded according to Table 62.

Table 62, Event Type and Event identifiers

Event Type		Recommended Button Label	Event Data (optional)
0x00	CACHING	-	-
0x01	CACHED_KEYS	-	ASCII bytes
0x02	ENTER	Enter	ASCII bytes
0x03	DISARM_ALL	Disarm	ASCII bytes
0x04	ARM_ALL	Arm	ASCII bytes
0x05	ARM_AWAY	Away	ASCII bytes
0x06	ARM_HOME	Home	ASCII bytes
0x07	EXIT_DELAY	Arm_Delay	ASCII bytes
0x08	ARM_1	Arm zone 1	ASCII bytes
0x09	ARM_2	Arm zone 2	ASCII bytes
0x0A	ARM_3	Arm zone 3	ASCII bytes
0x0B	ARM_4	Arm zone 4	ASCII bytes
0x0C	ARM_5	Arm zone 5	ASCII bytes
0x0D	ARM_6	Arm zone 6	ASCII bytes
0x0E	RFID	-	As advertised by the Data Type field
0x0F	BELL		-
0x10	FIRE	Fire	-
0x11	POLICE	Police	-
0x12	ALERT_PANIC		-
0x13	ALERT_MEDICAL		-
0x14	GATE_OPEN	Open, Up. 'O' or similar	ASCII bytes
0x15	GATE_CLOSE	Close, Down, 'C' or similar	ASCII bytes
0x16	LOCK		ASCII bytes
0x17	UNLOCK		ASCII bytes
0x18	TEST	Test	ASCII bytes
0x19	CANCEL	Cancel	ASCII bytes

All other values are reserved and MUST NOT be used by a sending device. Reserved values MUST be ignored by a receiving device.

Event Type CACHING

The CACHING Event Type is used to indicate to the central controller that the user has started entering credentials, and that caching is initiated. This allows the central controller to change the indications on the Entry Control device through the indicator Command Class, or to change the status of the central controller user interface.

If Key Cached Size it set to 1, the CACHING event notification MUST NOT be sent. Instead each individual credential byte MUST be sent in its own CACHED_KEYS event notification.

Event Type CACHED_KEYS

The CACHED_KEYS Event Type is used to send user inputs in a Notification Frame. The CACHED_KEYS is sent when the user input is terminated by one of the following reasons:

- The Key Cache Size is exceeded
- The Key Cache Timeout is exceeded
- A command button is pressed
- User data is received by other means, e.g. from an RFID tag

4.50 Firmware Update Meta Data Command Class, version 1 [DEPRECATED]

THIS VERSION HAS BEEN DEPRECATED

A device MAY implement support for this version of the command class, but it is RECOMMENDED that new implementations implement the Firmware Update Meta Data Command Class, version 3.

If implementing support for this version of the command class, it is RECOMMENDED that support for Firmware Update Meta Data Command Class, version 3 is also implemented.

The Firmware Update Meta Data Command Class may be used to transfer a firmware image to a Z-Wave device.

A device which supports the Firmware Update CC MUST support the Version CC and the Manufacturer Specific CC to enable other devices to select the correct firmware for a specific target. The Version Command Class may be used to verify that the intended firmware version is installed.

Devices implementing the Firmware Update CC MUST support a data rate of 40kbit/s or more.

A checksum SHOULD be appended to the firmware image to ensure the integrity of the image. A receiving device SHOULD verify the integrity of the received firmware image by matching the received firmware image checksum and the firmware image checksum that is indicated by the Firmware Meta Data Report when the firmware update is initiated.

It is RECOMMENDED that firmware update is enabled by out-of-band authentication (e.g. physical activation of a pushbutton). If out-of-band authentication is implemented, the device to receive a firmware update must be armed for firmware update before the Firmware Update Meta Data Request Get Command is issued.

4.50.1 Firmware Meta Data Get Command

The Firmware Meta Data Get Command is used to request information on the current firmware in the device.

The Firmware Meta Data Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD							
Command = FIRMWARE_MD_GET							

4.50.2 Firmware Meta Data Report Command

The Firmware Meta Data Report Command is used to advertise the status of the current firmware in the device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD							
Command = FIRMWARE_MD_REPORT							
Manufacturer ID 1							
Manufacturer ID 2							
Firmware ID 1							
Firmware ID 2							
Checksum 1							
Checksum 2							

Manufacturer ID (16 bits)

The Manufacturer ID is a unique ID identifying the manufacturer of the device.

Manufacturer identifiers can be found in [11].

The first byte MUST be the most significant byte.

Firmware ID (16 bits)

A manufacturer SHOULD assign a unique Firmware ID to each existing product variant. A product variant may be a particular hardware version for a particular world region. When combined with the Manufacturer ID, the Firmware ID is a unique identification of a firmware image that is guaranteed to work with a particular product among all Z-Wave enabled products in the world.
If the product has no Firmware ID, the value 0x0000 MUST be returned.

The first byte is the most significant byte.

A user may request Manufacturer Specific information to get more detailed information on the product.

A controlling device SHOULD match the advertised firmware ID for the existing firmware image to the firmware ID provided for a new image. If the firmware IDs do not match, the controlling device SHOULD abort the firmware update operation.

Checksum (16 bits)

The checksum field MAY be used to report a checksum value of the firmware image currently running in the Z-Wave chip. As an alternative, a value of zero MAY be returned.

The first byte is the most significant byte.

It is RECOMMENDED to use a checksum algorithm that implements a CRC-CCITT polynomial using initialization value equal to 0x1D0F and 0x1021 (normal representation).

4.50.3 Firmware Update Meta Data Request Get Command

The Firmware Update Meta Data Request Get Command is used to request that a firmware update is initiated.
The Firmware Update Meta Data Request Report Command MUST be returned in response to this command.

The firmware update MUST NOT be initiated if the Manufacturer ID and the Firmware ID do not match the actual firmware image values.
The firmware update MUST be aborted if the checksum does not match the calculated checksum after the firmware image has been transferred.

It is RECOMMENDED that firmware update is initiated by out-of-band authentication (e.g. physical activation of a pushbutton).

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing.
The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD							
Command = FIRMWARE_UPDATE_MD_REQUEST_GET							
Manufacturer ID 1							
Manufacturer ID 2							
Firmware ID 1							
Firmware ID 2							
Checksum 1							
Checksum 2							

Manufacturer ID (16 bits)

The Manufacturer ID is a unique ID identifying the manufacturer of the device.

Manufacturer identifiers can be found in [11].The first byte is the most significant byte.

Firmware ID (16 bits)

A manufacturer SHOULD assign a unique Firmware ID to each existing product variant. A product variant may be a particular hardware version for a particular world region. When combined with the

Manufacturer ID, the Firmware ID is a unique identification of a firmware image that is guaranteed to work with a particular product among all Z-Wave enabled products in the world.

A receiving application MUST check that the Manufacturer ID and Firmware ID fields match the Manufacturer ID and Firmware ID values of the current firmware. While it is NOT RECOMMENDED, one MAY design a the product that has no Firmware ID. A sending node MUST specify the Firmware ID value 0x0000 to indicate an absent Firmware ID. A receiving node MUST interpret the value 0x0000 as “No Firmware ID available”.

The first byte MUST be the most significant byte.

Checksum (16 bits)

The checksum field MUST carry the checksum of the firmware image about to be transferred. A receiving application MUST match this value and the checksum calculated from the received firmware image.

It is RECOMMENDED to use a checksum algorithm that implements a CRC-CCITT polynomium using initialization value equal to 0x1D0F and 0x1021 (normal representation). Refer to appendices in [2].

4.50.4 Firmware Update Meta Data Request Report Command

This command is used to advertise if the firmware update will be initiated.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD							
Command = FIRMWARE_UPDATE_MD_REQUEST_REPORT							
Status							

Status (8 bits)

The Status identifier can return the following values:

Status	Description
0x00	ERROR. The Manufacturer ID and/or Firmware ID fields are invalid. The firmware update will not be initiated.
0x01	ERROR. The Manufacturer ID and Firmware ID fields are valid but Firmware Update mode was not armed in the device before the Firmware Update Meta Data Request Get message was issued. The firmware update will not be initiated.
0xFF	OK. Manufacturer ID and Firmware ID fields are valid. The firmware update is about to begin.

4.50.5 Firmware Update Meta Data Get Command

The Firmware Update Meta Data Get Command is used to request one or more Firmware Update Meta Data Report Commands.

The Firmware Update Meta Data Report Command MUST be returned in response to this command.

The transmission of the next Firmware Update Meta Data Get Command MAY be delayed if time is required to store the most recent firmware fragment in temporary non-volatile memory. A node MAY request multiple Firmware Update Meta Data Report Commands to improve throughput.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Two exceptions may apply: A noise burst may interfere the transmission or the data source may stop returning Firmware Update Meta Data Report Commands.

To accommodate for bursts of RF noise, a device receiving data SHOULD repeatedly retransmit the same Firmware Update Meta Data Get Command every 10 seconds in case no Firmware Update Meta Data Report Commands are received.
A device receiving data SHOULD stop retransmitting Firmware Update Meta Data Get commands 2 minutes after the last successful reception of a Firmware Update Meta Data Report Command.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD							
Command = FIRMWARE_UPDATE_MD_GET							
Number of Reports							
0	Report number 1						
Report number 2							

Number of Reports (8 bits)

Number of Firmware Update Meta Data Report Commands to be received in response to this Firmware Update Meta Data Get Command.

Report number 1 .. 2 (15 bits)

The Report number field indicates the Firmware Update Meta Data Report Command to be requested. The report number values MUST be a sequence starting from 1. The first byte (Report number 1) is the most significant byte.

4.50.6 Firmware Update Meta Data Report Command

The Firmware Update Meta Data Report Command is used to transfer a firmware image fragment.

If sending more than a single Firmware Update Meta Data Report Command at a time, a node MUST apply a delay between each transmitted command. The minimum required time delay and number of frames before a delay must be inserted depends on the actual bit rate.

- 40 kbit/s: At least 35 ms if sending more than 1 frame back-to-back
- 100 kbit/s: At least 15 ms if sending more than 2 frames back-to-back

If needed, a data source SHOULD abort an ongoing transfer by responding to a Firmware Update Meta Data Get Command with a Firmware Update Meta Data Report Command with the Last bit enabled and the Data fields intentionally corrupted.
This will invalidate the calculated firmware checksum, which should eventually cause the receiving device to return a Firmware Update Meta Data Status Report Command with the status code <checksum error>.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD							
Command = FIRMWARE_UPDATE_MD_REPORT							
Last	Report number 1						
Report number 2							
Data 1							
…							
Data N							

Last (1 bit)

The Last flag indicates if the requested Firmware Update Meta Data Report Command carries the last firmware image fragment.
The flag MUST be set to ‘1’ if this is the last fragment. Otherwise the flag MUST be ‘0’.

Report number (15 bits)

The Report number field indicates the sequence number of the contained firmware fragment. The first firmware fragment MUST be identified by the Report number value 1. The sequence number of each following firmware fragment MUST be incremented.

Report number 1 is the most significant byte.

The report number may be used to calculate the offset of the data by the formula:

Offset = (Report number – 1) x Number of Data fields (N)

Except for the last frame, each Report MUST carry the same number of Data bytes as the first fragment. The last frame MAY carry a shorter Data field.

Data (N bytes)

The Data field is used to carry one firmware image fragment. Except for the last frame, each Firmware Update Meta Data Report Command MUST carry the same number of Data bytes as the first fragment. The last frame MAY carry a shorter Data field. A sending device MUST use a fragment size which matches the actual number of available Data bytes. The number of available Data bytes depends on the actual bit rate, the use of security encapsulation as well as the presence of Checksum bytes in the Firmware Update Meta Data Report. A receiving device MUST determine the number of Data bytes from the length of the first received frame.

4.50.7 Firmware Update Meta Data Status Report Command

This command is used to advertise the firmware update status.

The command MUST be issued when the firmware update is completed or aborted by the device receiving the firmware.

A node MUST NOT issue Firmware Update Meta Data Get Command after receiving a Firmware Update Meta Data Status Report.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD							
Command = FIRMWARE_UPDATE_MD_STATUS_REPORT							
Status							

Status (8 bits)

The Status identifier can return the following values:

Status	Description
0x00	The device was unable to receive the requested firmware data without checksum error. Number of retries and request sequence of missing frames are implementation specific. The image is not stored.
0x01	The device was unable to receive the requested firmware data. Number of retries and request sequence of missing frames are implementation specific.
…	*reserved*
0xFF	New firmware was successfully stored in temporary non-volatile memory. The device will now start storing the new image in primary non-volatile memory dedicated to executable code. Then the device will restart itself.

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

4.50.8 Detailed description of frame flow

Figure 22 shows a device (Node A) requesting the Manufacturer ID and Firmware ID of another device (Node B) by issuing the Firmware Meta Data Get Command.

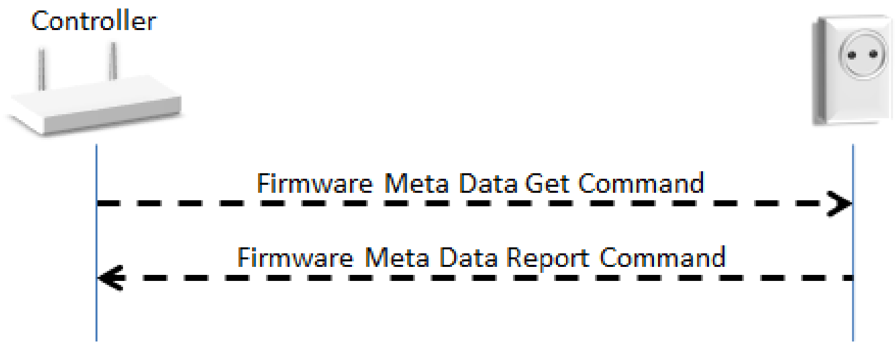


Figure 22, Requesting Manufacturer ID and Firmware ID of a device

Figure 23 outlines the actual firmware update message flow. Prior to the firmware update, Node B receives an out-of-band authentication (e.g. physical activation of a pushbutton). Node A sends a Firmware Meta Data Request Get Command to make Node B begin downloading a new firmware image. Node B returns a Firmware Meta Data Request Report Command report to confirm the update request. Node B begins pulling firmware image fragments from Node A. Finally a Firmware Update Meta Data Status Report is returned to Node A to indicate the success (or failure) of the update process.

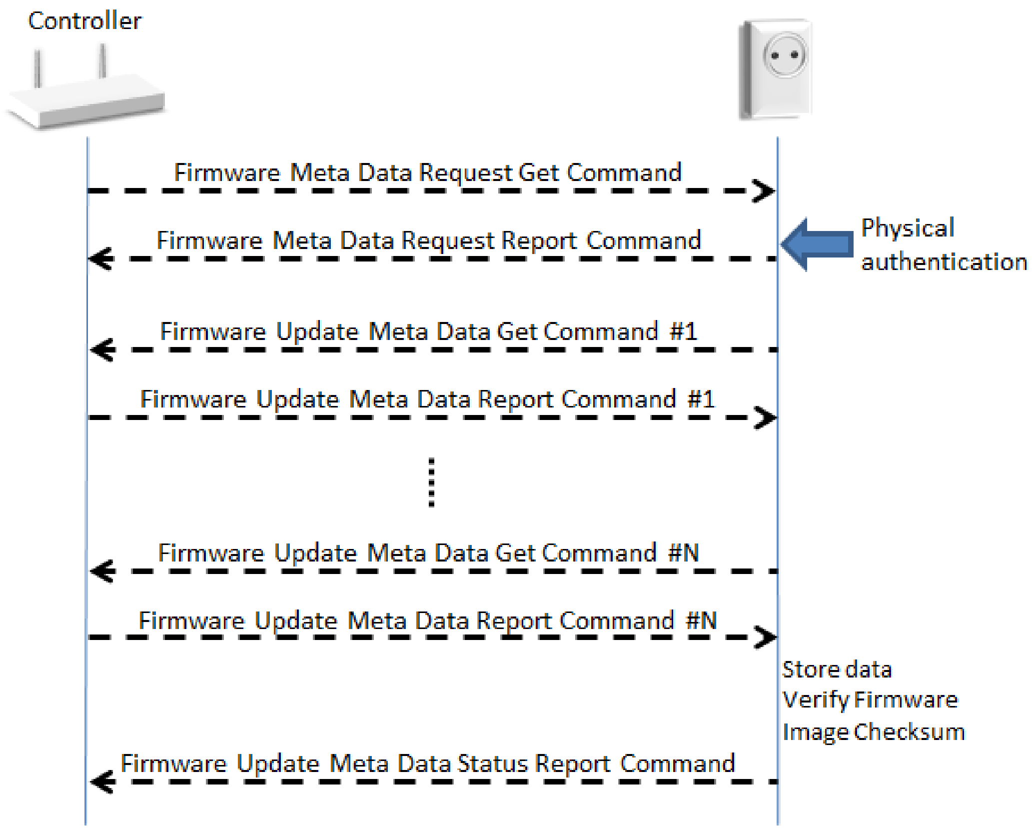


Figure 23, Transferring a firmware image to a device

4.51 Firmware Update Meta Data Command Class, version 2

Firmware Update Meta Data Command Class, version 2 updates the Firmware Update Meta Data Report Command.

While support for version 1 has been deprecated, a controlling device implementing Firmware Update Meta Data Command Class, version 2 MUST also be able to control Firmware Update Meta Data Command Class, version 1 based devices.

4.51.1 Compatibility considerations

4.51.1.1 Interoperability with v1 devices

Version 2 of the Firmware Update Meta Data Report command introduces a 16-bit Checksum field to follow the variable-length Data field. No fields follow the variable-length Data field in Version 1 of the Firmware Update Meta Data Report command.

At the same time, the specified method for determining the length of the variable-length Data field is that the length must be calculated from the length of the received frame.

The unintended consequence is that a version 1 implementation receiving a Firmware Update Meta Data Report v2 command will consider the 16-bit Checksum field to be the last two bytes of the Data field. Similarly, a version 2 implementation receiving a Firmware Update Meta Data Report v1 command will consider the last two bytes of the Data field to be the 16-bit Checksum field.
In either case, the transfer will fail.

Therefore, a controlling device implementing the Firmware Update Meta Data Command Class, version 2 MUST do the following when initiating the transfer of a firmware image:

1. Request the version of the Firmware Update Meta Data Command Class from the target device

2. Use the version of the Firmware Update Meta Data Report implemented by the target device

Determine the number of Data bytes that can fit into the Firmware Update Meta Data Report command so that the complete command can still fit into the payload field of the transport frame. The number of available payload bytes in the frame depends on the actual bit rate, the use of security encapsulation as well as the presence of Checksum bytes in the Firmware Update Meta Data Report.

The controller SHOULD accept any binary image format and transfer that without any modifications. The image may be encrypted or carry checksums.

4.51.1.2 Checksum calculation

Version 1 of the Firmware Update Meta Data Command Class introduced a 16-bit Checksum field used to verify firmware image integrity. It has been a recommendation to use the CRC-CCITT polynomial for calculating checksums. Nodes having implemented another method for calculating the checksum will find a non-matching checksum.

From Version 5 onwards, the checksum calculation method is mandatory.

For more details about the checksum calculation, refer to [2] Appendix B

4.51.2 Firmware Meta Data Report Command

The Firmware Meta Data Report Command is used to advertise the status of the current firmware in the device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_MD							
Command = FIRMWARE_MD_REPORT							
Manufacturer ID 1							
Manufacturer ID 2							
Firmware ID 1							
Firmware ID 2							
Checksum 1							
Checksum 2							

Manufacturer ID (16 bits)

The Manufacturer ID is a unique ID identifying the manufacturer of the device.

Manufacturer identifiers can be found in [11].

The first byte MUST be the most significant byte.

Firmware ID (16 bits)

A manufacturer SHOULD assign a unique Firmware ID to each existing product variant. A product variant may be a particular hardware version for a particular world region. When combined with the Manufacturer ID, the Firmware ID is a unique identification of a firmware image that is guaranteed to work with a particular product among all Z-Wave enabled products in the world.
If the product has no Firmware ID, the value 0x0000 MUST be returned.

The first byte is the most significant byte.

A user may request Manufacturer Specific information to get more detailed information on the product.

A controlling device SHOULD match the advertised firmware ID for the existing firmware image to the firmware ID provided for a new image. If the firmware IDs do not match, the controlling device SHOULD abort the firmware update operation.

Checksum (16 bits)

The checksum field MAY be used to report a checksum value of the firmware image currently running in the Z-Wave chip. As an alternative, a value of zero MAY be returned.

The first byte is the most significant byte.

It is RECOMMENDED to use a checksum algorithm that implements a CRC-CCITT polynomial using initialization value equal to 0x1D0F and 0x1021 (normal representation).

4.51.3 Firmware Update Meta Data Report Command

The Firmware Update Meta Data Report command is used to transfer a firmware image fragment.

If sending more than a single Firmware Update Meta Data Report Command at a time, a node MUST apply a delay between each transmitted command. The minimum required time delay and number of frames before a delay must be inserted depends on the actual bit rate.

- 40 kbit/s: At least 35 ms if sending more than 1 frame back-to-back
- 100 kbit/s: At least 15 ms if sending more than 2 frames back-to-back

If needed, a data source SHOULD abort an ongoing transfer by responding to a Firmware Update Meta Data Get Command with a Firmware Update Meta Data Report Command with the Last bit enabled and the Data fields intentionally corrupted while the Checksum field of the Firmware Update Meta Data Report Command is valid, i.e. calculated over the corrupted Data fields.
This will invalidate the calculated firmware checksum, which should eventually cause the receiving device to return a Firmware Update Meta Data Status Report Command with the status code <checksum error>.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD</td></tr>
<tr><td colspan="8">Command = FIRMWARE_UPDATE_MD_REPORT</td></tr>
<tr><td>Last</td><td colspan="7">Report number 1</td></tr>
<tr><td colspan="8">Report number 2</td></tr>
<tr><td colspan="8">Data 1</td></tr>
<tr><td colspan="8">…</td></tr>
<tr><td colspan="8">Data N</td></tr>
<tr><td colspan="8">Checksum 1</td></tr>
<tr><td colspan="8">Checksum 2</td></tr>
</table>

Last (1 bit)

The Last flag indicates if the requested Firmware Update Meta Data Report Command carries the last firmware image fragment.
The flag MUST be set to '1' if this is the last fragment. Otherwise the flag MUST be '0'.

Report number (15 bits)

The Report number field indicates the sequence number of the contained firmware fragment. The first firmware fragment MUST be identified by the Report number value 1. The sequence number of each following firmware fragment MUST be incremented.

The report number may be used to calculate the offset of the data by the formula:

Offset = (Report number – 1) x Number of Data fields (N)

Except for the last frame, each Report MUST carry the same number of Data bytes as the first fragment. The last frame MAY carry a shorter Data field.

Data (N bytes)

The Data field is used to carry one firmware image fragment. Except for the last frame, each Firmware Update Meta Data Report Command MUST carry the same number of Data bytes as the first fragment. The last frame MAY carry a shorter Data field. A sending device MUST use a fragment size which matches the actual number of available Data bytes. The number of available Data bytes depends on the actual bit rate, the use of security encapsulation as well as the presence of Checksum bytes in the Firmware Update Meta Data Report. A receiving device MUST determine the number of Data bytes from the length of the first received frame.

Checksum (16 bits)

The checksum field MUST be used to ensure the consistency of the entire command; including the command class and command identifiers. It is RECOMMENDED to use a checksum algorithm that implements the CRC-CCITT polynomium using initialization value equal to 0x1D0F and 0x1021 (normal representation).

The Checksum field is known to cause compatibility issues with version 1 devices. Refer to 4.51.1.

4.51.4 Firmware Update Meta Data Status Report Command

This command is used to advertise the firmware update status.

The command MUST be issued when the firmware update is completed or aborted by the device receiving the firmware.

A node MUST NOT issue Firmware Update Meta Data Get Command after receiving a Firmware Update Meta Data Status Report.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD							
Command = FIRMWARE_UPDATE_MD_STATUS_REPORT							
Status							

Status (8 bits)

The Status identifier can return the following values:

Status	Description
0x00	The device was unable to receive the requested firmware data without checksum error. Number of retries and request sequence of missing frames are implementation specific. The image is not stored. **(Version 1)**
0x01	The device was unable to receive the requested firmware data. Number of retries and request sequence of missing frames are implementation specific. The image is not stored. **(Version 1)**
…	*reserved*
0xFF	New firmware was successfully stored in temporary non-volatile memory. The device will now start storing the new image in primary non-volatile memory dedicated to executable code. Then the device will restart itself. **(Version 1)**

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

4.52 Firmware Update Meta Data Command Class, version 3

The Firmware Update Meta Data Command Class, version 3 allows a Z-Wave enabled device to receive one out of several firmware images.

As an example, one may construct a product comprising a Z-Wave chip and a secondary host processor that maintains a security certificate. With the capability to handle multiple firmware images enables the Z-Wave chip, the host processor and the security certificate may all be updated via individual firmware IDs.

Commands not explicitly mentioned remain the same as in Firmware Update Meta Data Command Class, version 2.

A device implementing Firmware Update Meta Data Command Class, version 3 MUST also implement Firmware Update Meta Data Command Class, version 2.
While support for version 1 has been deprecated, a controlling device implementing Firmware Update Meta Data Command Class, version 2 MUST also be able to control Firmware Update Meta Data Command Class, version 1 based devices.

Unintended modification of a firmware image SHOULD be prevented. It is RECOMMENDED that firmware update is enabled by out-of-band authentication (e.g. physical activation of a pushbutton).

A checksum MUST be appended to the Firmware 0 image to ensure the integrity of the image. A receiving device SHOULD verify the integrity of the received Firmware 0 image by matching the received Firmware 0 image checksum and the Firmware 0 image checksum that is indicated by the Firmware Meta Data Report when the firmware update is initiated.

Any image transferred via the Firmware Update Meta Data Command Class SHOULD include a fingerprint value for validation of the integrity of the entire image after the transfer. The image and the fingerprint value MUST be packed in one entity which can be stored in one file and transferred as one entity.

A manufacturer MUST assign a unique Firmware ID to each existing product variant. A product variant may be a particular hardware version for a particular world region. The combination of Manufacturer ID and Firmware ID provides unique identification of a firmware image that is guaranteed to work with a particular component of a particular product.

4.52.1 Compatibility considerations

4.52.1.1 New fields and values

The Firmware Update Command Class, Version 3, introduces WaitTime parameter and a new Status code 0xFE for the Firmware Update Meta Data Status Report Command.

The status code 0xFE is intended for confirming the successful transfer of images which do not necessitate a restart, e.g. security certificates. If this code was returned to a controller implementing a previous version, the controller would report that an error had occurred. Therefore, a receiving node MUST NOT return this code after having updated the firmware image for the Z-Wave chip image, i.e. the Firmware ID 0 target.

4.52.1.2 Interoperability with v1 devices

Version 2 of the Firmware Update Meta Data Report command introduces a 16-bit Checksum field to follow the variable-length Data field. No fields follow the variable-length Data field in Version 1 of the Firmware Update Meta Data Report command. Version 3 of the Firmware Update Meta Data Report command uses the v2 format.

At the same time, the specified method for determining the length of the variable-length Data field is that the length must be calculated from the length of the received frame.

The unintended consequence is that a version 1 implementation receiving a Firmware Update Meta Data Report v2 command will consider the 16-bit Checksum field to be the last two bytes of the Data field. Similarly, a version 2 implementation receiving a Firmware Update Meta Data Report v1 command will consider the last two bytes of the Data field to be the 16-bit Checksum field.
In either case, the transfer will fail.

Therefore, a controlling device implementing the Firmware Update Meta Data Command Class, version 3 MUST do the following when initiating the transfer of a firmware image:

1. Request the version of the Firmware Update Meta Data Command Class from the target device

2. Use the version of the Firmware Update Meta Data Report implemented by the target device

Determine the number of Data bytes that can fit into the Firmware Update Meta Data Report command so that the complete command can still fit into the payload field of the transport frame. The number of available payload bytes in the frame depends on the actual bit rate, the use of security encapsulation as well as the presence of Checksum bytes in the Firmware Update Meta Data Report

4.52.1.3 Checksum calculation

Version 1 of the Firmware Update Meta Data Command Class introduced a 16-bit Checksum field used to verify firmware image integrity. It has been a recommendation to use the CRC-CCITT polynomial for calculating checksums. Nodes having implemented another method for calculating the checksum will find a non-matching checksum.

From Version 5 onwards, the checksum calculation method is mandatory.

For more details about the checksum calculation, refer to [2] Appendix B

4.52.2 Firmware Meta Data Report Command

The Firmware Meta Data Report Command is used to advertise the status of the current firmware in the device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD							
Command = FIRMWARE_MD_REPORT							
Manufacturer ID 1							
Manufacturer ID 2							
Firmware 0 ID 1							
Firmware 0 ID 2							
Firmware 0 Checksum 1							
Firmware 0 Checksum 2							
Firmware Upgradable							
Number of Firmware Targets							
Max Fragment Size 1							
Max Fragment Size 2							
Firmware 1 ID 1							
Firmware 1 ID 2							
…							
Firmware N ID 1							
Firmware N ID 2							

Manufacturer ID (16 bits)

The Manufacturer ID is a unique ID identifying the manufacturer of the device. Manufacturer identifiers can be found in [11].

The first byte is the most significant byte.

Firmware 0 ID (16 bits)

The Firmware 0 ID field is dedicated to target 0, i.e. the Z-Wave chip.

A manufacturer MUST assign a unique Firmware ID to each existing product variant. A product variant may be a particular hardware version for a particular world region. The combination of Manufacturer ID and Firmware ID provides unique identification of a firmware image that is guaranteed to work with a particular component of a particular product.

The first byte is the most significant byte.

A user may request Manufacturer Specific information to get more detailed information on the product.

A controlling device SHOULD match the advertised firmware ID for the existing firmware image to the firmware ID provided for a new image. If the firmware IDs do not match, the controlling device SHOULD abort the firmware update operation.

Firmware 0 Checksum (16 bits)

The checksum field MUST be used to ensure consistency of the Firmware 0 image currently in the device. The first byte is the most significant byte. It is RECOMMENDED to use a checksum algorithm that implements a CRC-CCITT polynomial using initialization value equal to 0x1D0F and 0x1021 (normal representation).

Firmware Upgradable (8 bits)

The Firmware Upgradable field defines whether the Z-Wave chip is firmware Upgradable. If Firmware Upgradable value is 0x00 indicates that firmware image is not Upgradable and 0xFF indicates that the firmware image is Upgradable.

Number of Firmware Targets (8 bits)

The Number of Firmware Targets field MUST report the number of firmware IDs following this field. The Firmware 0 ID field is not included. The field MUST be zero if the device only implements a Firmware 0 target, i.e. the Z-Wave chip.

Max Fragment Size (16 bits)

The Max Fragment Size field MUST report the maximum number of Data bytes that a device is able to receive at a time. A sending device MAY send shorter fragments. The fragment size actually used is indicated in the Firmware Update Meta Data Request Get Command and confirmed in the Firmware Update Meta Data Request Report Command.

The Max Fragment Size may be longer than the supported frame length of Z-Wave if the image is to be transferred over e.g. IP. If the image is to be transferred over Z-Wave, the sending device MUST use a fragment size which matches the actual number of available Data bytes. The number of available Data bytes depends on the actual bit rate, the use of security encapsulation as well as the presence of Checksum bytes in the Firmware Update Meta Data Report.

Firmware n ID (16 bits)

The Firmware 1 ID field indicates the Firmware ID that must be used for target 1. The Firmware 2 ID field represents target 2. And so on.

The first byte is the most significant byte.

A controlling device SHOULD match the advertised firmware ID for the existing firmware image to the firmware ID provided for a new image. If the firmware IDs do not match, the controlling device SHOULD abort the firmware update operation.

4.52.3 Firmware Update Meta Data Request Get Command

The Firmware Update Meta Data Request Get Command is used to request that a firmware update is initiated by the node receiving this command.

The Firmware Update Meta Data Request Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

The firmware update MUST NOT be initiated if the Manufacturer ID and the Firmware ID do not match the actual firmware image values for the specified Firmware Target.
The firmware update MUST be aborted if the checksum does not match the calculated checksum after the firmware image has been transferred.

The node receiving this command MUST confirm the reception of this command by returning a Firmware Update Meta Data Request Report Command.

It is RECOMMENDED that firmware update is enabled by out-of-band authentication (e.g. physical activation of a pushbutton) prior to the transmission of this command.

Any image transferred via the Firmware Update Meta Data Command Class SHOULD include a fingerprint value to enable the validation of the integrity of the entire image after the transfer. The image and the fingerprint value MUST be packed in one entity which can be stored in one file and transferred as one entity.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD							
Command = FIRMWARE_UPDATE_MD_REQUEST_GET							
Manufacturer ID 1							
Manufacturer ID 2							
Firmware ID 1							
Firmware ID 2							
Checksum 1							
Checksum 2							
Firmware Target							
Fragment Size 1							
Fragment Size 2							

Manufacturer ID (16 bits)

The Manufacturer ID MUST match the specified target of the actual device. The firmware update MUST NOT be initiated if the Manufacturer ID does not match the specified target of the actual device.

The first byte is the most significant byte.

Firmware ID (16 bits)

The Firmware ID MUST match the specified target of the actual device.

The firmware update MUST NOT be initiated if the Firmware ID does not match the specified target of the actual device.

The first byte is the most significant byte.

Checksum (16 bits)

The checksum field MUST carry the checksum of the firmware image about to be transferred. A receiving application MUST match this value and the checksum calculated from the received firmware image before permanently storing the received image.

It is RECOMMENDED to use a checksum algorithm that implements a CRC-CCITT polynomial using initialization value equal to 0x1D0F and 0x1021 (normal representation).

Firmware Target (8 bits)

The Firmware Target field MUST be used to identify the firmware image to be updated. The firmware images are identified as follows:

Firmware Target	Description
0x00	Firmware image targeted for the Z-Wave chip.
0x01	Firmware image intended for target 1 defined by the manufacturer.
…	
0xFF	Firmware image intended for target 255 defined by the manufacturer.

Fragment Size (16 bits)

The Fragment Size field MUST report the fragment size that is to be used for firmware fragments. A receiving device MUST use this fragment size for the firmware update. The fragment size is not exchanged during the actual firmware update.
The Fragment Size MUST NOT exceed the Max Fragment Size value of the Firmware Meta Data Report.

A version 1 Firmware Update Meta Data Request Get Command does not carry a Fragment Size field. A receiving device MUST determine the number of Data bytes from the length of the first received frame, the use of security encapsulation as well as the presence of Checksum bytes in the Firmware Update Meta Data Report.

4.52.4 Firmware Update Meta Data Request Report Command

This command is used to advertise if the firmware update will be initiated.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD</td></tr>
<tr><td colspan="8">Command = FIRMWARE_UPDATE_MD_REQUEST_REPORT</td></tr>
<tr><td colspan="8">Status</td></tr>
</table>

Status (8 bits)

The Status identifier can return the following values:

Status	Description
0x00	ERROR. Invalid combination of Manufacturer ID and Firmware ID. The device will not initiate the firmware update.
0x01	ERROR. Device expected an authentication event to enable firmware update. The device will not initiate the firmware update.
0x02	ERROR. The requested Fragment Size exceeds the Max Fragment Size. The device will not initiate the firmware update.
0x03	ERROR. This firmware target is not upgradable. The device will not initiate the firmware update.
0xFF	OK. The device will initiate the firmware update of the target specified in the Firmware Update Meta Data Request Get Command.

4.52.5 Firmware Update Meta Data Get Command

The Firmware Update Meta Data Get Command is used to request one or more Firmware Update Meta Data Report Commands.

One or more Firmware Update Meta Data Report Commands MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

A node MUST verify the Manufacturer ID and Firmware ID fields received in a Firmware Update Meta Data Request Get Command before sending any Firmware Update Meta Data Get Commands.
A controlling node receiving a Firmware Update Meta Data Get Command MUST return fragments of the firmware image that was previously identified by the Manufacturer ID and Firmware ID fields received in a Firmware Update Meta Data Request Get Command.

The transmission of the next Firmware Update Meta Data Get Command MAY be delayed if time is required to store the most recent firmware fragment in temporary non-volatile memory. A node MAY request multiple Firmware Update Meta Data Report Commands to improve throughput.

RAM may be a limited resource. A node MAY adjust the size of incoming Firmware Update Meta Data Report Commands to the available first level RAM receive buffer by reporting the desired Max Fragment Size when returning a Firmware Meta Data Report Command in response to a Firmware Update Meta Data Get Command.

Two exceptions may apply: A noise burst may interfere the transmission or the data source may stop returning Firmware Update Meta Data Report Commands.

To accommodate for bursts of RF noise, a device receiving data SHOULD repeatedly retransmit the same Firmware Update Meta Data Get Command every 10 seconds in case no Firmware Update Meta Data Report Commands are received.
A device receiving data SHOULD stop retransmitting Firmware Update Meta Data Get commands 2 minutes after the last successful reception of a Firmware Update Meta Data Report Command.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD</td></tr>
<tr><td colspan="8">Command = FIRMWARE_UPDATE_MD_GET</td></tr>
<tr><td colspan="8">Number of Reports</td></tr>
<tr><td>0</td><td colspan="7">Report Number 1</td></tr>
<tr><td colspan="8">Report Number 2</td></tr>
</table>

Number of Reports (8 bits)

Number of Firmware Update Meta Data Report Commands to be received in response to a single Firmware Update Meta Data Get Command.
The requesting node MUST keep track of missing Firmware Update Meta Data Report Commands.

Report Number (15 bits)

The Report number field indicates the sequence number of the requested firmware fragment. The first firmware fragment MUST be identified by the Report Number value 1.

Report number 1 is the most significant byte.

If the Number of Reports field is larger than 1, the Report Number MUST identify the first firmware fragment in the requested sequence of firmware fragments.

4.52.6 Firmware Update Meta Data Report Command

The Firmware Update Meta Data Report command is used to transfer a firmware image fragment.

If sending more than a single Firmware Update Meta Data Report Command at a time, a node MUST apply a delay between each transmitted command. The minimum required time delay and number of frames before a delay must be inserted depends on the actual bit rate.

- 40 kbit/s: At least 35 ms if sending more than 1 frame back-to-back
- 100 kbit/s: At least 15 ms if sending more than 2 frames back-to-back

After receiving all Firmware Update Meta Data Report Commands required for a complete firmware image, a receiving node MUST store the firmware image in the Firmware Target specified in a previously received Firmware Update Meta Data Request Get Command.

If needed, a data source SHOULD abort an ongoing transfer by responding to a Firmware Update Meta Data Get Command with a Firmware Update Meta Data Report Command with the Last bit enabled and the Data fields intentionally corrupted while the Checksum field of the Firmware Update Meta Data Report Command is valid, i.e. calculated over the corrupted Data fields.
This will invalidate the calculated firmware checksum, which should eventually cause the receiving device to return a Firmware Update Meta Data Status Report Command with the status code <checksum error>.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD							
Command = FIRMWARE_UPDATE_MD_REPORT							
Last	Report number 1						
Report number 2							
Data 1							
…							
Data N							
Checksum 1							
Checksum 2							

Last (1 bit)

The Last field indicates if the Firmware Update Meta Data Report Command is the last one. The value '1' indicates that this is the last report. Otherwise the field has the value '1'.

On the reception of the last image fragment, the receiving node MUST verify that the transferred image does match the indicated Manufacturer ID, Firmware ID and Firmware Target values.

Report number (15 bits)

The Report number field indicates the sequence number of the contained firmware fragment. The first firmware fragment MUST be identified by the Report number value 1. The sequence number of each following firmware fragment MUST be incremented.

Report number 1 is the most significant byte.

The report number may be used to calculate the offset of the data by the formula:

Offset = (Report Number – 1) x (Fragment Size)

Except for the last frame, each Report MUST carry the same number of Data bytes as the first fragment. The last frame MAY carry a shorter Data field.

Data (N bytes)

The Data field is used to carry one firmware image fragment. Except for the last frame, each Firmware Update Meta Data Report Command MUST carry the same number of Data bytes as the first fragment. The last frame MAY carry a shorter Data field. A sending device MUST use a fragment size which matches the actual number of available Data bytes. The number of available Data bytes depends on the actual bit rate, the use of security encapsulation as well as the presence of Checksum bytes in the Firmware Update Meta Data Report. A receiving device MUST determine the number of Data bytes from the length of the first received frame.

Checksum (16 bits)

The checksum field MUST be used to ensure the consistency of the entire command; including the command class and command identifiers. It is RECOMMENDED to use a checksum algorithm that implements the CRC-CCITT polynomium using initialization value equal to 0x1D0F and 0x1021 (normal representation). The checksum algorithm used for the Firmware Update Meta Data Command Class, version 3 MUST be the same as the algorithm used for the Firmware Update Meta Data Command Class, version 2 in the actual product.

The Checksum field is known to cause compatibility issues with version 1 devices. Refer to 4.51.1.

4.52.7 Firmware Update Meta Data Status Report Command

This command is used to advertise the firmware update status.

The command MUST be issued when the firmware update is completed or aborted by the device receiving the firmware.

A device MUST NOT issue the Firmware Update Meta Data Get Command after receiving a Firmware Update Meta Data Status Report.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD</td></tr>
<tr><td colspan="8">Command = FIRMWARE_UPDATE_MD_STATUS_REPORT</td></tr>
<tr><td colspan="8">Status</td></tr>
<tr><td colspan="8">WaitTime MSB</td></tr>
<tr><td colspan="8">WaitTime LSB</td></tr>
</table>

Status (8 bits)

The Status identifier can return the following values:

Status	Description
0x00	The device was unable to receive the requested firmware data without checksum error. Number of retries and request sequence of missing frames are implementation specific. The image is not stored. **(Version 1)**
0x01	The device was unable to receive the requested firmware data. Number of retries and request sequence of missing frames are implementation specific. The image is not stored. **(Version 1)**
…	*Reserved*
0xFE	New image was successfully stored in temporary non-volatile memory. The device does not restart itself. This Status code MUST NOT be used when updating the Z-Wave chip image **(Version 3)**
0xFF	New image was successfully stored in temporary non-volatile memory. The device will now start storing the new image in primary non-volatile memory dedicated to executable code. Then the device will restart itself. **(Version 1)**

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

The status code 0xFE MAY be used for confirming the successful transfer of images which do not necessitate a restart, e.g. security certificates.

The status code 0xFE MUST NOT be advertised after the transfer of an image for the Firmware ID 0 target (the “Z-Wave chip” image).

Controlling nodes implementing earlier versions of the Firmware Update Meta Data CC do not support status codes defined for newer versions. Therefore, a device returning the Firmware Update Meta Data Status Report Command MUST comply with the version implemented on the controlling device (device sending the image). The device returning the Firmware Update Meta Data Status Report can identify the version of the controlling device based on the Firmware Update Meta Data Request Get. This may be done as follows:

- If the Firmware Update Meta Data Request Get does not include Firmware Target and Fragment Size (8 bytes), the controller is version 1 or 2

- If the Firmware Update Meta Data Request Get includes Firmware Target and Fragment Size but not Activation (11 bytes), the controller is version 3

WaitTime (16 bits)

The WaitTime field MUST report the time that is needed before the receiving node again becomes available for communication after the transfer of an image. The unit is the second.

The value 0 (zero) MUST indicate that the node is ready.

The value 0xFFFF is reserved and MUST NOT be returned.

A controlling application which receives this status command SHOULD wait for the number of seconds specified in the WaitTime field before trying to resume communication. When resuming communication, the controlling application SHOULD issue a NOP command and wait for acknowledgement. The controlling application MAY attempt resuming communication repeatedly. In that case, the NOP interval SHOULD be five seconds. The NOP interval MUST be at least one second.

4.53 Firmware Update Meta Data Command Class, version 4

The Firmware Update Meta Data Command Class may be used to transfer a firmware image to a Z-Wave device.

Commands not mentioned here remain the same as in version 3.

A device implementing Firmware Update Meta Data Command Class, version 4 MUST also implement Firmware Update Meta Data Command Class, versions 2 and 3.
While support for version 1 has been deprecated, a controlling device implementing Firmware Update Meta Data Command Class, version 2 MUST also be able to control Firmware Update Meta Data Command Class, version 1 based devices.

4.53.1 Compatibility considerations

4.53.1.1 New commands, fields and values

The Firmware Update Meta Data Command Class version 4 supports delaying and scheduling a firmware update after downloading the image.

Version 4 updates the Firmware Update Meta Data Request Get Command and introduces the following commands:

- New Firmware Update Activation Set Command
- New Firmware Update Activation Status Report

After downloading an image, a device may use the Firmware Update Meta Data Status Report Command to indicate to the controlling device that the device is waiting for an activation command.

4.53.1.2 Interoperability with v1 devices

Version 2 of the Firmware Update Meta Data Report command introduces a 16-bit Checksum field to follow the variable-length Data field. No fields follow the variable-length Data field in Version 1 of the Firmware Update Meta Data Report command. Version 4 of the Firmware Update Meta Data Report command uses the v2 format.

At the same time, the specified method for determining the length of the variable-length Data field is that the length must be calculated from the length of the received frame.

The unintended consequence is that a version 1 implementation receiving a Firmware Update Meta Data Report v2 command will consider the 16-bit Checksum field to be the last two bytes of the Data field. Similarly, a version 2 implementation receiving a Firmware Update Meta Data Report v1 command will consider the last two bytes of the Data field to be the 16-bit Checksum field.
In either case, the transfer will fail.

Therefore, a controlling device implementing the Firmware Update Meta Data Command Class, version 2 MUST do the following when initiating the transfer of a firmware image:

1. Request the version of the Firmware Update Meta Data Command Class from the target device
2. Use the version of the Firmware Update Meta Data Report implemented by the target device
3. Determine the number of Data bytes that can fit into the Firmware Update Meta Data Report command so that the complete command can still fit into the payload field of the transport frame. The number of available payload bytes in the frame depends on the actual bit rate, the use of security encapsulation as well as the presence of Checksum bytes in the Firmware Update Meta Data Report

4.53.1.3 Checksum calculation

Version 1 of the Firmware Update Meta Data Command Class introduced a 16-bit Checksum field used to verify firmware image integrity. It has been a recommendation to use the CRC-CCITT polynomial for calculating checksums. Nodes having implemented another method for calculating the checksum will find a non-matching checksum.

From Version 5 onwards, the checksum calculation method is mandatory.

For more details about the checksum calculation, refer to [2] Appendix B

4.

4.53.2 Firmware Update Meta Data Request Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD							
Command = FIRMWARE_UPDATE_MD_REQUEST_GET							
Manufacturer ID 1							
Manufacturer ID 2							
Firmware ID 1							
Firmware ID 2							
Checksum 1							
Checksum 2							
Firmware Target							
Fragment Size 1							
Fragment Size 2							
Reserved							Activation

All Parameters not described below are the same as in v3.

Activation (1 bit)

The Activation field is used to advertise if the receiving node may delay the actual firmware update.

The field MUST be interpreted as:

’1’: The receiving device MAY delay the actual firmware update.
If the receiving node delays the firmware update, the delay MUST be advertised via the Status code 0xFD in the Firmware Update Meta data Status Report Command.

’0’: The receiving device MUST NOT delay the firmware update.

4.53.3 Firmware Update Meta Data Status Report Command

This command is used to advertise the firmware update status.

The command MUST be issued when the firmware update is completed or aborted by the device receiving the firmware.

A device MUST NOT issue the Firmware Update Meta Data Get Command after receiving a Firmware Update Meta Data Status Report.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD							
Command = FIRMWARE_UPDATE_MD_STATUS_REPORT							
Status							
WaitTime MSB							
WaitTime LSB							

Status (8 bits)

The Status identifier can return the following values:

Status	Description
0x00	The device was unable to receive the requested firmware data without checksum error. Number of retries and request sequence of missing frames are implementation specific. The image is not stored. **(Version 1)**
0x01	The device was unable to receive the requested firmware data. Number of retries and request sequence of missing frames are implementation specific. The image is not stored. **(Version 1)**
0x02	The transferred image does not match the Manufacturer ID. The image is not stored. **(Version 4)**
0x03	The transferred image does not match the Firmware ID. The image is not stored. **(Version 4)**
0x04	The transferred image does not match the Firmware Target. The image is not stored. **(Version 4)**
0x05	Invalid file header information. The image is not stored. **(Version 4)**
0x06	Invalid file header format. The image is not stored. **(Version 4)**
0x07	Insufficient memory. The image is not stored. **(Version 4)**
…	*Reserved*
0xFD	Firmware image downloaded successfully, waiting for activation command. **(Version 4)**
0xFE	New image was successfully stored in temporary non-volatile memory. The device does not restart itself. This Status code MUST NOT be used when updating the Z-Wave chip image **(Version 3)**
0xFF	New image was successfully stored in temporary non-volatile memory. The device will now start storing the new image in primary non-volatile memory dedicated to executable code. Then the device will restart itself. **(Version 1)**

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

The status code 0xFD MAY be used by a version 4 device for confirming that an image has been successfully transferred but that the actual firmware update will not be performed until a Firmware Update Activation Set Command is received. The Firmware Update Activation Set Command MAY be delayed for any period of time. The delay MAY be controlled via the Schedule Command Class.

The status code 0xFE MAY be used for confirming the successful transfer of an image which does not necessitate a restart, e.g. security a certificate.
The status code 0xFE MUST NOT be advertised after the transfer of an image for the Firmware ID 0 target (the “Z-Wave chip” image).

Controlling nodes implementing earlier versions of the Firmware Update Meta Data CC do not support status codes defined for newer versions. Therefore, a device returning the Firmware Update Meta Data Status Report Command MUST comply with the version implemented on the controlling device (device sending the image). The device returning the Firmware Update Meta Data Status Report can identify the version of the controlling device based on the Firmware Update Meta Data Request Get. This may be done as follows:

- If the Firmware Update Meta Data Request Get does not include Firmware Target and Fragment Size (8 bytes), the controller is version 1 or 2
- If the Firmware Update Meta Data Request Get includes Firmware Target and Fragment Size but not Activation (11 bytes), the controller is version 3
- If the Firmware Update Meta Data Request Get includes Activation (12 bytes), the controller is version 4

WaitTime (16 bits)

The WaitTime field MUST report the time that is needed before the receiving node again becomes available for communication after the transfer of an image. The unit is the second.

The value 0 (zero) MUST indicate that the node is ready.

The value 0xFFFF is reserved and MUST NOT be returned.

A controlling application which receives this status command SHOULD wait for the number of seconds specified in the WaitTime field before trying to resume communication. When resuming communication, the controlling application SHOULD issue a NOP command and wait for acknowledgement. The controlling application MAY attempt resuming communication repeatedly. In that case, the NOP interval SHOULD be five seconds. The NOP interval MUST be at least one second.

4.53.4 Firmware Update Activation Set Command

This command is used to initiate the programming of a previously transferred firmware image. Refer to the Firmware Update Meta Data Status Report Command Status code 0xFD.

This command MAY be issued directly by a controlling device.
Further, the command MAY be scheduled for later execution via the Schedule Command Class.

The Firmware Update Activation Status Report Command MUST be returned in response to this command.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD							
Command = FIRMWARE_UPDATE_ACTIVATION_SET							
Manufacturer ID 1							
Manufacturer ID 2							
Firmware ID 1							
Firmware ID 2							
Checksum 1							
Checksum 2							
Firmware Target							

See description in previous commands.

4.53.5 Firmware Update Activation Status Report

This command is used to advertise the result of a firmware update operation initiated by the Firmware Update Activation Set Command.

The Firmware Update Activation Status Report fields MUST reflect the values specified in the Firmware Update Activation Set Command.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD							
Command = FIRMWARE_UPDATE_ACTIVATION_STATUS_REPORT							
Manufacturer ID 1							
Manufacturer ID 2							
Firmware ID 1							
Firmware ID 2							
Checksum 1							
Checksum 2							
Firmware Target							
Firmware Update Status							

See description in previous commands.

Firmware Update Status (8 bits)

Status	Description
0x00	Invalid combination of manufacturer ID and firmware ID or Firmware Target. The received image will not be stored. **(Version 4)**
0x01	Error activating the firmware. Last known firmware image has been restored. **(Version 4)**
…	*Reserved*
0xFF	Firmware update completed successfully. **(Version 4)**

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

4.54 Firmware Update Meta Data Command Class, version 5

The Firmware Update Meta Data Command Class may be used to transfer a firmware image to or from a Z-Wave node.

Commands not mentioned in this version remain unchanged from version 4.

4.54.1 Compatibility considerations

A device implementing Firmware Update Meta Data Command Class, version 5 MUST also implement Firmware Update Meta Data Command Class, version 4.

Firmware Update Meta Data Command Class, version 5 is backwards compatible with Firmware Update Meta Data Command Class, version 4.

4.54.1.1 New commands, fields and values

The Firmware Update Meta Data Command Class, version 5 introduces the support of a hardware identifier to uniquely identify firmware images that can be loaded on devices. This allows devices running identical software version on different hardware revisions to receive the correct firmware image.

The following commands are extended from version 4:

- Firmware Meta Data Report Command
- Firmware Update Meta Data Request Get Command
- Firmware Update Meta Data Request Report Command
- Firmware Update Meta Data Status Report Command
- Firmware Update Activation Set Command
- Firmware Update Activation Status Report Command

The Firmware Update Meta Data Command Class, version 5 also introduces the support firmware download from the Z-Wave node to the controller.

The following commands are introduced:

- Firmware Update Meta Data Prepare Get Command
- Firmware Update Meta Data Prepare Report Command

4.54.2 Interoperability Considerations

4.54.2.1 Interoperability with v1 devices

Version 2 of the Firmware Update Meta Data Report command introduces a 16-bit Checksum field to follow the variable-length Data field. No fields follow the variable-length Data field in Version 1 of the Firmware Update Meta Data Report command. Version 5 of the Firmware Update Meta Data Report command uses the version 2 format.

At the same time, the specified method for determining the length of the variable-length Data field is that the length must be calculated from the length of the received frame.

The unintended consequence is that a version 1 implementation receiving a Firmware Update Meta Data Report version 2 Command will consider the 16-bit Checksum field to be the last two bytes of the Data field.
Similarly, a version 2 implementation receiving a Firmware Update Meta Data Report version 1 Command will consider the last two bytes of the Data field to be the 16-bit Checksum field.
In either case, the transfer will fail.

Therefore, a controlling device implementing the Firmware Update Meta Data Command Class, version 2 MUST do the following when initiating the transfer of a firmware image:

5. Request the version of the Firmware Update Meta Data Command Class from the target device
6. Use the version of the Firmware Update Meta Data Report implemented by the target device
7. Determine the number of Data bytes that can fit into the Firmware Update Meta Data Report command so that the complete command can still fit into the payload field of the transport frame. The number of available payload bytes in the frame depends on the actual bit rate, the use of security encapsulation as well as the presence of Checksum bytes in the Firmware Update Meta Data Report

4.54.2.2 Checksum calculation

Version 1 of the Firmware Update Meta Data Command Class introduced a 16-bit Checksum field used to verify firmware image integrity. It has been a recommendation to use the CRC-CCITT polynomial for calculating checksums. There is no way to identify what method has been used for calculating the checksum for nodes implementing version 1 to version 4. Nodes having implemented another method for calculating the checksum will find a non-matching checksum.

From Version 5 onwards, the checksum calculation method is mandatory.

For more details about the checksum calculation, refer to [2] Appendix B

4.54.3 Firmware Meta Data Report Command

The Firmware Meta Data Report Command is used to advertise the status of the current firmware in the device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD							
Command = FIRMWARE_MD_REPORT							
Manufacturer ID 1							
Manufacturer ID 2							
Firmware 0 ID 1							
Firmware 0 ID 2							
Firmware 0 Checksum 1							
Firmware 0 Checksum 2							
Firmware Upgradable							
Number of Firmware Targets							
Max Fragment Size 1							
Max Fragment Size 2							
Firmware 1 ID 1							
Firmware 1 ID 2							
…							
Firmware N ID 1							
Firmware N ID 2							
Hardware Version							

All fields not described below are the same as in version 4.

Firmware 0 Checksum (16 bits)

The checksum field MUST carry the checksum of the Firmware 0 Image.

The checksum MUST be calculated using the CRC-CCITT polynomium using initialization value equal to 0x1D0F and 0x1021 (normal representation).

For more details, refer to [2] Appendix B

Hardware Version (8 bits)

The Hardware Version field MUST report a value which is unique to this particular version of the product. It MUST be possible to uniquely identify applicable firmware images via the Manufacturer ID, Firmware ID and the Hardware Version fields. This information allows selecting a firmware image that is guaranteed to work with this particular version of the product.

The Hardware Version field MUST apply to the entire product and not only to the version of the Z-Wave radio chip.

This field MUST report the same value as the Version CC Version Report Command: Hardware Version field.

4.54.4 Firmware Update Meta Data Request Get Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD							
Command = FIRMWARE_UPDATE_MD_REQUEST_GET							
Manufacturer ID 1							
Manufacturer ID 2							
Firmware ID 1							
Firmware ID 2							
Firmware Checksum 1							
Firmware Checksum 2							
Firmware Target							
Fragment Size 1							
Fragment Size 2							
Reserved							Activation
Hardware Version							

All fields not described below are the same as in version 4.

Firmware Checksum (16 bits)

The checksum field MUST carry the checksum of the firmware image about to be transferred.

The checksum MUST be calculated using the CRC-CCITT polynomium using initialization value equal to 0x1D0F and 0x1021 (normal representation).

For more details, refer to [2] Appendix B

Hardware Version (8 bits)

Refer to 4.54.3 Firmware Meta Data Report Command.

4.54.5 Firmware Update Meta Data Request Report Command

This command is used to advertise if the firmware update will be initiated.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD							
Command = FIRMWARE_UPDATE_MD_REQUEST_REPORT							
Status							

Status (8 bits)

The Status field MUST comply with Table 63.

Table 63, Firmware Update Meta Data Request Report::Status encoding

Status	Description
0x00	ERROR. Invalid combination of Manufacturer ID and Firmware ID. The device will not initiate the firmware update.**(Version 1)**
0x01	ERROR. Device expected an authentication event to enable firmware update. The device will not initiate the firmware update. **(Version 1)**
0x02	ERROR. The requested Fragment Size exceeds the Max Fragment Size. The device will not initiate the firmware update. **(Version 3)**
0x03	ERROR. This firmware target is not upgradable. The device will not initiate the firmware update. **(Version 3)**
0x04	ERROR. Invalid Hardware Version. The device will not initiate the firmware update. **(Version 5)**
0xFF	OK. The device will initiate the firmware update of the target specified in the Firmware Update Meta Data Request Get Command. **(Version 1)**

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

4.54.6 Firmware Update Meta Data Status Report Command

This command is used to advertise the firmware update status.

The command MUST be issued when the firmware update is completed or aborted by the device receiving the firmware.

A device MUST NOT issue the Firmware Update Meta Data Get Command after receiving a Firmware Update Meta Data Status Report.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD							
Command = FIRMWARE_UPDATE_MD_STATUS_REPORT							
Status							
WaitTime MSB							
WaitTime LSB							

All fields not described below are the same as in version 4.

Status (8 bits)

The Status field MUST comply with Table 64.

Table 64, Firmware Update Meta Data Status Report::Status encoding

Status	Description
0x00	The device was unable to receive the requested firmware data without checksum error. Number of retries and request sequence of missing frames are implementation specific. The image is not stored. **(Version 1)**
0x01	The device was unable to receive the requested firmware data. Number of retries and request sequence of missing frames are implementation specific. The image is not stored. **(Version 1)**
0x02	The transferred image does not match the Manufacturer ID. The image is not stored. **(Version 4)**
0x03	The transferred image does not match the Firmware ID. The image is not stored. **(Version 4)**
0x04	The transferred image does not match the Firmware Target. The image is not stored. **(Version 4)**
0x05	Invalid file header information. The image is not stored. **(Version 4)**
0x06	Invalid file header format. The image is not stored. **(Version 4)**
0x07	Insufficient memory. The image is not stored. **(Version 4)**
0x08	The transferred image does not match the Hardware version The image is not stored **(Version 5)**
…	*Reserved*
0xFD	Firmware image downloaded successfully, waiting for activation command. **(Version 4)**
0xFE	New image was successfully stored in temporary non-volatile memory. The device does not restart itself. This Status code MUST NOT be used when updating the Z-Wave chip image **(Version 3)**
0xFF	New image was successfully stored in temporary non-volatile memory. The device will now start storing the new image in primary non-volatile memory dedicated to executable code. Then the device will restart itself. **(Version 1)**

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

The status code 0xFD MAY be used by a version 4 device for confirming that an image has been successfully transferred but that the actual firmware update will not be performed until a Firmware Update Activation Set Command is received. The Firmware Update Activation Set Command MAY be delayed for any period of time. The delay MAY be controlled via the Schedule Command Class.

The status code 0xFE MAY be used for confirming the successful transfer of an image which does not necessitate a restart, e.g. security a certificate.
The status code 0xFE MUST NOT be advertised after the transfer of an image for the Firmware ID 0 target (the “Z-Wave chip” image).

Controlling nodes implementing earlier versions of the Firmware Update Meta Data CC do not support status codes defined for newer versions. Therefore, a device returning the Firmware Update Meta Data Status Report Command MUST comply with the version implemented on the controlling device (device sending the image). The device returning the Firmware Update Meta Data Status Report can identify the version of the controlling device based on the Firmware Update Meta Data Request Get. This MAY be done as follows:

- If the Firmware Update Meta Data Request Get does not include Firmware Target and Fragment Size (8 bytes), the controller is version 1 or 2
- If the Firmware Update Meta Data Request Get includes Firmware Target and Fragment Size but not Activation (11 bytes), the controller is version 3
- If the Firmware Update Meta Data Request Get includes Activation but not Hardware Version (12 bytes), the controller is version 4
- If the Firmware Update Meta Data Request Get includes Hardware Version (13 bytes), the controller is version 5.

4.54.7 Firmware Update Activation Set Command

This command is used to initiate the programming of a previously transferred firmware image.
Refer to the Firmware Update Meta Data Status Report Command Status code 0xFD.

This command MAY be issued directly by a controlling device.
Further, the command MAY be scheduled for later execution via the Schedule Command Class.

The Firmware Update Activation Status Report Command MUST be returned in response to this command.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD							
Command = FIRMWARE_UPDATE_ACTIVATION_SET							
Manufacturer ID 1							
Manufacturer ID 2							
Firmware ID 1							
Firmware ID 2							
Firmware Checksum 1							
Firmware Checksum 2							
Firmware Target							
Hardware Version							

All fields not described below are the same as in version 4.

Firmware Checksum (16 bits)

The checksum field MUST carry the checksum of the firmware image about to be activated.

The checksum MUST be calculated using the CRC-CCITT polynomium using initialization value equal to 0x1D0F and 0x1021 (normal representation).

For more details, refer to [2] Appendix B

Hardware Version (8 bits)

Refer to 4.54.3 Firmware Meta Data Report Command.

4.54.8 Firmware Update Activation Status Report Command

This command is used to advertise the result of a firmware update operation initiated by the Firmware Update Activation Set Command.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD							
Command = FIRMWARE_UPDATE_ACTIVATION_STATUS_REPORT							
Manufacturer ID 1							
Manufacturer ID 2							
Firmware ID 1							
Firmware ID 2							
Firmware Checksum 1							
Firmware Checksum 2							
Firmware Target							
Firmware Update Status							
Hardware Version							

All fields not described below are the same as in version 4.

Firmware Checksum (16 bits)

The checksum field MUST carry the checksum of the firmware image activated by the Firmware Update Activation Set Command.

The checksum MUST be calculated using the CRC-CCITT polynomium using initialization value equal to 0x1D0F and 0x1021 (normal representation).

For more details, refer to [2] Appendix B

Hardware Version (8 bits)

Refer to 4.54.3 Firmware Meta Data Report Command.

Firmware Update Status (8 bits)

The Firmware Update Status field MUST comply with Table 65.

Table 65, Firmware Update Activation Status Report::Firmware Update Status encoding

Status	Description
0x00	Invalid combination of manufacturer ID, firmware ID and Hardware Version or Firmware Target. The received image will not be stored. **(Version 5)**
0x01	Error activating the firmware. Last known firmware image has been restored. **(Version 4)**
…	*Reserved*
0xFF	Firmware update completed successfully. **(Version 4)**

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

4.54.9 Firmware Update Meta Data Prepare Get Command

The Firmware Update Meta Data Prepare Get Command is used to request that a firmware download is initiated by the node sending this command.

The Firmware Update Meta Data Prepare Report Command MUST be returned in response to this command when the receiving node is ready to send the requested firmware image.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

Any image transferred via the Firmware Update Meta Data Command Class SHOULD include a fingerprint value to enable the validation of the integrity of the entire image after the transfer. The image and the fingerprint value MUST be packed in one entity which can be stored in one file and transferred as one entity.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD							
Command = FIRMWARE_UPDATE_MD_PREPARE_GET							
Manufacturer ID 1							
Manufacturer ID 2							
Firmware ID 1							
Firmware ID 2							
Firmware Target							
Fragment Size 1							
Fragment Size 2							
Hardware Version							

Refer to 4.54.4 Firmware Update Meta Data Request Get Command for field description.

4.54.10 Firmware Update Meta Data Prepare Report Command

This command is used to advertise if the firmware image has been prepared and is ready to be transferred.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_FIRMWARE_UPDATE_MD							
Command = FIRMWARE_UPDATE_MD_PREPARE_REPORT							
Status							
Firmware Checksum 1							
Firmware Checksum 2							

Status (8 bits)

The Status field MUST comply with Table 66.

Table 66, Firmware Update Meta Data Prepare Report::Status encoding

Status	Description
0x00	ERROR. Invalid combination of Manufacturer ID and Firmware ID. The receiving node MUST NOT initiate the firmware download.**(Version 5)**
0x01	ERROR. Device expected an authentication event to enable firmware update. The receiving node MUST NOT initiate the firmware download. **(Version 5)**
0x02	ERROR. The requested Fragment Size exceeds the Max Fragment Size. The receiving node MUST NOT initiate the firmware download. **(Version 5)**
0x03	ERROR. This firmware target is not downloadable. The receiving node MUST NOT initiate the firmware download. **(Version 5)**
0x04	ERROR. Invalid Hardware Version. The receiving node MUST NOT initiate the firmware download. **(Version 5)**
0xFF	OK. The receiving node can initiate the firmware download of the target specified in the Firmware Update Meta Data Prepare Get Command. **(Version 5)**

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Firmware Checksum (16 bits)

The checksum field MUST carry the checksum of the firmware image about to be transferred.

The checksum MUST be calculated using the CRC-CCITT polynomium using initialization value equal to 0x1D0F and 0x1021 (normal representation).

For more details, refer to [2] Appendix B

4.54.11 Examples

4.54.11.1 Identifying firmware revisions

The different fields Manufacturer ID, Firmware ID and Hardware version are used to identify compatible firmware image with the product.

The Version Command Class is also used for identifying the Firmware version/subversion

An example of the different fields' usage for a wall outlet is shown in Table 67

Table 67, Labelling different Firmware revisions

Product	Firmware identification			
	Manufacturer ID	Hardware Version	Firmware ID	Firmware version /sub-version
Initial Wall Outlet	0x0001	0x01	0x0001	0x01 / 0x01
New revision				
Fixing previous bugs	0x0001	0x01	0x0001	0x01 / 0x02
New revision				
Introducing new features (e.g. multiple press)	0x0001	0x01	0x0001	0x02 / 0x01
New revision				
US version (initial version)	0x0001	0x01	0x0001	0x02 / 0x01
China version (matching local requirements)	0x0001	0x01	0x0002	0x02 / 0x01
New revision				
Proximity Sensor added (US)	0x0001	0x02	0x0001	0x02 / 0x01
Proximity Sensor added (China)	0x0001	0x02	0x0002	0x02 / 0x01
New revision				
Adding S2 Capability (without proximity sensor, US)	0x0001	0x01	0x0001	0x03 / 0x01
Adding S2 Capability (without proximity sensor, China)	0x0001	0x01	0x0002	0x03 / 0x01
Adding S2 Capability (with proximity sensor, US)	0x0001	0x02	0x0001	0x03 / 0x01
Adding S2 Capability (with proximity sensor, US)	0x0001	0x02	0x0002	0x03 / 0x01

An illustration of a controller retrieving the firmware information is given in Figure 24

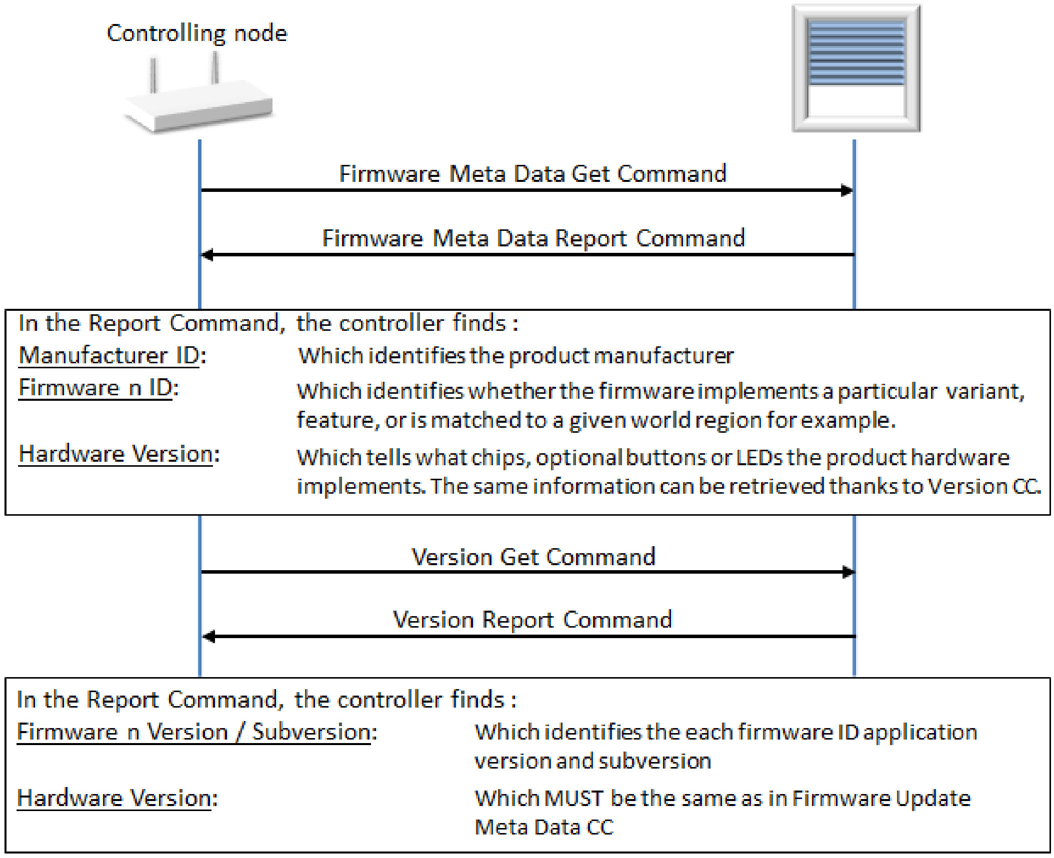


Figure 24, Identifying compatible firmware images

4.54.11.2 Firmware download

The use of the Firmware Update Meta Data Prepare Get and Firmware Update Meta Data Prepare Report commands is illustrated in Figure 25.

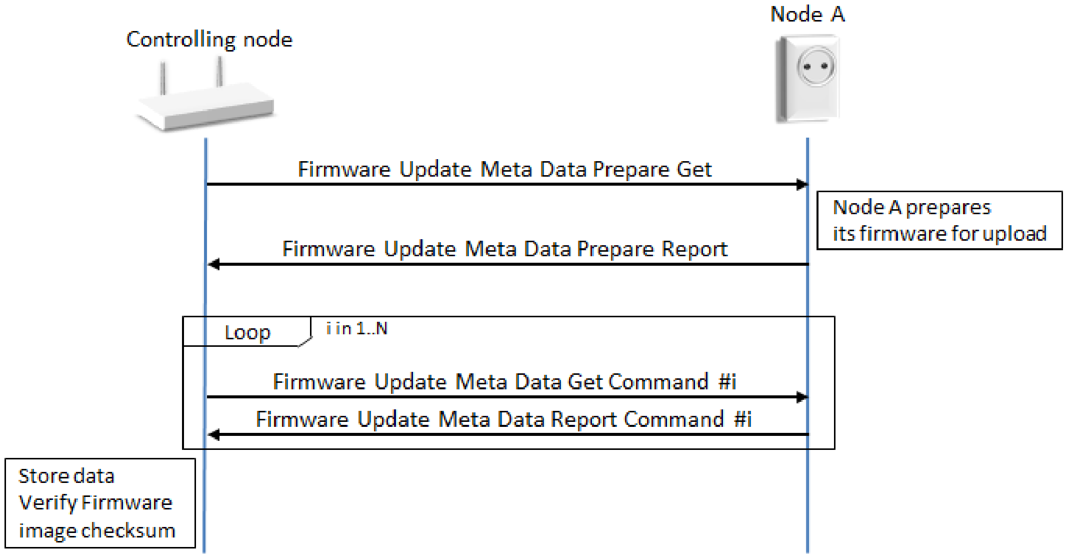


Figure 25, Firmware download flow diagram

4.55 Geographic Location Command Class, version 1

The Geographic Location Command Class is used to read latitude and longitude from another device. The latitude and longitude may also be set according to the geographic location in question. Date and geographic location may be used to calculate sunrise and sunset for e.g. automatic lighting control.

4.55.1 Geographic Location Set Command

The Geographic Location Set Command used to set latitude and longitude.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_GEOGRAPHIC_LOCATION							
Command = GEOGRAPHIC_LOCATION_SET							
Longitude Degrees							
Long. Sign	Longitude Minutes						
Latitude Degrees							
Lat. Sign	Latitude Minutes						

Longitude (16 bits)

The longitude determines one's location on the earth's surface, East or West of the Greenwich Meridian. The Greenwich Meridian is located at the Greenwich observatory, in Greenwich, England to be the geographic point for where East and West meet. Therefore, Greenwich Meridian is indicated as 0° longitude. Longitude values for points East of the Meridian are always positive, while points West of the Meridian are always negative. Valid ranges are for degrees (from -180 to 180) and minutes (0-59). Other values will be interpreted as 0.

Latitude (16 bits)

The latitude determines one's location on the earth's surface, North or South of the Equator. Latitude is measured between -90° South, and +90° North of the Equator point (0°). Valid ranges are for degrees (from -90 to 90) and minutes (0-59). Other values will be interpreted as 0.

4.55.2 Geographic Location Get Command

The Geographic Location Get Command is used to request latitude and longitude from a device.

The Geographic Location Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_GEOGRAPHIC_LOCATION							
Command = GEOGRAPHIC_LOCATION_GET							

4.55.3 Geographic Location Report Command

The Geographic Location Report Command returns latitude and longitude from a device in a Z-Wave network.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_GEOGRAPHIC_LOCATION							
Command = GEOGRAPHIC_LOCATION_REPORT							
Longitude Degrees							
Long. Sign	Longitude Minutes						
Latitude Degrees							
Lat. Sign	Latitude Minutes						

Refer to description under the Geographic Location Set Command.

4.56 Grouping Name Command Class, version 1 [DEPRECATED]

THIS COMMAND CLASS HAS BEEN DEPRECATED

A device MAY implement this command class, but it is RECOMMENDED that new implementations use the Association Group Information (AGI) Command Class for naming association groups.
If implementing this command class, it is RECOMMENDED that the Association Group Information (AGI) Command Class is also implemented.

The Grouping Name Command Class is used to transfer name of groupings (as defined by the grouping identifier in the Association Command Class).

4.56.1 Grouping Name Set Command

The Grouping Name Set Command used to set a grouping Identifier name.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_GROUPING_NAME							
Command = GROUPING_NAME_SET							
Grouping identifier							
Reserved					Char. Presentation		
Grouping Name 1							
…							
Grouping Name N							

Grouping Identifier (8 bits)

The field specifies the Grouping ID.

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Char. Presentation (3 bits)

The char presentation identifier MAY be set to the following values:

Char. Presentation	Description
0	Using standard ASCII codes, see appendices in [2] (values 128-255 are ignored)
1	Using standard and OEM Extended ASCII codes, see appendices in [2]
2	Unicode UTF-16

Note: Devices supporting Unicode UTF-16 characters can have strings of a maximum of 8 characters because each character is described by a 2 byte long decimal representation. The first byte is the most significant byte. I.e. if there is one Unicode character in the set frame the char 1 will be MSB and char 2 will be LSB of the Unicode character.

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Grouping Name (N bytes)

Grouping name using specified character representation.

The Grouping name MAY have a maximum of 16 characters and a minimum of 0 characters. The number of character fields transmitted MUST be determined from the frame length. If a frame with more than 16 characters is received, the receiving node MUST ignore any characters following the 16th character.

4.56.2 Grouping Name Get Command

The Grouping Name Get Command is used to request a grouping Identifier name.

The Grouping Name Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing.
The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_GROUPING_NAME							
Command = GROUPING_NAME_GET							
Grouping identifier							

Grouping Identifier (8 bits)

The field specifies the grouping identifier.

4.56.3 Group Name Report Command

The Grouping Name Report Command is used to report the grouping Identifier name.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_GROUPING_NAME							
Command = GROUPING_NAME_REPORT							
Grouping identifier							
Reserved					Char. Presentation		
Grouping Name 1							
…							
Grouping Name N							

Grouping Identifier (8 bits)

The field specifies the grouping identifier.

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Char. Presentation (3 bits)

Refer to description under the Grouping Name Set Command

Grouping Name (N bytes)

The Grouping Name fields contain the assigned group name. The Group Name can have a maximum of 16 characters and a minimum of 0 characters.

4.57 Hail Command Class, version 1 [DEPRECATED]

THIS COMMAND CLASS HAS BEEN DEPRECATED

The Hail Command Class used by applications to hail other devices in the Z-Wave network. The usage of the Hail Command Class is application specific.

4.57.1 Hail Command

Application can send unsolicited Hail Command to other devices in a Z-Wave network.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_HAIL							
Command = HAIL							

4.58 HRV Status Command Class, version 1

The residential Heat Recovery Ventilation (HRV) Status Command Class is used to read out a number of parameters in the ventilation system.

4.58.1 HRV Status Get Command

The residential HRV Status Get Command is used to request specific parameters from the ventilation system.

The HRV Status Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_HRV_STATUS							
Command = HRV_STATUS_GET							
Status Parameter							

Status Parameter (8 bits)

The status parameter used to indicate which status parameter that is requested.

Status Parameter	Value
Outdoor Air temperature	0
Supply Air (to room) temperature	1
Exhaust Air (from room) temperature	2
Discharge Air temperature	3
Room temperature	4
Relative Humidity in room	5
Remaining filter life	6

4.58.2 HRV Status Report Command

The residential HRV Status Report Command used to report a specific status parameter in response to a HRV Status Get.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_HRV_STATUS							
Command = HRV_STATUS_REPORT							
Status Parameter							
Precision			Scale		Size		
Value 1							
….							
Value N							

Status Parameter (8 bits)

The status parameter used to indicate which status parameter that is reported. Refer to 4.58.1 HRV Status Get for possible values.

Precision (3 bits)

The precision field describes what the precision of the setpoint value is. The number indicates the number of decimals. The decimal value 1025 with precision 2 is therefore equal to 10.25.

Scale (2 bits)

The scale field indicates the scale used the list of possible scales are given below:

Status Parameter	Scale	Value
Outdoor Air temperature	Celsius (C)	0
	Fahrenheit (F)	1
Supply Air (to room) temperature	Celsius (C)	0
	Fahrenheit (F)	1
Exhaust Air (from room) temperature	Celsius (C)	0
	Fahrenheit (F)	1
Discharge Air temperature	Celsius (C)	0
	Fahrenheit (F)	1
Room temperature	Celsius (C)	0
	Fahrenheit (F)	1
Relative Humidity in room	Percentage (%)	0
Remaining filter life	Percentage (%)	0

Size (3 bits)

The size field indicates the number of bytes used for the sensor value. This field can take values from 1 (001b), 2 (010b) or 4 (100b).

Value (N bytes)

The value is a signed field. The value MAY be 1, 2 or 4 bytes in size. This first byte is the most significant byte. The table below show examples of signed decimal value with their hexadecimal equivalents.

Signed 1 byte decimal value	Hexadecimal
127	0x7F
25	0x19
2	0x02
1	0x01
0	0x00
-1	0xFF
-2	0xFE
-25	0xE7
-128	0x80

Signed 2 bytes decimal value	Hexadecimal
32767	0x7FFF
1025	0x0401
2	0x0002
1	0x0001
0	0x0000
-1	0xFFFF
-2	0xFFFE
-1025	0xFBFF
-32768	0x8000

4.58.3 HRV Status Supported Get Command

The HRV Status Supported Get Command is used to request a bitmap of the supported status parameters.

The HRV Status Supported Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing.
The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_HRV_STATUS							
Command = HRV_STATUS_SUPPORTED_GET							

4.58.4 HRV Status Supported Report Command

The HRV Status Supported Report Command used to report a bitmap indicating the supported status parameters.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_HRV_STATUS							

Command = HRV_STATUS_SUPPORTED_REPORT
Bit Mask 1
…
Bit Mask N

Bit Mask (N bytes)

The Bit Mask fields describe the supported status parameters from the ventilation system. Bit 0 in the Bit Mask 1 field used to indicate if the status parameter “Outdoor Air Temperature” is supported, 0 indicating not supported and 1 indicating supported. Bit 1 in the Bit Mask 1 field used to indicate if the status parameter “Supply Air Temperature” is supported and so forth. All available status parameters are given in Section 4.58.1 HRV Status Get.

It is only necessary to send the Bit Mask fields 1 and up to the one indicating the last support status parameter. The number of Bit Mask fields transmitted MUST be determined from the length field in the frame.

4.59 HRV Control Command Class, version 1

The Heat Recovery Ventilation (HRV) Control Command Class is introducing control of Heat Recovery Ventilation systems via the Z-Wave interface.

4.59.1 HRV Mode Set

The HRV Mode Set Command used to set the desired mode in the device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_HRV_CONTROL							
Command = HRV_CONTROL_MODE_SET							
Reserved			Mode				

Mode (5 bits)

The mode identifier MAY be set to the following values:

Mode	Name	Description
0	Off	The HRV system is in the off state, frost protection can occur.
1	Demand / Automatic	The HRV system is controlled based on sensor input.
2	Schedule	The HRV system is controlled based on predefined input from the factory and/or user/installer.
3	Energy Savings Mode	The HRV system will be put into a reduced heat / ventilation mode.
4	Manual	The HRV system is in manual mode. The command HRV Manual Control Set may be used to manually control the device.

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

4.59.2 HRV Mode Get Command

The HRV Mode Get Command is used to request the current mode from the device.

The HRV Mode Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing.
The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_HRV_CONTROL							
Command = HRV_CONTROL_MODE_GET							

4.59.3 HRV Mode Report Command

The HRV Mode Report Command used to report the mode from the device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_HRV_CONTROL							
Command = HRV_CONTROL_MODE_REPORT							
Reserved			Mode				

Mode (8 bits)

Refer to description under the HRV Mode Set command.

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

4.59.4 HRV Bypass Set Command

The HRV Bypass Set Command used to set the bypass mode when the ventilation system is set to manual mode. If the system is not in manual mode while receiving this command it MUST be ignored.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_HRV_CONTROL							
Command = HRV_CONTROL_BYPASS_SET							
Bypass							

Bypass (8 bits)

The value MAY be 0x00 (close) or 0xFF (open).

Furthermore, the field MAY carry a percentage value between 1 to 99 (0x01 - 0x63). If the ventilation system supports modulated bypass, the percentage value will represent the aperture of the bypass. If the system does not support the full range of aperture steps, the values SHOULD be mapped linearly over the entire scale.

If ventilation system does support modulated bypass the values from 1 to 99 MUST be interpreted as fully open.

The value 254 (0xFE) MAY be used to set the bypass into automatic mode.

4.59.5 HRV Bypass Get Command

The HRV Bypass Get Command is used to request the current bypass setting.

The HRV Bypass Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_HRV_CONTROL							
Command = HRV_CONTROL_BYPASS _GET							

4.59.6 HRV Bypass Report Command

The HRV Bypass Report command used to report the current bypass parameters.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_HRV_CONTROL							
Command = HRV_CONTROL_BYPASS_REPORT							
Bypass							

Bypass (8 bits)

See description under Section 4.59.4 HRV Bypass Set.

4.59.7 HRV Ventilation Rate Set Command

The HRV Ventilation Rate Set Command used to set the ventilation rate when the ventilation system is set to manual mode. If the system is not in manual mode while receiving this command it MUST be ignored.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_HRV_CONTROL							
Command = HRV_CONTROL_VENTILATION_RATE_SET							
Ventilation Rate							

Ventilation Rate (8 bits)

The value MAY be 0x00 (off) or 0xFF (on).

The field MAY carry a percentage value between 1 to 99 (0x01 - 0x63). A ventilation system MAY map the values 1..99 to less than 99 steps.

4.59.8 HRV Ventilation Rate Get Command

The HRV Ventilation Rate Get Command is used to request the current ventilation rate setting.

The HRV Ventilation Rate Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_HRV_CONTROL							
Command = HRV_CONTROL_ VENTILATION_RATE _GET							

4.59.9 HRV Ventilation Rate Report Command

The HRV Ventilation Rate Report Command used to report the current ventilation rate setting.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_HRV_CONTROL							
Command = HRV_CONTROL_ VENTILATION_RATE _REPORT							
Ventilation Rate							

Ventilation Rate (8 bits)

See description under Section 4.59.7 HRV Ventilation Rate Set.

4.59.10 HRV Mode Supported Get Command

The HRV Mode Supported Get Command is used to request the supported modes from the device.

The HRV Mode Supported Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_HRV_CONTROL							
Command = HRV_CONTROL_MODE_SUPPORTED_GET							

4.59.11 HRV Mode Supported Report Command

The HRV Mode Supported Report Command used to report the supported modes from the device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_HRV_CONTROL							
Command = HRV_CONTROL_MODE_SUPPORTED_REPORT							
Reserved				Manual Control Supported			
Bit Mask 1							
…							
Bit Mask N							

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Manual Control Supported (3 bits)

The manual control supported bits describes the supported manual control modes of the ventilation system. The mode is supported if the bit is 1; if the bit is 0 then the mode is not supported.

The bits are mapped to the following controls:

Bit	Name
0	Bypass Open / Close
1	Bypass Auto
2	Modulated Bypass
3	Ventilation Rate

E.g. a ventilation system supporting only open and close would report Manual Control Supported = 0x01.

Bit Mask (N bytes)

The Bit Mask fields describe the supported modes by the device.

- Bit 0 in Bit Mask 1 field indicates if Mode = 0 (Off) is supported.
- Bit 1 in Bit Mask 1 field indicates if Mode = 1 (Demand / Automatic) is supported.
- …

If the Mode is supported the bit MUST be set to 1. If the Mode is not supported the bit MUST be set to 0. It is only necessary to send the Bit Mask fields from 1 and up to the one indicating the last supported mode. The number of Bit Mask fields transmitted MUST be determined from the length field in the frame.

All available modes are given in Section 4.59.1 HRV Mode Set.

4.60 Humidity Control Mode Command Class, version 1

The Humidity Control Mode Command Class is used to control a humidity control device.

4.60.1 Humidity Control Mode Set Command

The Humidity Control Mode Set Command is used to set the humidity control mode in the device.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_HUMIDITY_CONTROL_MODE</td></tr>
<tr><td colspan="8">Command = HUMIDITY_CONTROL_MODE_SET</td></tr>
<tr><td colspan="4">Reserved</td><td colspan="4">Mode</td></tr>
</table>

Reserved

The reserved field is for future use. Reserved bits MUST be set to zero by a sending node. Reserved bits MUST be ignored by a receiving node.

Mode (4 bits)

The encoding of the humidity control mode field MUST be according to Table 68.

Table 68, Mode

Mode	Description	Version
0	Off – Humidity control system is off.	1
1	Humidify – The system will attempt to raise humidity to the humidifier setpoint.	1
2	De-humidify – The system will attempt to lower the humidity to the de-humidifier setpoint.	1

All other values are reserved. Reserved values MUST NOT be used by a sending node and MUST be ignored by a receiving node.

4.60.2 Humidity Control Mode Get Command

The Humidity Control Mode Get Command is used to request the supported humidity control modes from the device. The Humidity Control Mode Report Command MUST be returned in response to a Humidity Control Mode Get Command.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_HUMIDITY_CONTROL_MODE							
Command = HUMIDITY_CONTROL_MODE_GET							

4.60.3 Humidity Control Mode Report Command

The Humidity Control Mode Report Command is used to report the humidity control mode from the device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_HUMIDITY_CONTROL_MODE							
Command = HUMIDITY_CONTROL_MODE_REPORT							
Reserved				Mode			

Reserved

The reserved field is for future use. Reserved bits MUST be set to zero by a sending node. Reserved bits MUST be ignored by a receiving node.

Mode (4 bits)

Refer to the description under the Humidity Control Mode Set Command.

4.60.4 Humidity Control Mode Supported Get Command

The Humidity Control Mode Supported Get Command is used to request the supported modes from the device. The Humidity Control Mode Supported Report Command MUST be returned in response to a Humidity Control Mode Supported Get Command.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_HUMIDITY_CONTROL_MODE							
Command = HUMIDITY_CONTROL_MODE_SUPPORTED_GET							

4.60.5 Humidity Control Mode Supported Report Command

The Humidity Control Mode Supported Report Command is used to report the supported humidity control modes from the device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_HUMIDITY_CONTROL_MODE							
Command = HUMIDITY_CONTROL_MODE_SUPPORTED_REPORT							
Bit Mask 1							
…							
Bit Mask N							

Bit Mask (N bytes)

The Bit Mask fields MUST advertise the supported humidity control modes by the device. The encoding of the Size field MUST be according to Table 69.

Table 69, Supported Humidity Control Mode

Field	Bit	Support for Control Mode	Version
Bit Mask 1	0	*Reserved*	1
Bit Mask 1	1	1 (Humidifier)	1
Bit Mask 1	2	2 (De-humidifer)	1

All Bit Mask fields and bits not specified above are reserved. Reserved bits MUST be set to zero by a sending node. Reserved bits MUST be ignored by a receiving node.

For each individual bit, The value ‘0’ MUST signify that the actual Mode is not supported.
The value ‘1’ MUST signify that the actual Mode is supported.

A sending node MAY omit trailing Bit Mask fields if they are not needed. The number of Bit Mask fields MUST be determined from the length field in the frame.

4.61 Humidity Control Operating State Command Class, version 1

The Humidity Control Operating State Command Class is used to obtain the operating state of the humidity control device.

4.61.1 Humidity Control Operating State Get Command

The Humidity Control Operating State Get Command is used to request the operating state of the humidity control device. The Humidity Control Operating State Report Command MUST be returned in response to a Humidity Control Operating State Get Command.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_HUMIDITY_CONTROL_OPERATING_STATE</td></tr>
<tr><td colspan="8">Command = HUMIDITY_CONTROL_OPERATING_STATE_GET</td></tr>
</table>

4.61.2 Humidity Control Operating State Report Command

The Humidity Control Operating State Report Command is used to report the operating state of the humidity control device.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_HUMIDITY_CONTROL_OPERATING_STATE</td></tr>
<tr><td colspan="8">Command = HUMIDITY_CONTROL_OPERATING_STATE_REPORT</td></tr>
<tr><td colspan="4">Reserved</td><td colspan="4">Operating State</td></tr>
</table>

Reserved

The reserved field is for future use. Reserved bits MUST be set to zero by a sending node. Reserved bits MUST be ignored by a receiving node.

Operating State (4 bits)

The Operating State field MUST advertise the current operating state.

The encoding of the Operating State field MUST be according to Table 70.

Table 70, Operating State

Operating State	Description	Version
0	Idle	1
1	Humidifying	1
2	De-humidifying	1

All other values are reserved. Reserved values MUST NOT be used by a sending node and MUST be ignored by a receiving node.

4.62 Humidity Control Setpoint Command Class, version 1

The Humidity Control Setpoint Command Class is used for humidity control setpoint handling.

4.62.1 Humidity Control Setpoint Set Command

The Humidity Control Setpoint Set Command is used to set the humidity control setpoint in the device.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_HUMIDITY_CONTROL_SETPOINT</td></tr>
<tr><td colspan="8">Command = HUMIDITY_CONTROL_SETPOINT_SET</td></tr>
<tr><td colspan="4">Reserved</td><td colspan="4">Setpoint Type</td></tr>
<tr><td colspan="3">Precision</td><td colspan="2">Scale</td><td colspan="3">Size</td></tr>
<tr><td colspan="8">Value 1</td></tr>
<tr><td colspan="8">…</td></tr>
<tr><td colspan="8">Value N</td></tr>
</table>

Reserved

The reserved field is for future use. Reserved bits MUST be set to zero by a sending node. Reserved bits MUST be ignored by a receiving node.

Setpoint Type (4 bits)

The Setpoint Type field MUST specify the humidity control setpoint to be set in the humidity control device. The encoding of the Setpoint Type field MUST be according to Table 50.

Table 71, Setpoint Type

Setpoint Type	Description	Version
1	Humidifier	1
2	De-humidifier	1

All other values are reserved. Reserved values MUST NOT be used by a sending node and MUST be ignored by a receiving node.

Precision (3 bits)

The precision field MUST specify the precision of the setpoint value. The number indicates the number of decimals. As an example, the decimal value 1025 with precision two (2) is equal to 10.25.

Scale (2 bits)

The Scale field MUST specify the humidity scale used. The encoding of the Scale field MUST be according to Table 72.

Table 72, Scale values

Scale	Description	Version
0	Percentage value	1
1	Absolute humidity (g/m^3)	1

All other values are reserved. Reserved values MUST NOT be used by a sending node and MUST be ignored by a receiving node.

Size (3 bits)

The Size field MUST specify the number of bytes used for the Value field. The encoding of the Size field MUST be according to Table 73.

Table 73, Size values

Size	Description	Version
1	Value field is 1 byte long	1
2	Value field is 2 bytes long	1
4	Value field is 4 bytes long	1

All other values are reserved. Reserved values MUST NOT be used by a sending node and MUST be ignored by a receiving node.

Value (N bytes)

The value is a signed field. The field MAY be one, two, or four bytes in size as specified by the Size field. Value 1 is the most significant byte.

4.62.2 Humidity Control Setpoint Get Command

The Humidity Control Setpoint Get Command is used to request the given humidity control setpoint type in a device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_HUMIDITY_CONTROL_SETPOINT							
Command = HUMIDITY_CONTROL_SETPOINT_GET							
Reserved				Setpoint Type			

Reserved

The reserved field is for future use. Reserved bits MUST be set to zero by a sending node. Reserved bits MUST be ignored by a receiving node.

Setpoint Type (4 bits)

Refer to the description under the Humidity Control Setpoint Set Command.

4.62.3 Humidity Control Setpoint Report Command

The Humidity Control Setpoint Report Command is used to report the value of the humidity control setpoint type in the device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_HUMIDITY_CONTROL_SETPOINT							
Command = HUMIDITY_CONTROL_SETPOINT_REPORT							
Reserved				Setpoint Type			
Precision			Scale		Size		
Value 1							
…							
Value N							

Reserved

The reserved field is for future use. Reserved bits MUST be set to zero by a sending node. Reserved bits MUST be ignored by a receiving node.

Setpoint Type (4 bits)

Refer to the description under the Humidity Control Setpoint Set Command.

Precision (3 bits)

Refer to the description under the Humidity Control Setpoint Set Command.

Scale (2 bits)

Refer to the description under the Humidity Control Setpoint Set Command.

Size (3 bits)

Refer to the description under the Humidity Control Setpoint Set Command.

Value (N bytes)

Refer to the description under the Humidity Control Setpoint Set Command.

4.62.4 Humidity Control Setpoint Supported Get Command

The Humidity Control Setpoint Supported Get Command is used to request the humidity control setpoint types supported by the device. The Humidity Control Setpoint Supported Report Command MUST be returned in response to a Humidity Control Setpoint Supported Get Command.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_HUMIDITY_CONTROL_SETPOINT							
Command = HUMIDITY_CONTROL_SETPOINT_SUPPORTED_GET							

4.62.5 Humidity Control Setpoint Supported Report Command

The Humidity Control Setpoint Supported Report Command is used to report the humidity control setpoint types supported by the device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_HUMIDITY_CONTROL_SETPOINT							
Command = HUMIDITY_CONTROL_SETPOINT_SUPPORTED_REPORT							
Bit Mask 1							
…							
Bit Mask N							

Bit Mask (N bytes)

The Bit Mask fields MUST advertise the humidity control setpoint types supported by the device.The encoding of the Size field MUST be according to Table 74.

Table 74, Humidity Control Setpoint Supported Report values

Field	Bit	Support for Setpoint Type	Version
Bit Mask 1	0	*Reserved*	1
Bit Mask 1	1	1 (Humidifier)	1
Bit Mask 1	2	2 (De-humidifer)	1

All Bit Mask fields and bits not specified above are reserved. Reserved bits MUST be set to zero by a sending node. Reserved bits MUST be ignored by a receiving node.

For each individual bit, The value ‘0’ MUST signify that the actual Setpoint Type is not supported. The value ‘1’ MUST signify that the actual Setpoint Type is supported.

A sending node MAY omit trailing Bit Mask fields if they are not needed. The number of Bit Mask fields MUST be determined from the length field in the frame.

4.62.6 Humidity Control Setpoint Scale Supported Get Command

The Humidity Control Setpoint Scale Supported Get Command is used to retrieve the supported scales of the humidity control setpoints in the device. The Humidity Control Scale Supported Report Command MUST be returned in response to a Humidity Control Scale Supported Get Command.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_HUMIDITY_CONTROL_SETPOINT							
Command = HUMIDITY_CONTROL_SETPOINT_SCALE_SUPPORTED_GET							
Reserved				Setpoint Type			

Reserved

The reserved field is for future use. Reserved bits MUST be set to zero by a sending node. Reserved bits MUST be ignored by a receiving node.

Setpoint Type (4 bits)

Refer to the description under the Humidity Control Setpoint Set Command.

4.62.7 Humidity Control Setpoint Scale Supported Report Command

The Humidity Control Setpoint Scale Supported Report Command indicates the supported scales of the humidity control setpoint in a bit mask format.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_HUMIDITY_CONTROL_SETPOINT							
Command = HUMIDITY_CONTROL_SETPOINT_SCALE_SUPPORTED_REPORT							
Reserved				Scale Bit Mask			

Reserved

The reserved field is for future use. Reserved bits MUST be set to zero by a sending node. Reserved bits MUST be ignored by a receiving node.

Scale Bit Mask (4 bits)

Refer to the description under the Humidity Control Setpoint Set Command.

4.62.8 Humidity Control Setpoint Capabilities Get Command

The Humidity Control Setpoint Capabilities Get Command is used to request the minimum and maximum setpoint values for a given humidity control Setpoint Type. The Humidity Control Setpoint Capabilities Report Command MUST be returned in response to a Humidity Control Setpoint Capabilities Get Command.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_HUMIDITY_CONTROL_SETPOINT							
Command = HUMIDITY_CONTROL_SETPOINT_CAPABILITIES_GET							
Reserved				Setpoint Type			

Reserved

The reserved field is for future use. Reserved bits MUST be set to zero by a sending node. Reserved bits MUST be ignored by a receiving node.

Setpoint Type (4 bits)

Refer to the description under the Humidity Control Setpoint Set Command.

4.62.9 Humidity Control Setpoint Capabilities Report Command

The Humidity Control Setpoint Capabilities Report Command is used to report the minimum and maximum values of the requested humidity control setpoint type.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_HUMIDITY_CONTROL_SETPOINT							
Command = HUMIDITY_CONTROL_SETPOINT_CAPABILITIES_REPORT							
Reserved				Setpoint Type			
Precision			Scale		Size		
Minimum Value 1							
…							
Minimum Value N							
Precision			Scale		Size		
Maximum Value 1							
…							
Maximum Value N							

Refer to the description under the Humidity Control Setpoint Set Command for other parameter/field descriptions.

4.63 Indicator Command Class, version 1

The Indicator Command Class is used to show the actual state, level etc. on a device.

4.63.1 Indicator Set Command

The Indicator Set Command may be used to enable or disable an indicator.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_INDICATOR							
Command = INDICATOR_SET							
Value							

Value (8 bits)

The value MAY be 0x00 (off/disable) or 0xFF (on/enable).

The field MAY carry values from 1 to 99. In case the indicator does not have the capability to show the range from 1 to 99, the values 1 to 99 MUST be presented as (on/enable).

4.63.2 Indicator Get Command

The Indicator Get Command is used to request the state of an indicator.

The Indicator Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing.
The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_INDICATOR							
Command = INDICATOR_GET							

4.63.3 Indicator Report Command

The Indicator Report Command may be used to advertise the state of an indicator.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_INDICATOR							
Command = INDICATOR_REPORT							
Value							

Value (8 bits)

Refer to description under the Indicator Set Command.

4.64 Indicator Command Class, version 2

The Indicator Command Class, version 2 is used to manipulate indicator resources in a supporting device. An indicator may be an LED, an LCD display or a buzzer.

One example of use is a central entry control application that uses the indicators of an entry control keypad device to advertise system status information.

If the Indicator Command Class is implemented in a devices based on Role Type RSS (using Wake Up Command Class), the control of indicators must be synchronized with the Wake Up Notification. The controller must therefore wait for the Wake Up Notification, before it can control the indicators. The device implementing the indicator Command Class must therefore send the Wake Up Notification in case it expects to be controlled.

4.64.1 Compatibility Considerations

A supporting device MAY use its indicator resources for local status reporting until the first Indicator Set command is received. After receiving an Indicator Set command, the supporting device MUST NOT use its indicator resources for local status reporting until it has been reset to default or re-included in the network. An exception to this MAY be interface feedback functionality like a short beep on each button press. Likewise, transmission failure to the central controller MAY be advertised via the use of local indications.
As an example, an entry control keypad device may feature an LCD backlight. If the backlight is advertised as an indicator resource, a central control application also has to control the backlight of the keypad, e.g. in response to local user activity and to temporarily draw attention to all keypads when an alarm is enabled.

The Indicator Command Class, version 1 provides a single unspecified indicator resource.

The Indicator Command Class, version 2 supports multiple indicator resources. Each indicator resource may implement a number of properties such as on/off and color.

The Indicator Command Class, version 2 renames the Value field of version 1 to "Indicator 0 Value". The Indicator 0 Value field properties are unspecified as in version 1.

A supporting version 2 device MUST ignore the Indicator 0 Value of the Indicator Set command if other Value fields are specified.
A supporting version 2 device MUST map the Indicator 0 Value of the Indicator Set command to a supported indicator if no other fields are specified.

Since the Indicator 0 Value field is always present in Indicator Set commands, a controlling device MUST send the correct value in every Indicator Set command.
A controlling version 2 device MUST ignore the Indicator 0 Value of the Indicator Report command if other Value fields are specified.
A controlling version 2 device MUST interpret the Indicator 0 Value of the Indicator Report command the same way as in the Indicator Set command.

4.64.2 Service Discovery Considerations

A control application may discover available indicators and their properties via the Indicator Supported Report Command. Color properties allow the application to discover the color of actual indicators. A single indicator, e.g. ARMED_NOT_ARMED, may provide two colors; red and green.

Likewise, a control application may detect the availability of indicator resources using LCD, e-paper or similar technologies to implement an indicator resource that keeps its state when the device goes to sleep. The device may advertise the property “Low_power” to indicate this.

4.64.3 Indicator Set Command

The Indicator Set Command is used to manipulate one or more indicator resources of a supporting device.

A supporting device MAY use its indicator resources for local status reporting until the first Indicator Set command is received. After receiving an Indicator Set command, the supporting device MUST NOT use its indicator resources for local status reporting until it has been reset to default or re-included in the network. An exception to this MAY be interface feedback functionality like a short beep on each button press. Likewise, transmission failure to the central controller MAY be advertised via the use of local indications.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_INDICATOR							
Command = INDICATOR_SET							
Indicator 0 Value (Indicator ID 0 = 0x00, Property ID 0 = 0x01)							
Reserved			Indicator Object Count				
Indicator ID 1							
Property ID 1							
Value 1							
Indicator ID 2							
Property ID 2							
Value 2							
…….							
Indicator ID N							
Property ID N							
Value N							

Indicator 0 Value (1 byte)

This field provides backwards compatibility with devices implementing the Indicator Command Class, version 1.

A receiving device MUST ignore this field if other indicator resources are specified.
A receiving device MUST map this value to a supported indicator resource if no other fields are specified.

Refer to the compatibility considerations found in 4.64.1.

Reserved

This field MUST be set to 0 by a sending device and MUST be ignored by a receiving device.

Indicator Object Count (1 byte)

This field is used to advertise the number of indicator objects carried in the actual command. An indicator object MUST comprise an Indicator ID, Property ID and Value field. There MAY be several indicator objects carrying the same Indicator ID but different Property IDs, e.g. Color and Blink pattern.

A receiving device MUST ignore an indicator object if an unsupported Indicator ID or Property ID is specified. The receiving device MUST continue parsing the command if more fields follow the ignored indicator object.

Indicator ID n (1 byte)

This field is used to identify the actual indicator resource.

There MAY be several indicator objects carrying the same Indicator ID but different Property IDs, e.g. Color and Blink pattern.

Property ID n (1 byte)

This field is used to identify the specific property of the indicator resource identified by the Indicator ID.

There MAY be several indicator objects carrying the same Indicator ID but different Property IDs, e.g. Color and Blink pattern.

Value n (1 byte)

This field is used to specify the value to assign to the specific property identified by the Indicator ID and the Property ID fields.

4.64.3.1 Indicator IDs

The identifiers used in Indicator ID fields MUST comply with the definitions found in Table 75.

Table 75, Indicator CC :: Indicator IDs

Indicator ID	Description	Appearance and use
0x00	NA	Not Applicable. MUST NOT be used for other purposes than service discovery and rejecting invalid requests.
0x01	ARMED	Use to indicate that alarm is armed (RECOMMENDED color: Red)
0x02	NOT_ARMED	Use to indicate that alarm is disarmed (RECOMMENDED color: Green)
0x03	READY	Use to indicate that device is ready (RECOMMENDED color: Green)
0x04	FAULT	Use to indicate a general error (RECOMMENDED color: Red)
0x05	BUSY	Use to indicate that device is temporarily busy (RECOMMENDED color: Yellow)
0x06	ENTER_ID	Use to signal that device is waiting for ID (RECOMMENDED color: Yellow)
0x07	ENTER_PIN	Use to signal that device is waiting for PIN code (RECOMMENDED color: Yellow)
0x08	OK	Use to indicate OK e.g. the entered code is accepted (RECOMMENDED color: Green)
0x09	NOT_OK	Use to indicate NOT OK e.g. the entered code is NOT accepted (RECOMMENDED color: Red)
…		
0x20	ZONE1_ARMED	Use to indicate that alarm zone 1 is armed (RECOMMENDED color: Red)
0x21	ZONE2_ARMED	Use to indicate that alarm zone 2 is armed (RECOMMENDED color: Red)
0x22	ZONE3_ARMED	Use to indicate that alarm zone 3 is armed (RECOMMENDED color: Red)
0x23	ZONE4_ARMED	Use to indicate that alarm zone 4 is armed (RECOMMENDED color: Red)
0x24	ZONE5_ARMED	Use to indicate that alarm zone 5 is armed (RECOMMENDED color: Red)
0x25	ZONE6_ARMED	Use to indicate that alarm zone 6 is armed (RECOMMENDED color: Red)
…		
0x30	LCD_BACKLIGHT	Use to turn on LCD backlight, e.g. to shortly draw attention when alarm is activated from another entry control keypad.
…		
0x40	BUTTON_BACKLIGHT_ LETTERS	Use to indicate that buttons are ready for user input

Indicator ID	Description	Appearance and use
0x41	BUTTON_BACKLIGHT_ DIGITS	Use to indicate that buttons are ready for user input
0x42	BUTTON_BACKLIGHT_ COMMAND	Use to indicate that buttons are ready for user input
0x43	BUTTON1_INDICATION	Use to draw attention to button 1 (RECOMMENDED for button backlight or LED next to button)
0x44	BUTTON2_INDICATION	Use to draw attention to button 2 (RECOMMENDED for button backlight or LED next to button)
0x45	BUTTON3_INDICATION	Use to draw attention to button 3 (RECOMMENDED for button backlight or LED next to button)
0x46	BUTTON4_INDICATION	Use to draw attention to button 4 (RECOMMENDED for button backlight or LED next to button)
0x47	BUTTON5_INDICATION	Use to draw attention to button 5 (RECOMMENDED for button backlight or LED next to button)
0x48	BUTTON6_INDICATION	Use to draw attention to button 6 (RECOMMENDED for button backlight or LED next to button)
0x49	BUTTON7_INDICATION	Use to draw attention to button 7 (RECOMMENDED for button backlight or LED next to button)
0x4A	BUTTON8_INDICATION	Use to draw attention to button 8 (RECOMMENDED for button backlight or LED next to button)
0x4B	BUTTON9_INDICATION	Use to draw attention to button 9 (RECOMMENDED for button backlight or LED next to button)
0x4C	BUTTON10_INDICATION	Use to draw attention to button 10 (RECOMMENDED for button backlight or LED next to button)
0x4D	BUTTON11_INDICATION	Use to draw attention to button 11 (RECOMMENDED for button backlight or LED next to button)
0x4E	BUTTON12_INDICATION	Use to draw attention to button 12 (RECOMMENDED for button backlight or LED next to button)
…		
0xF0	Buzzer	Use to draw attention or provide user feedback

All other values are reserved and MUST NOT be used by a sending device. Reserved values MUST be ignored by a receiving device.

4.64.3.2 Property IDs

The identifiers used in Property ID fields MUST comply with the definitions found in Table 76.

Table 76, Indicator CC :: Property IDs

Property ID	General Description	Specific use
0x01	Multilevel	Light level, sound level 0x00 = OFF 0x01 – 0x63 = Lowest non-zero level - 100% 0xFF = Restore most recent (non-zero) level.
0x02	Binary	Turn indication On or Off 0x00 = OFF 0x01 – 0x63, 0xFF = ON
0x03	On_Off_Period	The Period in seconds of one ON/OFF period 0x00 – 0xFF = 0 - 25,5 sec
0x04	On_Off_Cycles	Number of On_Off_Period to run 0x00 – 0xFE = 0 – 254 times 0xFF = Run until stopped by On_Off
…		
0x10	Low_power -	ADVERTISE ONLY: This property MAY be used to advertise that the indicator can continue working in sleep mode. The indicator may use LCD, e-paper or similar technologies to achieve this. If this property is not advertised, the indicator MUST be switch Off doing power down. The indicator MUST keep the Off state at next wakeup.

All other values are reserved and MUST NOT be used by a sending device. Reserved values MUST be ignored by a receiving device.

If the On_Off_Period property is specified, the On_Off_Cycles property MUST also be specified. A receiving device MUST apply an On_Off_Cycles property value of 5 times if the On_Off_Period property is specified and the On_Off_Cycles property is not specified.

If the On_Off_Cycles property is specified, the On_Off_Period property MUST also be specified. A receiving device MUST apply an On_Off_Period property value of 1 second if the On_Off_Cycles property is specified and the On_Off_Period property is not specified.

The Multilevel and Binary properties can be used anytime to stop an alternating period configured the On_Off_Period and On_Off_Cycles properties.

Properties marked as “ADVERTISE ONLY” MUST be treated as unsupported by the Indicator Set and Indicator Get Commands.

4.64.4 Indicator Get Command

The Indicator Get Command may be used to request the state of an indicator.

The Indicator Report Command MUST be returned in response to the Indicator Get Command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_INDICATOR							
Command = INDICATOR_GET							
Indicator ID							

Indicator ID n (1 byte)

This field is used to specify the actual indicator resource.

If an unsupported Indicator ID is specified, a receiving device MUST advertise an indicator object with all fields set to zero in the Indicator Report Command that is returned in response to this command.

4.64.5 Indicator Report Command

The Indicator Report Command is used to advertise the state of an indicator resource.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_INDICATOR							
Command = INDICATOR_REPORT							
Indicator 0 Value (Indicator ID 0 = 0x00, Property ID 0 = 0x01)							
Reserved			Indicator Object Count				
Indicator ID							
PropertyID 1							
Value 1							
Indicator ID							
PropertyID 2							
Value 2							
…….							
Indicator ID							
PropertyID N							
Value N							

This field provides backwards compatibility with devices implementing the Indicator Command Class, version 1. Refer to section 4.64.1 for details.

A controlling version 2 device MUST ignore the Indicator 0 Value field if other values are advertised.

Indicator 0 Value (1 byte)

Refer to the Indicator Set Command.

Refer to the compatibility considerations found in 4.64.1.

Reserved

This field MUST be set to 0 by a sending device and MUST be ignored by a receiving device.

Indicator Object Count (1 byte)

This field is used to advertise the number of indicator objects carried in this command. An indicator object MUST comprise an Indicator ID, Property ID and Value field. Only one Indicator may be advertised at a time. Therefore, all indicator objects MUST carry the same Indicator ID while Property IDs MUST be different, e.g. Color and Blink pattern. This format is compatible w. the Indicator Set Command.

Indicator ID (1 byte)

This field is used to identify the actual indicator resource.

All indicator objects MUST carry the same Indicator ID.

Property ID n (1 byte)

Refer to the Indicator Set Command.

Value n (1 byte)

Refer to the Indicator Set Command.

4.64.6 Indicator Supported Get Command

The Indicator Supported Get Command may be used to request the supported properties of an indicator.

The Indicator Supported Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_INDICATOR							
Command = INDICATOR_SUPPORTED_GET							
Indicator ID							

Indicator ID (1 byte)

This field is used to specify the actual indicator resource.

A sending device SHOULD set this field to zero to discover supporting Indicator IDs. A device receiving an Indicator ID value of zero MUST advertise the first supported Indicator ID in the Indicator Supported Report Command that is returned in response to this command.

A device receiving a non-zero Indicator ID that is not supported MUST advertise the value zero in all fields of the Indicator Supported Report Command.

4.64.7 Indicator Supported Report Command

The Indicator Supported Report Command is used to advertise the supported properties for a given indicator.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_INDICATOR							
Command = INDICATOR_SUPPORTED_REPORT							
Indicator ID							
Next Indicator ID							
Reserved			Property Supported Bit Mask Length				
Property Supported Bit Mask 1							
...							
Property Supported Bit Mask N							

Indicator ID (1 byte)

This field is used to advertise the actual indicator resource.

Reserved

This field MUST be set to 0 by a sending device and MUST be ignored by a receiving device.

Next Indicator ID (1 byte)

This field is used to advertise if more Indicator IDs are supported.

If a sending device supports additional Indicator IDs, this field MUST advertise the next supported Indicator ID.
If the sending device supports only one Indicator ID, or if this command carries the last supported Indicator ID, this field MUST advertise the value 0.

A receiving device SHOULD continue requesting supported Indicator IDs via the Indicator Supported Get Command until this field advertises the value zero.

Property Supported Bit Mask Length (1 byte)

This field is used to advertise the length of the Property Supported Bit Mask field.

The value MUST be in the range 0..32.

Property Supported Bit Mask (variable length)

The Property Supported Bit Mask is used to advertise the properties supported by the actual Indicator ID.

- Bit 0 in Bit Mask 1 is not allocated to any property and MUST be set to zero.
- Bit 1 in Bit Mask 1 indicates if Property ID = 1 (Multilevel) is supported.
- Bit 2 in Bit Mask 1 indicates if Property ID = 2 (On/Off) is supported.
- …

If the Property ID is supported, the corresponding bit MUST be set to 1. If the Property ID is not supported the corresponding bit MUST be set to 0.

Note that this Command Class maps bit 1 to Property ID=1. Some Command Classes map bit 0 to the first property

4.65 IP Association Command Class, version 1

The IP Association Command Class is used to create and maintain Z-Wave application bindings between IPv6 hosts. An association group sends a predefined command to the configured destinations when triggered by an event. The parameters of the association group command may be dynamic, e.g. the temperature of a sensor reading or the light level for a dimmer.

IP Association identifies nodes via IPv6 addresses.
IP Association is natively Z-Wave Multi Channel End Point aware.
IP Association groups are specific to a particular End Point or Root Device.

4.65.1 Compatibility considerations

The IP Association Command Class extends the functionality of the Z-Wave Multi Channel Association Command Class. Z-Wave nodes may be represented as Z/IP nodes in an IP subnet. IP Associations may be mapped to classic Z-Wave Multi Channel associations.

The IP Association Remove Command extends the Multi Channel Association Remove Command by adding the ability to clear all groups in all End Points of a node. Only one IP association can be created or maintained with a single command.

In order to respond quickly and to conserve battery, it is RECOMMENDED that a Z/IP Gateway caches IP associations and corresponding Z-Wave associations and only looks up all associations when explicitly instructed to do so.

4.65.2 Terminology

The IP Association Command Class operates on IP primitives but works for all generations of Z-Wave technology. In the following, the term “Classic Z-Wave” refers to operations that are possible with Z-Wave-only releases.

The term “Z-Wave node” is used to identify a Z-Wave node as used in Classic Z-Wave while the term “Z/IP node” denotes an IP enabled Z-Wave node that implements the ICMP echo service and the Z-Wave UDP service. A Z/IP node may implement other IP services. Such IP services are out of scope of this document.

An IP association is created by sending an IP Association Set command from a configuration tool to the association source, instructing the association source to add an association destination to an association group maintained in the association source. The association destination has no knowledge that an association is created.

A Z/IP Gateway represents classic Z-Wave nodes as Z/IP nodes in an IP environment. The Z/IP Gateway therefore maps IP association commands from Z/IP clients to relevant Association and Multi Channel Association commands for Z-Wave nodes.

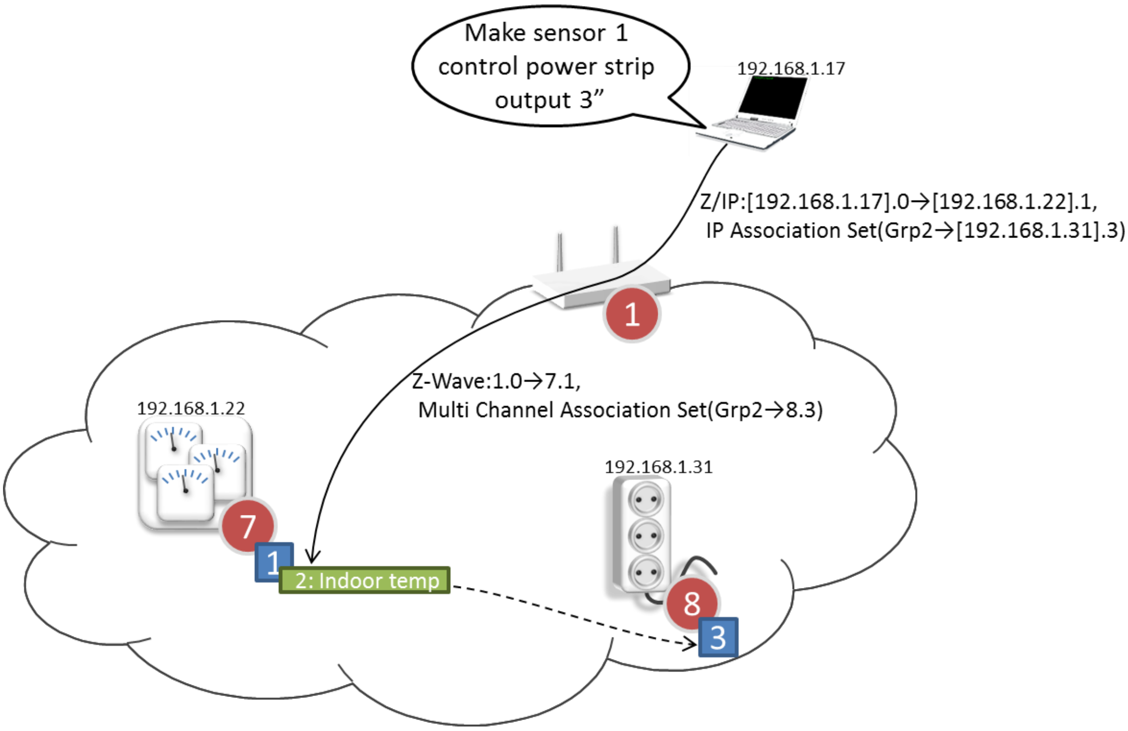


Figure 26, IP Association example

4.65.3 Z-Wave Multi Channel compatibility considerations

IP Association is natively Multi Channel aware. The same message format is used to create an association to a logical End Point as to the Root Device.

In the Z/IP framework, End Point 0 represents the physical entity.

IP Associations are conceptually created between two Z/IP resources identified by an IPv6 address and an End Point. In case one or both parties are Z-Wave nodes represented as Z/IP nodes, the Z/IP Gateway MUST create Z-Wave associations in the following way, based on the Resource Directory information pertaining to the actual nodes.

A Z/IP gateway MUST map IP associations to Z-Wave associations according to Table 77.

Table 77, IP association mapping to Z Wave association

Association source Multi Channel Capability	Association destination Multi Channel Capability	Association type	Creating the association
-	-	Root Device → Root Device	An Association Set Command MUST be sent un-encapsulated to the association source node
✓	✓	End Point m → End Point n m, n MAY be 0	A Multi Channel Association Set Command MUST be sent Multi Channel encapsulated to the association source End Point. A Multi Channel Association from the Lifeline association group of a sensor Root Device enables the transmission of sensor reports with different Source End Points to the Lifeline destination.
✓	-	End Point m → Root Device	An Association Set Command MUST be sent Multi Channel encapsulated to the association source End Point.
-	✓	Root Device → End Point n	An Association Set Command MUST be sent un-encapsulated to the association source node. This first association MUST target a NodeID owned by the Z/IP Gateway. A second association MUST be created from the abovementioned Z/IP Gateway to the Multi Channel End Point specified in the IP Association Set Command.

4.65.4 Advertising the IP Association Command Class

The Node Information Frame emitted by Classic Z-Wave nodes does not include IP Association. To allow IP clients to issue IP Association commands, the Z/IP Gateway MUST rewrite the list of supported command classes from the node, replacing the Association and Multi Channel Association Command Classes with the IP Association Command Class in the following situations:

- When a Node Info Cached Report is sent to an IP client.
- When a Node Add Status is being sent to an IP client.

In both cases, rewriting MUST NOT be performed if the report is being sent to a native Z-Wave node.

Furthermore, mDNS responses to queries for nodes supporting the IP Association Command Class MUST also advertise nodes supporting the Association and Multi Channel Association Command Classes (but the command classes must be replaced with COMMAND_CLASS_IP_ASSOCIATION before advertised via mDNS).

4.65.5 IP Association Set Command

This command is used to add one resource to a given association group.

The receiving IPv6 host SHOULD add the specified destination resource to the specified association group.
This command MAY be ignored if the association group is already full.

Unless the association destination is a gateway, a controlling node SHOULD NOT create an association if the association destination node does not support the commands that the actual association group will be sending. The AGI Command Class SHOULD be used to probe the commands that a given association group will be sending.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_IP_ASSOCIATION							
Command = IP_ASSOCIATION_SET							
Grouping identifier							
IPv6 Address 1							
..							
IPv6 Address 16							
End Point							

Grouping identifier (8 bits)

This field is used to specify the actual association group.
Grouping Identifiers MUST be assigned in a consecutive range starting from 1.

IPv6 Address (16 bytes)

This field specifies the full IPv6 address for the association destination.
The IPv6 address MUST NOT be compressed.
The IPv6 address SHOULD be in the ULA IPv6 prefix or in a globally routable IPv6 prefix.
The IPv6 address MAY be an IPv4-mapped IPv6 address.
The field MUST NOT carry a link-local IPv6 address.

End Point (8 bits)

This field is used in combination with the IPv6 Address to identify the actual resource.

The field represents the Z/IP Bit Address flag (bit 7) as well as the Z/IP 7-bit End Point identifier (bits 6..0).

The receiving node MUST treat this field as one scalar value when adding or removing associations; i.e. any 8-bit End Point value causes one association to be added.

4.65.6 IP Association Get Command

This command is used to request active associations for a given association group.The IP Association Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing.
The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_IP_ASSOCIATION							
Command = IP_ASSOCIATION_GET							
Grouping identifier							
Index							

Grouping identifier (8 bits)

This field is used to specify the actual association group.
Grouping Identifiers MUST be assigned in a consecutive range starting from 1.

If an unsupported Grouping Identifier is specified, the IP Association Report Command returned in response to this command MUST carry IPv6 address and End Point fields which are set to all-zeros.

Index (8 bits)

This field is used to specify an entry in the association table for the group identified by the Grouping Identifier.

A requesting host SHOULD start specifying the index value 1.
The actual number of nodes in the association group may be determined from the Actual Nodes field of the IP Association Report.

4.65.7 IP Association Report Command

This command is used to advertise one association group entry.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_IP_ASSOCIATION							
Command = IP_ASSOCIATION_REPORT							
Grouping identifier							
Index							
Actual Nodes							
IPv6 Address 1							
..							
IPv6 Address 16							
End Point							

Grouping identifier (8 bits)

This field is used to identify the actual group of nodes.

Index (8 bits)

This field is used to advertise the actual entry in the association group identified by the Grouping Identifier.

Actual Nodes (8 bits)

This value indicates the number of nodes in the association group advertised by the Grouping identifier field.

IPv6 Address (16 bytes)

This field carries a full IPv6 address with no compression. The address SHOULD be in the ULA IPv6 prefix or in a globally routable IPv6 prefix. The address MAY be an IPv4-mapped IPv6 address. The field MUST NOT carry a link-local IPv6 address.

If the Actual Nodes value is zero, the IPv6 Address field MUST be all zeros.

End Point (8 bits)

This field is used in combination with the IPv6 Address to identify the actual resource

If the Actual Nodes value is zero, the End Point field MUST be all zeros.

The field represents the Z/IP Bit Address flag (bit 7) as well as the Z/IP 7-bit End Point identifier (bits 6..0).

4.65.8 IP Association Remove Command

This command is used to remove one resource from a given association group.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_IP_ASSOCIATION							
Command = IP_ASSOCIATION_REMOVE							
Grouping identifier							
IPv6 Address 1							
..							
IPv6 Address 16							
End Point							

Grouping identifier (8 bits)

This field is used to specify the actual association group. Grouping Identifiers MUST be assigned in a consecutive range starting from 1.

A receiving node MUST ignore an unsupported Grouping Identifier.

This field MUST be interpreted in combination with the IPv6 Address and End Point fields.

IPv6 Address (16 bytes)
End Point (1 byte)

These fields specify the resources that are to be removed.
The Grouping identifier and these fields MUST be interpreted as follows.

Table 78, IP Association Remove, V1 :: Parameter interpretation

Command parameters			**Removal operation**
Grouping identifier	**IPv6 Address**	**End Point ID**	
> 0	<> 0	= 0 (Root Device)	Remove destination from association group in the association source Root Device
> 0	<> 0	> 0 (End Point)	Remove destination from association group in the specified association source End Point
> 0	= 0	= 0 (Root Device)	Remove all destinations from association group in the association source Root Device
> 0	= 0	> 0 (End Point)	Remove all destinations from association group in the

			specified association source End Point
= 0	= 0	= 0 (Root Device)	Remove all destinations from all association groups in the association source Root Device and in all association source End Points
= 0	= 0	> 0 (End Point)	Remove all destinations from all groups in the specified association source End Point
= 0	<> 0	>= 0	*Reserved*

*) The End Point field represents the Z/IP Bit Address flag (bit 7) as well as the Z/IP 7-bit End Point identifier (bits 6..0).
The receiving node MUST treat the End Point field as one scalar value when removing associations; i.e. any 8-bit End Point value causes one association to be removed.

4.66 IP Configuration Command Class, version 1 [OBSOLETED]

THIS COMMAND CLASS HAS BEEN OBSOLETED

New implementations MUST NOT use the IP configuration Command Class. Refer to the Z/IP and Network management Command Classes.

The IP Configuration Command Class is used to configure network identifiers for IPV4 devices. The intended use of the command class is illustrated in the figure below.

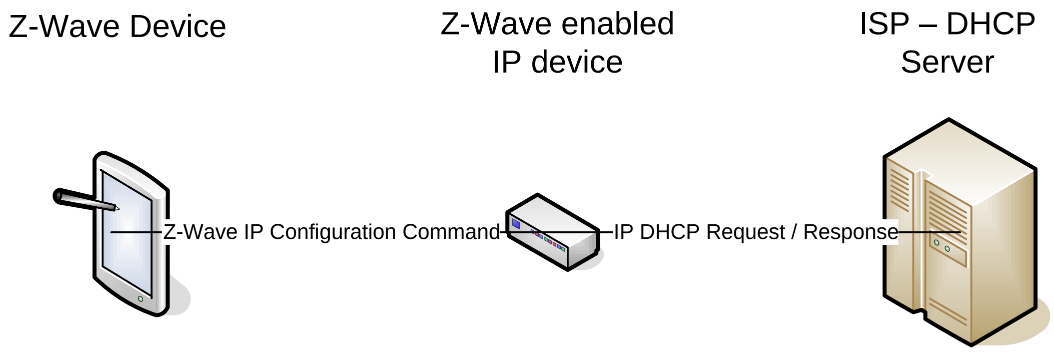


Figure 27, Configuration of network identifiers for IPV4 devices

In the figure the Z-Wave Remote to the left, sends an IP Configuration Command to the Z-Wave enabled IP device, telling it to acquire its configuration using DHCP. The Z-Wave enabled IP device will now perform a standard DHCP IP request to the DHCP server over an IP based network.

Another example might be where the Z-Wave Remote statically configures the Z-Wave enabled IP device with fixed IP, subnet, DNS etc. by sending an IP Configuration Command.

Note that this class is only intended for IPV4 and not IPV6 support.

4.66.1 IP Configuration Set Command

The IP Configuration Set Command used to configure IPV4 settings in a device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_IP_CONFIGURATION							
Command = IP_CONFIGURATION_SET							
Reserved						Auto IP	Auto DNS
IP Address 1							
IP Address 2							
IP Address 3							
IP Address 4							
Subnet Mask 1							
Subnet Mask 2							
Subnet Mask 3							
Subnet Mask 4							
Gateway 1							
Gateway 2							
Gateway 3							
Gateway 4							
DNS1 1							
DNS1 2							
DNS1 3							
DNS1 4							
DNS2 1							
DNS2 2							
DNS2 3							
DNS2 4							

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Auto IP (1 bit)

If Auto IP bit is set, the following fields are ignored: IP Address, Subnet Mask, and Gateway. And are allocated by DHCP or BOOTP instead.

Auto DNS (1 bit)

The Auto DNS if set indicates to ignore DNS1 and DNS2 and allocate DNS by DHCP instead. Note that some devices might not support Auto DNS without Auto IP set.

IP Address (32 bits)

The IP Address indicates the static IP address of the device itself. The first byte is the most significant byte.

Subnet mask (32 bits)

The Subnet Mask determines the portion of the IP address that represents the subnet. The first byte is the most significant byte.

Gateway (32 bits)

The Gateway indicates the default gateway that serves as an access point to another network. The first byte is the most significant byte.

DNS1 (32 bits)

The DNS1 allows the use of domain name system (DNS) server names instead of using numerical IP addresses for management packet routing. In case the device will not need DNS, and SHOULD NOT query it from DHCP then leave field as all zeroes. The first byte is the most significant byte.

DNS2 (32 bits)

The DNS2 provides a secondary DNS server name. In case only one DNS server is available or the device will not need DNS then leave field as all zeroes. The first byte is the most significant byte.

4.66.2 IP Configuration Get Command

The IP Configuration Get Command is used to request the IPV4 settings in a device.

The IP Configuration Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_IP_CONFIGURATION							
Command = IP_CONFIGURATION_GET							

4.66.3 IP Configuration Report Command

The IP Configuration Report Command used to return IPV4 settings in a device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_IP_CONFIGURATION							
Command = IP_CONFIGURATION_REPORT							
Reserved						Auto IP	Auto DNS
IP Address1							
IP Address2							
IP Address3							
IP Address4							
Subnet Mask1							
Subnet Mask2							
Subnet Mask3							
Subnet Mask4							
Gateway1							
Gateway2							
Gateway3							
Gateway4							
DNS11							
DNS12							
DNS13							
DNS14							
DNS21							
DNS22							
DNS23							
DNS24							
LeaseTime1							
LeaseTime2							
LeaseTime3							
LeaseTime4							

Refer to explanation of parameters in IP Configuration Set Command description.

Lease Time (32 bits)

The lease time specifies the time the IP address has been granted, if Auto IP is being used (in seconds). If the device does not know its lease period it MUST return 0 for the lease time fields.

4.66.4 IP Configuration DHCP Release Command

The IP Configuration DHCP Release Command used to release the DHCP lease.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_IP_CONFIGURATION							
Command = IP_CONFIGURATION_RELEASE							

4.66.5 IP Configuration DHCP Renew Command

The IP Configuration DHCP Renew Command used to force the renewal of the DHCP lease.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_IP_CONFIGURATION							
Command = IP_CONFIGURATION_RENEW							

4.67 Irrigation Command Class, version 1

The Irrigation Command Class provides commands to support irrigation control functionality. Note, the reader should familiarize themselves with the section on Terminology before proceeding.

4.67.1 Terminology

4.67.1.1 Zone Valve

In an irrigation system, a zone valve is turned on to allow a certain group of sprinklers to water an area (or zone) for a specified duration. Through this document, the term “zone valve” and “valve” are used synonymously, whereas a “master valve” is always identified as such.

A single channel of an irrigation control device is controlled as a Multi Channel End Point representing the Valve Device Type.

A Valve Device Type must support the following features:

- Valve CC

A Valve Device Type should support the following features:

- Schedule CC
  - Supporting at least one schedule
  - Supporting Start time, alternating days and Odd/Even dates
  - Supporting Binary Switch CC
  - Supporting Schedule enable/disable

4.67.1.2 Master Valve

In an irrigation system, a master valve allows water to flow to a set of zone valves, and must be turned on before water flows to the zone valves. During a scheduled run, a master valve remains on for the entire scheduled run, so that all zone valves will be able to provide water to their associated sprinklers. So when a master valve is turned on, it must remain on for the duration of the scheduled run, and then be shut off after the scheduled run is completed.

NOTE: In some cases a master valve switch may drive a pump relay rather than an actual irrigation valve. In this case, there may be a configurable delay between the master valve (pump) being turned on and the rest of the zone valves being turned on during the scheduled run.

NOTE: When a given zone is turned on, a Master Valve (if present) should automatically turn on. At the end of a scheduled run, a System Off command should turn off everything.

4.67.1.3 Scheduled Run

In an irrigation system, a scheduled run is when a set of zone valves are run sequentially, one after another. So, each zone valve will be turned on for a specified duration and once complete, that valve will be turned off, then the next zone valve will be turned on. In a scheduled run, each zone valve has its own specified run duration, which is unique to that zone valve and that specific scheduled run.

A scheduled run can be a one-time event or a periodic event depending on how it is configured.

Also, in some irrigation systems, a master valve will be turned on before turning on individual zone valves. So when a master valve is turned on, it must remain on for the duration of the scheduled run and then be shut off after the scheduled run is complete.

4.67.1.4 Valve Table

A valve table is a list of zone valves and their corresponding run durations. A valve table can be created, configured and stored on an irrigation control device. The valve table can be executed (or run) through the Irrigation Valve Table Run Command, which can be sent directly to the device or scheduled via the Schedule Set Command.

4.67.1.5 Scheduling

Schedule CC must support alternating days and Odd/Even date processing.

4.67.1.5.1 Gateway Driven Scheduling

An irrigation control device can be driven directly from a Z-Wave gateway, where the gateway turns on the master valve and sequentially turns on each valve for a specified duration, turning it off at the end of its run. The day, time, valves and durations are driven by the gateway as configured by the user.

4.67.1.5.1.1 Individual Valves

The most basic option is for the gateway to directly use the Basic Set Command to turn on specific valves. When the duration has expired, the gateway will need to turn the actual valve off. Depending on the irrigation system, the gateway may need to turn on the master valve using the Basic Set Command.

4.67.1.5.1.2 Valve Tables

The gateway can also use the concept of a valve table to turn on a set of valves sequentially. A valve table allows a single command, the Irrigation Valve Table Run Command, to trigger a scheduled run. Initially the gateway must create the valve tables, using the command Irrigation Valve Table Set Command. Then the gateway can trigger a scheduled run by sending the Irrigation Valve Table Run Command at any specified time. When received, the device will sequentially run the selected valves, in the specified valve table, for their associated runtimes. Depending on the irrigation system, the gateway may need to turn on the master valve using the Irrigation Valve Run Command.

4.67.1.5.2 Configured Scheduling

An irrigation control device can be configured with a number of weekly schedules that will be run at a specified time on specified days of the week. This allows the irrigation control device to operate independently during normal irrigation operations. The gateway can also update the schedules periodically or under unique circumstances, such as wet weather. If the gateway loses connection with the irrigation device, any enabled schedules must continue to run. The number of schedules available can be determined through Schedule Command Class.

Schedules are configured by the gateway. Initially, the gateway will create a valve table, using the command Irrigation Valve Table Set Command. Then the gateway will set up a scheduled run by using the Schedule Set Command. Each schedule triggers an Irrigation Valve Table Run Command. Note that depending on the irrigation system, the gateway may need to turn on master valves using the Basic Set Command.

Alternatively, a schedule can be created by listing individual Basic Set commands. Zones controllable through the same Root Device that is the core of the system can be controlled Basic Set commands to the individual End Points. Independent zones can be controlled by direct Basic_Set commands. This methodology also supports an independent implementation where a single zone valve supports Schedule CC. A schedule entry with a single Basic Set(On) command and a duration allows a single valve to participate in a complex irrigation system.

Note that some irrigation devices may not be equipped with a real time clock. If the device loses connection with the gateway, it should attempt to run the schedules as accurately as possible. In some cases, the scheduling accuracy may be severely degraded due to loss of power, clock drift, etc. The product documentation should address this scenario.

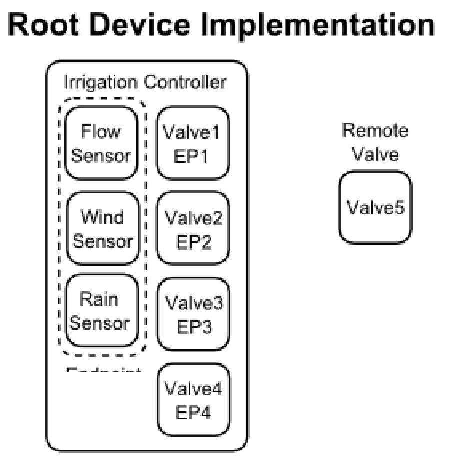
4.67.1.5.3 Schedule Examples

4.67.1.5.3.1 Root Device Irrigation control device

This example shows an Irrigation control device with 4 channels (EP1-EP4) and an external valve controlled through Association Group 5. When the Valve Table is executed, the controller will need to send Basic Set(On) to AG5 to start the zone, then Basic Set(Off) at the end of the run to turn off the zone.

Valve Table1
  EP1, Valve (On), Duration - Master Valve
  EP2, Valve (On), Duration - Zone1
  EP3, Valve (On), Duration - Zone2
  EP4, Valve (On), Duration - Zone3
  Association Group5, Valve (On), Duration - Zone4

Schedule1
  TableRun(Table1)

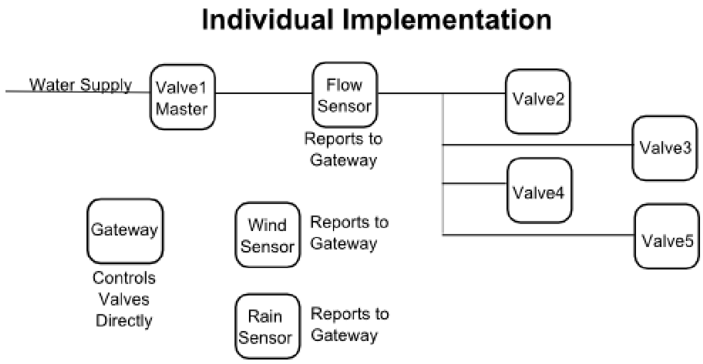


Irrigation Controller can control external Remote Valves

4.67.1.5.3.2 *Individual Valve Devices*

This example shows 5 independent valves being controlled by a central Gateway. 1 or more of these valves may be controlled by an Irrigation control device. Any sensors in the system reports to the Gateway for processing.

Valve1 (Master Valve): Schedule1 Valve (On) - Master Valve Schedule5 Valve (Off) - Master Valve	Valve4 (Zone 3): Schedule3 Valve (On) Schedule4 Valve (Off)
Valve2 (Zone 1): Schedule1 Valve (On) Schedule2 Valve (Off)	Valve4 (Zone 4): Schedule4 Valve (On) Schedule5 Valve (Off)
Valve3 (Zone 2): Schedule2 Valve (On) Schedule3 Valve (Off)	



4.67.1.5.4 Backup Schedule

A backup schedule is a schedule which may be used when all other schedules have ended or are inactive.

If any other schedules are active, the backup schedule will be suspended.

On an irrigation device the backup schedule will be identified using the schedule ID of 0xFF. The backup schedule can be accessed using the Schedule Report Command, can be disabled or removed using the Schedule State Set Command, and can be added or changed using the Schedule Set Command.

4.67.1.6 Irrigation Command Class Overview

Irrigation System Info Get Command
Irrigation System Info Report Command
Irrigation SystemValve Status Get Command
Irrigation SystemValve Status Report Command
Irrigation System Config Set Command
Irrigation System Config Get Command
Irrigation System Config Report Command
Irrigation Valve Info Get Command
Irrigation Valve Info Report Command
Irrigation Valve Config Set Command
Irrigation Valve Config Get Command
Irrigation Valve Config Report Command
Irrigation Valve Run Command
Irrigation Valve Table Set Command
Irrigation Valve Table Get Command
Irrigation Valve Table Report Command
Irrigation Valve Table Run Command
Irrigation System Shutoff Command

4.67.2 Compatibility Considerations

No issues.

4.67.3 Interoperability Considerations

A system integrator should be aware that if an irrigation control device is controlled via direct End Point control and via the Root Device Irrigation control functionality at the same time, this may cause unpredictable behavior. This applies whether commands are issued instantly or scheduled. It is RECOMMENDED that an irrigation control system is managed either via direct End Point control or via the Irrigation Command Class.

4.67.4 Irrigation System Info Get Command

The Irrigation System Info Get Command is used to request general information about the irrigation system.

The Irrigation System Info Report Command MUST be returned in response to this command.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_IRRIGATION							
Command = IRRIGATION_SYSTEM_INFO_GET							

4.67.5 Irrigation System Info Report Command

The Irrigation System Info Report Command provides general information about the irrigation system.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_IRRIGATION</td></tr>
<tr><td colspan="8">Command = IRRIGATION_SYSTEM_INFO_REPORT</td></tr>
<tr><td colspan="3">Reserved</td><td colspan="2">Reserved</td><td colspan="2">Reserved</td><td>Master Valve</td></tr>
<tr><td colspan="8">Total Number of Valves</td></tr>
<tr><td colspan="8">Total Number of Valve Tables</td></tr>
<tr><td colspan="4">Reserved</td><td colspan="4">Valve Table Max Size</td></tr>
</table>

Reserved fields MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Master Valve (1 bit)

Indicates that a master valve is supported by the device.

Total Number of Valves (8 bits)

The total number of valves supported by the device. The valves MUST be identified sequentially from 1 to the total number.

Total Number of Valve Tables (8 bits)

The total number of valve tables that can be created/stored on the device. The tables MUST be identified sequentially from 1 to the total number.

Valve Table Max Size (4 bits)

The maximum size a valve table can be. The value MUST be in the range 1..15.

4.67.6 Irrigation System Status Get Command

The Irrigation System Status Get Command is used to request status information about the irrigation system.

The Irrigation System Status Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_IRRIGATION</td></tr>
<tr><td colspan="8">Command = IRRIGATION_SYSTEM_STATUS_GET</td></tr>
</table>

4.67.7 Irrigation System Status Report Command

The Irrigation System Status Report Command provides status information about the irrigation system.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_IRRIGATION</td></tr>
<tr><td colspan="8">Command = IRRIGATION_SYSTEM_STATUS_REPORT</td></tr>
<tr><td colspan="8">System Voltage</td></tr>
<tr><td colspan="8">Sensor Status</td></tr>
<tr><td colspan="3">Flow Precision</td><td colspan="2">Flow Scale=l/h</td><td colspan="3">Flow Size</td></tr>
<tr><td colspan="8">Flow Value 1</td></tr>
<tr><td colspan="8">..</td></tr>
<tr><td colspan="8">Flow Value N</td></tr>
<tr><td colspan="3">Pressure Precision</td><td colspan="2">Pressure Scale=l/h</td><td colspan="3">Pressure Size</td></tr>
<tr><td colspan="8">Pressure Value 1</td></tr>
<tr><td colspan="8">..</td></tr>
<tr><td colspan="8">Pressure Value N</td></tr>
<tr><td colspan="8">Shutoff Duration</td></tr>
<tr><td colspan="8">System Error Status</td></tr>
<tr><td colspan="7">Reserved</td><td>Master Valve</td></tr>
<tr><td colspan="8">Valve ID</td></tr>
</table>

System Voltage (8 bits)

The voltage level for the device. The value MUST be in Volt with no indication of the voltage type (AC or DC).

Sensor Status (8 bits)

This bit mask field indicates which sensors are detected by the device. The field MUST be encoded according to Table 79, Sensor Status.

Flow Reading

If the Sensor Status indicates that a flow sensor is detected, this reading is valid. Otherwise, this field MUST be set to 0. This field MUST be expressed in l/h.

The format of the Flow fields MUST comply with the format used by the Multilevel Sensor Command Class.

Pressure Reading

If the Sensor Status indicates that a pressure sensor is detected, this reading is valid. Otherwise, this field MUST be set to 0. This field MUST be expressed in kPa.

The format of the Pressure fields MUST comply with the format used by the Multilevel Sensor Command Class.

Shutoff Duration (8 bits)

See “Duration” field in the Irrigation System Shutoff Command for more details.

System Error Status (8 bits)

This bit mask provides system error status fields. The field MUST be encoded according to these rules.

- Bit 0 indicates that the device has not been programmed.
- Bit 1 indicates that the device has experienced an emergency shutdown.
- Bit 2 indicates that the system pressure high threshold has been triggered.
- Bit 3 indicates that the system pressure low threshold has been triggered.
- Bit 4 indicates that a valve (master or zone) is reporting errors. The valves can be checked using the Irrigation Valve Info Get Command.
- Bits 5-7 are reserved.

In some cases, errors are no longer active. Any inactive errors (except valve errors) can be cleared by sending an Irrigation System Config Set Command. See the Irrigation Valve Info Report Command for more details on valve specific errors.

Master Valve (1 bit)

Indicates if the master valve is currently on.

Valve ID (8 bits)

Indicates if a zone valve is currently on, and identifies the valve. Valves are identified sequentially from 1 to the total number available on the device. If no valve is running, this field must report zero.

4.67.7.1 Sensor Status Fields

Table 79, Sensor Status

Bit	Sensor Status
0	Flow Sensor Detected
1	Pressure Sensor Detected
2	Rain Sensor Detected
3	Moisture Sensor Detected
4 – 7	Reserved

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Flow Sensor Detected

If the flow sensor is detected then the Flow Reading field is valid.

Pressure Sensor Detected

If the pressure sensor is detected then the Pressure Reading field is valid.

Rain Sensor Detected

If the rains sensor is detected then the rain sensor is reporting rain detected.

Moisture Sensor Detected

If the moisture sensor is detected then the moistures sensor is signaling moisture detected.

4.67.8 Irrigation System Config Set Command

The Irrigation System Config Set Command allows the irrigation system to be configured accordingly.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_IRRIGATION</td></tr>
<tr><td colspan="8">Command = IRRIGATION_SYSTEM_CONFIG_SET</td></tr>
<tr><td colspan="8">Master Valve Delay</td></tr>
<tr><td colspan="3">High Pressure Threshold Precision</td><td colspan="2">High Pressure Threshold Scale=kPa</td><td colspan="3">High Pressure Threshold Size</td></tr>
<tr><td colspan="8">High Pressure Threshold Value 1</td></tr>
<tr><td colspan="8">…</td></tr>
<tr><td colspan="8">High Pressure Threshold Value N</td></tr>
<tr><td colspan="3">Low Pressure Threshold Precision</td><td colspan="2">Low Pressure Threshold Scale=kPa</td><td colspan="3">Low Pressure Threshold Size</td></tr>
<tr><td colspan="8">Low Pressure Threshold Value 1</td></tr>
<tr><td colspan="8">…</td></tr>
<tr><td colspan="8">Low Pressure Threshold Value N</td></tr>
<tr><td colspan="8">Sensor Polarity</td></tr>
</table>

Master Valve Delay (8 bits)

Delay (seconds) between turning on the master valve and turning on any “zone” valve. This field is only valid if a master valve is supported by the irrigation device. This value MUST be expressed in seconds, where a value of 0 indicates that the valve should be turned on immediately after the master valve is turned on.

Pressure High Threshold

The device’s pressure high threshold. If the pressure crosses above this threshold it MUST trigger the pressure high threshold notification. This field is only valid if a pressure sensor is connected to the device. This field MUST be expressed in kPa.

The format of the Pressure fields MUST comply with the format used by the Multilevel Sensor Command Class.

Pressure Low Threshold

The device’s pressure low threshold. If the pressure crosses below this threshold it MUST trigger the pressure low threshold notification. This field is only valid if a pressure sensor is connected to the device. This field MUST be expressed in kPa.

The format of the Pressure fields MUST comply with the format used by the Multilevel Sensor Command Class.

Sensor Polarity (8 bits)

This bit mask field allows the command to set the polarity of the following sensors. If the valid bit is not set to 1, the rest of the fields must be ignored. The field MUST be encoded according to Table 80**,** Sensor Polarity.

Table 80, **Sensor Polarity**

Bit	Sensor Polarity
0	Rain Sensor Polarity (0 LOW, 1 HIGH)
1	Moisture Sensor Polarity (0 LOW, 1 HIGH)
2 – 6	Reserved
7	Valid

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

4.67.9 Irrigation System Config Get Command

The Irrigation System Config Get Command is used to request the current configuration of the irrigation system.

The Irrigation System Config Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_IRRIGATION							
Command = IRRIGATION_SYSTEM_CONFIG_GET							

4.67.10 Irrigation System Config Report Command

The Irrigation System Config Report Command provides details about the current configuration of the irrigation system.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_IRRIGATION</td></tr>
<tr><td colspan="8">Command = IRRIGATION_SYSTEM_CONFIG_REPORT</td></tr>
<tr><td colspan="8">Master Valve Delay</td></tr>
<tr><td colspan="3">High Pressure Threshold Precision</td><td colspan="2">High Pressure Threshold Scale=kPa</td><td colspan="3">High Pressure Threshold Size</td></tr>
<tr><td colspan="8">High Pressure Threshold Value 1</td></tr>
<tr><td colspan="8">…</td></tr>
<tr><td colspan="8">High Pressure Threshold Value N</td></tr>
<tr><td colspan="3">Low Pressure Threshold Precision</td><td colspan="2">Low Pressure Threshold Scale=kPa</td><td colspan="3">Low Pressure Threshold Size</td></tr>
<tr><td colspan="8">Low Pressure Threshold Value 1</td></tr>
<tr><td colspan="8">…</td></tr>
<tr><td colspan="8">Low Pressure Threshold Value N</td></tr>
<tr><td colspan="8">Sensor Polarity</td></tr>
</table>

Master Valve Delay (8 bits)

Delay (seconds) between turning on the master valve and turning on any “zone” valve. This field is only valid if a master valve is supported by the irrigation device, otherwise it should report zero. The value MUST be expressed in seconds, where a value of 0 indicates that the valve should be turned on immediately after the master valve is turned on.

Pressure High Threshold

The device’s pressure high threshold. If the pressure crosses above this threshold it will trigger the pressure high threshold notification. This field is only valid if a pressure sensor is connected to the device, otherwise it should report zero. This field MUST be expressed in kPa.

The format of the Pressure fields MUST comply with the format used by the Multilevel Sensor Command Class.

Pressure Low Threshold

The device’s pressure low threshold. If the pressure crosses below this threshold it will trigger the pressure low threshold notification. This field is only valid if a pressure sensor is connected to the device, otherwise it should report zero. This field MUST be expressed in kPa.

The format of the Pressure fields MUST comply with the format used by the Multilevel Sensor Command Class.

Sensor Polarity (8 bits)

This bit mask field allows the command to report the polarity of the following sensors. If the valid bit is not set to 1, the rest of the fields must be ignored. The field MUST be encoded according to Table 80**,** Sensor Polarity.

4.67.11 Irrigation Valve Info Get Command

The Irrigation Valve Info Get Command is used to request general information about the specified valve.

The Irrigation Valve Info Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_IRRIGATION							
Command = IRRIGATION_VALVE_INFO_GET							
Reserved							Master Valve
Valve ID							

Reserved fields MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Master Valve (1 bit)

Indicates if the valve ID refers to a master or standard “zone” valve id.

Valve ID (8 bits)

The valve ID of interest. Valves (including master) MUST be identified sequentially from 1 to the total number available on the device.

4.67.12 Irrigation Valve Info Report Command

The Irrigation Valve Info Report Command provides general information about the specified valve.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_IRRIGATION							
Command = IRRIGATION_VALVE_INFO_REPORT							
Reserved						Conn-ected	Master Valve
Valve ID							
Nominal Current							
Valve Error Status							

Reserved fields MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Master Valve (1 bit)

Indicates if the valve ID refers to a master or standard "zone" valve id.

Connected (1 bit)

Indicates if the valve is connected or not.

Valve ID (8 bits)

The valve ID of interest. Valves (including master) are identified sequentially from 1 to the total number available on the device.

Nominal Current (8 bits)

The valve's nominal current (1 unit equals 10 mA) when turned on.

Valve Error Status (8 bits)

This bit mask provides valve error status fields. The field MUST be encoded according to these rules.

- Bit 0 indicates short circuit has been detected.
- Bit 1 indicates current high threshold has been detected.
- Bit 2 indicates current low threshold has been detected.
- Bit 3 indicates maximum flow has been detected (zone valves only).
- Bit 4 indicates flow high threshold has been detected (zone valves only).
- Bit 5 indicates flow low threshold has been detected (zone valves only).
- Bits 6-7 are reserved.

Any inactive errors can be cleared by sending an Irrigation Valve Config Set Command. Also, if the valve is turned on again, any inactive errors should be cleared as the valve is turned on.

4.67.13 Irrigation Valve Config Set Command

The Irrigation Valve Config Set Command allows an irrigation valve to be configured accordingly.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_IRRIGATION</td></tr>
<tr><td colspan="8">Command = IRRIGATION_VALVE_CONFIG_SET</td></tr>
<tr><td colspan="7">Reserved</td><td>Master Valve</td></tr>
<tr><td colspan="8">Valve ID</td></tr>
<tr><td colspan="8">Nominal Current High Threshold</td></tr>
<tr><td colspan="8">Nominal Current Low Threshold</td></tr>
<tr><td colspan="3">Maximum Flow Precision</td><td colspan="2">Maximum Flow Scale=l/h</td><td colspan="3">Maximum Flow Size</td></tr>
<tr><td colspan="8">Maximum Flow Value 1</td></tr>
<tr><td colspan="8">…</td></tr>
<tr><td colspan="8">Maximum Flow Value N</td></tr>
<tr><td colspan="3">Flow High Threshold Precision</td><td colspan="2">Flow High Threshold Scale=l/h</td><td colspan="3">Flow High Threshold Size</td></tr>
<tr><td colspan="8">Flow High Threshold Value 1</td></tr>
<tr><td colspan="8">…</td></tr>
<tr><td colspan="8">Flow High Threshold Value N</td></tr>
<tr><td colspan="3">Flow Low Threshold Precision</td><td colspan="2">Flow Low Threshold Scale=l/h</td><td colspan="3">Flow Low Threshold Size</td></tr>
<tr><td colspan="8">Flow Low Threshold Value 1</td></tr>
<tr><td colspan="8">…</td></tr>
<tr><td colspan="8">Flow Low Threshold Value N</td></tr>
<tr><td colspan="8">Sensor Usage</td></tr>
</table>

Reserved fields MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Master Valve (1 bit)

Indicates if the valve ID refers to a master or standard “zone” valve id.

Valve ID (8 bits)

The valve ID of interest. Valves (including master) MUST be identified sequentially from 1 to the total number available on the device.

Nominal Current High Threshold (8 bits)

The valve's nominal current (1 unit equals 10 mA) high threshold. If the current crosses above this threshold it MUST trigger the current high threshold notification.

Nominal Current Low Threshold (8 bits)

The valve's nominal current (1 unit equals 10 mA) low threshold. If the current crosses below this threshold it MUST trigger the current low threshold notification.

Maximum Flow

The maximum flow allowed for the valve. This field is only valid if a flow sensor is connected to the device. This field does not apply to a master valve. This field MUST be expressed l/h.

Flow High Threshold

The valve's flow high threshold. If the flow crosses above this threshold it will trigger the flow high threshold notification. This field is only valid if a flow sensor is connected. This field does not apply to a master valve. This field MUST be expressed in l/h.

The format of the Flow fields MUST comply with the format used by the Multilevel Sensor Command Class.

Flow Low Threshold

The valve's flow low threshold. If the flow crosses below this threshold it will trigger the flow low threshold notification. This field is only valid if a flow sensor is connected to the device. This field does not apply to a master valve. This field MUST be expressed in l/h.

The format of the Flow fields MUST comply with the format used by the Multilevel Sensor Command Class.

Sensor Usage (8 bits)

This field allows the valve to be configured to use certain sensors when running. For example, if the valve is configured to use the rain sensor, the valve will shut off (or not run) if the rain sensor is turned on (or active). This field does not apply to a master valve. The field MUST be encoded according to Table 81**,** Sensor Usage.

Table 81, **Sensor Usage**

Bit	Sensor Usage
0	Use Rain Sensor
1	Use Moisture Sensor
2 – 7	Reserved

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

4.67.14 Irrigation Valve Config Get Command

The Irrigation Valve Config Get Command is used to request the current configuration of an irrigation valve.

The Irrigation Valve Config Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_IRRIGATION							
Command = IRRIGATION_VALVE_CONFIG_GET							
Reserved							Master Valve
Valve ID							

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Master Valve (1 bit)

Indicates if the valve ID refers to a master or standard “zone” valve id.

Valve ID (8 bits)

The valve ID of interest. Valves (including master) MUST be identified sequentially from 1 to the total number available on the device.

4.67.15 Irrigation Valve Config Report Command

The Irrigation Valve Config Report Command provides details about the valve’s current configuration.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_IRRIGATION</td></tr>
<tr><td colspan="8">Command = IRRIGATION_VALVE_CONFIG_REPORT</td></tr>
<tr><td colspan="7">Reserved</td><td>Master Valve</td></tr>
<tr><td colspan="8">Valve ID</td></tr>
<tr><td colspan="8">Nominal Current High Threshold</td></tr>
<tr><td colspan="8">Nominal Current Low Threshold</td></tr>
<tr><td colspan="3">Maximum Flow Precision</td><td colspan="2">Maximum Flow Scale=l/h</td><td colspan="3">Maximum Flow Size</td></tr>
<tr><td colspan="8">Maximum Flow Value 1</td></tr>
<tr><td colspan="8">…</td></tr>
<tr><td colspan="8">Maximum Flow Value N</td></tr>
<tr><td colspan="3">Flow High Threshold Precision</td><td colspan="2">Flow High Threshold Scale=l/h</td><td colspan="3">Flow High Threshold Size</td></tr>
<tr><td colspan="8">Flow High Threshold Value 1</td></tr>
<tr><td colspan="8">…</td></tr>
<tr><td colspan="8">Flow High Threshold Value N</td></tr>
<tr><td colspan="3">Flow Low Threshold Precision</td><td colspan="2">Flow Low Threshold Scale=l/h</td><td colspan="3">Flow Low Threshold Size</td></tr>
<tr><td colspan="8">Flow Low Threshold Value 1</td></tr>
<tr><td colspan="8">…</td></tr>
<tr><td colspan="8">Flow Low Threshold Value N</td></tr>
<tr><td colspan="8">Sensor Usage</td></tr>
</table>

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Master Valve (1 bit)

Indicates if the valve ID refers to a master or standard “zone” valve id.

Valve ID (8 bits)

The valve ID of interest. Valves (including master) MUST be identified sequentially from 1 to the total number available on the device.

Nominal Current High Threshold (8 bits)

The valve's nominal current (1 unit equals 10 mA) high threshold. If the current crosses above this threshold it will trigger the current high threshold notification.

Nominal Current Low Threshold (8 bits)

The valve's nominal current (1 unit equals 10 mA) low threshold. If the current crosses below this threshold it will trigger the current low threshold notification.

Maximum Flow

The maximum flow allowed for the valve. This field is only valid if a flow sensor is connected to the device. This field does not apply to a master valve. This field MUST be expressed in l/h.

The format of the Flow fields MUST comply with the format used by the Multilevel Sensor Command Class.

Flow High Threshold

The valve's flow high threshold. If the flow crosses above this threshold it will trigger the flow high threshold notification. This field is only valid if a flow sensor is connected to the device. This field does not apply to a master valve. This field MUST be expressed in l/h.

The format of the Flow fields MUST comply with the format used by the Multilevel Sensor Command Class.

Flow Low Threshold

The valve's flow low threshold. If the flow crosses below this threshold it will trigger the flow low threshold notification. This field is only valid if a flow sensor is connected to the device. This field does not apply to a master valve. This field MUST be expressed in l/h.

The format of the Flow fields MUST comply with the format used by the Multilevel Sensor Command Class.

Sensor Usage (8 bits)

This field allows the valve to be configured to use certain sensors when running. For example, if the valve is configured to use the rain sensor, the valve will shut off (or not run) if the rain sensor is turned on (or active). The field MUST be encoded according to Table 81**,** Sensor Usage.

4.67.16 Irrigation Valve Run Command

The Irrigation Valve Run Command will run the specified valve for a specified duration.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_IRRIGATION							
Command = IRRIGATION_VALVE_RUN							
Reserved							Master Valve
Valve ID							
Duration MSB							
Duration LSB							

Reserved fields MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Master Valve (1 bit)

Indicates if the valve ID refers to a master or standard “zone” valve id.

Valve ID (8 bits)

The valve ID of interest. Valves (including master) MUST be identified sequentially from 1 to the total number available on the device.

Duration (16 bits)

In seconds, where a value of 0 indicates that the valve should be shut off, if running.

4.67.17 Irrigation Valve Table Set Command

The Irrigation Valve Table Set Command is used to set a valve table with a list of valves and durations.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_IRRIGATION							
Command = IRRIGATION_VALVE_TABLE_SET							
Valve Table ID							
Valve ID 1							
Duration MSB 1							
Duration LSB 1							
…							
Valve ID N							
Duration MSB N							
Duration LSB N							

Valve Table ID (8 bits)

The valve table ID. Valves tables MUST be identified sequentially from 1 to the total number available on the device.

Valve ID and Duration (N * (8 bits+16 bits))

These fields contain a variable list of valve table entries, composed of a valve ID and a duration in seconds. Valves MUST be identified sequentially from 1 to the total number available on the device. This list MUST NOT exceed 15 entries (45 bytes). This list MAY be empty (0 bytes).

4.67.18 Irrigation Valve Table Get Command

The Irrigation Valve Table Get Command is used to request the contents of the specified Valve Table ID.

The Irrigation Valve Table Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_IRRIGATION							
Command = IRRIGATION_VALVE_TABLE_GET							
Valve Table ID							

Valve Table ID (8 bits)

The valve table ID. Valves tables MUST be identified sequentially from 1 to the total number available on the device.

4.67.19 Irrigation Valve Table Report Command

The Irrigation Valve Table Report Command provides the contents of the specified Valve Table ID.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_IRRIGATION							
Command = IRRIGATION_VALVE_TABLE_REPORT							
Valve Table ID							
Valve ID 1							
Duration MSB 1							
Duration LSB 1							
…							
Valve ID N							
Duration MSB N							
Duration LSB N							

Valve Table ID (8 bits)

The valve table ID. Valves tables MUST be identified sequentially from 1 to the total number available on the device.

Valve ID and Duration (N * (8 bits+16 bits))

These fields contain a variable list of valve table entries, composed of a valve ID and a duration in seconds. Valves MUST be identified sequentially from 1 to the total number available on the device. This list MUS T NOT exceed 15 entries (45 bytes). This list MAY be empty (0 bytes).

4.67.20 Irrigation Valve Table Run Command

The Irrigation Valve Table Run Command is used to run the specified valve tables sequentially.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_IRRIGATION							
Command = IRRIGATION_VALVE_TABLE_RUN							
Valve Table ID 1							
…							
Valve Table ID N							

Valve Table ID (N bytes)

These fields contain a variable list of valve table IDs that will be run sequentially in the order they are listed. This list MUST NOT exceed 46 entries.

4.67.21 Irrigation System Shutoff Command

The Irrigation System Shutoff Command is used to shut off the irrigation system for a specified duration.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_IRRIGATION							
Command = IRRIGATION_SYSTEM_SHUTOFF							
Duration							

Duration (8 bits)

The duration for which all irrigation activity must be shut off. The duration MUST be in the range 1..254 hours (approximately 10 days). The shutoff duration MUST temporarily prevent all enabled schedules, including the backup schedule, from running. Once the duration expires, any enabled schedules MUST run as configured.

If the duration is set to 0x00, the command MUST immediately turn off any running valves (including master valves), as well as cancel any active Irrigation Valve Table Run Commands or currently running schedules. Any future schedules MUST continue to run as configured.

If the duration is set to 0xFF, the command MUST prevent all enabled schedules, including the backup schedule, from running until the shutoff duration is set to a lesser value. The MAY be enabled immediately by sending the command again with a duration of 0x00.

If the device receives any the following commands the shutoff duration MUST be reset to zero. Note, the gateway SHOULD NOT send any of these commands if the device is expected to be in a “shutoff” mode.

- Irrigation Valve Run Command
- Irrigation Valve Table Run Command

4.68 Language Command Class, version 1

The Language Command Class is used to specify the language settings on a device.

4.68.1 Language Set Command

The Language Set Command used to transfer data to a device. The device uses the default language in case the selected language is not supported.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_LANGUAGE							
Command = LANGUAGE_SET							
Language 1							
Language 2							
Language 3							
Country 1							
Country 2							

Language (24 bits)

The code definition of the languages may be found in ISO 639-2:1998 ‘Codes for the representation of names of languages – Part 2: Alpha-3 code’. In the table below are some examples of the alpha-3 codes listed:

Language	Language 1	Language 2	Language 3
English	e	n	g
Flemish; Dutch	d	u	t
French	f	r	e
German	g	e	r
Italian	i	t	a
Polish	p	o	l
Russian	r	u	s
Walloon	w	l	n

Country (16 bits)

The code definition of the countries may be found in ISO 3166-1 ‘Country Codes: Alpha-2 codes’. The use of the country field is OPTIONAL and only defined in case it is necessary for distinguishing geographical variants. The number of data fields transmitted MUST be determined from the length field in the frame. In the table below are some examples of the alpha-2 codes listed:

Language	Country 1	Country 2
Belgium	B	E
Italy	I	T
Netherlands	N	L
Poland	P	L
United Kingdom	G	B
United States	U	S

4.68.2 Language Get Command

The Language Get Command is used to request the current language setting in a device.

The Language Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_LANGUAGE							
Command = LANGUAGE_GET							

4.68.3 Language Report Command

The Language Report returns the current language setting in a device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_LANGUAGE							
Command = LANGUAGE_REPORT							
Language 1							
Language 2							
Language 3							
Country 1							
Country 2							

Refer to explanation under the Language Set Command.

4.69 Lock Command Class, version 1 [DEPRECATED]

<u>THIS COMMAND CLASS HAS BEEN DEPRECATED</u>

A device MAY implement this command class, but it is RECOMMENDED that new implementations use the Door Lock Command Class.
If implementing this command class, it is RECOMMENDED that the Door Lock Command Class is also implemented.

The Lock Command Class is used to lock and unlock a “lock” type device, e.g. a door or window lock

4.69.1 Lock Set Command

The Lock Set Command used to set the lock state in a device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_LOCK							
Command = LOCK_SET							
Lock State							

Lock State (8 bits)

The lock state field used to set the lock state of the device. The value 0 indicates that the device is unlocked. The value 1 indicates that the device is locked.

4.69.2 Lock Get Command

The Lock Get Command is used to request the lock state from a device.

The Lock Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing.
The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_LOCK							
Command = LOCK_GET							

4.69.3 Lock Report Command

The Lock Report Command used to report the lock state of a device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_LOCK							
Command = LOCK_REPORT							
Lock State							

Lock State (8 bits)

The Lock state field used to report the lock state of the device. The value 0 indicates that the device is unlocked. The value 1 indicates that the device is locked.

4.70 Mailbox Command Class, version 1

The Mailbox Command Class is intended for IP based gateway deployments with distributed mailbox resources. One example is a constrained gateway device which is offloaded by another IP host with sufficient memory to host the Mailbox Service. The Mailbox Service may be hosted by a LAN host or an Internet server.

The Mailbox Command Class allows any mailbox capable device to either make itself into a Mailbox Service, or utilize another Mailbox Service in the network.

4.70.1 Mailbox Framework

The Mailbox Command Class describes a framework that consists of two specific Mailbox Modes described below:

1. **Mailbox Proxy,** which forwards mailbox requests to a Mailbox Service.

2. **Mailbox Service,** which accepts the forwarded mailbox requests and stores them until the designated recipient announces that it is awake.

A mailbox device MAY support one or both of the two Mailbox Modes. However, a mailbox device MUST NOT take both Mailbox Modes in a network.

Before configuring Mailbox Proxy forwarding, a configuring node MUST ensure that the forwarding and receiving devices support their respective required modes. The information can be found using the Mailbox Configuration Get Command and Mailbox Configuration Report Command.

4.70.1.1 Mailbox Proxy

The Mailbox Proxy device forwards all received frames that are destined for a non-listening node to the configured Mailbox Service. Before forwarding the frame, it MUST be attempted to send the frame to the node first as it may be awake following a manual activation or inclusion. If the Mailbox Proxy can deliver the frame to the non-listening node, the Mailbox Proxy MUST NOT forward the frame to the Mailbox Service.

The Mailbox Proxy MUST support the Wake Up Command Class.

4.70.1.2 Mailbox Service

The Mailbox Service serves as a conventional mailbox, with the addition that it may receive forwarded frames from a Mailbox Proxy. A Mailbox Service may have a finite mailbox queue capacity, which is reported in the Mailbox Configuration Report. The Mailbox Service MUST NOT communicate with a Z/IP client directly, since it may not be able to route messages to the client.

4.70.1.3 Frameflow

Figure 28 illustrates the communication between a Z/IP Client (1) attempting communication to a non-listening node (4). The communication is passing through the Mailbox Proxy (2) which initially will attempt direct communication with (4). If failing to reach (4), the frame will be forwarded to the Mailbox Service (3) using the Mailbox Queue Command with Push Operation.

Following the Mailbox Queue push, the Mailbox Service will send a Mailbox Queue Command with Waiting Operation to the proxy, piggybacking the original UDP command on the message. The Proxy will build a “NACK Waiting” Z/IP Command targeted for the Z/IP node, based on the piggy backed message from the Proxy Service. The Proxy Service MUST also append the Expected Delay header extension to the “NACK Waiting” Z/IP Command. This step MUST be repeated every 60s seconds as long as the message is in the mailbox.

Upon wake-up, the non-listening node (4) will transmit a Wake Up Notification to the Mailbox Proxy (2), which must be configured using the Wake Up Command Class. Whenever the Mailbox Proxy (2) receives a Wake Up Notification, the notification will be forwarded as a Z/IP Packet to the Mailbox Service (4). The Mailbox Service inspects the queue to see if there are any frames for (4) and responds with either an empty Mailbox Queue Command Pop operation with “Last” bit set to 1 or any frames that may be in queue, finishing with the last frame having “Last” bit set to 1.

Mailbox Proxy (2) receives the Mailbox Queue Pop frame on which it performs a Virtual Node Rewrite to match the original sender of the UDP frame of the Mailbox Queue Pop command. The frame is sent from the virtual node to (4) followed by a “Wake Up No More Information” Command. Any eventual reports will be replied to the virtual node that forwards them to (1). The proxy MUST send a Mailbox Queue Command with ACK operation to the Proxy Service when it has delivered the frame and potentially the “No more information”

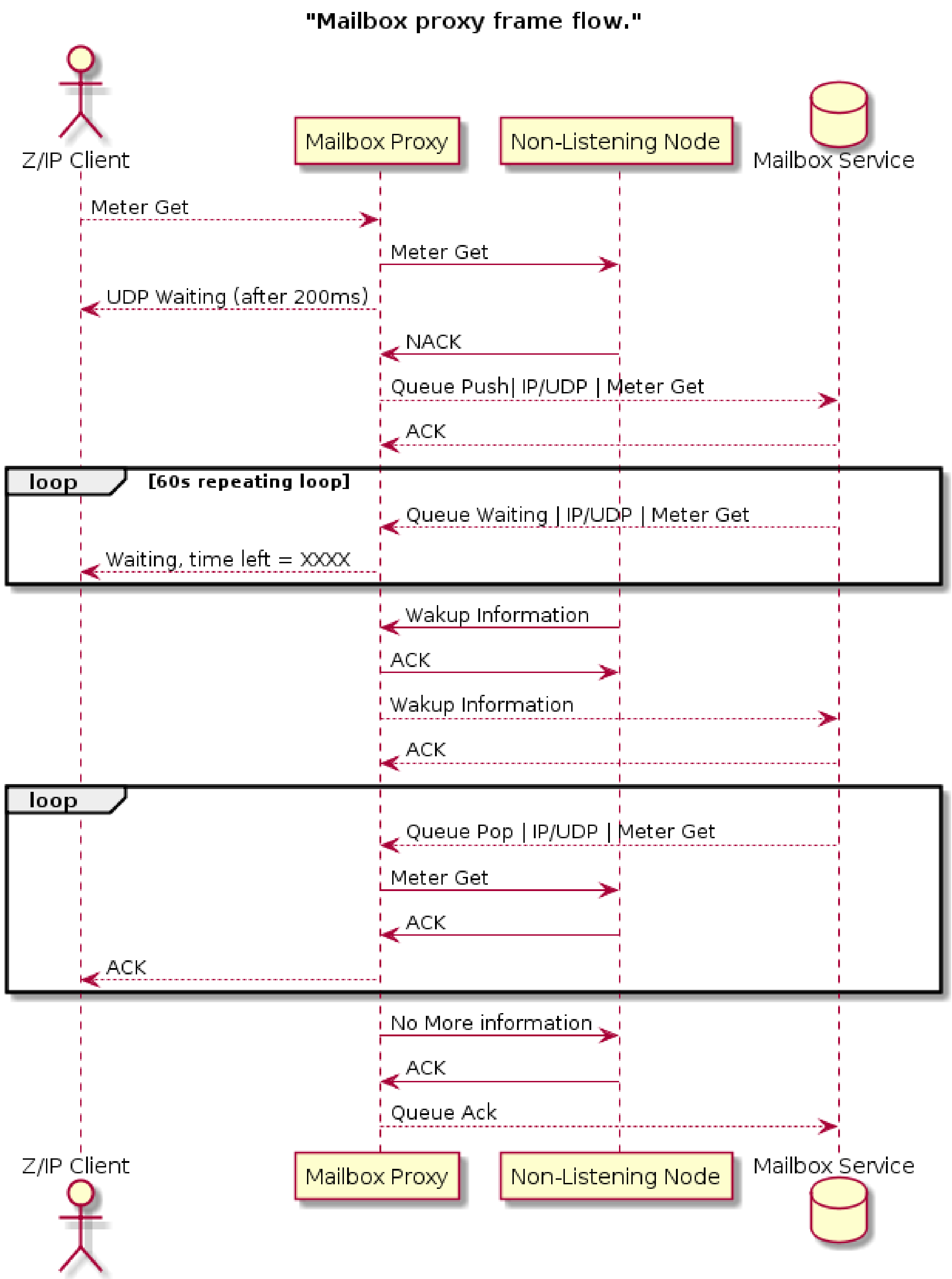


Figure 28, Mailbox Frameflow

4.70.2 Mailbox Configuration Get Command

The Mailbox Configuration Get Command is used to request the Mailbox configuration from a supporting device.

The Mailbox Configuration Report command MUST be returned in response to a Mailbox Configuration Get command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_MAILBOX							
Command = MAILBOX_CONFIGURATION_GET							

4.70.3 Mailbox Configuration Set Command

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_MAILBOX							
Command = MAILBOX_ CONFIGURATION_SET							
Reserved					Mode		
Forwarding Destination IPv6 Address – Byte 1							
…							
Forwarding Destination IPv6 Address – Byte 16							
UDP Port Number - Byte 1							
UDP Port Number - Byte 2							

Reserved (5 bits)

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Mode (3 bits)

The Mode field is used to advertise the Mailbox mode to be configured in the node. This field MUST be encoded according to Table 82.

Table 82, Mailbox Configuration Set::Mode encoding

Value	Description
0x00	Disable Mailbox Service Disable Mailbox Proxy forwarding
0x01	Enable Mailbox Service
0x02	Enable Mailbox Proxy forwarding

Forwarding Destination IPv6 Address (16 bytes)

If the Mailbox Proxy Forwarding is enabled in the Mode field, the Forwarding Destination IPv6 Address field MUST specify the Forwarding Destination IPv6 Address. The field MUST specify an IPv6 formatted address of the Mailbox Service to receive forwarded mailbox packages. If the Forwarding Destination is identified by an IPv4 address this field MUST be formatted as an IPv4-mapped IPv6 address [7].

If the Mailbox Proxy Forwarding is not enabled in the Mode field, the Forwarding Destination IPv6 Address MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

UDP Port Number (2 bytes)

This field indicates the UDP Port number of the Mailbox Service running at the Forwarding Destination.

If the Mailbox Proxy Forwarding is not enabled in the Mode field, this field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

4.70.4 Mailbox Configuration Report Command

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_MAILBOX</td></tr>
<tr><td colspan="8">Command = MAILBOX_ CONFIGURATION_REPORT</td></tr>
<tr><td colspan="3">Reserved</td><td colspan="2">Supported Modes</td><td colspan="3">Mode</td></tr>
<tr><td colspan="8">Mailbox Capacity – Byte 1</td></tr>
<tr><td colspan="8">Mailbox Capacity – Byte 2</td></tr>
<tr><td colspan="8">Forwarding Destination IPv6 Address – Byte 1</td></tr>
<tr><td colspan="8">…</td></tr>
<tr><td colspan="8">Forwarding Destination IPv6 Address – Byte 16</td></tr>
<tr><td colspan="8">UDP Port Number – Byte 1</td></tr>
<tr><td colspan="8">UDP Port Number – Byte 2</td></tr>
</table>

Reserved (5 bits)

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Supported Modes (2 bits)

The Supported Modes bit field is used to advertise the functionalities supported by the node. This field MUST be encoded according to Table 83

Table 83, Mailbox Configuration Report::Supported Modes encoding

Bit Value	Description
0x01	Mailbox Service supported
0x02	Mailbox Proxy supported

Mode (3 bits)

Refer to 4.70.3 Mailbox Configuration Set Command.

Mailbox Capacity (2 bytes)

This field advertises the number of frames (at a maximum of 1280 bytes per frame) that may be stored in the mailbox while waiting for a Wake Up Notification.

A value of 0 MUST indicate that the mailbox will only support mailbox forwarding to another Mailbox Service.

A value of 0xFFFF MUST indicate that the mailbox in effect have no storage limitation.

Forwarding Destination IPv6 Address (16 bytes)

Refer to 4.70.3 Mailbox Configuration Set Command.

UDP Port Number (2 bytes)

Refer to 4.70.3 Mailbox Configuration Set Command.

4.70.5 Mailbox Queue Command

The Mailbox Queue Command is a container for various operations between a mailbox proxy and a Mailbox Service.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_MAILBOX</td></tr>
<tr><td colspan="8">Command = MAILBOX_QUEUE</td></tr>
<tr><td colspan="4">Reserved</td><td>Last</td><td colspan="3">Operation</td></tr>
<tr><td colspan="8">Queue Handle</td></tr>
<tr><td colspan="8">Mailbox Entry – Byte 1</td></tr>
<tr><td colspan="8">…</td></tr>
<tr><td colspan="8">Mailbox Entry – Byte N</td></tr>
</table>

Reserved (6 Bit)

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Last (1 bit)

The Last field is used to indicate if the current mailbox frame is the last in the queue for the specific device. The Last bit only applies when the “Pop” Operation is used.

The value 1 MUST indicate that the frame is the last on the queue.

The value 0 MUST indicate that more frames will follow.

Operation (3 bits)

The encoding of the Operation field MUST be according to Table 84.

Table 84, Mailbox Queue::Operation

Value	Description
0x00	Push. Queue a message from the proxy to the Mailbox Service
0x01	Pop. Dequeue a message from the Mailbox Service to the Mailbox Proxy for delivery on the PAN
0x02	Waiting. Service->Proxy: send waiting messages to the client.
0x03	Ping. Service->Proxy: send UDP ping messages to the client.
0x04	ACK. Proxy->Service: Frame has been delivered. Service->Proxy: Frame has been queued.
0x05	NACK. Proxy->Service: Frame was not queued. Wait for ACK before attempting queuing. Service->Proxy: Node is not responding. Keep in queue.
0x06	Queue Full. Proxy->Service: The capacity of the Mailbox Service has been reached. Wait until queue has been emptied.

All other values are reserved and MUST NOT be used by a sending node.
Reserved values MUST be ignored by a receiving node.

Queue Handle (8 bits)

The Queue Handle field is used to identify the queue this message belongs to. A service uses this handle with the source IP of the MAILBOX_QUEUE message to identify the queue to which a message belongs to.

Mailbox Entry (N Bytes)

The Mailbox Entry field contains the entire received UDP Package. Including, ZIP headers and Z-Wave Payload.

To avoid duplicate entries, the Mailbox Service MUST maintain a list of CRC16 checksums for each mailbox entry. All mailbox entries MUST be unique, if a matching CRC16 exists for an incoming package, the incoming package MUST be discarded.

When WAITING timer elapses the mailbox MUST send a WAITING message to all clients that has posted entries to the mailbox.

4.70.6 Mailbox Wake Up Notification Command

This command allows a mailbox proxy resource to notify a Mailbox Service resource that a wake up device is currently awake.

A Mailbox Proxy resource MAY send this command to a Mailbox Service resource.
A Mailbox Service resource MUST NOT send this command to a mailbox proxy resource.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_MAILBOX							
Command = MAILBOX_WAKEUP_NOTIFICATION							
Queue Handle							

Queue Handle (8 bits)

This field is used to specify the actual queue handle to send notification to.

4.70.7 Mailbox Failing Node Command

This command allows a mailbox proxy resource to notify a Mailbox Service resource that a wake up device is no longer available.

A Mailbox Proxy resource MAY send this command to a Mailbox Service resource.
A Mailbox Service resource MUST NOT send this command to a mailbox proxy resource.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_MAILBOX							
Command = MAILBOX_NODE_FAILING							
Queue Handle							

Queue Handle (8 bits)

This field is used to specify the actual queue.

A receiving Mailbox Service resource MUST discard all state information and enqueued messages for the actual queue.

4.70.8 Frame Flow diagrams Examples

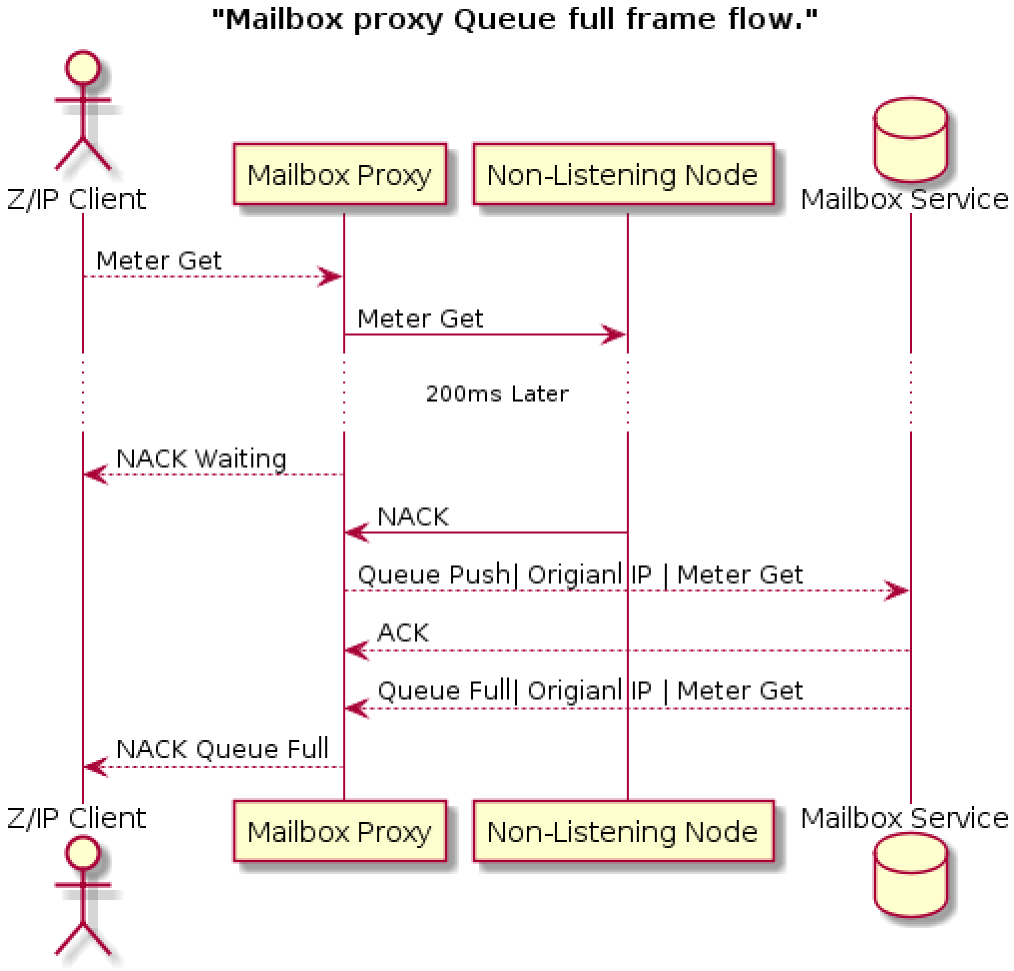


Figure 29, Mailbox proxy queue full frame flow

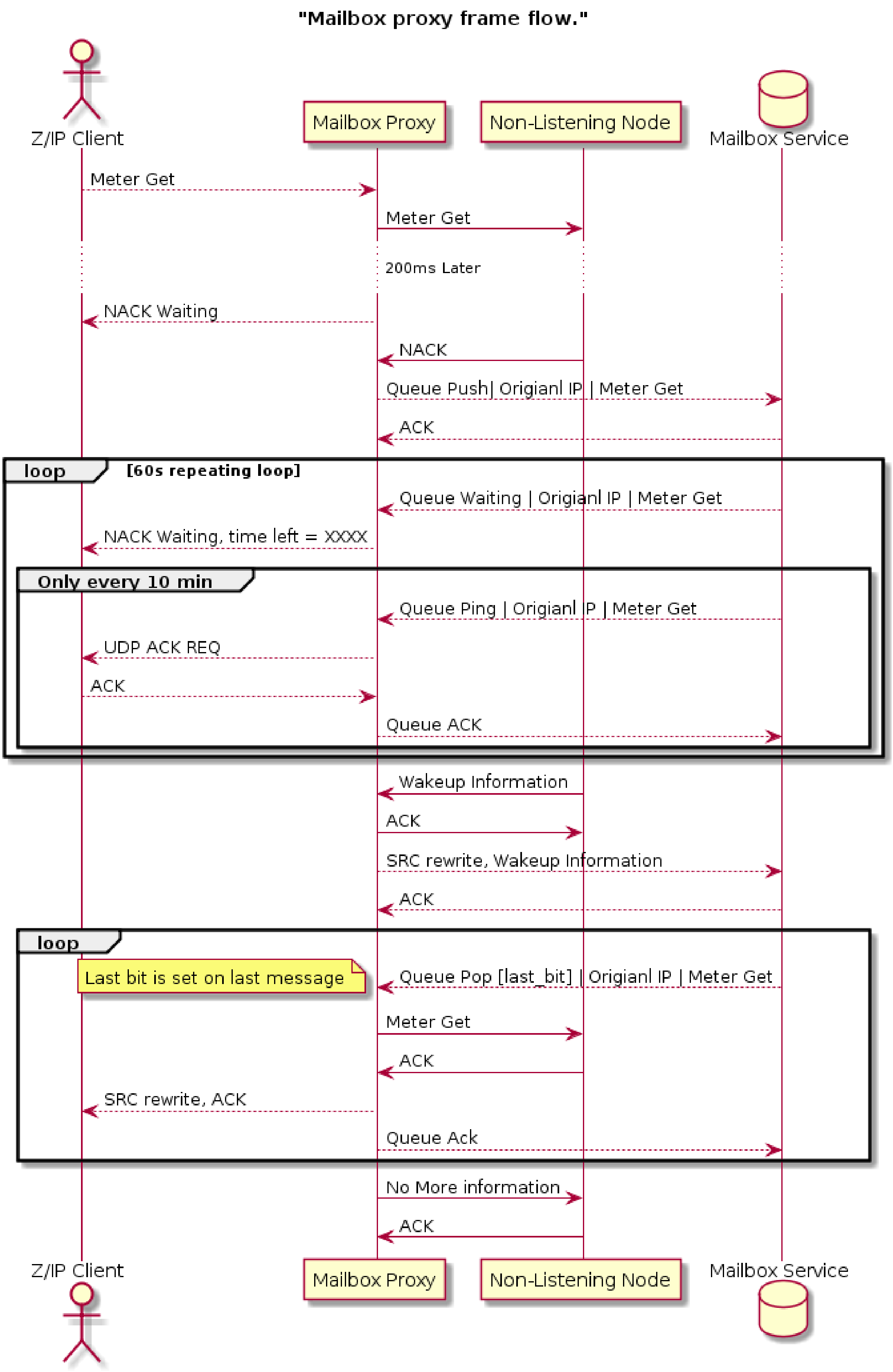


Figure 30, Normal frame flow

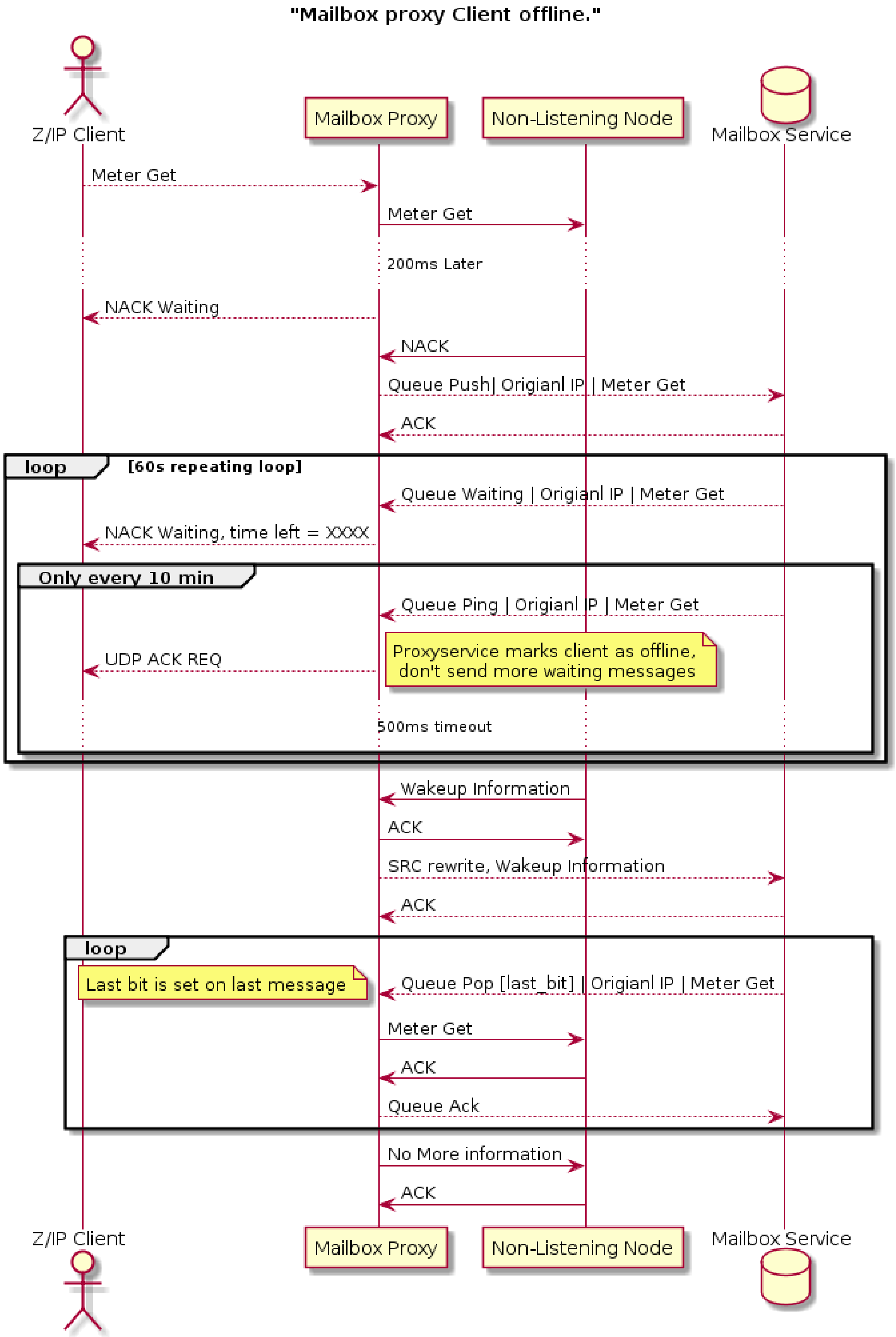


Figure 31, Z/IP Client goes offline and stops replying to UDP ping

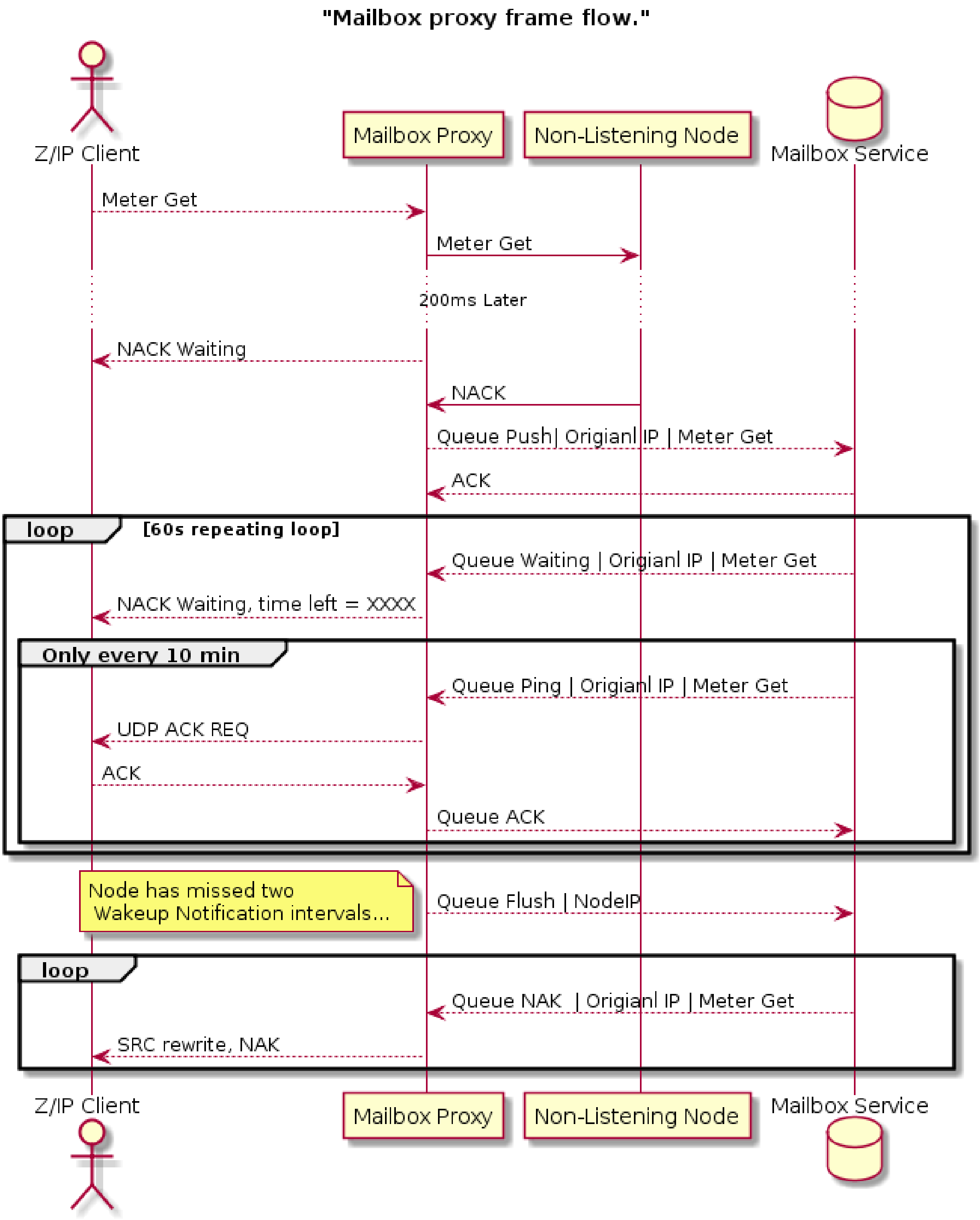


Figure 32, Sleeping node misses 2 wakeup intervals and proxy tells service to flush queue

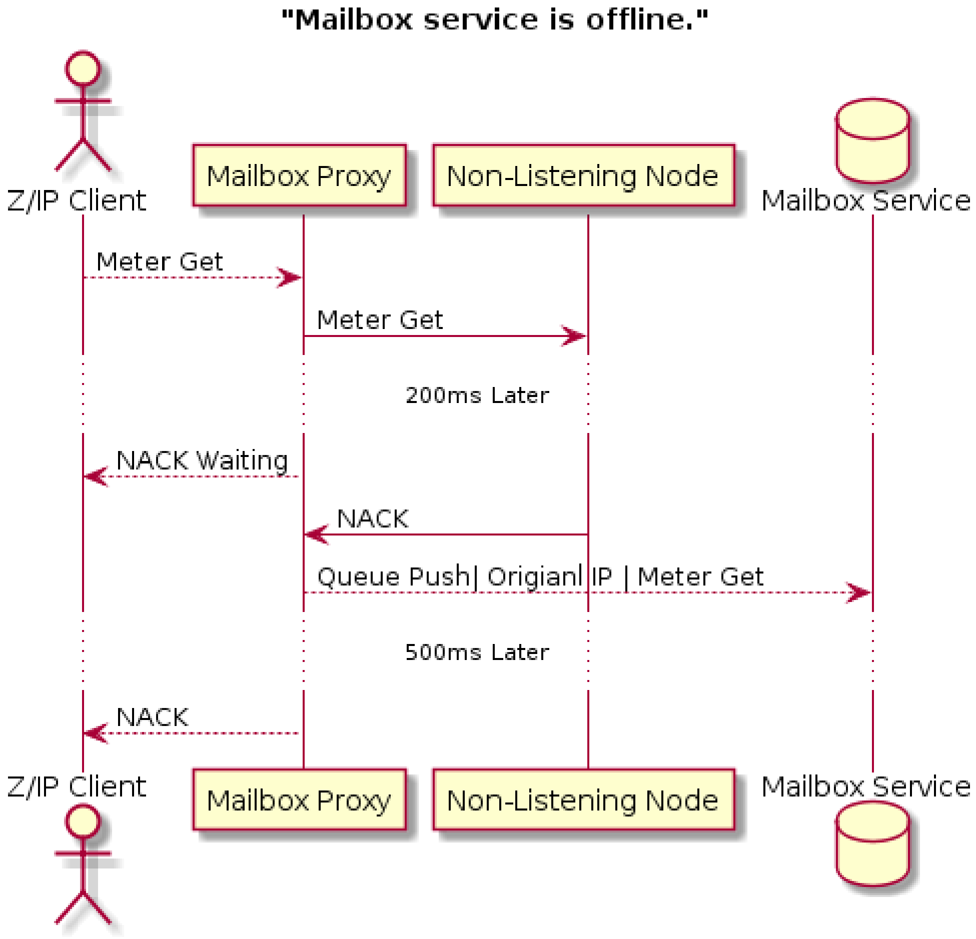


Figure 33 Mailbox Service is offline

4.71 Manufacturer Proprietary Command Class, version 1

The Manufacturer Proprietary Command Class is used to transfer data between devices in the Z-Wave network. The data content MUST be vendor specific and MUST be non-value added with respect to the Home Automation application in general. An example could be data used to diagnose the hardware in a device.

Note: Do not use the Manufacturer Proprietary Command Class without written approval from Sigma Designs.

4.71.1 Manufacturer Proprietary Command

The Manufacturer Proprietary Command is used to transfer proprietary commands between devices. The Command features a manufacturer specific identifier to allow the receiving device to check if this Command can be interpreted. The vendor is responsible for establishing a Command structure to differ between the set of Commands supported.

In order not to congest the Z-Wave network, large data transfers MUST leave transmit opportunities for other nodes in the network. If sending a command longer than two frames, a node MUST implement a delay between every transmitted frame. The minimum required time delay and number of frames before a delay must be inserted depends on the actual bit rate.

- 40 kbit/s: At least 35 ms if sending more than 2 frames back-to-back
- 100 kbit/s: At least 15 ms if sending more than 2 frames back-to-back

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_MANUFACTURER_PROPRIETARY							
Manufacturer ID 1							
Manufacturer ID 2							
Data 1							
…							
Data N							

Manufacturer ID (16 bits)

The Manufacturer ID is a unique ID identifying the manufacturer of the device. Manufacturer identifiers can be found in [11].

The first byte (Manufacturer ID 1) is the most significant byte.

Data (N bytes)

The data fields may be used for data transfer etc. The number of data fields transmitted MUST be determined from the length field in the frame.

4.72 Manufacturer Specific Command Class, version 1

Use this Command Class to advertise manufacturer specific information. Version 2 of this command class further allows device specific information to be advertised.

4.72.1 Security Considerations

The Manufacturer Specific Command Class provides information about the device implementation. This information can potentially be used by an attacker to find vulnerabilities.
If a node supports this Command Class, it MUST comply with Table 85.

Table 85, Manufacturer Specific Command Class support

	After Non-Secure inclusion	After Secure inclusion
Non-Secure or Less-Secure communication	The node MUST support the Manufacturer Specific Command Class and advertise it in the NIF.	The node MUST NOT support the Manufacturer Specific Command Class and MUST NOT advertise it in the NIF or Security Command Class Supported Capabilities commands.
Secure communication at the highest Security Class	N/A.	The node MUST support the Manufacturer Specific Command Class and advertise it in the Security Command Supported Capabilities commands.

4.72.2 Manufacturer Specific Get Command

A node MAY use the Manufacturer Specific Get Command to request manufacturer specific information from another node.

The Manufacturer Specific Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing.
The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_MANUFACTURER_SPECIFIC							
Command = MANUFACTURER_SPECIFIC_GET							

4.72.3 Manufacturer Specific Report Command

This command is used to advertise manufacturer specific device information.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_MANUFACTURER_SPECIFIC							
Command = MANUFACTURER_SPECIFIC_REPORT							
Manufacturer ID 1							
Manufacturer ID 2							
Product Type ID 1							
Product Type ID 2							
Product ID 1							
Product ID 2							

Manufacturer ID (16 bits)

The Manufacturer ID field MUST carry the unique ID identifying the manufacturer of the device.

Manufacturer identifiers can be found in [11].The first byte is the most significant byte.

Product Type ID (16 bits)

The Product Type ID field MUST carry a unique ID identifying the actual product type.

The first byte is the most significant byte.

A specific Product Type ID MUST be defined by the manufacturer for each type of product.

Product ID (16 bits)

The Product ID field MUST carry a unique ID identifying the actual product.

The first byte is the most significant byte.

A specific Product ID MUST be defined by the manufacturer for each product of a given product type. Thus, the same Product ID value may appear for different Product Type ID values.

4.73 Manufacturer Specific Command Class, version 2

Manufacturer Specific Command Class, version 2 adds a set of commands to communicate unique identification, e.g. the serial number, of the product.

Commands not mentioned here remains unchanged as specified for Manufacturer Specific Command Class, Version 1.

4.73.1 Security Considerations

The Manufacturer Specific Command Class provides information about the device implementation. This information can potentially be used by an attacker to find vulnerabilities.
If a node supports this Command Class, it MUST comply with Table 85.

4.73.2 Device Specific Get Command

This command is used to request device specific information.

The Device Specific Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing.
The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_MANUFACTURER_SPECIFIC</td></tr>
<tr><td colspan="8">Command = DEVICE_SPECIFIC_GET</td></tr>
<tr><td colspan="5">Reserved</td><td colspan="3">Device ID Type</td></tr>
</table>

Device ID Type (3 bits)

This field contains values for the Device ID Type.

Device ID Type	Value
Return OEM factory default Device ID Type	0
Serial Number	1
Pseudo Random	2
Reserved	3-7

A requesting node SHOULD specify a value of zero when issuing the Device Specific Get command since the responding node may only be able to return one Device ID Type.

4.73.3 Device Specific Report Command

This command is used to advertise device specific information.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_MANUFACTURER_SPECIFIC</td></tr>
<tr><td colspan="8">Command = DEVICE_SPECIFIC_REPORT</td></tr>
<tr><td colspan="5">Reserved</td><td colspan="3">Device ID Type</td></tr>
<tr><td colspan="3">Device ID Data Format</td><td colspan="5">Device ID Data Length</td></tr>
<tr><td colspan="8">Device ID Data 1</td></tr>
<tr><td colspan="8">...</td></tr>
<tr><td colspan="8">Device ID Data N</td></tr>
</table>

Device ID Data Format (3 bits)

This command field defines the format used for Device ID Data.

Device ID Data Format	Value	Description
UTF-8	0	The Device ID Data MUST be in UTF-8 format.
Binary	1	The Device ID Data is in plain binary format and MUST be displayed as hexadecimal values e.g. 0x30, 0x31, 0x32, 0x33 MUST be displayed as h'30313233.

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Device ID Type (3 bits)

This field contains values for the Device ID Type. See Device Specific Get Command for details.

In case the Device ID Type specified in a Device Specific Get command is not supported by the responding node, the responding node MAY return the factory default Device ID Type (as if receiving the value 0 in the Device Specific Get command).

Device ID Data Length(5 bits)

This field contains the length of the Device ID Data field. The field MUST NOT carry the value zero.

Device ID Data (N bytes)

Data fields for Device ID. “Device ID Data Format” defines data format and “Device ID Data Length Indicator” defines length.

The Manufacturer ID and Device ID combination MUST be globally unique.

4.74 Meter Command Class, version 1

The Meter Command Class defines the Commands necessary to read accumulated values in physical units from a water meter or metering device (gas, electric etc.) and thereby enabling automatic meter reading capabilities.

Automatic meter reading (AMR), is the technology of automatically collecting data from water meter or energy metering devices and transferring that data to a central database for billing and/or analyzing.

4.74.1 Terminology

A meter is used to monitor a resource. The meter accumulates the resource flow over time. As an option, the meter may report not only the most recent accumulated reading but also the previous reading and the time that elapsed since then. A meter may also be able to report the current resource flow. This is known as the instant value.

Meters may report two rate types. A utility installs production meters in its production facilities, while it installs consumption meters at consumer premises. The difference is the mapping of the physical flow direction to the forward direction of the meter:

- The accumulated value of a production meter grows when more resources are produced
- The accumulated value of a production meter drops when more resources are consumed

- The accumulated value of a consumption meter grows when more resources are consumed
- The accumulated value of a consumption meter drops when more resources are produced

If a meter only supports one rate type, the meter may run backwards. If a meter runs backwards, the accumulated value may eventually become negative. While the production meter of a power generator rarely runs backwards, the consumption meter in a private household with solar panels runs backwards every time the solar panels deliver more than is consumed in the household. The actual reading of a one-rate type meter reflects the net accumulated value since the meter was installed with a factory default reading of zero. The accumulated value over an arbitrary interval may be calculated by subtracting a previous accumulated value from the current accumulated value. Normal arithmetic rules ensure that this also works in case of negative values.

A production meter may be used as a consumption meter by applying a sign change to all production meter readings before using the values as consumption meter readings.

A meter device may advertise that it implements two separate registers for the production and consumption, respectively. In that case, both of these meters are always running forward: Production makes the production meter run forward while consumption makes the consumption meter run forward. Two meter reports must be used to report the values of the respective meters.

4.74.2 Meter Get Command

The Meter Get Command is used to request the accumulated consumption in physical units from a metering device.

The Meter Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER							
Command = METER_GET							

4.74.3 Meter Report Command

The Meter Report Command is used to advertise a meter reading.

The Meter Report Command MUST NOT be transmitted using broadcast.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER							
Command = METER_REPORT							
Meter Type							
Precision			Scale		Size		
Meter Value 1							
…							
Meter Value N							

Meter Type (8 bits)

Meter Type specifies what type of metering device this Command originates from. Refer to the table below with respect to defined metering devices.

Table 86, Meter Report v1::Meter Type encoding

Value	Meter Type
0x01	Electric meter
0x02	Gas meter
0x03	Water meter

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Precision (3 bits)

The precision field describes what the precision of the value is. The number indicates the number of decimals. The decimal value 1025 with precision 2 is therefore equal to 10.25.

The value of the precision field MUST be in the range 0..7.

Scale (2 bits)

The Scale field MUST advertise the unit used for this report. Refer to the table below with respect to defined scales for the relevant metering devices.

Table 87, Meter Report v1::Scale encoding

Meter Type	Scale	Value
Electric meter	kWh	0x00
Electric meter	*reserved*	0x01
Electric meter	*reserved*	0x02
Electric meter	*reserved*	0x03
Gas meter	Cubic meters	0x00
Gas meter	*reserved*	0x01
Gas meter	*reserved*	0x02
Gas meter	*reserved*	0x03
Water meter	Cubic meters	0x00
Water meter	Cubic feet	0x01
Water meter	US gallons	0x02
Water meter	*reserved*	0x03

Reserved values MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Size (3 bits)

The size field indicates the number of bytes that used for the meter value. This field can take values from 1 (001b), 2 (010b) or 4 (100b).

Meter Value (N bytes)

The Meter Value is a signed field. The field MAY be 1, 2 or 4 bytes in size. The first byte is the most significant byte. The table below shows signed decimal values together with their hexadecimal equivalents.

Table 88, Meter Report v1::Meter Value encoding

Signed 1 byte decimal value	Hexadecimal	Signed 2 bytes decimal value	Hexadecimal
127	0x7F	32767	0x7FFF
25	0x19	1025	0x0401
2	0x02	2	0x0002
1	0x01	1	0x0001
0	0x00	0	0x0000
-1	0xFF	-1	0xFFFF
-2	0xFE	-2	0xFFFE
-25	0xE7	-1025	0xFBFF
-128	0x80	-32768	0x8000

Notice: The metering device receiving the Meter Report MUST always show the value even though the Metering Device Type and/or Scale are not supported.

4.75 Meter Command Class, version 2

The Meter Command Class is intended for devices capable of reporting energy measurements in addition to any main functionality or features e.g. an appliance module reporting the current consumption of the connected load. This command class is not intended for residential utility sub-metering such as a water meter counting total consumption.

Meter Command Class (Version 2) is improved with the following functionalities:

- Commands to interview the device for supported Meter types
- New field ‘Previous Meter Value’ added to the Meter Report to allow for easy calculation of consumption since previous measurement
- Reset of accumulated consumption
- Add capability to communicate current consumption (W)
- ‘Scale’ entry for pulse meters along with W, kVAh and Cubic feet

The commands not mentioned are the same as specified for Meter Command Class (Version 1).

4.75.1 Meter Supported Get Command

The Meter Supported Get Command is used to request the supported scales in a sub meter.

The Meter Supported Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER							
Command = METER_SUPPORTED_GET							

4.75.2 Meter Supported Report Command

This command is used to advertise supported scales in a sub meter.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_METER</td></tr>
<tr><td colspan="8">Command = METER_SUPPORTED_REPORT</td></tr>
<tr><td>Meter Reset</td><td colspan="2">Reserved</td><td colspan="5">Meter Type</td></tr>
<tr><td colspan="4">Reserved</td><td colspan="4">Scale Supported</td></tr>
</table>

Meter Reset (1 bit)

The Meter Reset field set to "1" indicates support for the Meter Reset Command.

Meter Type (5 bits)

See Meter Type defined in Meter Report Command.

Scale Supported (4 bits)

This field MUST advertise the supported scales of the device. The field MUST be formatted and interpreted as a bitmask.
The bit value '1' MUST indicate support for the actual scale. The bit value '0' MUST indicate that there is no support for the actual scale.

The Scale Supported bitmask MUST be interpreted in combination with the Meter Type.

Table 89, Meter Supported Report v2::Scale Supported encoding

Meter Type	Scale	Scale Supported Bit position	Measurement mode
Electric meter	kWh	Bit 0	Accumulated
	kVAh	Bit 1	Accumulated
	W	Bit 2	Instant
	Pulse count	Bit 3	Accumulated
Gas meter	Cubic meters	Bit 0	Accumulated
	Cubic feet	Bit 1	Accumulated
	Reserved	Bit 2	n/a
	Pulse count	Bit 3	Accumulated
Water meter	Cubic meters	Bit 0	Accumulated
	Cubic feet	Bit 1	Accumulated
	US gallons	Bit 2	Accumulated
	Pulse count	Bit 3	Accumulated

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

4.75.3 Meter Reset Command

The Meter Reset Command used to reset ALL accumulated values stored in the meter device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER							
Command = METER_RESET							

4.75.4 Meter Get Command

The Meter Get Command is used to request a device supported measurement in physical units from a metering device.

The Meter Report Command MUST be returned in response to this command if the requested scale is supported.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing.
The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER							
Command = METER_GET							
Reserved			Scale		Reserved		

Scale (2 bits)

See description of Meter Report Command.

If the Scale is not present (v1) or set to 0x00, a receiving node MUST return a Report using its default scale. A Meter Get Command containing a non-supported Scale MUST be ignored by a receiving device.

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

4.75.4.1 Backwards compatibility

When devices supporting Meter Command Class (Version 1) receives a Meter Get Command of Version 2 it MUST report its implemented scale.

The manufacturer MUST define which scale is the default scale and it MUST be described in the product manual.

4.75.5 Meter Report Command

The Meter Report Command is used to advertise a meter reading.

The Meter Report Command MUST NOT be transmitted using broadcast.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER							
Command = METER_REPORT							
Reser-ved	Rate Type		Meter Type				
Precision			Scale		Size		
Meter Value 1							
…							
Meter Value N							
Delta Time 1							
Delta Time 2							
Previous Meter Value 1							
…							
Previous Meter Value N							

Rate Type (2 bits)

Rate Type specifies if it is import or export values to be read. Setting the Rate Type to Import on the Meter Report Command is an indication that the Meter Value is a consumed measurement. In contrary when the Rate Type is set to Export the indication of the Meter Value is a produced measurement.

Table 90, Meter Report v2::Rate Type encoding

Value	Rate Type
0x00	Reserved
0x01	Import (consumed)
0x02	Export (produced)
0x03	Reserved

Meter Type (5 bits)

Meter Type specifies what type of metering device this command originates from. Refer to the table below with respect to defined metering devices.

Table 91, Meter Report v2::Meter Type encoding

Value	Meter Type
0x00	Reserved
0x01	Electric meter
0x02	Gas meter
0x03	Water meter
0x04-0x1F	Reserved

Precision (3 bits)

The Precision field describes what the precision of the value is. The number indicates the number of decimals. The decimal value 1025 with precision 1 MUST be interpreted as 102.5 and if precision is 3 then the interpreted value is 1.025.

Scale (2 bits)

The Scale field field MUST advertise the unit used for this report. Refer to the table below with respect to defined scales for the relevant metering devices.

Table 92, Meter Report v2::Scale encoding

Meter Type	Scale	Value
Electric meter	kWh	0x00
	kVAh	0x01
	W	0x02
	Pulse count	0x03
Gas meter	Cubic meters	0x00
	Cubic feet	0x01
	Reserved	0x02
	Pulse count	0x03
Water meter	Cubic meters	0x00
	Cubic feet	0x01
	US gallons	0x02
	Pulse count	0x03

Size (3 bits)

The Size field MUST advertise the number of bytes that is used for the Meter Value. The value of the Size field MUST be in the range (1,2,4). All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Meter Value (N bytes)

The Meter Value field is a signed field. The Meter Value field MUST have the size advertised by the Size field. The first byte is the most significant byte.

The table below shows signed decimal values together with the hexadecimal equivalents.

Table 93, Meter Report v2::Meter Value encoding

Signed 1 byte		Signed 2 bytes		Signed 4 bytes	
Dec	Hex	Dec	Hex	Dec	Hex
127	0x7F	32767	0x7FFF	2147483647	0x7FFFFFFF
...	...	...	...	...	...
63	0x3F	16383	0x3FFF	1073741823	0x3FFFFFFF
...	...	...	...	...	...
1	0x01	1	0x0001	1	0x00000001
0	0x00	0	0x0000	0	0x00000000
-1	0xFF	-1	0xFFFF	-1	0xFFFFFFFF
...	...	...	...	...	...
-63	0xC1	-16383	0xC001	-1073741823	0xC0000001
...	...	...	...	...	...
-128	0x80	-32768	0x8000	-2147483648	0x80000000

Delta Time (16 bits)

The Delta Time field MUST advertise the elapsed time in seconds between the ‘Meter Value’ and the ‘Previous Meter Value’ measurements. Values MUST be in the range 0 to 65536 seconds.

Table 94, Meter Report v2::Delta Time encoding

Delta Time	Value
No Previous Meter Value field included in the Meter Report	0x0000
1 sec. between Meter Value and Previous Meter Value measurements	0x0001
…	…
65534 sec. between Meter Value and Previous Meter Value measurements	0xFFFE
Unknown time between Meter Value and Previous Meter Value measurements	0xFFFF

If the Delta Time is 0, the Previous Meter Value field MUST be omitted.

Previous Meter Value (N bytes)

The Previous Meter Value MUST be omitted if the Delta Time field is set to the value 0.
The Previous Meter Value MUST be included if the Delta Time field is set to a non-zero value.

The Previous Meter Value field MUST have the same size and format as the Meter Value field. Refer to the description of the Meter Value field.

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

4.75.5.1 Examples

To retrieve accumulated power consumption from a device supporting Meter Command Class (Version 2), the Meter Get Command has to indicate what scale is requested.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">0x32 (COMMAND_CLASS_METER)</td></tr>
<tr><td colspan="8">0x01 (METER_GET)</td></tr>
<tr><td colspan="3">0</td><td colspan="2">0 (kWh)</td><td colspan="3">0</td></tr>
</table>

As a response to the above Meter Get Command an example of accumulated power consumption can be reported with the below Meter Report indicating a current measurement of 12065.298 kWh and 10 minutes ago the previous measurement was 12060.678 kWh.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">0x32 (COMMAND_CLASS_METER)</td></tr>
<tr><td colspan="8">0x02 (METER_REPORT)</td></tr>
<tr><td>0</td><td colspan="2">2 (export)</td><td colspan="5">0 (Electric meter)</td></tr>
<tr><td colspan="3">3 (3 decimals)</td><td colspan="2">0 (kWh)</td><td colspan="3">4 (4 bytes)</td></tr>
<tr><td colspan="8">0x00 (Meter Value 1)</td></tr>
<tr><td colspan="8">0xB8 (Meter Value 2)</td></tr>
<tr><td colspan="8">0x1A (Meter Value 3)</td></tr>
<tr><td colspan="8">0x12 (Meter Value 4)</td></tr>
<tr><td colspan="8">0x02 (Delta Time 1)</td></tr>
<tr><td colspan="8">0x58 (Delta Time 2)</td></tr>
<tr><td colspan="8">0x00 (Previous Meter Value 1)</td></tr>
<tr><td colspan="8">0xB8 (Previous Meter Value 2)</td></tr>
<tr><td colspan="8">0x08 (Previous Meter Value 3)</td></tr>
<tr><td colspan="8">0x06 (Previous Meter Value 4)</td></tr>
</table>

To retrieve instant power consumption from a device supporting Meter Command Class (Version 2), the Meter Get Command MUST indicate what scale is requested.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">0x32 (COMMAND_CLASS_METER)</td></tr>
<tr><td colspan="8">0x01 (METER_GET)</td></tr>
<tr><td colspan="3">0</td><td colspan="2">2 (W)</td><td colspan="3">0</td></tr>
</table>

An example of instant power consumption can be reported with the below Meter Report indicating a current measurement of 39.99 W.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">0x32 (COMMAND_CLASS_METER)</td></tr>
<tr><td colspan="8">0x02 (METER_REPORT)</td></tr>
<tr><td>0</td><td colspan="2">2 (export)</td><td colspan="5">0 (Electric meter)</td></tr>
<tr><td colspan="3">2 (2 decimals)</td><td colspan="2">2 (W)</td><td colspan="3">2 (2 bytes)</td></tr>
<tr><td colspan="8">0x0F (Meter Value 1)</td></tr>
<tr><td colspan="8">0x9F (Meter Value 2)</td></tr>
<tr><td colspan="8">0x00 (Delta Time 1)</td></tr>
<tr><td colspan="8">0x00 (Delta Time 2)</td></tr>
</table>

4.76 Meter Command Class, version 3

The Meter Command Class is intended for Z-Wave enabled devices capable of reporting energy measurements in addition to any main functionality or features e.g. an appliance module reporting the current consumption of the connected load. This command class is not intended for residential utility sub-metering such as a water meter counting total consumption.

Meter Command Class (Version 3) is improved with the following functionalities:

- ‘Scale’ entry has been enlarged to support 8 scales

The commands not mentioned are the same as specified for Meter Command Class (Version 2).

4.76.1 Meter Supported Report Command

This command is used to advertise supported scales in a sub meter.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_METER</td></tr>
<tr><td colspan="8">Command = METER_SUPPORTED_REPORT</td></tr>
<tr><td>Meter Reset</td><td colspan="2">Reserved</td><td colspan="5">Meter Type</td></tr>
<tr><td colspan="8">Scale Supported</td></tr>
</table>

Meter Reset (1 bit)

The Meter Reset field set to “1” indicates support for the Meter Reset Command.

Meter Type (5 bits)

See Meter Type defined in Meter Report Command.

Scale Supported (8 bits)

This field MUST advertise the supported scales of the device. The field MUST be formatted and interpreted as a bitmask.
The bit value ‘1’ MUST indicate support for the actual scale. The bit value ‘0’ MUST indicate that there is no support for the actual scale.

The Scale Supported bitmask MUST be interpreted in combination with the Meter Type.

Table 95, Meter Supported Report v3::Scale Supported encoding

Meter Type	Scale	Scale Supported Bit position	Measurement mode
Electric meter	kWh	Bit 0	Accumulated
	kVAh	Bit 1	Accumulated
	W	Bit 2	Instant
	Pulse count	Bit 3	Accumulated

	V	Bit 4	Instant
	A	Bit 5	Instant
	Power factor	Bit 6	Instant
	reserved	Bit 7	reserved
Gas meter	Cubic meters	Bit 0	Accumulated
	Cubic feet	Bit 1	Accumulated
	Reserved	Bit 2	n/a
	Pulse count	Bit 3	Accumulated
	Reserved	Bit 4-7	reserved
Water meter	Cubic meters	Bit 0	Accumulated
	Cubic feet	Bit 1	Accumulated
	US gallons	Bit 2	Accumulated
	Pulse count	Bit 3	Accumulated
	Reserved	Bit 4-7	reserved

Reserved values MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

4.76.2 Meter Get Command

The Meter Get Command is used to request a device supported measurement in physical units from a metering device.

The Meter Report Command MUST be returned in response to this command if the requested scale is supported.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER							
Command = METER_GET							
Reserved		Scale			Reserved		

Scale (3 bits)

See description of Meter Report Command.

If the Scale is not present (v1) or set to 0x00, a receiving node MUST return a Report using its default scale.

A Meter Get Command containing a non-supported Scale MUST be ignored by a receiving device.

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

4.76.2.1 Backwards compatibility

The manufacturer MUST define which scale is the default scale and it MUST be described in the product manual. The default scale MUST be in the range 0..3 as Version 1 and 2 devices will not be able to interpret Scale values in the range 4..7.

4.76.3 Meter Report Command

The Meter Report Command is used to advertise a meter reading.

The Meter Report Command MUST NOT be transmitted using broadcast.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_METER</td></tr>
<tr><td colspan="8">Command = METER_REPORT</td></tr>
<tr><td>Scale (2)</td><td colspan="2">Rate Type</td><td colspan="5">Meter Type</td></tr>
<tr><td colspan="3">Precision</td><td colspan="2">Scale (1:0)</td><td colspan="3">Size</td></tr>
<tr><td colspan="8">Meter Value 1</td></tr>
<tr><td colspan="8">…</td></tr>
<tr><td colspan="8">Meter Value N</td></tr>
<tr><td colspan="8">Delta Time 1</td></tr>
<tr><td colspan="8">Delta Time 2</td></tr>
<tr><td colspan="8">Previous Meter Value 1</td></tr>
<tr><td colspan="8">…</td></tr>
<tr><td colspan="8">Previous Meter Value N</td></tr>
</table>

Rate Type (2 bits)

Rate Type specifies if it is import or export values to be read. Setting the Rate Type to Import on the Meter Report Command is an indication that the Meter Value is a consumed measurement. In contrary when the Rate Type is set to Export the indication of the Meter Value is a produced measurement.

Table 96, Meter Report v3::Rate Type encoding

Value	Rate Type
0x00	Reserved
0x01	Import (consumed)
0x02	Export (produced)
0x03	Reserved

Reserved values MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Meter Type (5 bits)

Meter Type specifies what type of metering device this command originates from. Refer to the table below with respect to defined metering devices.

Table 97, Meter Report v3::Meter Type encoding

Value	Meter Type
0x00	Reserved
0x01	Electric meter
0x02	Gas meter
0x03	Water meter
0x04-0x1F	Reserved

Reserved values MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Precision (3 bits)

The Precision field describes what the precision of the value is. The number indicates the number of decimals. The decimal value 1025 with precision 1 MUST be interpreted as 102.5 and if precision is 3 then the interpreted value is 1.025.

The value of the precision field MUST be in the range 0..7.

Scale (3 bits)

The Scale field field MUST advertise the unit used for this report. The Scale field is composed of two sub-fields Scale (2) and Scale (1:0) which MUST be composed and interpreted as one unit.

Refer to the table below with respect to defined scales for the relevant metering devices.

Table 98, Meter Report v3::Scale encoding

Meter Type	Scale	Value
Electric meter	kWh	0x00
	kVAh	0x01
	W	0x02
	Pulse count	0x03
	V	0x04
	A	0x05
	Power Factor	0x06
	reserved	0x07
Gas meter	Cubic meters	0x00
	Cubic feet	0x01
	Reserved	0x02
	Pulse count	0x03
	reserved	0x04- 0x07
Water meter	Cubic meters	0x00
	Cubic feet	0x01
	US gallons	0x02
	Pulse count	0x03
	reserved	0x04-0x07

Reserved values MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Size (3 bits)

The Size field MUST advertise the number of bytes that is used for the Meter Value. The value of the Size field MUST be in the range (1,2,4). All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Meter Value (N bytes)

The Meter Value field is a signed field. The Meter Value field MUST have the size advertised by the Size field. The first byte MUST be the most significant byte.

This field MUST comply with Table 93.

Delta Time (16 bits)

The Delta Time field MUST advertise the elapsed time in seconds between the ‘Meter Value’ and the ‘Previous Meter Value’ measurements.

This field MUST comply with Table 94

If the Delta Time is 0, the Previous Meter Value field MUST be omitted.

Previous Meter Value (N bytes)

The Previous Meter Value MUST be omitted if the Delta Time field is set to the value 0.
The Previous Meter Value MUST be included if the Delta Time field is set to a non-zero value.

The Previous Meter Value field MUST have the same size and format as the Meter Value field. Refer to the description of the Meter Value field.

4.76.3.1 Examples of Meter Report Commands

To retrieve Instant Voltage from a device supporting Meter Command Class (Version 3), the Meter Get Command has to indicate what scale is requested.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">0x32 (COMMAND_CLASS_METER)</td></tr>
<tr><td colspan="8">0x01 (METER_GET)</td></tr>
<tr><td colspan="2">0</td><td colspan="3">4 (V)</td><td colspan="3">0</td></tr>
</table>

As a response to the above Meter Get Command an example of instant Voltage can be reported with the below Meter Report indicating a current measurement of 108.5 V and 10 minutes ago the previous measurement was 109.1V.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">0x32 (COMMAND_CLASS_METER)</td></tr>
<tr><td colspan="8">0x02 (METER_REPORT)</td></tr>
<tr><td>1 (V)</td><td colspan="2">2 (export)</td><td colspan="5">0 (Electric meter)</td></tr>
<tr><td colspan="3">1 (1 decimals)</td><td colspan="2">0 (V)</td><td colspan="3">4 (4 bytes)</td></tr>
<tr><td colspan="8">0x00 (Meter Value 1)</td></tr>
<tr><td colspan="8">0x00 (Meter Value 2)</td></tr>
<tr><td colspan="8">0x04 (Meter Value 3)</td></tr>
<tr><td colspan="8">0x3D (Meter Value 4)</td></tr>
<tr><td colspan="8">0x02 (Delta Time 1)</td></tr>
<tr><td colspan="8">0x58 (Delta Time 2)</td></tr>
<tr><td colspan="8">0x00 (Previous Meter Value 1)</td></tr>
<tr><td colspan="8">0x00 (Previous Meter Value 2)</td></tr>
<tr><td colspan="8">0x04 (Previous Meter Value 3)</td></tr>
<tr><td colspan="8">0x43 (Previous Meter Value 4)</td></tr>
</table>

To retrieve instant power consumption from a device supporting Meter Command Class (Version 2), the Meter Get Command has to indicate what scale is requested.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">0x32 (COMMAND_CLASS_METER)</td></tr>
<tr><td colspan="8">0x01 (METER_GET)</td></tr>
<tr><td colspan="3">0</td><td colspan="2">2 (W)</td><td colspan="3">0</td></tr>
</table>

An example of instant power consumption can be reported with the below Meter Report indicating a current measurement of 39.99 W.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">0x32 (COMMAND_CLASS_METER)</td></tr>
<tr><td colspan="8">0x02 (METER_REPORT)</td></tr>
<tr><td>0</td><td colspan="2">2 (export)</td><td colspan="5">0 (Electric meter)</td></tr>
<tr><td colspan="3">2 (2 decimals)</td><td colspan="2">2 (W)</td><td colspan="3">2 (2 bytes)</td></tr>
<tr><td colspan="8">0x0F (Meter Value 1)</td></tr>
<tr><td colspan="8">0x9F (Meter Value 2)</td></tr>
<tr><td colspan="8">0x00 (Delta Time 1)</td></tr>
<tr><td colspan="8">0x00 (Delta Time 2)</td></tr>
</table>

4.77 Meter Command Class, version 4

The Meter Command Class is intended for Z-Wave enabled devices capable of reporting energy measurements in addition to any main functionality or features e.g. an appliance module reporting the current consumption of the connected load. This command class is not intended for residential utility metering such as a water meter counting total consumption.

Meter Command Class, Version 4 is improved with the following functionalities:

- The Rate Type is included in the Meter Supported Report and the Meter Get Command
- Added support for Meter Scale Types kVar and kVarh

The commands not mentioned are the same as specified for Meter Command Class, Version 3.

4.77.1 Meter Supported Report Command

This command is used to advertise supported capabilities of a sub meter.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_METER</td></tr>
<tr><td colspan="8">Command = METER_SUPPORTED_REPORT</td></tr>
<tr><td>Meter Reset</td><td colspan="2">Rate Type</td><td colspan="5">Meter Type</td></tr>
<tr><td>M.S.T</td><td colspan="7">Scale Supported “Byte 1”</td></tr>
<tr><td colspan="8">Number of Scale Supported Bytes to Follow (Version 4 Extension)</td></tr>
<tr><td colspan="8">Scale Supported 1 “Byte 2”</td></tr>
<tr><td colspan="8">…</td></tr>
<tr><td colspan="8">Scale Supported N “Byte N+1”</td></tr>
</table>

Meter Reset (1 bit)

The Meter Reset field set to “1” indicates support for the Meter Reset Command.

Rate Type (2 bits)

Rate Type specifies if import and export consumption is supported. Import is an indication that the Meter Value is a consumed measurement and Export is an indication that the Meter Value is a produced measurement.

Table 99, Meter Supported Report::Rate Type encoding

Value	Rate Type
0x00	Reserved
0x01	Import only (consumed)
0x02	Export only (produced)
0x03	Both Import and Export

Meter Type (5 bits)

See Meter Type defined in Meter Report Command.

Scale Supported (7 bits)

This field MUST advertise the supported scales of the device. The field MUST be formatted and interpreted as a bitmask.
The bit value ‘1’ MUST indicate support for the actual scale. The bit value ‘0’ MUST indicate that there is no support for the actual scale.

The Scale Supported bitmask MUST be interpreted in combination with the Meter Type.

Meter Type	Scale	Scale Supported Byte number	Scale Supported Bit position	Measurement mode
Electric meter	kWh	Byte 1	Bit 0	Accumulated
	kVAh	Byte 1	Bit 1	Accumulated
	W	Byte 1	Bit 2	Instant
	Pulse count	Byte 1	Bit 3	Accumulated
	V	Byte 1	Bit 4	Instant
	A	Byte 1	Bit 5	Instant
	Power factor	Byte 1	Bit 6	Instant
	kVar	Byte 2	Bit 0	Instant
	kVarh	Byte 2	Bit 1	Accumulated
	Reserved	Byte 2	Bit 2-7	Reserved
Gas meter	Cubic meters	Byte 1	Bit 0	Accumulated
	Cubic feet	Byte 1	Bit 1	Accumulated
	Reserved	Byte 1	Bit 2	n/a
	Pulse count	Byte 1	Bit 3	Accumulated
	Reserved	Byte 1	Bit 4-6	Reserved
	Reserved	Byte 2	Bit 0-7	Reserved
Water meter	Cubic meters	Byte 1	Bit 0	Accumulated
	Cubic feet	Byte 1	Bit 1	Accumulated
	US gallons	Byte 1	Bit 2	Accumulated
	Pulse count	Byte 1	Bit 3	Accumulated
	Reserved	Byte 1	Bit 4-6	Reserved
	Reserved	Byte 2	Bit 0-7	Reserved

M.S.T. (More Scale Types) (1 bit)

This bit MUST be set to ‘1’ when more than one Scale Supported byte is carried in the Meter Supported Report. In this case the Number of Scale Supported Bytes to Follow field MUST advertise the total number of Scale Supported Bytes. The bitmask of the following Scale Supported Bytes is described in the table of Scale Supported.

The M.S.T bit MUST be set to ‘0’ if the Meter Supported Report only carries the Scale Supported “Byte 1” field. In this case the report MUST NOT include the “Number of Scale Supported Bytes to Follow” field or any “Scale Supported Bytes 1 – N” fields.

Number of Scale Supported to Follow (8 bits)

The number of Scale Supported Bytes to follow.

Scale Supported (bytes 2-N+1) (N bytes)

These fields MUST advertise the supported scales of the device not covered by byte 1. The fields MUST be formatted and interpreted as bitmasks.
The bit value ‘1’ MUST indicate support for the actual scale. The bit value ‘0’ MUST indicate that there is no support for the actual scale.

The Scale Supported bitmask MUST be interpreted in combination with the Meter Type.

See table in Scale Supported for more information.

4.77.2 Meter Get Command

The Meter Get Command is used to request a device supported measurement in physical units from a metering device.

The Meter Report Command MUST be returned in response to this command if the requested scale and rate type are supported.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing.
The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_METER</td></tr>
<tr><td colspan="8">Command = METER_GET</td></tr>
<tr><td colspan="2">Rate Type</td><td colspan="3">Scale</td><td colspan="3">Reserved</td></tr>
<tr><td colspan="8">Scale 2</td></tr>
</table>

Rate Type (2 bits)

Rate Type specifies if it is import or export values to be requested. Import is an indication that the Meter Value is a consumed measurement and Export is an indication that the Meter Value is a produced measurement. If both Import and Export is needed, two Meter Get Commands must be send.

Table 100, Meter Get v4::Rate Type encoding

Value	Rate Type
0x00	Reserved
0x01	Import (consumed)
0x02	Export (produced)
0x03	Not to be used

Reserved values MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

If the Rate Type is not present (v1-v3) or set to 0x00, a receiving node MUST return a Report using its default Rate Type.

Scale (3 bits)

See description of Meter Report Command. This field MUST specify the scale of the requested meter report. The Scale Value 7 MUST be used to indicate that the scale of the requested meter reading is advertised by the Scale 2 field.

For valid Scale values, refer to the Meter Report Command V4.

If the Scale is not present (v1) or set to 0x00, a receiving node MUST return a Report using its default scale.

A Meter Get Command containing a non-supported Scale MUST be ignored by a receiving device.

Scale 2 (8 bits)

The Scale 2 field MUST be appended to the Meter Get Command if the Scale field is set to 7 (M.S.T). The Scale 2 field MUST NOT be appended to the Meter Get Command if the Scale field carries any other value then 7.

For valid Scale 2 values, refer to the Meter Report Command V4.

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

4.77.2.1 Backwards compatibility

The manufacturer MUST define which scale and rate type are the default and it MUST be described in the product manual. The default scale MUST be in the range 0..3 as Version 1 and 2 devices will not be able to interpret Scale values in the range 4..7.

4.77.3 Meter Report Command

The Meter Report Command is used to advertise a meter reading.

The Meter Report Command MUST NOT be transmitted using broadcast.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_METER</td></tr>
<tr><td colspan="8">Command = METER_REPORT</td></tr>
<tr><td>Scale (2)</td><td colspan="2">Rate Type</td><td colspan="5">Meter Type</td></tr>
<tr><td colspan="3">Precision</td><td colspan="2">Scale (1:0)</td><td colspan="3">Size</td></tr>
<tr><td colspan="8">Meter Value 1</td></tr>
<tr><td colspan="8">…</td></tr>
<tr><td colspan="8">Meter Value N</td></tr>
<tr><td colspan="8">Delta Time 1</td></tr>
<tr><td colspan="8">Delta Time 2</td></tr>
<tr><td colspan="8">Previous Meter Value 1</td></tr>
<tr><td colspan="8">…</td></tr>
<tr><td colspan="8">Previous Meter Value N</td></tr>
<tr><td colspan="8">Scale 2</td></tr>
</table>

Rate Type (2 bits)

Rate Type specifies if it is import or export values to be read. Setting the Rate Type to Import on the Meter Report Command is an indication that the Meter Value is a consumed measurement. In contrary when the Rate Type is set to Export the indication of the Meter Value is a produced measurement.

Table 101, Meter Report v4::Rate Type encoding

Value	Rate Type
0x00	Reserved
0x01	Import (consumed)
0x02	Export (produced)
0x03	Not to be used

Reserved values MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Meter Type (5 bits)

Meter Type specifies what type of metering device this command originates from. Refer to the table below with respect to defined metering devices. Metering device types are defined by the Z-Wave Alliance.

Table 102, Meter Report v4::Meter Type encoding

Value	Meter Type
0x00	Reserved
0x01	Electric meter
0x02	Gas meter
0x03	Water meter

Reserved values MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Precision (3 bits)

The Precision field describes what the precision of the value is. The number indicates the number of decimals. The decimal value 1025 with precision 1 must be interpreted as 102.5 and if precision is 3 then the interpreted value is 1.025.

The value of the precision field MUST be in the range 0..7.

Scale (3 bits)

The Scale field MUST advertise the unit used for this report. The Scale field is composed of two sub-fields Scale (2) and Scale (1:0) which MUST be composed and interpreted as one unit.

Refer to the table below with respect to defined scales for the relevant metering devices.

The Scale Value 7 MUST be used to indicate that the scale of the advertised meter reading is advertised through the Scale 2 field.

Table 103, Meter Report v4::Scale encoding

Meter Type	Scale	Value
Electric meter	kWh	0x00
	kVAh	0x01
	W	0x02
	Pulse count	0x03
	V	0x04
	A	0x05
	Power Factor	0x06
	M.S.T	0x07
Gas meter	Cubic meters	0x00
	Cubic feet	0x01
	Reserved	0x02
	Pulse count	0x03
	Reserved	0x04- 0x06

	M.S.T	0x07
Water meter	Cubic meters	0x00
	Cubic feet	0x01
	US gallons	0x02
	Pulse count	0x03
	Reserved	0x04-0x06
	M.S.T	0x07

Size (3 bits)

The Size field MUST advertise the number of bytes that is used for the Meter Value. The value of the Size field MUST be in the range (1,2,4). All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Meter Value (N bytes)

The Meter Value field is a signed field. The Meter Value field MUST have the size advertised by the Size field. The first byte is the most significant byte.

This field MUST comply with Table 93.

Delta Time (16 bits)

The Delta Time field MUST advertise the elapsed time in seconds between the ‘Meter Value’ and the ‘Previous Meter Value’ measurements.

This field MUST comply with Table 94

If the Delta Time is 0, the Previous Meter Value field MUST be omitted.

Previous Meter Value (N bytes)

The Previous Meter Value MUST be omitted if the Delta Time field is set to the value 0.
The Previous Meter Value MUST be included if the Delta Time field is set to a non-zero value.

The Previous Meter Value field MUST have the same size and format as the Meter Value field. Refer to the description of the Meter Value field.

Scale 2 (8 bits)

The Scale 2 field MUST be appended to the Meter Report Command if the Scale field is set to 7 (M.S.T).
The Scale 2 field MUST NOT be appended to the Meter Report Command if the Scale field carries any other value then 7.

The below table shows valid Scale 2 values.

Table 104, Meter Report v4::Scale 2 encoding

Meter Type	Scale	Value
Electric meter	kVar	0x00
	kVarh	0x01
	Reserved	0x02 – 0xFF
Gas meter	Reserved	0x00 – 0xFF
Water meter	Reserved	0x00 – 0xFF

Reserved values MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

4.78 Meter Table Configuration Command Class, version 1

The Meter Table Configuration Command Class defines the Commands necessary to configure the fundamental properties of the meter.

The Meter Table configuration commands are separated from the Meter Table monitoring commands in the Meter Table Monitor Command Class, allowing the classes to be optionally supported at different Z-Wave security levels. (E.g. Meter table monitoring commands could be supported in any device, while enabling a strict and certificate based security solution for the Meter Table Configuration Command class). Please refer to the Hybrid Security Command class for more details regarding Z-Wave security levels.

4.78.1 Meter Table Point Adm Number Set Command

The Meter Table Point Adm Number Set Command is used to set the Meter Point Administration Number in the metering device. The Meter Point Administration Number is used to identify the customer.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_METER_TBL_CONFIG</td></tr>
<tr><td colspan="8">Command = METER_TBL_TABLE_POINT_ADM_NO_SET</td></tr>
<tr><td colspan="3">Reserved</td><td colspan="5">Number of Meter Point Adm Number Characters</td></tr>
<tr><td colspan="8">Meter Point Adm Number Character 1</td></tr>
<tr><td colspan="8">…</td></tr>
<tr><td colspan="8">Meter Point Adm Number Character N</td></tr>
</table>

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Number of Meter Point Adm Number Characters (5 bits)

Number of characters in the meter point administration number(1…32).

Meter Point Adm Number Character (N bytes)

The Meter Point Adm Number character fields hold the string identifying the customer. The character presentation uses standard ASCII codes (values 128-255 are ignored).

4.79 Meter Table Monitor Command Class, version 1

The Meter Table Monitor Command Class defines the Commands necessary to read historical and accumulated values in physical units from a water meter or other metering device (gas, electric etc.) and thereby enabling automatic meter reading capabilities

4.79.1 Meter Table Point Adm. Number Get Command

The Meter Table Point Adm. Number Get Command is used to request the Meter Point Administration Number to identify customer.

The Meter Table Adm. Number Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER_TBL_MONITOR							
Command = METER_TBL_TABLE_POINT_ADM_NO_GET							

4.79.2 Meter Table Point Adm. Number Report Command

The Meter Table Point Adm. Number Report Command reports parameters used for identification of customer and metering device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER_TBL_MONITOR							
Command = METER_TBL_TABLE_POINT_ADM_NO_REPORT							
Reserved			Number of Meter Point Adm. Number Characters				
Meter Point Adm. Number Character 1							
…							
Meter Point Adm. Number Character N							

Refer to description of fields under the Meter Table Point Adm. Number Set Command (section 4.78.1).

4.79.3 Meter Table ID Get Command

The Meter Table ID Get Command is used to request the parameters used for identification of customer and metering device.

The Meter Table ID Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER_TBL_MONITOR							
Command = METER_TBL_TABLE_ID_GET							

4.79.4 Meter Table ID Report Command

The Meter Table ID Report Command reports parameters used for identification of customer and metering device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER_TBL_MONITOR							
Command = METER_TBL_TABLE_ID_REPORT							
Reserved			Number of Meter ID Characters				
Meter ID Character 1							
…							
Meter ID Character N							

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Number of Meter ID Characters (5 bits)

Number of characters defining the meter ID (1..32).

Meter ID Character (N bytes)

The Meter ID character fields hold the string identifying the individual metering device. The character presentation uses standard ASCII codes (values 128-255 are ignored). In addition one can use the Manufacturer Specific Command Class in conjunction as product identification.

4.79.5 Meter Table Capability Get Command

The Meter Table Capability Get Command is used to request the capabilities of a metering device.

The Meter Table Capability Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER_TBL_MONITOR							
Command = METER_TBL_TABLE_CAPABILITY_GET							

4.79.6 Meter Table Capability Report Command

This command is used to advertise meter table capabilities.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER_TBL_MONITOR							
Command = METER_TBL_REPORT							
Rate Type		Meter Type					
Reserved				Pay Meter			
Dataset Supported 1							
Dataset Supported 2							
Dataset Supported 3							
Dataset History Supported 1							
Dataset History Supported 2							
Dataset History Supported 3							
Data History Supported 1							
Data History Supported 2							
Data History Supported 3							

Rate Type (2 bits)

Rate Type specifies the type of parameters in the report. Rate Type defined as the *Meter Rate Type* variable; refer to section 3.5.3 for a definition of the variable.

Meter Type (6 bits)

Meter Type specifies the type of metering device the command originates. Meter Type defined as the *Meter Type* variable; refer to section 3.5.5 for a definition of the variable.

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Pay Meter (4 bits)

Pay Meter specifies the way settling of account is done. Refer to the table below with respect to defined payment meters.

Table 105, Meter Table Capability Report::Pay Meter encoding

Value	Pay Meter
0x00	Reserved
0x01	Creditmeter
0x02	Prepayment meter
0x03	Prepayment meter with debt recovery

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Dataset Supported / Dataset History Supported (24 bits / 24 bits)

Dataset Supported specifies which parameters, are available to be requested from the metering device. Dataset Supported Parameters defined as the *Meter Dataset* variable; refer to section 3.5.2 for a definition of the variable.

Dataset History Supported specifies the type of data/values, which may be requested through the Meter Table Historical data Get command (see 4.79.14) from the metering device. Dataset History Supported Parameters defined as the *Meter Dataset* variable; refer to section 3.5.2 for a definition of the variable.

Data History Supported (24 bits)

Data History Supported specifies number of possible entries in the buffer holding the historical values. Historical data cannot be retrieved when Data History Supported is equal to 0.

4.79.7 Meter Table Status Supported Get Command

The Meter Table Status Supported Get Command is used to request the supported operating status event parameters and logging depth of the metering device.

The Meter Table Status Supported Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER_TBL_MONITOR							
Command = METER_TBL _STATUS_SUPPORTED_GET							

4.79.8 Meter Table Status Supported Report Command

The Meter Table Status Report Command is used to report the supported operation status' and logging depth of these in the meter.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER_TBL_MONITOR							
Command = METER_TBL _STATUS_SUPPORTED_REPORT							
Supported Operating Status 1							
Supported Operating Status 2							
Supported Operating Status 3							
Status Event Log Depth							

Supported Operating Status (24 bits)

This command parameter is defined in a bitmap format and holds values and combinations of the meters operating status. If no "Operating Status Event" is supported by the application all fields of this parameter MUST be set to '0' as an indication of the meter is at "Normal/Idle" operating status. Supported Operating Status supports the following combinations of operating status:

Table 106, Meter Table Status Supported Report::Supported Operating Status encoding

Operating Status	Bit Map	Operating Status Event Identifier	Description
1	Bit 0	0x00	Disconnected The meter has been disconnected.
1	Bit 1	0x01	Leak detection A leak has been detected.
1	Bit 2	0x02	Power failure All phases / meter without power.
1	Bit 3	0x03	Power failure – Phase 1 Power failure on phase 1 detected.
1	Bit 4	0x04	Power failure – Phase 2 Power failure on phase 2 detected.
1	Bit 5	0x05	Power failure – Phase 3 Power failure on phase 3 detected.
1	Bit 6	0x06	Voltage Quality Voltage quality not within limits.
1	Bit 7	0x07	Tamper detection Tamper switch has been activated.
2	Bit 0	0x08	Magnetic tampering Magnetic tempering detected.
2	Bit 1	0x09	Meter Clock Set The clock has been set in the meter.
2	Bit 2	0x0A	Meter Clock Adjusted

			The clock has been adjusted in the meter.
2	Bit 3	0x0B	Out of Credit The meter is out of credit.
2	Bit 4	0x0C	Emergency credit The meter is running on emergency credit.
2	Bit 5	0x0D	Transformer Ratio Changed “Current transformation ratio” in meter changed.
2	Bit 6	0x0E	Temperature Sensor 1 – Out of range Temperature Sensor 1 outside specifications.
2	Bit 7	0x0F	Temperature Sensor 2 – Out of range Temperature Sensor 2 outside specifications.
3	Bit 0	0x10	Temperature Sensor 3 – Out of range Temperature Sensor 3 outside specifications.
3	Bit 1	0x11	Burst Detection Heat Burst detected in the heating network causing potential flooding.
3	Bit 2	0x12	Error – Check Meter Any other error condition not mentioned above.
3	Bit 4 – 7	0x13 – 0x1F	Reserved Values reserved for future expansion.

Status Event Log Depth (8 bits)

The *status event log depth* indicates the supported depth of the event log. If the meter only supports reporting the current status the status event log depth MUST be set to 0.

4.79.9 Meter Table Status Depth Get Command

The Meter Table Status Depth Get Command is used to request the current operating status of the metering device or to request a number of the latest status event from the logs.

The Meter Table Status Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER_TBL_MONITOR							
Command = METER_TBL _STATUS_DEPTH_GET							
Status Event Log Depth							

Status Event Log Depth (8 bits)

The *status event log depth* indicates the number of latest recorded events that should be returned in the corresponding report. If the log depth is set to 0 the meter will only return the current status; if the log depth is set to 0xFF the meter SHOULD return the entire status log.

4.79.10 Meter Table Status Date Get Command

The Meter Table Status Date Get Command is used to request a number of status events recorded in a certain time interval.

The Meter Table Status Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

If the meter does not support a status event log history it MUST return the current state of the meter (see Meter Table Status Report Command).

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER_TBL_MONITOR							
Command = METER_TBL _STATUS_DATE_GET							
Maximum Reports							
Start - Year 1							
Start - Year 2							
Start - Month							
Start - Day							
Start - Hour Local Time							
Start - Minute Local Time							
Start - Second Local Time							
Stop - Year 1							
Stop - Year 2							
Stop - Month							
Stop - Day							
Stop - Hour Local Time							
Stop - Minute Local Time							
Stop - Second Local Time							

Maximum Reports (8 bits)

The maximum reports parameter is used to indicate the maximum number of reports to return based on the get. Reports are always returned with the most recently recorded log entry first. If set to 0x00 the meter will return all reports based on the request.

Start/Stop - Year (16 bits)

Specify the year in the usual Gregorian calendar. The first byte (Year 1) is the most significant byte.

Start/Stop - Month (8 bits)

Specify the month of the year between 01 (January) and 12 (December).

Start/Stop - Day (8 bits)

Specify the day of the month between 01 and 31.

Start/Stop - Hour Local Time (8 bits)

Specify the number of complete hours that have passed since midnight (00-23) in local time.

Start/Stop - Minute Local Time (8 bits)

Specify the number of complete minutes that have passed since the start of the hour (00-59) in local time.

Start/Stop - Second Local Time (8 bits)

Specify the number of complete seconds since the start of the minute (00-59) in local time. The value 60 used to keep UTC from wandering away is not supported.

4.79.11 Meter Table Status Report Command

This command is used to advertise the current status of the meter.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_METER_TBL_MONITOR</td></tr>
<tr><td colspan="8">Command = METER_TBL _STATUS _REPORT</td></tr>
<tr><td colspan="8">Reports to follow</td></tr>
<tr><td colspan="8">Current Operating Status 1</td></tr>
<tr><td colspan="8">Current Operating Status 2</td></tr>
<tr><td colspan="8">Current Operating Status 3</td></tr>
<tr><td>Event 1 - Type</td><td colspan="2">Reserved</td><td colspan="5">Event 1 - Operating Status Event ID</td></tr>
<tr><td colspan="8">Event 1 - Year 1</td></tr>
<tr><td colspan="8">Event 1 - Year 2</td></tr>
<tr><td colspan="8">Event 1 - Month</td></tr>
<tr><td colspan="8">Event 1 - Day</td></tr>
<tr><td colspan="8">Event 1 - Hour Local Time</td></tr>
<tr><td colspan="8">Event 1 - Minute Local Time</td></tr>
<tr><td colspan="8">Event 1 - Second Local Time</td></tr>
<tr><td colspan="8">…</td></tr>
<tr><td>Event N -Type</td><td colspan="2">Reserved</td><td colspan="5">Event N - Operating Status Event ID</td></tr>
<tr><td colspan="8">Event N - Year 1</td></tr>
<tr><td colspan="8">Event N - Year 2</td></tr>
<tr><td colspan="8">Event N - Month</td></tr>
<tr><td colspan="8">Event N - Day</td></tr>
<tr><td colspan="8">Event N - Hour Local Time</td></tr>
<tr><td colspan="8">Event N -Minute Local Time</td></tr>
<tr><td colspan="8">Event N - Second Local Time</td></tr>
</table>

Reports to follow (8 bits)

This value indicates how many report frames there are left, the value 0xFF means that the number of reports have not been calculated yet or that there is more than 255 reports to follow.

Current Operating Status (24 bits)

This command parameter is defined in a bitmap format and holds values and combinations of the meters operating status. If no “Operating Status Event” has been registered for the period requested by the Meter Table Status Date Get Command, all fields of this parameter MUST be set to ‘0’ AND no “Event” parameters are to be included in the Meter Table Status Report Command i.e. the command parameters ends after "Current Operating Status" . Refer to "Supported Operating Status" in section 4.79.8 for details.

Event – Type (1 bit)

The parameter reports type of event number *n* in the event log. The event type MAY be 0 (the meter entered the state described by the operating event status ID at the time reported), or 1 (the meter left the state described by the operating event status ID).

Event – Operating Status Event ID (5 bits)

The operating status event identifier of event number *n* in the event log. The identifiers available are given in the table in section 4.79.8.

Event - Year (16 bits)

Specify the year in the usual Gregorian calendar. The first byte (Year 1) is the most significant byte.

Event - Month (8 bits)

Specify the month of the year between 01 (January) and 12 (December).

Event - Day (8 bits)

Specify the day of the month between 01 and 31.

Event - Hour Local Time (8 bits)

Specify the number of complete hours that have passed since midnight (00-23) in local time.

Event - Minute Local Time (8 bits)

Specify the number of complete minutes that have passed since the start of the hour (00-59) in local time.

Event - Second Local Time (8 bits)

Specify the number of complete seconds since the start of the minute (00-59) in local time. The value 60 used to keep UTC from wandering away is not supported.

4.79.12 Meter Table Current Data Get Command

The Meter Table Current Data Get Command is used to request a number of time stamped values (current) in physical units according to the dataset mask.

The Meter Table Current Data Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER_TBL_MONITOR							
Command = METER_TBL_CURRENT_DATA_GET							
Dataset Requested 1							
Dataset Requested 2							
Dataset Requested 3							

Dataset Requested (24 bits)

The Dataset Requested is used to indicate which parameters are requested from the meter. The Dataset Requested parameter defined as the *Meter Dataset* variable; refer to section 3.5.2 for a definition of the variable.

4.79.13 Meter Table Current Data Report Command

This command is used to report a number of time stamped values.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_METER_TBL_MONITOR</td></tr>
<tr><td colspan="8">Command = METER_TBL_CURRENT_DATA_REPORT</td></tr>
<tr><td colspan="8">Reports to Follow</td></tr>
<tr><td colspan="6">Reserved</td><td colspan="2">Rate Type</td></tr>
<tr><td colspan="8">Dataset 1</td></tr>
<tr><td colspan="8">Dataset 2</td></tr>
<tr><td colspan="8">Dataset 3</td></tr>
<tr><td colspan="8">Year 1</td></tr>
<tr><td colspan="8">Year 2</td></tr>
<tr><td colspan="8">Month</td></tr>
<tr><td colspan="8">Day</td></tr>
<tr><td colspan="8">Hour Local Time</td></tr>
<tr><td colspan="8">Minute Local Time</td></tr>
<tr><td colspan="8">Second Local Time</td></tr>
<tr><td colspan="3">Current Meter Precision 1</td><td colspan="5">Current Meter Scale 1</td></tr>
<tr><td colspan="8">Current Value 1,1</td></tr>
<tr><td colspan="8">Current Value 1,2</td></tr>
<tr><td colspan="8">Current Value 1,3</td></tr>
<tr><td colspan="8">Current Value 1,4</td></tr>
<tr><td colspan="8">…</td></tr>
<tr><td colspan="3">Current Meter Precision N</td><td colspan="5">Current Meter Scale N</td></tr>
<tr><td colspan="8">Current Value N,1</td></tr>
<tr><td colspan="8">Current Value N,2</td></tr>
<tr><td colspan="8">Current Value N,3</td></tr>
<tr><td colspan="8">Current Value N,4</td></tr>
</table>

Reports to follow (8 bits)

This value indicates how many report frames there are left, the value 0xFF means that the number of reports have not been calculated yet or that there is more than 255 reports to follow.

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Rate Type (2 bits)

Rate Type specifies the type of parameters in the report. Rate Type defined as the *Meter Rate Type* variable; refer to section 3.5.3 for a definition of the variable.

Dataset (24 bits)

The dataset parameter indicates which data is included in the report. Dataset parameters defined as the *Meter Dataset* variable; refer to section 3.5.2 for a definition of the variable.

Year (16 bits)

Specify the year in the usual Gregorian calendar. The first byte (Year 1) is the most significant byte. A year equal to 0x0000 indicates that an accumulated value is not determined yet.

Month (8 bits)

Specify the month of the year between 01 (January) and 12 (December).

Day (8 bits)

Specify the day of the month between 01 and 31.

Hour Local Time (8 bits)

Specify the number of complete hours that have passed since midnight (00-23) in local time.

Minute Local Time (8 bits)

Specify the number of complete minutes that have passed since the start of the hour (00-59) in local time.

Second Local Time (8 bits)

Specify the number of complete seconds since the start of the minute (00-59) in local time. The value 60 used to keep UTC from wandering away is not supported.

Meter Precision (3 bits)

The Meter Precision field describes what the precision of the value is. The number indicates the number of decimals. The decimal value 1025 with precision 2 is therefore equal to 10.25.

Meter Scale (5 bits)

The Meter Scale used to indicate the scale (unit) of the reported parameter.

Table 107, Meter Table Current Data Report::Meter Scale encoding

Meter Type	Meter Scale	Value
Electric meter	kWh	0x00
	kVARh	0x01
	%	0x02
	Pulse count	0x03
	kVAR	0x04
	Voltage (V)	0x05
	Amperes (A)	0x06
	kW	0x07
	Ratio	0x08
	Reserved	0x09- 0x1F
Gas and water meter	Cubic meter	0x00
	Cubic feet	0x01
	US gallon	0x02
	Pulse count	0x03
	IMP gallon	0x04
	Liter	0x05
	kPa	0x06
	Centum cubic feet	0x07
	Cubic meter per hour	0x08
	Liter per hour	0x09
	kWh	0x0A
	MWh	0x0B
	KW	0x0C
	Hours	0x0D
	Reserved	0x0E-0x1F
Heating and Cooling Meter	Cubic meter (m^3)	0x00
	Metric Ton (tonne) (t)	0x01
	Cubic meter per hour (m^3/h)	0x02
	Liter per hour (l/h)	0x03
	kW	0x04
	MW	0x05

	kWh	0x06
	MWh	0x07
	Giga Joule (GJ)	0x08
	Giga Calorie (Gcal)	0x09
	Celsius (C^{o})	0x0A
	Fahrenheit (^{o}F)	0x0B
	Hours	0x0C
	Reserved	0x0D-0x1F

Current Value (32 bits)

The Current Value is a 32 bit signed field defined by dataset requested field. The first byte (Value 1) is the most significant byte. The table below shows signed decimal values together with their hexadecimal equivalents.

Signed 4 bytes	
Decimal	**Hexadecimal**
2147483647	0x7FFFFFFF
...	...
1073741823	0x3FFFFFFF
...	...
1	0x00000001
0	0x00000000
-1	0xFFFFFFFF
…	...
-1073741823	0xC0000001
…	...
-2147483648	0x80000000

NOTICE: The device receiving the Meter Table Current Data Report MUST show the value even though the scale is not supported.

4.79.14 Meter Table Historical Data Get Command

The Meter Table Historical Data Get Command is used to request a number of time stamped values (historical) in physical units according to rate type, dataset mask and time interval.

The Meter Table Historical Data Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER_TBL_MONITOR							
Command = METER_TBL_HISTORICAL_DATA_GET							
Maximum Reports							
Historical Dataset Requested 1							
Historical Dataset Requested 2							
Historical Dataset Requested 3							
Start Year 1							
Start Year 2							
Start Month							
Start Day							
Start Hour Local Time							
Start Minute Local Time							
Start Second Local Time							
Stop Year 1							
Stop Year 2							
Stop Month							
Stop Day							
Stop Hour Local Time							
Stop Minute Local Time							
Stop Second Local Time							

Maximum Reports (8 bits)

The maximum reports parameter is used to indicate the maximum number of reports to return based on the get. Reports are always returned with the most recently recorded value first. If set to 0x00 the meter will return all reports based on the request.

Dataset History (24 bits)

The Historical Dataset Requested parameter is use to indicate which parameters are requested from the meter. Historical Dataset Requested parameters defined as the *Meter Dataset* variable; refer to section 3.5.2 for a definition of the variable.

Start/Stop Year (16 bits)

Specify the year in the usual Gregorian calendar. The first byte (Year 1) is the most significant byte.

Start/Stop Month (8 bits)

Specify the month of the year between 01 (January) and 12 (December).

Start/Stop Day (8 bits)

Specify the day of the month between 01 and 31.

Start/Stop Hour Local Time (8 bits)

Specify the number of complete hours that have passed since midnight (00-23) in local time.

Start/Stop Minute Local Time (8 bits)

Specify the number of complete minutes that have passed since the start of the hour (00-59) in local time.

Start/Stop Second Local Time (8 bits)

Specify the number of complete seconds since the start of the minute (00-59) in local time. The value 60 used to keep UTC from wandering away is not supported.

4.79.15 Meter Table Historical Data Report Command

This command is used to report a number of time stamped values.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER_TBL_MONITOR							
Command = METER_TBL_HISTORICAL_DATA_REPORT							
Reports to Follow							
Reserved						Rate Type	
Dataset 1							
Dataset 2							
Dataset 3							
Historical Year 1							
Historical Year 2							
Historical Month							
Historical Day							
Historical Hour Local Time							
Historical Minute Local Time							
Historical Second Local Time							
Historical Precision 1			Historical Scale 1				
Historical Value 1,1							
Historical Value 1,2							
Historical Value 1,3							
Historical Value 1,4							
…							
Historical Precision N			Historical Scale N				
Historical Value N,1							
Historical Value N,2							
Historical Value N,3							
Historical Value N,4							

Reports to Follow (8 bits)

This value indicates how many report frames there are left, the value 0xFF means that the number of reports have not been calculated yet or that there is more than 255 reports to follow.

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Rate Type (2 bits)

Rate Type specifies the type of parameters in the report. Rate Type defined as the *Meter Rate Type* variable; refer to section 3.5.3 for a definition of the variable.

Dataset (24 bits)

This command parameter is defined in a bitmap format and holds values and combinations of the meters dataset. If no historical data has been registered for the period requested by the Meter Table Historical Data Get Command, all fields of this parameter MUST be set to ‘0’ AND no “Historical” parameters are to be included in the Meter Table Historical Data Report Command i.e. the command parameters ends after “Dataset”. Refer to “Meter Dataset” in section 3.5.2 for details.

Historical Year (16 bits)

Specify for the dataset the year in the usual Gregorian calendar. The first byte (Year 1) is the most significant byte.

Historical Month (8 bits)

Specify for the dataset the month of the year between 01 (January) and 12 (December). A year equal to 0x0000 indicates that an accumulated value is not determined yet.

Historical Day (8 bits)

Specify for the dataset the day of the month between 01 and 31.

Historical Hour Local Time (8 bits)

Specify for the dataset the number of complete hours that have passed since midnight (00-23) in local time.

Historical Minute Local Time (8 bits)

Specify for the dataset the number of complete minutes that have passed since the start of the hour (00-59) in local time.

Historical Second Local Time (8 bits)

Specify for the dataset the number of complete seconds since the start of the minute (00-59) in local time. The value 60 used to keep UTC from wandering away is not supported.

Historical Precision (3 bits)

The Historical Precision field describes what the precision of the value is. The number indicates the number of decimals. The decimal value 1025 with precision 2 is therefore equal to 10.25.

Historical Scale (5 bits)

The Historical Scale used to indicate the scale (unit) of the reported parameter. The Historical Scale parameter is of the variable type *Meter Scale*; refer to Section 4.79.13 for a definition of the variable.

Historical Value

The Historical Value is a 32 bit signed field defined by dataset requested field. The first byte (Value 1) is the most significant byte. The table below shows signed decimal values together with their hexadecimal equivalents.

Signed 4 bytes	
Decimal	**Hexadecimal**
2147483647	0x7FFFFFFF
...	...
1073741823	0x3FFFFFFF
...	...
1	0x00000001
0	0x00000000
-1	0xFFFFFFFF
...	...
-1073741823	0xC0000001
...	...
-2147483648	0x80000000

NOTICE: The device receiving the Meter Table Historical Data Report MUST always show the value even though the Scale is not supported.

4.80 Meter Table Monitor Command Class, version 2

The Meter Table Monitor Command Class defines the Commands necessary to read historical and accumulated values in physical units from a water meter or other metering device (gas, electric etc.) or electric sub-metering device and thereby enabling automatic meter reading capabilities

4.80.1 Meter Table Point Adm. Number Get Command

The Meter Table Point Adm. Number Get Command is used to request the Meter Point Administration Number to identify customer.

The Meter Table Point Adm. Number Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER_TBL_MONITOR							
Command = METER_TBL_TABLE_POINT_ADM_NO_GET							

4.80.2 Meter Table Point Adm. Number Report Command

This command reports parameters used for identification of customer and metering device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER_TBL_MONITOR							
Command = METER_TBL_TABLE_POINT_ADM_NO_REPORT							
Reserved			Number of Meter Point Adm. Number Characters				
Meter Point Adm. Number Character 1							
…							
Meter Point Adm. Number Character N							

Refer to description of fields under the Meter Table Point Adm. Number Set Command in Meter Table Configuration Command Class, version 1.

If Meter Table Point Adm, Number is not Set using the Meter Table Point Adm, Number Set Command in Meter Table Configuration Command Class, version 1 the Reported Number of Meter Point Adm, Number must be 0x00 and no Meter Table Point Adm, Numbers must be reported.

4.80.3 Meter Table ID Get Command

The Meter Table ID Get Command is used to request the parameters used for identification of customer and metering device.

The Meter Table ID Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER_TBL_MONITOR							
Command = METER_TBL_TABLE_ID_GET							

4.80.4 Meter Table ID Report Command

This command reports parameters used for identification of customer and metering device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER_TBL_MONITOR							
Command = METER_TBL_TABLE_ID_REPORT							
Reserved			Number of Meter ID Characters				
Meter ID Character 1							
…							
Meter ID Character N							

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Number of Meter ID Characters (5 bits)

Number of characters defining the meter ID (1..32).

Meter ID Character (N bytes)

The Meter ID character fields hold the string identifying the individual metering device. The character presentation uses standard ASCII codes (values 128-255 are ignored). In addition one can use the Manufacturer Specific Command Class in conjunction as product identification.

Note: If the Meter Table ID is not used Number of Meter ID Characters must be reported as 0x00 and no Meter ID Characters must be reported.

4.80.5 Meter Table Capability Get Command

The Meter Table Capability Get Command is used to request the capabilities of a metering device.

The Meter Table Capability Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER_TBL_MONITOR							
Command = METER_TBL_TABLE_CAPABILITY_GET							

4.80.6 Meter Table Capability Report Command

This command is used to advertise meter table capabilities.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER_TBL_MONITOR							
Command = METER_TBL_REPORT							
Rate Type		Meter Type					
Reserved				Pay Meter			
Dataset Supported 1							
Dataset Supported 2							
Dataset Supported 3							
Dataset History Supported 1							
Dataset History Supported 2							
Dataset History Supported 3							
Data History Supported 1							
Data History Supported 2							
Data History Supported 3							

Rate Type (2 bits)

Rate Type specifies the type of parameters in the report. Rate Type defined as the *Meter Rate Type* variable; refer to section 3.5.3 for a definition of the variable.

If the Meter Type is Submeter the Rate Type must be Import to indicate consumption.

Meter Type (6 bits)

Meter Type specifies the type of metering device the command originates. Meter Type defined as the *Meter Type* variable; refer to section 3.5.5 for a definition of the variable.

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Pay Meter (4 bits)

Pay Meter specifies the way settling of account is done.

Pay Meter	Value
Reserved	0x00
Creditmeter	0x01
Prepayment meter	0x02
Prepayment meter with debt recovery	0x03
Reserved	0x04-0x07

Dataset Supported / Dataset History Supported (24 bits / 24 bits)

Dataset Supported specifies which parameters, are available to be requested from the metering device. Dataset Supported Parameters defined as the *Meter Dataset* variable; refer to section 3.5.2 for a definition of the variable.

Dataset History Supported specifies the type of data/values, which may be requested through the Meter Table Historical data Get command (see 4.80.14) from the metering device. Dataset History Supported Parameters defined as the *Meter Dataset* variable; refer to section 3.5.2 for a definition of the variable.

Data History Supported (24 bits)

Data History Supported specifies number of possible entries in the buffer holding the historical values. Historical data cannot be retrieved when Data History Supported is equal to 0.

4.80.7 Meter Table Status Supported Get Command

The Meter Table Status Supported Get Command is used to request the supported operating status event parameters and logging depth of the metering device.

The Meter Table Status Supported Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER_TBL_MONITOR							
Command = METER_TBL _STATUS_SUPPORTED_GET							

4.80.8 Meter Table Status Supported Report Command

The Meter Table Status Report Command is used to report the supported operation status' and logging depth of these in the meter.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER_TBL_MONITOR							
Command = METER_TBL _STATUS_SUPPORTED_REPORT							
Supported Operating Status 1							
Supported Operating Status 2							
Supported Operating Status 3							
Status Event Log Depth							

Supported Operating Status (1–4 Bytes)

This command parameter is defined in a bitmap format and holds values and combinations of the meters operating status. If no "Operating Status Event" is supported by the application all fields of this parameter must be set to '0' as an indication of the meter is at "Normal/Idle" operating status. Supported Operating Status supports the following combinations of operating status:

Status Event Log Depth (8 bits)

The *status event log depth* indicates the supported depth of the event log. If the meter only supports reporting the current status the status event log depth must be set to 0.

4.80.9 Meter Table Status Depth Get Command

The Meter Table Status Depth Get Command is used to request the current operating status of the metering device or to request a number of the latest status event from the logs.

The Meter Table Status Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER_TBL_MONITOR							
Command = METER_TBL _STATUS_DEPTH_GET							
Status Event Log Depth							

Status Event Log Depth (8 bits)

The *status event log depth* indicates the number of latest recorded events that should be returned in the corresponding report. If the log depth is set to 0 the meter will only return the current status; if the log depth is set to 0xFF the meter should return the entire status log.

4.80.10 Meter Table Status Date Get Command

The Meter Table Status Date Get Command is used to request a number of status events recorded in a certain time interval.

The Meter Table Status Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

If the meter does not support a status event log history it must return the current state of the meter (see section 4.80.11)

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER_TBL_MONITOR							
Command = METER_TBL _STATUS_DATE_GET							
Maximum Reports							
Start - Year 1							
Start - Year 2							
Start - Month							
Start - Day							
Start - Hour Local Time							
Start - Minute Local Time							
Start - Second Local Time							
Stop - Year 1							
Stop - Year 2							
Stop - Month							
Stop - Day							
Stop - Hour Local Time							
Stop - Minute Local Time							
Stop - Second Local Time							

Maximum Reports (8 bits)

The maximum reports parameter is used to indicate the maximum number of reports to return based on the get. Reports are always returned with the most recently recorded log entry first. If set to 0x00 the meter will return all reports based on the request.

Start / Stop - Year (16 bits)

Specify the year in the usual Gregorian calendar. The first byte (Year 1) is the most significant byte.

Start / Stop - Month (8 bits)

Specify the month of the year between 01 (January) and 12 (December).

Start / Stop - Day (8 bits)

Specify the day of the month between 01 and 31.

Start / Stop - Hour Local Time (8 bits)

Specify the number of complete hours that have passed since midnight (00-23) in local time.

Start / Stop - Minute Local Time (8 bits)

Specify the number of complete minutes that have passed since the start of the hour (00-59) in local time.

Start / Stop - Second Local Time (8 bits)

Specify the number of complete seconds since the start of the minute (00-59) in local time. The value 60 used to keep UTC from wandering away is not supported.

4.80.11 Meter Table Status Report Command

This command is used to advertise the current status of the meter.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER_TBL_MONITOR							
Command = METER_TBL _STATUS _REPORT							
Reports to follow							
Current Operating Status 1							
Current Operating Status 2							
Current Operating Status 3							
Event 1 - Type	Reserved		Event 1 - Operating Status Event ID				
Event 1 - Year 1							
Event 1 - Year 2							
Event 1 - Month							
Event 1 - Day							
Event 1 - Hour Local Time							
Event 1 - Minute Local Time							
Event 1 - Second Local Time							
…							
Event n -Type	Reserved		Event n - Operating Status Event ID				
Event n - Year 1							
Event n - Year 2							
Event n - Month							
Event n - Day							
Event n - Hour Local Time							
Event n -Minute Local Time							
Event n - Second Local Time							

Reports to follow (8 bits)

This value indicates how many report frames there are left, the value 0xFF means that the number of reports have not been calculated yet or that there is more than 255 reports to follow.

Current Operating Status (24 bits)

This command parameter is defined in a bitmap format and holds values and combinations of the meters operating status. If no “Operating Status Event” has been registered for the period requested by the Meter Table Status Date Get Command, all fields of this parameter must be set to ‘0’ AND no “Event” parameters are to be included in the Meter Table Status Report Command i.e. the command parameters ends after “Current Operating Status” . Refer to “Supported Operating Status” in section 4.79.8 for details.

Event n – Type (1 bit)

The parameter reports type of event number *n* in the event log. The event type MAY be 0 (the meter entered the state described by the operating event status ID at the time reported), or 1 (the meter left the state described by the operating event status ID).

Event – Operating Status Event ID

The operating status event identifier of event number *n* in the event log. The identifiers available are given in the table in section 4.79.8.

Event - Year (16 bits)

Specify the year in the usual Gregorian calendar. The first byte (Year 1) is the most significant byte.

Event - Month (8 bits)

Specify the month of the year between 01 (January) and 12 (December).

Event - Day (8 bits)

Specify the day of the month between 01 and 31.

Event - Hour Local Time (8 bits)

Specify the number of complete hours that have passed since midnight (00-23) in local time.

Event - Minute Local Time (8 bits)

Specify the number of complete minutes that have passed since the start of the hour (00-59) in local time.

Event - Second Local Time (8 bits)

Specify the number of complete seconds since the start of the minute (00-59) in local time. The value 60 used to keep UTC from wandering away is not supported.

4.80.12 Meter Table Current Data Get Command

The Meter Table Current Data Get Command is used to request a number of time stamped values (current) in physical units according to the dataset mask.

The Meter Table Current Data Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER_TBL_MONITOR							
Command = METER_TBL_CURRENT_DATA_GET							
Dataset Requested 1							
Dataset Requested 2							
Dataset Requested 3							

Dataset Requested (24 bits)

The Dataset Requested is used to indicate which parameters are requested from the meter. The Dataset Requested parameter defined as the *Meter Dataset* variable; refer to section 3.5.2 for a definition of the variable.

4.80.13 Meter Table Current Data Report Command

This command is used to report a number of time stamped values.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER_TBL_MONITOR							
Command = METER_TBL_CURRENT_DATA_REPORT							
Reports to Follow							
Operating Status Indication	Reserved					Rate Type	
Dataset 1							
Dataset 2							
Dataset 3							
Year 1							
Year 2							
Month							
Day							
Hour Local Time							
Minute Local Time							
Second Local Time							
Current Meter Precision 1				Current Meter Scale 1			
Current Value 1,1							
Current Value 1,2							
Current Value 1,3							
Current Value 1,4							
…							
Current Meter Precision N				Current Meter Scale N			
Current Value N,1							
Current Value N,2							
Current Value N,3							
Current Value N,4							

Reports to follow (8 bits)

This value indicates how many report frames there are left, the value 0xFF means that the number of reports have not been calculated yet or that there is more than 255 reports to follow.

Operating Status Indication (1 bit)

The Operating Status Indication bit is used to indicate that the reported meter data is measured while the meter is in a operating status different from Normal e.g. accuracy warning or clock not accurate.

Setting the bit to 1 means that the meter is operating in a status different from Normal. Setting the bit to 0 means that the meter is operating in Normal mode.

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Rate Type (2 bits)

Rate Type specifies the type of parameters in the report. Rate Type defined as the *Meter Rate Type* variable; refer to section 3.5.3 for a definition of the variable.

If the Meter Type is Submeter the Rate Type must be Import to indicate consumption.

Dataset (24 bits)

The dataset parameter indicates which data is included in the report. Dataset parameters defined as the *Meter Dataset* variable; refer to section 3.5.2 for a definition of the variable.

Year (16 bits)

Specify the year in the usual Gregorian calendar. The first byte (Year 1) is the most significant byte. A year equal to 0x0000 indicates that an accumulated value is not determined yet.

Month (8 bits)

Specify the month of the year between 01 (January) and 12 (December).

Day (8 bits)

Specify the day of the month between 01 and 31.

Hour Local Time (8 bits)

Specify the number of complete hours that have passed since midnight (00-23) in local time.

Minute Local Time (8 bits)

Specify the number of complete minutes that have passed since the start of the hour (00-59) in local time.

Second Local Time (8 bits)

Specify the number of complete seconds since the start of the minute (00-59) in local time. The value 60 used to keep UTC from wandering away is not supported.

Meter Precision (3 bits)

The Meter Precision field describes what the precision of the value is. The number indicates the number of decimals. The decimal value 1025 with precision 2 is therefore equal to 10.25.

Meter Scale (5 bits)

The Meter Scale used to indicate the scale (unit) of the reported parameter. Meter Scale variables; refer to section 3.5.4 for a definition of the variable.

Current Value (32 bits)

The Current Value is a 32 bit signed field defined by dataset requested field. The first byte (Value 1) is the most significant byte. The table below shows signed decimal values together with their hexadecimal equivalents.

Signed 4 bytes	
Decimal	**Hexadecimal**
2147483647	0x7FFFFFFF
...	...
1073741823	0x3FFFFFFF
...	...
1	0x00000001
0	0x00000000
-1	0xFFFFFFFF
...	...
-1073741823	0xC0000001
...	...
-2147483648	0x80000000

NOTICE: The device receiving the Meter Table Current Data Report must always show the value even though the scale is not supported.

4.80.14 Meter Table Historical Data Get Command

The Meter Table Historical Data Get Command is used to request a number of time stamped values (historical) in physical units according to rate type, dataset mask and time interval.

The Meter Table Historical Data Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER_TBL_MONITOR							
Command = METER_TBL_HISTORICAL_DATA_GET							
Maximum Reports							
Historical Dataset Requested 1							
Historical Dataset Requested 2							
Historical Dataset Requested 3							
Start Year 1							
Start Year 2							
Start Month							
Start Day							
Start Hour Local Time							
Start Minute Local Time							
Start Second Local Time							
Stop Year 1							
Stop Year 2							
Stop Month							
Stop Day							
Stop Hour Local Time							
Stop Minute Local Time							
Stop Second Local Time							

Maximum Reports (8 bits)

The maximum reports parameter is used to indicate the maximum number of reports to return based on the get. Reports are always returned with the most recently recorded value first. If set to 0x00 the meter will return all reports based on the request.

Dataset History (24 bits)

The Historical Dataset Requested parameter is use to indicate which parameters are requested from the meter. Historical Dataset Requested parameters defined as the *Meter Dataset* variable; refer to section 3.5.2 for a definition of the variable.

Start/Stop Year (16 bits)

Specify the year in the usual Gregorian calendar. The first byte (Year 1) is the most significant byte.

Start/Stop Month (8 bits)

Specify the month of the year between 01 (January) and 12 (December).

Start/Stop Day (8 bits)

Specify the day of the month between 01 and 31.

Start/Stop Hour Local Time (8 bits)

Specify the number of complete hours that have passed since midnight (00-23) in local time.

Start/Stop Minute Local Time (8 bits)

Specify the number of complete minutes that have passed since the start of the hour (00-59) in local time.

Start/Stop Second Local Time (8 bits)

Specify the number of complete seconds since the start of the minute (00-59) in local time. The value 60 used to keep UTC from wandering away is not supported.

4.80.15 Meter Table Historical Data Report Command

This command is used to report a number of time stamped values.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER_TBL_MONITOR							
Command = METER_TBL_HISTORICAL_DATA_REPORT							
Reports to Follow							
Operating Status Indication	Reserved					Rate Type	
Dataset 1							
Dataset 2							
Dataset 3							
Historical Year 1							
Historical Year 2							
Historical Month							
Historical Day							
Historical Hour Local Time							
Historical Minute Local Time							
Historical Second Local Time							
Historical Precision 1			Historical Scale 1				
Historical Value 1,1							
Historical Value 1,2							
Historical Value 1,3							
Historical Value 1,4							
…							
Historical Precision N			Historical Scale N				
Historical Value N,1							
Historical Value N,2							
Historical Value N,3							
Historical Value N,4							

Reports to Follow (8 bits)

This value indicates how many report frames there are left, the value 0xFF means that the number of reports have not been calculated yet or that there is more than 255 reports to follow.

Operating Status Indication (1 bit)

The Operating Status Indication bit is used to indicate that the reported meter data is measured while the meter is in a operating status different from Normal e.g. accuracy warning or clock not accurate.

Setting the bit to 1 means that the meter is operating in a status different from Normal. Setting the bit to 0 means that the meter is operating in Normal mode.

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Rate Type (2 bits)

Rate Type specifies the type of parameters in the report. Rate Type defined as the *Meter Rate Type* variable; refer to section 3.5.3 for a definition of the variable.

If the Meter Type is Submeter the Rate Type must be Import to indicate consumption.

Dataset (24 bits)

This command parameter is defined in a bitmap format and holds values and combinations of the meters dataset. If no historical data has been registered for the period requested by the Meter Table Historical Data Get Command, all fields of this parameter must be set to '0' AND no "Historical" parameters are to be included in the Meter Table Historical Data Report Command i.e. the command parameters ends after "Dataset". Refer to "Meter Dataset" in section 3.5.2 for details.

Historical Year (16 bits)

Specify for the dataset the year in the usual Gregorian calendar. The first byte (Year 1) is the most significant byte.

Historical Month (8 bits)

Specify for the dataset the month of the year between 01 (January) and 12 (December). A year equal to 0x0000 indicates that an accumulated value is not determined yet.

Historical Day (8 bits)

Specify for the dataset the day of the month between 01 and 31.

Historical Hour Local Time (8 bits)

Specify for the dataset the number of complete hours that have passed since midnight (00-23) in local time.

Historical Minute Local Time (8 bits)

Specify for the dataset the number of complete minutes that have passed since the start of the hour (00-59) in local time.

Historical Second Local Time (8 bits)

Specify for the dataset the number of complete seconds since the start of the minute (00-59) in local time. The value 60 used to keep UTC from wandering away is not supported.

Historical Precision (3 bits)

The Historical Precision field describes what the precision of the value is. The number indicates the number of decimals. The decimal value 1025 with precision 2 is therefore equal to 10.25.

Historical Scale (5 bits)

The Historical Scale used to indicate the scale (unit) of the reported parameter. The Historical Scale parameter is of the variable type *Meter Scale*; refer to Section 4.79.13 for a definition of the variable.

Historical Value

The Historical Value is a 32 bit signed field defined by dataset requested field. The first byte (Value 1) is the most significant byte. The table below shows signed decimal values together with their hexadecimal equivalents.

Signed 4 bytes	
Decimal	**Hexadecimal**
2147483647	0x7FFFFFFF
...	...
1073741823	0x3FFFFFFF
...	...
1	0x00000001
0	0x00000000
-1	0xFFFFFFFF
...	...
-1073741823	0xC0000001
...	...
-2147483648	0x80000000

NOTICE: The device receiving the Meter Table Historical Data Report must always show the value even though the Scale is not supported.

4.81 Meter Table Push Configuration Command Class version 1

The Meter Table Push Configuration Command Class is used to configure the meter to send a Current Data Report at a given interval. The meter may be configured to return different data sets at different intervals using both the primary and secondary push commands.

4.81.1 Meter Table Push Configuration Set Command

The Meter Table Push Configuration Set Command is used to request the meter to send a Current Data Report at a given interval. The meter may be configured to return different data sets at different intervals using both the primary and secondary push commands.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_METER_TBL_PUSH</td></tr>
<tr><td colspan="8">Command = METER_TBL_PUSH_CONFIGURATION_SET</td></tr>
<tr><td colspan="3">Reserved</td><td>P/S</td><td colspan="4">Operating Status Push Mode</td></tr>
<tr><td colspan="8">Push Dataset 1</td></tr>
<tr><td colspan="8">Push Dataset 2</td></tr>
<tr><td colspan="8">Push Dataset 3</td></tr>
<tr><td colspan="8">Interval Months</td></tr>
<tr><td colspan="8">Interval Days</td></tr>
<tr><td colspan="8">Interval Hours</td></tr>
<tr><td colspan="8">Interval Minutes</td></tr>
<tr><td colspan="8">Push Node ID</td></tr>
</table>

Operating Status Push Mode (4 bits)

The Operating Status Push Mode is use to configure if the Meter Table Status Report Command (please refer to 4.79.11) participates in the Push Functionality

Operating Status Push Mode Identifier	Description
0x00	Operating Status push disabled
0x01	Operating Status push based on Interval
0x02	Operating Status push based on Status Change
0x03	Operating Status push Based on Interval AND status change
0x04-0x0F	Reserved

P/S (1 bit)

P/S	Description
0x00	Primary push configuration
0x01	Secondary push configuration

Push Dataset (24 bits)

The Push Dataset parameter is use to indicate which parameters are requested to be pushed from the meter. The Push Dataset parameters defined as the *Meter Dataset* variable; refer to section 3.5.2 for a definition of the variable.

Interval Months (8 bits)

Specify the number of months between pushing the push dataset.

Interval Day (8 bits)

Specify the number of days between pushing the push dataset.

Interval Hours (8 bits)

Specify the number of hours between pushing the push dataset.

Interval Minute (8 bits)

Specify the number of minutes between pushing the push dataset.

Push Node ID (8 bits)

Specify the node ID of the node to receive push dataset in the given interval.

4.81.2 Meter Table Push Configuration Get Command

The Meter Table Push Configuration Get Command is used to request the meters push configuration

The Meter Table Push Configuration Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_METER_TBL_PUSH							
Command = METER_TBL_PUSH_CONFIGURATION_GET							

4.81.3 Meter Table Push Configuration Report Command

The Meter Table Push Configuration Report Command is used report the current Push Configuration

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_METER_TBL_PUSH</td></tr>
<tr><td colspan="8">Command = METER_TBL_PUSH_CONFIGURATION_REPORT</td></tr>
<tr><td colspan="3">Reserved</td><td>P/S</td><td colspan="4">Operating Status Push Mode</td></tr>
<tr><td colspan="8">Push Dataset 1</td></tr>
<tr><td colspan="8">Push Dataset 2</td></tr>
<tr><td colspan="8">Push Dataset 3</td></tr>
<tr><td colspan="8">Interval Months</td></tr>
<tr><td colspan="8">Interval Days</td></tr>
<tr><td colspan="8">Interval Hours</td></tr>
<tr><td colspan="8">Interval Minutes</td></tr>
<tr><td colspan="8">Push Node ID</td></tr>
</table>

Please refer to Meter Table Push Configuration Set Command (section 4.81.1) for detailed description of the fields.

4.82 Move To Position Window Covering Command Class, version 1 [OBSOLETED]

<u>THIS COMMAND CLASS HAS BEEN OBSOLETED</u>

New implementations MUST NOT support this command class.

Window covering device implementations SHOULD support the Window Covering Command Class.

The Move To Position Window Covering Command Class is used to control the position of a window covering device.

The Move To Position Window Covering Command Class is an actuator control command class. Refer to 3.7.

4.82.1 Move To Position Set Command

The Move To Position Set command, version 1 is used to instruct a window covering to move to a new position.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_MTP_WINDOW_COVERING							
Command = MOVE_TO_POSITION_SET							
Value							

Value (8 bits)

The encoding of the Value field MUST be according to Table 108.

Table 108, Move To Position Set :: Value

Value	Level	State
0 (0x00)	0%	Closed
1..99 (0x01..0x63)	Almost closed .. 100% Open	Open
…	*reserved*	*reserved*
255 (0xFF)	100% Open	Open

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

4.82.2 Move To Position Get Command

The Move To Position Get command, version 1 is used to request the status of a window covering device.

The Move To Position Report command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing.
The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_MTP_WINDOW_COVERING							
Command = MOVE_TO_POSITION_GET							

4.82.3 Move To Position Report Command

The Move To Position Report Command, version 1 is used to advertise the status of a window covering device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_MTP_WINDOW_COVERING							
Command = MOVE_TO_POSITION_REPORT							
Value							

Value (8 bits)

The encoding of the Value field MUST be according to Table 108.

The Value field SHOULD advertise the current value of the device hardware; also while in transition to a new target value.

A controlling device MUST NOT assume that the Value is identical to a value previously issued with a Set command when a transition has ended.

4.83 Multi Channel Command Class, version 1-2 [OBSOLETED]

THIS COMMAND CLASS HAS BEEN OBSOLETED

New implementations MUST use the Multi Channel Command Class, version 3 or newer.

4.84 Multi Channel Command Class, version 3

The Multi Channel command class is used to address one or more end points in a Multi Channel device. An application implementing this command class MUST set the Optional Functionality bit in the NIF.

A Multi Channel device MAY implement from 1 to 127 end points.

A controlling devices MAY use Multi Channel encapsulation to communicate with Multi Channel End Points in other devices. If such a controlling device does not implement any End Points, the device MUST NOT advertise the Multi Channel Command Class in the Node Information Frame.

Refer to 3.6 for an introduction to the Multi Channel concept.

4.84.1 Dynamic End Point considerations

An End Point MAY be dynamic. Dynamic End points are intended End Points able to change their capabilities or that can be added and removed from a Multi Channel device.

When creating a new dynamic End Point, it SHOULD be assigned an End Point identifier which has not been used recently to allow applications to discover the removal of a dynamic End Point.
When a dynamic End Point is removed, all other End Points MUST maintain their current End Point identifiers.

A supporting device implementing dynamic End Points MUST advertise the creation, change or removal of a dynamic End Point via the Root Device Lifeline association group by issuing a Multi Channel Capability Report.

A node MUST NOT advertise changes to dynamic End Points via broadcast transmission. A receiving node MUST ignore such broadcasted advertisements.

A node receiving dynamic End Point modifications SHOULD learn again about the sending node End Points. The RECOMMENDED procedure is shown in Figure 34

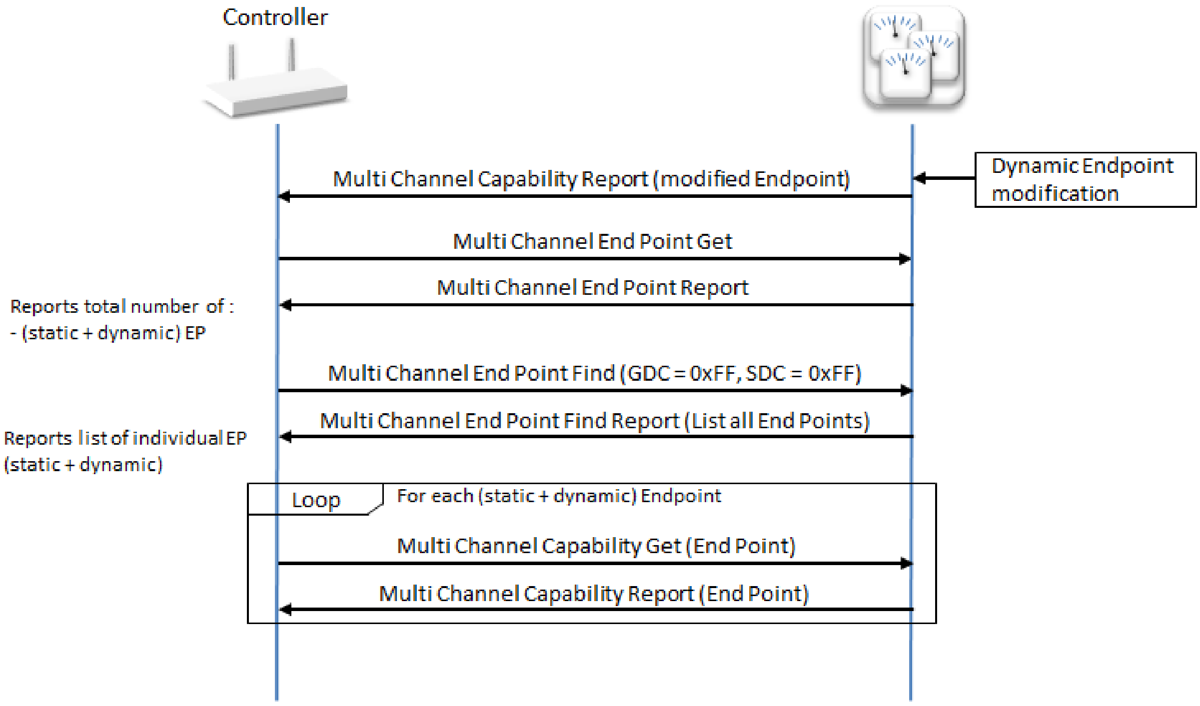


Figure 34, Dynamic End Point modification rediscovery (version 3)

After advertising the removal, a removed dynamic End Point MUST ignore all commands.

A controlling device SHOULD periodically verify the existence and capabilities of dynamic End Points.

4.84.2 Interoperability considerations

In order to communicate with Multi Channel End Points in other devices, a device MAY support the Multi Channel Command Class, version 3 without actually implementing any Multi Channel End Points. Such a device SHOULD NOT advertise support for the Multi Channel Command Class.

A typical example of such a device would be a gateway which can control End Points in other devices via commands from the Root Device of the gateway. Likewise, the Root Device of the gateway may receive unsolicited commands from End Points in other nodes.

4.84.3 Compatibility considerations

A Multi Channel device MUST implement all application functionality in End Points.

End Point 1 MUST implement the primary functionality of the actual Multi Channel device. Additional End Points MAY implement the same functionality, As an example, a power strip device may implement five End Points with identical functionality.

For backwards compatibility, the Root Device MUST mirror the functionality of End Point 1. Further, the Root Device MAY mirror the functionality of additional End Points. As an example, Basic Off and On commands for the Root Device may control all outlets of the power strip with 5 outlets.

With the exception of dynamic End Points, the Root Device MUST advertise all End Point functionality which is mirrored by the Root Device.
The Root Device SHOULD NOT advertise the application functionality of any dynamic End Point.

The Root Device of a Multi Channel device MUST NOT advertise any application functionality that cannot be reached via one or more End Points.

It MUST NOT be possible to limit the functionality, or enable non-compliant behavior by any End Point in the device by sending a command to the Root Device.

Multi Channel Command Class support MUST NOT be advertised by Multi Channel End Points.

When End Point functionality is advertised in the Root Device NIF, service discovery mechanisms like mDNS and installer-style GUIs risk presenting Root Device functionality which is actually a mirror representation of End Point functionality.
Therefore, application-style command classes of the Root Device NIF which are also advertised by at least one End Point SHOULD be filtered out by controllers or gateways before presenting the NIF contents via service discovery mechanisms like mDNS or to users in a GUI.

4.84.4 Multi Channel End Point Get Command

The Multi Channel End Point Get Command is used to query the number of End Points implemented by the node.

The Multi Channel End Point Report command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing.
The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_MULTI_CHANNEL</td></tr>
<tr><td colspan="8">Command = MULTI_CHANNEL_END_POINT_GET</td></tr>
</table>

4.84.5 Multi Channel End Point Report Command

The Multi Channel End Point Report Command is used to advertise the number of End Points implemented by the node.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_MULTI_CHANNEL</td></tr>
<tr><td colspan="8">Command = MULTI_CHANNEL_END_POINT_REPORT</td></tr>
<tr><td>Dyna-
mic</td><td>Iden-
tical</td><td colspan="6">Res</td></tr>
<tr><td>Res</td><td colspan="7">End Points</td></tr>
</table>

Dynamic (1 bit)

This field is used to advertise if the node implements a dynamic number of End Points.

The value 1 MUST be used to indicate that the number of End Points is dynamic. The value 0 MUST be used to indicate that the number of End Points is static.

Identical (1 bit)

This bit MUST be set to 1 if all End Points advertise the same generic and specific device class and support the same optional command classes.

The bit MUST be set to 0 if End Points do not advertise the same device class and command class information.

Res

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

End Points (7 bits)

Number of End Points implemented by the node. The value MUST be in the range 0..127.

If the device implements dynamic End Points, this field MUST advertise the number of End Points currently instantiated by the node. A dynamic End Point MAY be assigned any End Point identifier in the range 1..127.

4.84.6 Multi Channel Capability Get Command

The Multi Channel Capability Get Command is used to query the capabilities of End Points.

The Multi Channel Capability Report command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing.
The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_MULTI_CHANNEL							
Command = MULTI_CHANNEL_CAPABILITY_ GET							
Res	End Point						

Res

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

End Point (7 bits)

This field MUST specify the End Point which this capability request applies to.

If the specified End Point does not exist, this command MUST be ignored.
If the specified End Point represents a removed dynamic End Point, this command MUST be ignored.

4.84.7 Multi Channel Capability Report Command

The Multi Channel Capability Report Command is used to advertise the generic and specific device class of the End Point and the supported command classes.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_MULTI_CHANNEL</td></tr>
<tr><td colspan="8">Command = MULTI_CHANNEL_CAPABILITY_REPORT</td></tr>
<tr><td>Dynamic</td><td colspan="7">End Point</td></tr>
<tr><td colspan="8">Generic Device Class</td></tr>
<tr><td colspan="8">Specific Device Class</td></tr>
<tr><td colspan="8">Command Class 1</td></tr>
<tr><td colspan="8">Command Class 2</td></tr>
<tr><td colspan="8">…</td></tr>
<tr><td colspan="8">Command Class N</td></tr>
</table>

Dynamic (1 bit)

This field is used to advertise if the advertised End Point is dynamic.

This field MUST be set to 1 if this is a dynamic End Point.

When advertising the removal of a dynamic End Point, the Multi Channel Capability Report MUST carry the following values:

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_MULTI_CHANNEL</td></tr>
<tr><td colspan="8">Command = MULTI_CHANNEL_CAPABILITY_REPORT</td></tr>
<tr><td>Dynamic = 1</td><td colspan="7">End Point = (actual End Point)</td></tr>
<tr><td colspan="8">Generic Device Class =
GENERIC_TYPE_NON_INTEROPERABLE</td></tr>
<tr><td colspan="8">Specific Device Class =
SPECIFIC_TYPE_NOT_USED</td></tr>
</table>

The command MUST NOT advertise any Command Classes when advertising the removal of a dynamic End Point.

End Point (7 bits)

This field MUST advertise the End Point which this command relates to.

Generic Device class (8 bits)

The Generic Device Class of the advertised End Point.

Specific Device class (8 bits)

The Specific Device Class of the advertised End Point.

Command Class (N bytes)

This field is used to advertise command classes implemented by the device.
The field MAY advertise zero command classes. The number of Command Class bytes MUST be determined from the length field of the frame.

4.84.8 Multi Channel End Point Find Command

The Multi Channel End Point Find Command is used to search for a specific set of generic and specific device class in End Points.

The Multi Channel End Point Find Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_MULTI_CHANNEL							
Command = MULTI_CHANNEL_END_POINT_FIND							
Generic Device Class							
Specific Device Class							

Generic Device Class (8 bits)

The Generic Device Class that is to be searched in the Multi Channel device.

The value 0xFF MAY be specified for the Generic Device Class field to indicate that all existing End Points are to be reported.
If 0xFF is specified, the Specific Device Class field MUST also be set to 0xFF.

Specific Device Class (8 bits)

The Specific Device Class that is to be searched in the Multi Channel device.

The value 0xFF MAY be specified.
If 0xFF is specified, all devices with the specified Generic Device Class MUST be reported.

4.84.9 Multi Channel End Point Find Report Command

The Multi Channel End Point Find Report Command is used to advertise End Points which implement a given combination of Generic and Specific Device Classes.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_MULTI_CHANNEL</td></tr>
<tr><td colspan="8">Command = MULTI_CHANNEL_END_POINT_FIND_REPORT</td></tr>
<tr><td colspan="8">Reports to Follow</td></tr>
<tr><td colspan="8">Generic Device Class</td></tr>
<tr><td colspan="8">Specific Device Class</td></tr>
<tr><td>Res</td><td colspan="7">End Point 1</td></tr>
<tr><td colspan="8">…</td></tr>
<tr><td>Res</td><td colspan="7">End Point n</td></tr>
</table>

Reports to Follow (8 bits)

Multiple End Points may match the requested generic and specific device classes.

This field is used to advertise the number of outstanding End Point Find Report commands.

Generic Device Class (8 bits)

The Generic Device Class of all advertised End Points in this report.

The value 0xFF MUST be advertised, if this value was specified in the Multi Channel End Point Find Command.
If 0xFF is advertised, the Specific Device Class field MUST also advertise the value 0xFF.

If the value 0xFF is advertised, the advertised End Points MAY implement different generic and specific device classes.

Specific Device Class (8 bits)

The Specific Device Class of all advertised End Points in this report.

If the value 0xFF is advertised, the advertised End Points MAY implement different specific device classes.

The value 0xFF MUST be advertised, if the report advertises multiple End Points which implement the same generic device class but different specific device classes.

The value 0xFF MUST be advertised if the advertised generic device class is 0xFF.

Res

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

End Point (N * 7 bits)

The End Point(s) that matches the advertised generic and specific device class values.

If no End Point matches the advertised Generic Device Class and/or Specific Device Class requested in the Multi Channel End Point Find Command, the responding node MUST return a Multi Channel End Point Find Report Command with the End Point 1 field set to 0.

4.84.10 Multi Channel Command Encapsulation Command

The Multi Channel Command Encapsulation Command is used to encapsulate commands to of from a Multi Channel End Point.

The Multi Channel Command Encapsulation Command MUST NOT carry Source End Point and Destination End Point fields that are both zero.

A receiving node MAY respond to a Multi Channel encapsulated command if the Destination End Point field specifies a single End Point. In that case, the response MUST be Multi Channel encapsulated.

A receiving node MUST NOT respond to a Multi Channel encapsulated command if the Destination End Point field specifies multiple End Points via bit mask addressing.

A node MUST NOT return a Multi Channel Encapsulated command in response to a non-encapsulated command.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_MULTI_CHANNEL</td></tr>
<tr><td colspan="8">Command = MULTI_CHANNEL_CMD_ENCAP</td></tr>
<tr><td>Res</td><td colspan="7">Source End Point</td></tr>
<tr><td>Bit address</td><td colspan="7">Destination End Point</td></tr>
<tr><td colspan="8">Command Class</td></tr>
<tr><td colspan="8">Command</td></tr>
<tr><td colspan="8">Parameter 1</td></tr>
<tr><td colspan="8">...</td></tr>
<tr><td colspan="8">Parameter N</td></tr>
</table>

Res

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Source End Point (7 bits)

This field is used to advertise the originating End Point. The Source End Point MUST be in the range 0..127.

An End Point responding to a Multi Channel encapsulated command MUST swap the Source and Destination End Point identifiers in the Multi Channel encapsulated response that is returned.

If a sending device does not implement Multi Channel End Points or if the Root Device of the Multi Channel device is originating a command, the Source End Point MUST be set to 0.

Bit address (1 bit)

This bit is used to advertise if the destination End Point is specified as a bit mask.

The value 1 MUST indicate that the Destination End Point field is specified as a bit mask.

The value 0 MUST indicate that the Destination End Point field is specified as a single End Point.

Destination End Point (7 bits)

This field is used to advertise the destination End Point.

This field MUST be interpreted based on the “Bit address” value.

If the Bit Address field is set to 0, the Destination End Point field MUST carry a single End Point identifier value in the range 0..127.

If the Bit Address field is set to 1, the Destination End Point MUST use the following encoding.

- Bit 0 in the Destination End Point indicates if End Point 1 is a target
- Bit 1 in the Destination End Point indicates if End Point 2 is a target
- …

The bit value 0 MUST be used to advertise that the corresponding End Point is not a target.
The bit value 1 MUST be used to advertise that the corresponding End Point is a target.

Note that the mapping of bit 0 in bit masks is not consistent throughout Command Classes. Some map bit 0 to object identifier 0; others map bit 0 to object identifier 1.

Command Class (8 bits)

This field carries the Command Class identifier of the embedded command.

Command (8 bits)

This field carries the Command identifier of the embedded command.

Parameter (N bytes)

This field carries the parameter(s) of the embedded command. The number of parameters fields MUST be determined from the length field in the frame.

4.84.10.1 Supporting Dynamic End Points

When implementing a device supporting dynamic End Points the following guide lines must be followed for adding and removing End Points.

Adding an End Point

When adding a new End Point it MUST be numbered in succession. If the next consecutive number is 128, the device MUST search from the start of the list for the first empty entry.

Examples adding end points

The following examples show how End Points are inserted into the list of End Points.

Adding an End Point

End Point list before

1	Occupied
2	Occupied
3	Free
…	
126	Free
127	Free

End Point list after

1	Occupied
2	Occupied
3	Occupied
…	
126	Free
127	Free

Adding an End Point

End Point list before

1	Occupied
2	Free
3	Occupied
…	
126	Occupied
127	Free

End Point list after

1	Occupied
2	Free
3	Occupied
…	
126	Occupied
127	Occupied

Adding an End Point

End Point list before

1	Occupied
2	Free
3	Occupied
…	
126	Occupied
127	Occupied

End Point list after

1	Occupied
2	Occupied
3	Occupied
…	
126	Occupied
127	Occupied

Removing an End Point

When removing an End Point, the End Point MUST leave an empty spot in the End Point list, thus making sure there is no change in the numbering of the other End Points supported by the device.

Example removing end point

The following example show how End Points are removed from the list of End Points.

Remove End Point #2

End Point list before

1	Occupied
2	Occupied
3	Occupied
…	
126	Free
127	Free

End Point list after

1	Occupied
2	Free
3	Occupied
…	
126	Free
127	Free

4.85 Multi Channel Command Class, version 4

The Multi Channel command class is used to address one or more end points in a Multi Channel device. An application implementing this command class MUST set the Optional Functionality bit in the NIF.

A Multi Channel device MAY implement from 1 to 127 end points.

A controlling devices MAY use Multi Channel encapsulation to communicate with Multi Channel End Points in other devices. If such a controlling device does not implement any End Points, the device SHOULD NOT advertise the Multi Channel Command Class in the Node Information Frame.

Refer to 3.6 for an introduction to the Multi Channel concept.

4.85.1 Interoperability considerations

In order to communicate with Multi Channel End Points in other devices, a device MAY support the Multi Channel Command Class, version 3 without actually implementing any Multi Channel End Points. Such a device SHOULD NOT advertise support for the Multi Channel Command Class.

A typical example of such a device would be a gateway which can control End Points in other devices via commands from the Root Device of the gateway. Likewise, the Root Device of the gateway may receive unsolicited commands from End Points in other nodes.

4.85.2 Compatibility considerations

The Multi Channel Command Class, version 4 introduces Aggregated End Points.

Multi Channel Command Class, version 4 is backwards compatible with Multi Channel Command Class, version 3.

Aggregated End Points are assigned End Point identifiers following immediately after the identifiers allocated to individual End Points. Thus,

- Aggregated End Points are invisible to devices supporting Multi Channel Command Class, version 3 or earlier.
- Individual End Points are identical in version 3 and version 4.
- A version 3 device may discover and control individual End Points.
- A version 4 device may discover and control individual End Points and Aggregated End Points.

4.85.3 Aggregated End Point design principles

An Aggregated End Point MUST implement a function which relates to multiple individual End Points.

An Aggregated End Point MUST NOT forward commands to individual End Points. In other words, communication to a number of individual End Points MUST be done via multiple singlecast commands or via bit mask addressing.

A command issued to an Aggregated End Point MUST NOT cause any individual End Point to return a command in response.

Aggregated End Point MUST be assigned End Point identifiers from a continuous range starting immediately after the last individual End Point.

An Aggregated End Point MUST NOT implement other Command Classes and types than the ones explicitly listed in Table 109.

Table 109, Aggregated End Point Command Class support

Command Class	Type	Measurement mode
Meter	Electricity	Instant, Accumulated
Meter	Gas	Instant, Accumulated
Meter	Water	Instant, Accumulated
Multilevel Sensor	Power	Instant
Multilevel Sensor	Current	Instant
Multilevel Sensor	Air flow	Instant
Multilevel Sensor	Tank Capacity	Instant

4.85.4 Dynamic End Point considerations

In the case a node implements both Dynamic and Aggregated End Points, the Aggregated End Points will vary accordingly to the last active dynamic End Point. An illustration is given in Figure 35

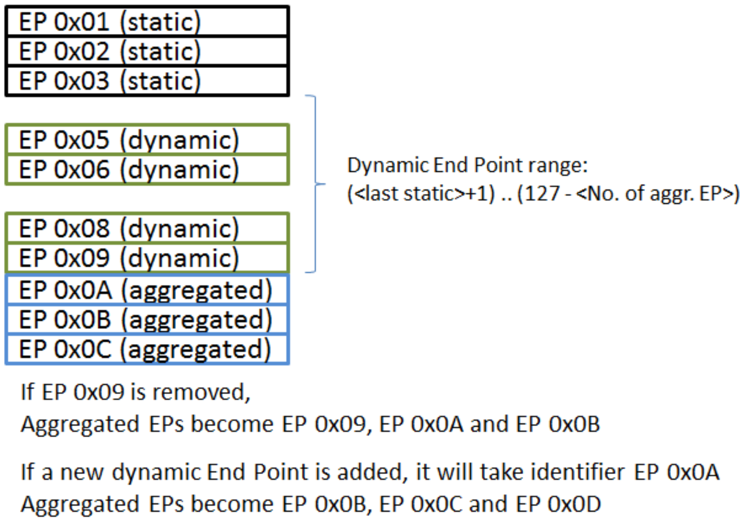


Figure 35, Static, dynamic and aggregated End Point layout example

A node receiving dynamic End Point modifications SHOULD learn again about the sending node End Points. The RECOMMENDED procedure is shown in Figure 36

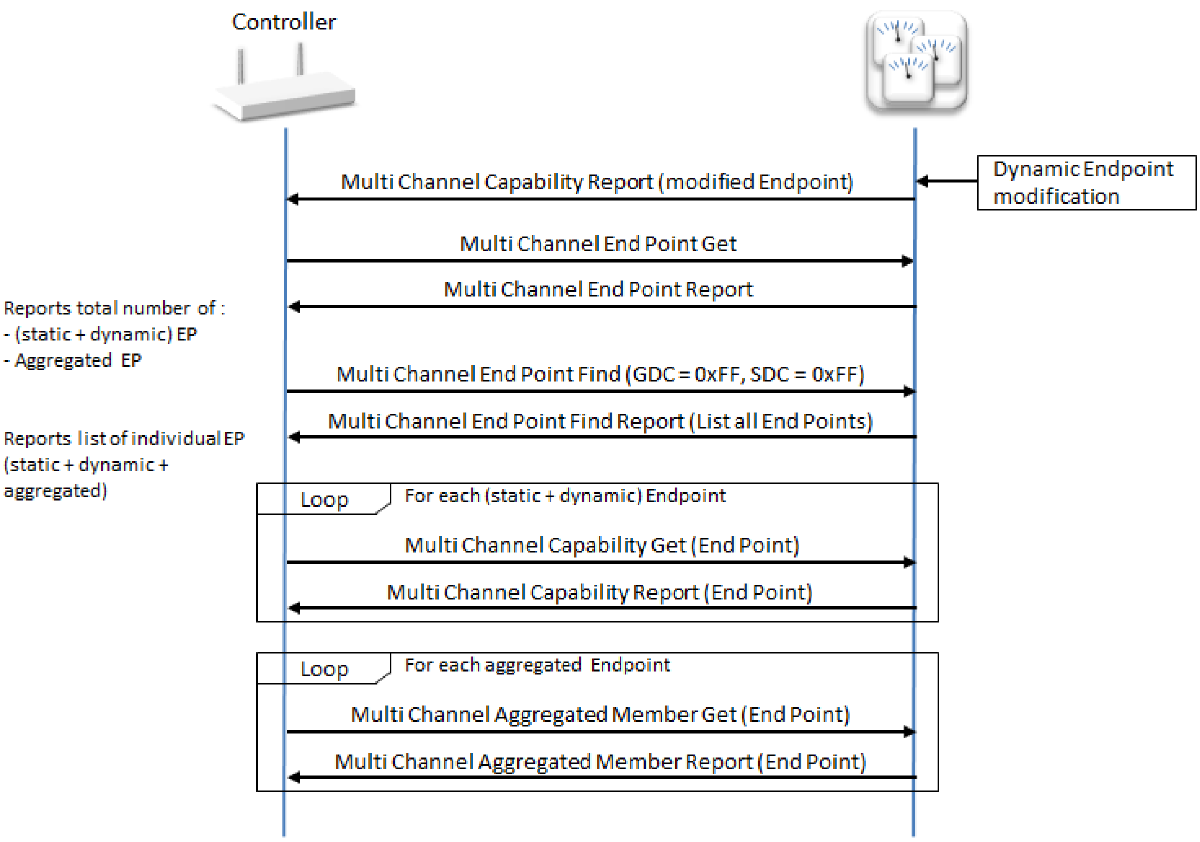


Figure 36, Dynamic End Point modification rediscovery (version 4)

4.85.5 Multi Channel End Point Get Command

The Multi Channel End Point Get Command is used to query the number of Multi Channel End Points and other relevant Multi Channel attributes.

The Multi Channel End Point Report MUST be returned in response to the Multi Channel End Point Get command.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_MULTI_CHANNEL							
Command = MULTI_CHANNEL_END_POINT_GET							

4.85.6 Multi Channel End Point Report Command

The Multi Channel End Point Report Command is used to advertise the number of Multi Channel End Points and other relevant Multi Channel attributes.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_MULTI_CHANNEL							
Command = MULTI_CHANNEL_END_POINT_REPORT							
Dyna-mic	Iden-tical	Res					
Res	Individual End Points						
Res	Aggregated End Points						

Dynamic (1 bit)

This field is set to 1 if the device has a dynamic number of end points. When the dynamic bit is set the number of end points in the device can change over time. Care should be taken when communicating with dynamic end points as the transmitter cannot be entirely sure the specific end point exists.

Identical (1 bit)

This bit is set to 1 if all End Points advertise the same generic and specific Device Class and optional Command Classes.

Res

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Individual End Points (7 bits)

This field MUST advertise the number of individual End Points implemented by this node.

The value MUST be in the range 1..127.
The sum of the values advertised by the Individual End Points and Aggregated End Points fields MUST be in the range 1..127.

Aggregated End Points (7 bits)

This field MUST advertise the number of Aggregated End Points implemented by this node.

The value MUST be in the range 0..127.
The sum of the values advertised by the Individual End Points and Aggregated End Points fields MUST be in the range 1..127.

If no Aggregated End Points are implemented, this field MUST advertise the value 0 (zero).

4.85.7 Multi Channel Capability Get Command

The Multi Channel Capability Get Command is used to query the capabilities of one individual End Point or Aggregated End Point.

The Multi Channel Capability Report command MUST be returned in response to the Multi Channel Capability Get command.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_MULTI_CHANNEL</td></tr>
<tr><td colspan="8">Command = MULTI_CHANNEL_CAPABILITY_ GET</td></tr>
<tr><td>Res</td><td colspan="7">End Point</td></tr>
</table>

Res

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

End Point (7 bits)

This field MUST specify a valid End Point as advertised by the Multi Channel End Point Report.

4.85.8 Multi Channel Capability Report Command

The Multi Channel Capability Report Command is used to advertise the generic and specific device class and the supported command classes of one End Point.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_MULTI_CHANNEL							
Command = MULTI_CHANNEL_CAPABILITY_REPORT							
Dynamic	End Point						
Generic Device Class							
Specific Device Class							
Command Class 1							
Command Class 2							
…							
Command Class N							

Dynamic (1 bit)

This field is set to one if this end point is a dynamic end point. When this bit is set in an end point, it cannot be assumed that it will reply to commands send to it because it could be gone again when a command is send to it.

Note: End point 1 cannot be dynamic.

End Point (7 bits)

This field MUST advertise a valid End Point as advertised by the Multi Channel End Point Report.

Generic Device class (8 bits)

The generic Device Class of the advertised End Point.

Specific Device class (8 bits)

The specific Device Class of the advertised End Point.

Command Class (N bytes)

This field advertises Command Classes supported or controlled by the End Point in question.

The Basic Command Class SHOULD NOT be advertised. Support for the Basic Command Class is implicitly expected for all Root Devices as well as End Points.

The number of Command Class fields MUST be determined from the length field in the frame.

4.85.9 Multi Channel End Point Find Command

The Multi Channel End Point Find Command is used to query individual End Points as well as Aggregated End Points for a specified set of generic and specific Device Class identifiers.

The Multi Channel End Point Find Report Command MUST be returned in response to the Multi Channel End Point Find command.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_MULTI_CHANNEL							
Command = MULTI_CHANNEL_END_POINT_FIND							
Generic Device Class							
Specific Device Class							

Generic Device Class (8 bits)

This field MUST specify the generic Device Class which an End Point has to match.

The value 0xFF MAY be specified for the Generic Device Class field to indicate that all existing End Points are to be reported.
If 0xFF is specified, the Specific Device Class field MUST also be set to 0xFF.

Specific Device Class (8 bits)

This field MUST specify the specific Device Class which an End Point has to match.

The value 0xFF MAY be specified. If specified, a responding device MUST treat the value as a wildcard; advertising all End Points which match the advertised Generic Device Class, regardless of the Specific Device Class of the individual End Point. If specified, the value 0xFF MUST also be advertised in the Multi Channel End Point Find Report Command.

4.85.10 Multi Channel End Point Find Report Command

The Multi Channel End Point Find Report Command used to reply to a Multi Channel End Point Find command.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_MULTI_CHANNEL							
Command = MULTI_CHANNEL_END_POINT_FIND_REPORT							
Reports to Follow							
Generic Device Class							
Specific Device Class							
Res	End Point 1						
…							
Res	End Point N						

For field description, refer to version 3 : 4.84.9 Multi Channel End Point Find Report Command.

4.85.11 Multi Channel Aggregated Members Get Command

The Multi Channel Aggregated Members Get Command is used to query the members of an Aggregated End Point.

The Multi Channel Aggregated Members Report MUST be returned in response to the Multi Channel Aggregated Members Get command.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_MULTI_CHANNEL							
Command = MULTI_CHANNEL_AGGREGATED_MEMBERS_GET							
Res	Aggregated End Point						

Res

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Aggregated End Point (7 bits)

This field MUST specify an Aggregated End Point.

The value MUST be in the range of advertised Aggregated End Points. If the value is not valid, a receiving node MUST return a Multi Channel Aggregated Members Report with the Number of Bit Masks field set to zero and no Bit Mask bytes.

4.85.12 Multi Channel Aggregated Members Report Command

The Multi Channel Aggregated Members Report Command is used to advertise the members of an Aggregated End Point.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_MULTI_CHANNEL							
Command = MULTI_CHANNEL_AGGREGATED_MEMBERS_REPORT							
Res	Aggregated End Point						
Number of Bit Masks							
Aggregated Members Bit Mask 1							
…							
Aggregated Members Bit Mask N							

Res

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Aggregated End Point (7 bits)

This field MUST advertise an Aggregated End Point.

If the command is returned in response to a Multi Channel Aggregated Members Get, this field MUST advertise the same value as was received in the Multi Channel Aggregated Members Get command.

Number of Bit Masks (8 bits)

This field MUST advertise the number of Aggregated Members Bit Mask bytes.

Aggregated Members Bit Mask (N bytes)

This bit mask is used to advertise the End Point members of the Aggregated End Point advertised in the Aggregated End Point field. The number of bit mask bytes MUST be determined from the Number of Bit Masks field.

This field MUST use the following encoding for advertising members.

- Bit 0 in Bit Mask 1 indicates if End Point 1 is a member
- Bit 1 in Bit Mask 1 indicates if End Point 2 is a member
- …

The bit value 0 MUST be used to advertise that the corresponding End Point is not a member.
The bit value 1 MUST be used to advertise that the corresponding End Point is a member.

The first byte of the Aggregated Members Bit Mask MUST represent End Points 1..8.

Note that the mapping of bit 0 in bit masks is not consistent throughout Command Classes. Some map bit 0 to object identifier 0; others map bit 0 to object identifier 1.

4.85.13 Multi Channel Command Encapsulation Command

The Multi Channel Command Encapsulation Command may be used to encapsulate commands. Any command supported by a Multi Channel End Point MAY be encapsulated using this command.

The Multi Channel Command Encapsulation Command MUST NOT carry Source End Point and Destination End Point fields that are both zero.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_MULTI_CHANNEL</td></tr>
<tr><td colspan="8">Command = MULTI_CHANNEL_CMD_ENCAP</td></tr>
<tr><td>Res</td><td colspan="7">Source End Point</td></tr>
<tr><td>Bit address</td><td colspan="7">Destination End Point</td></tr>
<tr><td colspan="8">Command Class</td></tr>
<tr><td colspan="8">Command</td></tr>
<tr><td colspan="8">Parameter 1</td></tr>
<tr><td colspan="8">...</td></tr>
<tr><td colspan="8">Parameter N</td></tr>
</table>

Res

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Source End Point (7 bits)

This field is used to advertise the originating End Point. The Source End Point MUST be in the range 0..127.

An End Point receiving a Multi Channel encapsulated Get command MUST swap the Source and Destination End Point identifiers in the Multi Channel encapsulated response returned to the requester.

If the Root Device is originating a command to an End Point in another node, the Source End Point MUST be set to 0.

Bit address (1 bit)

This bit is used to advertise if multiple Destination End Points(s) are specified as an End Point multicast bit mask.

This bit MUST be set to 1 if multicast addressing is used.
This bit MUST be set to 0 if one Destination End Point is specified individually.

End Point multicast bit mask addressing MAY be used for Destination End Points 1..7.
End Point multicast bit mask addressing MUST NOT be used for Destination End Points 8..127.

End Point multicast bit mask addressing MUST NOT be used for Get commands. An End Point receiving a command via End Point multicast bit mask addressing MUST NOT respond to the command.

Destination End Point (7 bits)

This field is used to advertise the destination End Point.

This field MUST be interpreted based on the “Bit address” value.

If the Bit Address field is set to 0, the Destination End Point field MUST carry a single End Point identifier value in the range 0..127.

If the Bit Address field is set to 1, the Destination End Point MUST use the following encoding.

- Bit 0 in the Destination End Point indicates if End Point 1 is a target
- Bit 1 in the Destination End Point indicates if End Point 2 is a target
- …

The bit value 0 MUST be used to advertise that the corresponding End Point is not a target.
The bit value 1 MUST be used to advertise that the corresponding End Point is a target.

Note that the mapping of bit 0 in bit masks is not consistent throughout Command Classes. Some map bit 0 to object identifier 0; others map bit 0 to object identifier 1.

Command Class (8 bits)

This field carries the Command Class identifier of the embedded command.

Command (8 bits)

This field carries the Command identifier of the embedded command.

Parameter (N bytes)

This field carries the parameter(s) of the embedded command. The number of Parameter fields MUST be determined from the length field in the frame.

If the encapsulated command is a request, the reply MUST be also be encapsulated.

It is allowed to encapsulate this command into another type of encapsulation, and to encapsulate another type of encapsulation into this command. Multiple encapsulations of the same type e.g. Multi Channel Command Encapsulations or Multi Command Encapsulations nested into each other MUST NOT be used.

4.86 Multi Channel Association Command Class, version 1 [OBSOLETED]

THIS VERSION HAS BEEN OBSOLETED

New implementations MUST use the Multi Channel Association Command Class, version 2 or newer.

4.87 Multi Channel Association Command Class, version 2

The Multi Channel Association Command Class is used to manage associations to Multi Channel End Point destinations as well as to NodeID destinations.

An association group sends an unsolicited command to the configured destinations when triggered by an event. The parameters of the command may be dynamic, e.g. the temperature of a sensor reading or the light level for a dimmer.

A NodeID association identifies its destination by a NodeID.
An End Point association identifies its destination by a combination of a NodeID and an End Point.

Refer to 3.6 for an introduction to the Multi Channel concept.

4.87.1 Compatibility considerations

The Multi Channel Association Command Class extends the functionality of the Association command class. A device supporting this Command Class version MUST also support the (non-Multi Channel) Association Command Class, version 2.

A controlling device SHOULD NOT create End Point associations to dynamic Multi Channel End Points.

The Association Group Information (AGI) Command Class SHOULD be supported to enable automated discovery of association group properties.

The Association and Multi Channel Association Command Classes MUST access the same collection of association groups. Further, the following applies:
The two command classes MUST advertise the total number of association groups.
The two command classes MUST advertise the total number of supported nodes for a given association group.
Any advertised association group MUST support NodeID destinations as well as End Point destinations.

NodeID associations maintained via the Multi Channel Association Command Class MUST be identical to NodeID associations maintained via the Association Command Class.

The Multi Channel Command Class specifies that, for backwards compatibility with non-Multi Channel devices, the Root Device of a Multi Channel device MUST mirror the functionality of End Point 1 and it MAY mirror the functionality of more End Points. This principle also applies to association groups advertised by the Root Device.
Each association group advertised by the Root Device, except for association group 1, SHOULD mirror an association group of an End Point.
If a Root Device association group mirrors an End Point association group, the Root Device SHOULD map all Association commands for that group to the mirrored End Point association group.

Except for association group 1 (the Z-Wave Plus Lifeline), a Multi Channel aware controlling device SHOULD ignore all Root Device association groups since they are just mirrored End Point association groups.

The use of encapsulation MUST comply with Table 110.

Table 110, V2 Associations and the transmissions they may trigger

Source → Destination		Association type	Allowed transmissions
NodeID → NodeID	(V2)	NodeID	Non-encapsulated
End Point → NodeID	(V2)	NodeID	Non-encapsulated
NodeID → End Point	(V2)	End Point	Encapsulated (src=0, dst>0)
End Point → End Point	(V2)	End Point	Encapsulated (src>0, dst>0)
~~NodeID → Root Device~~		~~End Point~~	*None *)*

*) Version 2 of this command class does not allow the destination End Point to be zero.

4.87.2 Z-Wave Plus considerations

The Z-Wave Plus certification program mandates that Association group 1 is reserved for the Lifeline association group. Group 1 MUST NOT be assigned to any other use than the Lifeline group. The actual Device Type specifies a mandatory list of commands which the device must be able to send to all lifeline group destinations. A manufacturer MAY add additional commands to the lifeline group.

End Points SHOULD NOT implement the Lifeline Association Group. End Points SHOULD report that zero NodeIDs are supported for association group 1.

The Z-Wave Plus certification program mandates support for the Association Group Information (AGI) Command Class if a device supports the Multi Channel Association Command Class.

4.87.3 Security considerations

A device which is included securely MUST NOT accept Association commands unless the commands are received via the highest security key assigned to the device.

4.87.4 Multi Channel Association Set Command

This command is used to request that one or more destinations are added to a given association group.

The destinations MAY be a mix of NodeID destinations and End Point destinations.

The receiving node SHOULD add the specified destinations to the specified association group.
This command MAY be ignored if the association group is already full.

Routing slaves MUST have return routes assigned to all association destinations.

Unless the association destination is a gateway, a controlling node SHOULD NOT create an association if the association destination node does not support the commands that the actual association group will be sending. The AGI Command Class SHOULD be used to probe the commands that a given association group will be sending.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_MULTI_CHANNEL_ASSOCIATION</td></tr>
<tr><td colspan="8">Command = MULTI_CHANNEL_ASSOCIATION_SET</td></tr>
<tr><td colspan="8">Grouping Identifier</td></tr>
<tr><td colspan="8">NodeID 1</td></tr>
<tr><td colspan="8">...</td></tr>
<tr><td colspan="8">NodeID M</td></tr>
<tr><td colspan="8">Marker = MULTI_CHANNEL_ASSOCIATION_SET_MARKER</td></tr>
<tr><td colspan="8">Multi Channel NodeID 1</td></tr>
<tr><td>Bit Address 1</td><td colspan="7">End Point 1</td></tr>
<tr><td colspan="8">…</td></tr>
<tr><td colspan="8">Multi Channel NodeID N</td></tr>
<tr><td>Bit Address N</td><td colspan="7">End Point N</td></tr>
</table>

Grouping Identifier (8 bits)

This field is used to specify the actual association group. Grouping Identifiers MUST be assigned in a consecutive range starting from 1.

A node that receives an unsupported Grouping Identifier MUST ignore this command

NodeID (M bytes)

This field specifies a list of NodeID destinations that are to be added to the specified association group as a NodeID association.

A NodeID association created via this field MUST be identical to a NodeID association created with the (non-Multi Channel) Association Set command.

Marker (8 bits)

This field is used to indicate the end of NodeID destinations and the start of End Point destinations. The Marker field MUST be set to the value MULTI_CHANNEL_ASSOCIATION_SET_MARKER. The field MAY be omitted if no End Point destinations are specified.

Multi Channel NodeID (N bytes)

The Multi Channel NodeID, Bit Address and End Point fields specify a list of End Points which are to be added to the specified association group as an End Point association.

The complete identification of an End Point destination requires a NodeID as well as an End Point identifier. This command MAY carry multiple copies of the same Multi Channel NodeID in combination with different End Point identifiers.

Bit Address + End Point (N bytes)

These fields MUST be processed in combination with the Multi Channel NodeID field.

The receiving node SHOULD treat the Bit Address flag and the End Point identifier as one scalar value; thus creating only one association group entry. Using a single scalar value enables:

- Better utilization of association group capacity
- Simple transmission of command for a multi-End Point destination
- Well-defined removal of bit addressed End Points

Refer to the Multi Channel Command Class for the actual encoding of bit addressed End Points.

The 7-bit End Point value MUST be in the range 1..127.

4.87.5 Multi Channel Association Get Command

This command is used to request the current destinations of a given association group.

The Multi Channel Association Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_MULTI_CHANNEL_ASSOCIATION							
Command = MULTI_CHANNEL_ASSOCIATION_GET							
Grouping Identifier							

Grouping Identifier (8 bits)

This field is used to specify the actual association group. Grouping Identifiers MUST be assigned in a consecutive range starting from 1.

A node that receives an unsupported Grouping Identifier SHOULD return information relating to Grouping Identifier 1.

4.87.6 Multi Channel Association Report Command

This command is used to advertise the current destinations for a given association group.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_MULTI_CHANNEL_ASSOCIATION							
Command = MULTI_CHANNEL_ASSOCIATION_REPORT							
Grouping Identifier							
Max Nodes Supported							
Reports to Follow							
NodeID 1							
…							
NodeID M							
Marker = MULTI_CHANNEL_ASSOCIATION_REPORT_MARKER							
Multi Channel NodeID 1							
Bit address 1	End Point 1						
…							

Multi Channel NodeID N	
Bit address N	End Point N

Grouping Identifier (8 bits)

This field is used to advertise the actual association group. Grouping Identifiers MUST be assigned in a consecutive range starting from 1.

Max Nodes Supported (8 bits)

The maximum number of destinations supported by the advertised association group. Each destination MAY be a NodeID destination or an End Point destination.

Reports to Follow (8 bits)

The entire list destinations of the advertised association group may be too long for one command. This field MUST advertise how many report frames will follow this report.

NodeID (M bytes)

This field advertises a list of NodeID destinations of the advertised association group.
The list of NodeIDs MUST be empty if there are no NodeID destinations configured for the advertised association group.

Marker (8 bits)

Refer to description under the Multi Channel Association Set command.

Multi Channel NodeID (N bytes)

The Multi Channel NodeID, Bit Address and End Point fields specify a list of End Points which are currently in the advertised association group.

The complete identification of an End Point requires a NodeID as well as an End Point identifier. The Multi Channel Association Report command MAY carry multiple copies of the same Multi Channel NodeID in combination with different End Point identifiers.

The list of Multi Channel NodeID and End Point identifiers MUST be empty if there are no End Point destinations configured for the advertised association group.

Bit address + End Point (N bytes)

These fields MUST be processed in combination with the Multi Channel NodeID field.

Refer to the Multi Channel Command Class for the actual encoding of bit addressed End Points.

4.87.7 Multi Channel Association Remove Command

This command is used to remove NodeID and End Point destinations from a given association group.

This command MUST manipulate the same list of NodeID destinations as the Association Remove Command.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_MULTI_CHANNEL_ASSOCIATION</td></tr>
<tr><td colspan="8">Command = MULTI_CHANNEL_ASSOCIATION_REMOVE</td></tr>
<tr><td colspan="8">Grouping Identifier</td></tr>
<tr><td colspan="8">NodeID 1</td></tr>
<tr><td colspan="8">..</td></tr>
<tr><td colspan="8">NodeID M</td></tr>
<tr><td colspan="8">Marker = MULTI_CHANNEL_ASSOCIATION_REMOVE_MARKER</td></tr>
<tr><td colspan="8">Multi Channel NodeID 1</td></tr>
<tr><td>Bit address 1</td><td colspan="7">End Point 1</td></tr>
<tr><td colspan="8">…</td></tr>
<tr><td colspan="8">Multi Channel NodeID N</td></tr>
<tr><td>Bit address N</td><td colspan="7">End Point N</td></tr>
</table>

Grouping Identifier (8 bits)

This field is used to specify from which association group the specified destinations should be removed.

Grouping Identifiers MUST be assigned in a consecutive range starting from 1.

A receiving node MUST ignore an unsupported Grouping Identifier; except for the value 0.

This field MUST be interpreted in combination with the NodeID and End Point fields.

(NodeID Destination) NodeID (M bytes)
(End Point Destination) Multi Channel NodeID (N bytes)
(End Point Destination) End Point (N bytes)

These fields specify the destinations that are to be removed.
The Grouping Identifier and these fields MUST be interpreted as follows.

“NodeID Destinations” denote the NodeID fields.
“End Point destinations” denote the combined Multi Channel NodeID and End Point fields found after the marker.

A receiving node MAY interpret the empty command (only Command Class and Command fields) as an instruction to Remove all NodeID destinations and End Point destinations from all association groups.

Table 111, Multi Channel Association Remove, V2 :: Parameter interpretation

Grouping identifier	Number of NodeID Destinations	Number of End Point Destinations	
> 0	> 0	(don't care)	Remove NodeID destinations from association group. Then remove End Points (next line) – if any
> 0	(don't care)	> 0 *)	Remove End Point destinations from association group *).
> 0	= 0	= 0	Remove all NodeID destinations and End Point destinations from association group.
= 0	> 0	(don't care)	Remove NodeID destinations from all association groups. Also remove End Point destinations (next line).
= 0	(don't care)	> 0 *)	Remove End Point destinations from all association groups *).
= 0	= 0	= 0	Remove all NodeID destinations and End Point destinations from all association groups.

*) The receiving node MUST treat the Bit Address flag and the End Point identifier as one scalar value. The receiving node MUST remove End Points if an exact match can be found, while it MAY ignore the removal request if an exact match cannot be found; even though there may exist associations for individual End Point destinations which are covered by the End Point bitmap address.

Marker (8 bits)

Refer to the description in 4.87.4 in the Multi Channel Association Set Command.

4.87.7.1 Examples

Remove specified NodeID from Group 3

Get association settings for Group 3

7	6	5	4	3	2	1	0
COMMAND_CLASS_ MULTI_CHANNEL_ASSOCIATION							
MULTI_CHANNEL_ASSOCIATION_GET							
Grouping Identifier = 3							

Current association settings for Group 3

7	6	5	4	3	2	1	0
COMMAND_CLASS_ MULTI_CHANNEL_ASSOCIATION							
MULTI_CHANNEL_ASSOCIATION_REPORT							
Grouping Identifier = 3							
Max Nodes Supported = 5							
Reports to Follow = 0							
NodeID = 11							
NodeID = 12							
MULTI_CHANNEL_ ASSOCIATION_REPORT_MARKER							

Multi Channel NodeID = 16
End Point = 2
Multi Channel NodeID = 16
End Point = 3
Multi Channel NodeID = 17
End Point = 2

Remove NodeID = 12 from Group 3

7	6	5	4	3	2	1	0
COMMAND_CLASS_ MULTI_CHANNEL_ASSOCIATION							
MULTI_CHANNEL_ASSOCIATION_REMOVE							
Grouping Identifier = 3							
NodeID = 12							

New association settings for Group 3

7	6	5	4	3	2	1	0
COMMAND_CLASS_ MULTI_CHANNEL_ASSOCIATION							
MULTI_CHANNEL_ASSOCIATION_REPORT							
Grouping Identifier = 3							
Max Nodes Supported = 5							
Reports to Follow = 0							
NodeID = 11							
MULTI_CHANNEL_ ASSOCIATION_REPORT_MARKER							
Multi Channel NodeID = 16							
End Point = 2							
Multi Channel NodeID = 16							
End Point = 3							
Multi Channel NodeID = 17							
End Point = 2							

Remove specified End Point from Group 3

Get association settings for Group 3

7	6	5	4	3	2	1	0
COMMAND_CLASS_ MULTI_CHANNEL_ASSOCIATION							
MULTI_CHANNEL_ASSOCIATION_GET							
Grouping Identifier = 3							

Current association settings for Group 3

7	6	5	4	3	2	1	0
COMMAND_CLASS_ MULTI_CHANNEL_ASSOCIATION							
MULTI_CHANNEL_ASSOCIATION_REPORT							
Grouping Identifier = 3							
Max Nodes Supported = 5							
Reports to Follow = 0							
NodeID = 11							
NodeID = 12							
MULTI_CHANNEL_ ASSOCIATION_REPORT_MARKER							
Multi Channel NodeID = 16							
End Point = 2							
Multi Channel NodeID = 16							
End Point = 3							
Multi Channel NodeID = 17							
End Point = 2							

Remove Multi Channel NodeID = 16,3

7	6	5	4	3	2	1	0
COMMAND_CLASS_ MULTI_CHANNEL_ASSOCIATION							
MULTI_CHANNEL_ASSOCIATION_REMOVE							
Grouping Identifier = 3							
MULTI_CHANNEL_ ASSOCIATION_REPORT_MARKER							
Multi Channel NodeID = 16							
End Point = 3							

New association settings for Group 3

7	6	5	4	3	2	1	0
COMMAND_CLASS_ MULTI_CHANNEL_ASSOCIATION							
MULTI_CHANNEL_ASSOCIATION_REPORT							
Grouping Identifier = 3							
Max Nodes Supported = 5							
Reports to Follow = 0							
NodeID = 11							
NodeID = 12							
MULTI_CHANNEL_ ASSOCIATION_REPORT_MARKER							
Multi Channel NodeID = 16							
End Point = 2							
Multi Channel NodeID = 17							
End Point = 2							

Remove bit addressable End Points from Group 3

Get association settings for Group 3

7	6	5	4	3	2	1	0
COMMAND_CLASS_ MULTI_CHANNEL_ASSOCIATION							
MULTI_CHANNEL_ASSOCIATION_GET							
Grouping Identifier = 3							

Current association settings for Group 3

7	6	5	4	3	2	1	0
COMMAND_CLASS_ MULTI_CHANNEL_ASSOCIATION							
MULTI_CHANNEL_ASSOCIATION_REPORT							
Grouping Identifier = 3							
Max Nodes Supported = 5							
Reports to Follow = 0							
NodeID = 11							
NodeID = 12							
MULTI_CHANNEL_ ASSOCIATION_REPORT_MARKER							
Multi Channel NodeID = 16							
End Point = 134 (b1000 0110)							
Multi Channel NodeID = 17							
End Point = 2							

Remove Multi Channel NodeID = 16,2 and 16,3

7	6	5	4	3	2	1	0
COMMAND_CLASS_ MULTI_CHANNEL_ASSOCIATION							
MULTI_CHANNEL_ASSOCIATION_REMOVE							
Grouping Identifier = 3							
MULTI_CHANNEL_ ASSOCIATION_REPORT_MARKER							
Multi Channel NodeID = 16							
End Point = 134 (b1000 0110)							

Note: The Multi Channel Node MUST report end points as bit addressable in order to be removed in the same manners.

New association settings for Group 3

7	6	5	4	3	2	1	0
COMMAND_CLASS_ MULTI_CHANNEL_ASSOCIATION							
MULTI_CHANNEL_ASSOCIATION_REPORT							
Grouping Identifier = 3							
Max Nodes Supported = 5							
Reports to Follow = 0							
NodeID = 11							
NodeID = 12							
MULTI_CHANNEL_ ASSOCIATION_REPORT_MARKER							
Multi Channel NodeID = 17							
End Point = 2							

Remove specified NodeID and End Point from Group 3

Get association settings for Group 3

7	6	5	4	3	2	1	0
COMMAND_CLASS_ MULTI_CHANNEL_ASSOCIATION							
MULTI_CHANNEL_ASSOCIATION_GET							
Grouping Identifier = 3							

Current association settings for Group 3

7	6	5	4	3	2	1	0
COMMAND_CLASS_ MULTI_CHANNEL_ASSOCIATION							
MULTI_CHANNEL_ASSOCIATION_REPORT							
Grouping Identifier = 3							
Max Nodes Supported = 5							
Reports to Follow = 0							
NodeID = 11							
NodeID = 12							
MULTI_CHANNEL_ ASSOCIATION_REPORT_MARKER							
Multi Channel NodeID = 16							
End Point = 2							
Multi Channel NodeID = 16							
End Point = 3							
Multi Channel NodeID = 17							
End Point = 2							

Remove NodeID = 12 & Multi Channel NodeID = 16,3

7	6	5	4	3	2	1	0
COMMAND_CLASS_ MULTI_CHANNEL_ASSOCIATION							
MULTI_CHANNEL_ASSOCIATION_REMOVE							
Grouping Identifier = 3							
NodeID = 12							
MULTI_CHANNEL_ ASSOCIATION_REPORT_MARKER							
Multi Channel NodeID = 16							
End Point = 3							

New association settings for Group 3

7	6	5	4	3	2	1	0
COMMAND_CLASS_ MULTI_CHANNEL_ASSOCIATION							
MULTI_CHANNEL_ASSOCIATION_REPORT							
Grouping Identifier = 3							
Max Nodes Supported = 5							
Reports to Follow = 0							
NodeID = 11							
MULTI_CHANNEL_ ASSOCIATION_REPORT_MARKER							
Multi Channel NodeID = 16							
End Point = 2							
Multi Channel NodeID = 17							
End Point = 2							

Remove all associations from Group 3

Get association settings for Group 3

7	6	5	4	3	2	1	0
COMMAND_CLASS_ MULTI_CHANNEL_ASSOCIATION							
MULTI_CHANNEL_ASSOCIATION_GET							
Grouping Identifier = 3							

Current association settings for Group 3

7	6	5	4	3	2	1	0
COMMAND_CLASS_ MULTI_CHANNEL_ASSOCIATION							
MULTI_CHANNEL_ASSOCIATION_REPORT							
Grouping Identifier = 3							
Max Nodes Supported = 5							
Reports to Follow = 0							
NodeID = 11							
NodeID = 12							
MULTI_CHANNEL_ ASSOCIATION_REPORT_MARKER							
Multi Channel NodeID = 16							
End Point = 2							
Multi Channel NodeID = 16							
End Point = 3							
Multi Channel NodeID = 17							
End Point = 2							

Remove all NodeIDs from Group 3

7	6	5	4	3	2	1	0
COMMAND_CLASS_ MULTI_CHANNEL_ASSOCIATION							
MULTI_CHANNEL_ASSOCIATION_REMOVE							
Grouping Identifier = 3							

New association settings for Group 3

7	6	5	4	3	2	1	0
COMMAND_CLASS_ MULTI_CHANNEL_ASSOCIATION							
MULTI_CHANNEL_ASSOCIATION_REPORT							
Grouping Identifier = 3							
Max Nodes Supported = 5							
Reports to Follow = 0							

Remove specific NodeID from all Groups

7	6	5	4	3	2	1	0
COMMAND_CLASS_ MULTI_CHANNEL_ASSOCIATION							
MULTI_CHANNEL_ASSOCIATION_REMOVE							
Grouping Identifier = 0							
NodeID = 12							

Remove specific Multi Channel NodeID from all Groups

7	6	5	4	3	2	1	0
COMMAND_CLASS_ MULTI_CHANNEL_ASSOCIATION							
MULTI_CHANNEL_ASSOCIATION_REMOVE							
Grouping Identifier = 0							
MULTI_CHANNEL_ ASSOCIATION_REPORT_MARKER							
Multi Channel NodeID = 16							
End Point = 3							

Remove all associations from all Groups

7	6	5	4	3	2	1	0
COMMAND_CLASS_ MULTI_CHANNEL_ASSOCIATION							
MULTI_CHANNEL_ASSOCIATION_REMOVE							

or

7	6	5	4	3	2	1	0
COMMAND_CLASS_ MULTI_CHANNEL_ASSOCIATION							
MULTI_CHANNEL_ASSOCIATION_REMOVE							
Grouping Identifier = 0							

4.87.8 Multi Channel Association Supported Groupings Get Command

This command is used to request the number of association groups that this node supports.

The Multi Channel Association Supported Groupings Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing.
The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_MULTI_CHANNEL_ASSOCIATION							
Command = MULTI_CHANNEL_ASSOCIATION_GROUPINGS_GET							

4.87.9 Multi Channel Association Supported Groupings Report Command

This command is used to advertise the maximum number of association groups implemented by this node.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_MULTI_CHANNEL_ASSOCIATION							
Command = MULTI_CHANNEL_ASSOCIATION_GROUPINGS_REPORT							
Supported Groupings							

Supported Groupings (8 bits)

This field is used to advertise the number of association groups that this node supports.

Grouping Identifiers MUST be assigned in a consecutive range starting from 1.

The value advertised by this field MUST be the total number of supported association groups; whether associations are managed via the Association Command Class or the Multi Channel Association Command Class. Each of these association groups MUST support NodeID destinations as well as End Point destinations.

4.88 Multi Channel Association Command Class, version 3

The Multi Channel Association Command Class is used to manage associations to Multi Channel End Point destinations as well as to NodeID destinations.

The following sections specify commands which were extended or added in version 3.

4.88.1 Compatibility considerations

The considerations of section 4.87.1 also apply to this version the following additions:

A device supporting this Command Class version MUST also

- support the Multi Channel Association Command Class, version 2
- support the (non-Multi Channel) Association Command Class, version 2

This version introduces support for the creation of an End Point Association to the Root Device of another device, e.g. a gateway, by allowing the destination End Point 0 as a valid value.
This allows a gateway to create a single Lifeline association from a Multi Channel device and subsequently receive status messages or sensor reports from the Root Device as well as the End Points of that device. Refer to section 4.88.2.

The use of Multi Channel encapsulation MUST comply with Table 112.

Table 112, V3 Associations and the transmissions they may trigger

Source → Destination		Association type	Allowed transmissions
NodeID → NodeID	(V2)	NodeID	Non-encapsulated
End Point → NodeID	(V2)	NodeID	Non-encapsulated
NodeID → End Point	(V2)	End Point	Encapsulated (src=0, dst>0)
End Point → End Point	(V2)	End Point	Encapsulated (src>0, dst>0)
End Point → Root Device	(V3)	End Point	Encapsulated (src>0, dst=0)
Root Device → Root Device	(V3)	End Point	Non-encapsulated *) or Encapsulated (src>0, dst=0)

*) Command must be sent non-encapsulated if both End Points are zero. However, the End Point association from a Root Device Lifeline group to a Root Device also allows a source End Point to send encapsulated commands to the Root Device Lifeline destination. Refer to 4.88.2.

4.88.2 Z-Wave Plus considerations

The considerations of version 2 also apply to this version.

Version 3 adds support for creation of an End Point Association to the Root Device of another device, e.g. a gateway.

A controlling device MUST verify that a device supports Multi Channel Association Command Class, Version 3 before creating an End Point Association to the Root Device of another device.

Multi Channel End Points may implement functionality relevant to a Lifeline destination, e.g. individual meter reports from a power strip. Multi Channel encapsulation allows the Lifeline destination to distinguish between apparently identical commands from different End Points.

As specified by the Multi Channel Command Class, a Root Device must not use Multi Channel encapsulation when communicating to another Root Device, i.e. if both End Points are zero. It does however still make sense to create an End Point Association from one Root Device to another Root Device, e.g. a gateway.

Consider, as an example, a two-channel temperature sensor. By creating an End Point association from the Lifeline Association Group to the Root Device of a gateway, the gateway may receive unsolicited Multi Channel encapsulated sensor reports from each of the sensor End Points as well as non-encapsulated battery status messages from the sensor Root Device.

The Root Device of a Multi Channel device SHOULD implement support for the Multi Channel Association Command Class if any of the End Points provide association capabilities. If the Root Device implements such support, the Root Device MAY forward Multi Channel encapsulated commands to the association destination on behalf of End Points.

A NodeID association MUST NOT be created in response to a request for an End Point association to the Root Device (destination End Point 0).
If a NodeID Association is created from the Root Device Lifeline Association Group, End Point commands MUST NOT be transmitted via the Lifeline Association Group of the root device. All commands originating from the Root Device (Tamper Alarm, Device Reset Locally, etc.) MUST still be sent non-encapsulated to the Lifeline destination.
If a NodeID association is created, two End Points MUST NOT forward identical commands (e.g. Multilevel Sensor Report::Temperature) via the Lifeline group, as the lack of source End Point

information will prevent a receiving application from distinguishing the individual commands from each other.

If an End Point Association is created from the Lifeline group, End Points MUST send relevant Multi Channel encapsulated commands to the Lifeline group destination.

A Multi Channel device MAY forward commands from multiple Multi Channel End Points to the Lifeline group destination.

4.88.3 Security considerations

The considerations of section 4.87.3 also apply to this version.

4.88.4 Multi Channel Association Set Command

This command is used to request that one or more destinations are added to a given association group.

The destinations MAY be a mix of NodeID destinations and End Point destinations.

The receiving node SHOULD add the specified destinations to the specified association group.
This command MAY be ignored if the association group is already full.

Routing slaves MUST have return routes assigned to all association destinations.

Unless the association destination is a gateway, a controlling node SHOULD NOT create an association if the association destination node does not support the commands that the actual association group will be sending. The AGI Command Class SHOULD be used to probe the commands that a given association group will be sending.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_MULTI_CHANNEL_ASSOCIATION							
Command = MULTI_CHANNEL_ASSOCIATION_SET							
Grouping Identifier							
NodeID 1							
...							
NodeID M							
Marker = MULTI_CHANNEL_ASSOCIATION_SET_MARKER							
Multi Channel NodeID 1							
Bit address 1	End Point 1						
…							
Multi Channel NodeID N							
Bit address N	End Point N						

Grouping Identifier (8 bits)

This field is used to specify the actual association group.

A node that receives an unsupported Grouping Identifier MUST ignore this command.

NodeID (M bytes)

The NodeID fields specify a list of NodeID destinations that are to be added to the specified association group as a NodeID association.

A NodeID association created via this field MUST be identical to a NodeID association created with the (non-Multi Channel) Association Set command.

Marker (8 bits)

This field is used to indicate the end of NodeID destinations and the start of End Point destinations. The Marker field MUST be set to the value MULTI_CHANNEL_ASSOCIATION_SET_MARKER. The field MAY be omitted if no End Point destinations are specified.

Multi Channel NodeID (N bytes)

The Multi Channel NodeID, Bit Address and End Point fields specify a list of End Point destinations which are to be added to the specified association group as an End Point association.

The complete identification of an End Point destination requires a NodeID as well as an End Point identifier. This command MAY carry multiple copies of the same Multi Channel NodeID in combination with different End Point identifiers.

Bit address + End Point (N bytes)

These fields MUST be processed in combination with the Multi Channel NodeID field.

The receiving node SHOULD treat the Bit Address flag and the End Point identifier as one scalar value; thus creating only one association group entry. Using a single scalar value enables:

- Better utilization of association group capacity
- Simple transmission of command for a multi-End Point destination
- Well-defined removal of bit addressed End Points

Refer to the Multi Channel Command Class for the actual encoding of bit addressed End Points.

The 7-bit End Point value MUST be in the range 0..127.

4.89 Multi Command Command Class, version 1

This Multi Command Command Class is used to encapsulate multiple Commands in one Command. The purpose for this command class is to limit the number of transmissions when a number of Commands need to be executed sequentially in a device. This command class may be used to extend battery lifetime.

A device that supports the Multi Command Command Class MAY receive a combination of encapsulated and normal non-encapsulated requests over time. The response MUST be as follows:

a) If the request is sent encapsulated, the response MUST be returned encapsulated.

b) If the request is sent non-encapsulated, the response MUST also be sent non-encapsulated.

Before sending an encapsulated command to another device, the sending device MUST ensure, that the destination supports the Multi Command Command Class. When associations are created to another destination, the controlling device MUST ensure, that the association destination supports the Multi Command Command Class.

A device MUST also be able to decode the encapsulated responses that may be returned in response to encapsulated request.

A device supporting the Multi Command Command Class MUST support all command classes advertised in the NIF, regardless of whether the commands are received as plain commends or encapsulated in the Multi Command Encapsulated Command.

4.89.1 Multi Command Encapsulated Command

The Multi Command Encapsulated Command used to contain multiple Commands. The encapsulated Commands MUST be executed in the order they are received. In case get Commands in a Multi Command Encapsulated Command are received by a device the reports MUST be replied in a Multi Command Encapsulated Command in the same order as the gets were received. Be aware of the payload limitations with respect to a routed single cast frame.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_MULTI_CMD							
Command = MULTI_CMD_ENCAP							
Number of Commands							
Command Length 1							
Command Class 1							
Command 1							
Data 1,1							
…							
Data 1,N							
…							
Command Length X							
Command Class X							
Command X							
Data X,1							
…							
Data X,N							

Number of Commands (8 bits)

This field indicates the number of encapsulated Commands.

Command Length (8 bits)

The Command Length field indicates the number of bytes used by the following encapsulated Command including Command Class identifiers, Command identifiers and the data.

Command Class (8 bits)

The Command Class field indicates the command class identifier of the encapsulated Command.

Command (8 bits)

The Command field indicates the Command identifier of the encapsulated Command.

Data (N bytes)

The Data fields in the encapsulated Command of the respective command class and Command identifiers.

4.89.2 Example

This example shows how a battery operated device can be instructed to go to sleep immediately after an internal parameter have been changed by one frame using the Multi Command Encapsulated Command.

7	6	5	4	3	2	1	0
COMMAND_CLASS_MULTI_CMD							
MULTI_CMD_ENCAP							
Number of Commands = 0x02							
Length = 0x05							
COMMAND_CLASS_CONFIGURATION							
CONFIGURATION_SET							
Parameter Number = 0x01							
Size = 0x01							
Configuration Value = 0x07							
Length = 0x02							
COMMAND_CLASS_WAKE_UP							
WAKE_UP_NO_MORE_INFORMATION							

In this example the Multi Command Encapsulated Command contains two encapsulated Commands. First the internal parameter no. 1 is changed to 0x07 by the Configuration Set Command and afterwards the Wake Up No More Information Command instructs the battery operated device to go to sleep immediately after, thereby minimizing the power consumption.

4.90 Multilevel Sensor Command Class, version 1-4

The Multilevel Sensor Command Class is used to control a multilevel sensor.

4.90.1 Multilevel Sensor Get Command

The Multilevel Sensor Get Command is used to request the level of a multilevel sensor.

The Multilevel Sensor Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SENSOR_MULTILEVEL							
Command = SENSOR_MULTILEVEL_GET							

4.90.2 Multilevel Sensor Report Command

The Multilevel Sensor Report Command is used by a multilevel sensor to advertise a sensor reading.

The Multilevel Sensor Report Command version 2 and 3 are extensions with respect to Sensor Types and associated Scales.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SENSOR_MULTILEVEL							
Command = SENSOR_MULTILEVEL_REPORT							
Sensor Type							
Precision			Scale		Size		
Sensor Value 1							
…							
Sensor Value N							

Sensor Type (8 bits)

Sensor Type specifies what type of sensor this Command originates from. Refer to Multilevel Sensor version 5 with respect to defined sensors. Sensor types are defined by the Z-Wave Alliance.

Precision (3 bits)

The precision field describes what the precision of the sensor value is. The number indicates the number of decimals. The decimal value 1025 with precision 2 is therefore equal to 10.25.

Scale (2 bits)

The Scale used to indicate what unit the sensor uses. Refer to Multilevel Sensor version 5 with respect to defined scales for the relevant sensors. Scales are defined by the Z-Wave Alliance.

Size (3 bits)

The size field indicates the number of bytes that used for the sensor value. This field can take values from 1 (001b), 2 (010b) or 4 (100b).

Sensor Value (N bytes)

The Sensor Value is a signed field. The field MAY be 1, 2 or 4 bytes in size. The first byte is the most significant byte. The first byte is the most significant byte. The table below shows signed decimal values together with their hexadecimal equivalents.

Signed 1 byte decimal value	Hexadecimal		Signed 2 bytes decimal value	Hexadecimal
127	0x7F		32767	0x7FFF
25	0x19		1025	0x0401
2	0x02		2	0x0002
1	0x01		1	0x0001
0	0x00		0	0x0000
-1	0xFF		-1	0xFFFF
-2	0xFE		-2	0xFFFE
-25	0xE7		-1025	0xFBFF
-128	0x80		-32768	0x8000

Notice: The device receiving the Multilevel Sensor Report MUST always show the sensor value even though the Sensor Type and/or Scale are not supported.

4.91 Multilevel Sensor Command Class, Version 5-10

The Multilevel Sensor Command Class allows a sensor device to issue readings to another device.

4.91.1 Compatibility considerations

Version 5 of this command class is extended with the following functionality:

- A “get-supported” mechanism for the controlling device to interview the multilevel sensor for its supported sensor types and/or scales
- Additional sensor type and scale fields to the Multilevel Sensor Get command to request for a specific sensor report
- Additional sensor types and/or scales to the list of multilevel sensors

Version 6 of this command class is extended with the following functionality:

- Additional sensor types

Version 7 of this command class is extended with the following functionality:

- Additional sensor types

Version 8 of this command class is extended with the following functionality:

- New Sensor Types and Scale Values for rotation and linear movement.
- New Sensor Types and Scale Values for Smoke Density

Multilevel Sensor Command Class, version 8 deprecates the Sensor Type “Angle Position”.

Version 9 of this command class is extended with the following functionality:

- New Sensor Types and Scale Values for Water Flow and Water Pressure
- New Sensor Types and Scale Values for RF Signal Strength

Version 10 of this command class is extended with the following functionality:

- New Sensor Types and Scale Values for Particulate Matter 10 and Respiratory rate
- New Scale Values for CO and VOC Sensors Types

4.91.2 Multilevel Sensor Get Supported Sensor Command

This command is introduced in v5 of the Multilevel Sensor command class and used to retrieve the supported sensor types from the multilevel sensor device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SENSOR_MULTILEVEL							
Command = SENSOR_MULTILEVEL_SUPPORTED_GET_SENSOR							

4.91.3 Multilevel Sensor Supported Sensor Report Command

This command is used to advertise the supported sensor types of a multilevel sensor device.

This command is introduced in v5 of the Multilevel Sensor command class. The command MUST be returned in response to the Multilevel Sensor Get Supported Sensor command.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SENSOR_MULTILEVEL							
Command = SENSOR_MULTILEVEL_SUPPORTED_SENSOR_REPORT							
Bit Mask 1							
…							
Bit Mask N							

Bit Mask (N bytes)

The Bit Mask field describes the supported sensor types by the multilevel sensor device and refers to the Sensor Type table of Multilevel Sensor Report command.

The Bit Mask fields advertise the supported Sensor Type(s).

- Bit 0 in Bit Mask 1 indicates if Sensor Type = Air Temperature (0x01) is supported
- Bit 1 in Bit Mask 1 indicates if Sensor Type = General Purpose (0x02) is supported
- Bit 2 in Bit Mask 1 indicates if Sensor Type = luminance (0x03) is supported
- …

The Notification Type is supported if the corresponding bit is set to '1' and the opposite if set to '0'. Notification Type = 0xFF MUST NOT be indicated by the Bit Masks fields.

It is only necessary to send the Bit Mask fields from 1 and up to the Bit Mask N indicating the last supported Sensor Type. The number of Bit Mask fields transmitted MUST be determined from the length field in the frame.

Note that the mapping of bit 0 to Sensor Type = 1 differs from the support mapping used by the Notification Command Class. The Notification Command Class maps bit 1 to Notification Type =1.

Example:

To indicate sensor type support for Relative Humidity, Luminance and Temperature the Multilevel Sensor Supported Sensor Report command MUST be structured as illustrated below.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SENSOR_MULTILEVEL							
Command = SENSOR_MULTILEVEL_SUPPORTED_SENSOR_REPORT							
Bit Mask 1							
0	0	0	1	0	1	0	1

4.91.4 Multilevel Sensor Get Supported Scale Command

This command is introduced in v5 of the Multilevel Sensor command class and used to retrieve the supported scales of the specific sensor type from the Multilevel Sensor device.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_SENSOR_MULTILEVEL</td></tr>
<tr><td colspan="8">Command = SENSOR_MULTILEVEL_SUPPORTED_GET_SCALE</td></tr>
<tr><td colspan="8">Sensor Type</td></tr>
</table>

Sensor Type (8 bits)

See Multilevel Sensor Report command.

4.91.5 Multilevel Sensor Supported Scale Report Command

This command is used to advertise the supported scales of a specified multilevel sensor type.

This command is introduced in version 5 of the Multilevel Sensor Command Class. The command MUST be returned in response to the Multilevel Sensor Get Supported Scale Command.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_SENSOR_MULTILEVEL</td></tr>
<tr><td colspan="8">Command = SENSOR_MULTILEVEL_SUPPORTED_SCALE_REPORT</td></tr>
<tr><td colspan="8">Sensor Type</td></tr>
<tr><td colspan="4">Reserved</td><td colspan="4">Scale Bit Mask</td></tr>
</table>

Sensor Type (8 bits)

See Multilevel Sensor Report command.

Scale Bit Mask (4 bits)

The Scale Bit Mask field describes the supported scales for the requested sensor type by the multilevel sensor device and refers to the Scale table of Multilevel Sensor Report command.

First bit (Bit 0) in Scale Bit Mask indicates support for the first Scale of the requested Sensor Type e.g. if Sensor Type = Voltage and the Bit 0 = 1, then the Volt scale is supported.

Second bit (Bit 1) in Scale Bit Mask indicates support for the second scale and so forth.

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Example:

To indicate scale support for mV and V of a sensor type = Voltage, the Multilevel Sensor Supported Scale Report command MUST be structured as illustrated below.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SENSOR_MULTILEVEL							
Command = SENSOR_MULTILEVEL_SUPPORTED_SCALE_REPORT							
Sensor Type = Voltage							
Reserved				Scale Bit Mask			
0	0	0	0	0	0	1	1

4.91.6 Multilevel Sensor Get Command

The Multilevel Sensor Get Command is used to request the level of a multilevel sensor.

The Multilevel Sensor Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

The Multilevel Sensor command class, version 5 adds fields to the Multilevel Sensor Get Command to request for the specific and supported sensor type and scale for its measured value. Versions prior to v5 do not specify the additional sensor type and scale fields, why the Multilevel Sensor device can only support one type of sensor when implementing solely Multilevel Sensor command class v1-4.

The transmitter MUST ensure that the receiver supports the requested Sensor Types and/or Scales by calling Multilevel Sensor Get Supported Sensor/Scale commands. If the receiver does not support a Sensor Type, it MUST reply with a manufacturer-selected default Multilevel Sensor Report command. Supporting Sensor Type but the receiver does not support the Scale, it MUST reply with a Multilevel Sensor Report command using a supported scale.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SENSOR_MULTILEVEL							
Command = SENSOR_MULTILEVEL_GET							
Sensor Type							
Reserved			Scale		Reserved		

Sensor Type (8 bits) & Scale (2 bits)

See Multilevel Sensor Report command.

Reserved

These fields MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

4.91.7 Multilevel Sensor Report Command

This command is used to advertise a multilevel sensor reading.

The Multilevel Sensor Report Command v5 is extended with additional Sensor Types and the associated Scales.

In case the received Multilevel Sensor Get command contains Sensor Type and/or Scale values not supported by the device, the device MUST reply with a manufacturer-defined and factory default Multilevel Sensor Report command. This behavior MUST also be implemented if the received Multilevel Sensor Get command is from command class versions prior to Multilevel Sensor Command Class v5.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SENSOR_MULTILEVEL							
Command = SENSOR_MULTILEVEL_REPORT							
Sensor Type							
Precision			Scale		Size		
Sensor Value 1							
…							
Sensor Value N							

Sensor Type (8 bits) & Scale (2 bits)

Sensor Type specifies what type of sensor this command originates from. The Scale is used to indicate what unit the sensor uses. Refer to the table below with respect to defined sensor types and scales.

Sensor types are defined by the Z-Wave Alliance. Values not defined below are reserved and MUST NOT be used.

Table 113, Multilevel Sensor: Sensor Type and Scale identifiers

Sensor Type (8 bits)				Scale (2 bits)	
Label	Value	Supported Bitmask		Label	Value
		Byte #	Bit #		
Reserved	0x00	N/A	N/A	Reserved	0x00-0x03
Air Temperature (v1)	0x01	Byte 1	Bit 0	Celsius (C)	0x00
				Fahrenheit (F)	0x01
				Reserved	0x02-0x03
General Purpose (v1)	0x02	Byte 1	Bit 1	Percentage value	0x00
				Dimensionless value	0x01
				Reserved	0x02-0x03
Luminance (v1)	0x03	Byte 1	Bit 2	Percentage value	0x00
				Lux	0x01
				Reserved	0x02-0x03
Power (v2)	0x04	Byte 1	Bit 3	Watt	0x00
				Btu/h	0x01
				Reserved	0x02-0x03

Sensor Type (8 bits)				Scale (2 bits)	
Label	**Value**	**Supported Bitmask**		**Label**	**Value**
		Byte #	**Bit #**		
Humidity (v2 + v5)	0x05	Byte 1	Bit 4	Percentage value	0x00
				Absolute humidity (g/m³) – (v5)	0x01
				Reserved	0x02-0x03
Velocity (v2)	0x06	Byte 1	Bit 5	m/s	0x00
				Mph	0x01
				Reserved	0x02-0x03
Direction (v2)	0x07	Byte 1	Bit 6	0 to 360 degrees. 0 = no wind, 90 = east, 180 = south, 270 = west, and 360 = north	0x00
				Reserved	0x01-0x03
Atmospheric Pressure (v2)	0x08	Byte 1	Bit 7	kPa (kilopascal)	0x00
				Inches of Mercury	0x01
				Reserved	0x02-0x03
Barometric Pressure (v2)	0x09	Byte 2	Bit 0	kPa (kilopascal)	0x00
				Inches of Mercury	0x01
				Reserved	0x02-0x03
Solar Radiation (v2)	0x0A	Byte 2	Bit 1	W/m²	0x00
				Reserved	0x01-0x03
Dew point (v2)	0x0B	Byte 2	Bit 2	Celsius (C)	0x00
				Fahrenheit (F)	0x01
				Reserved	0x02-0x03
Rain rate (v2)	0x0C	Byte 2	Bit 3	mm/h (millimeter/hour)	0x00
				in/h (inch/hour)	0x01
				Reserved	0x02-0x03
Tide level (v2)	0x0D	Byte 2	Bit 4	m (Meter)	0x00
				Feet	0x01
				Reserved	0x02-0x03
Weight (v3)	0x0E	Byte 2	Bit 5	Kg	0x00
				Pounds	0x01
				Reserved	0x02-0x03
Voltage (v3)	0x0F	Byte 2	Bit 6	V	0x00
				mV	0x01
				Reserved	0x02-0x03
Current (v3)	0x10	Byte 2	Bit 7	A	0x00
				mA	0x01
				Reserved	0x02-0x03
Carbon Dioxide CO_2-level (v3)	0x11	Byte 3	Bit 0	Ppm (Parts/million)	0x00
				Reserved	0x01-0x03
Air flow (v3)	0x12	Byte 3	Bit 1	m³/h (cubic meter/hour)	0x00
				cfm (cubic feet/minute)	0x01

Sensor Type (8 bits)				Scale (2 bits)	
Label	**Value**	**Supported Bitmask**		**Label**	**Value**
		Byte #	**Bit #**		
				Reserved	0x02-0x03
Tank capacity (v3)	0x13	Byte 3	Bit 2	l (liter)	0x00
				m³ (cubic meter)	0x01
				gallons	0x02
				Reserved	0x03
Distance (v3)	0x14	Byte 3	Bit 3	m (meter)	0x00
				cm	0x01
				feet	0x02
				Reserved	0x03
Angle Position (v4) [DEPRECATED by v8]	The use of this Sensor Type is NOT RECOMMENDED. The Direction (0x07) Sensor Type SHOULD be used for reporting polar positions. A device implementing the Angle Position Sensor Type SHOULD also implement the Direction (0x07) Sensor Type.				
	0x15	Byte 3	Bit 4	Percentage value	0x00
				Degrees relative to north pole of standing eye view	0x01
				Degrees relative to south pole of standing eye view	0x02
				Reserved	0x03
Rotation (v5)	0x16	Byte 3	Bit 5	rpm (revolutions per minute)	0x00
				Hz (Hertz)	0x01
				Reserved	0x02-0x03
Water Temperature (v5)	0x17	Byte 3	Bit 6	Celsius (C)	0x00
				Fahrenheit (F)	0x01
				Reserved	0x02-0x03
Soil Temperature (v5)	0x18	Byte 3	Bit 7	Celsius (C)	0x00
				Fahrenheit (F)	0x01
				Reserved	0x02-0x03
Seismic Intensity (v5)	0x19	Byte 4	Bit 0	Mercalli	0x00
				European Macroseismic	0x01
				Liedu	0x02
				Shindo	0x03
Seismic Magnitude (v5)	0x1A	Byte 4	Bit 1	Local (M_L)	0x00
				Moment (M_W)	0x01
				Surface wave (M_S)	0x02
				Body wave (M_B)	0x03
Ultraviolet (v5)	0x1B	Byte 4	Bit 2	UV index	0x00
				Reserved	0x01-0x03
Electrical Resistivity (v5)	0x1C	Byte 4	Bit 3	ohm metre (Ωm)	0x00
				Reserved	0x01-0x03
Electrical Conductivity (v5)	0x1D	Byte 4	Bit 4	siemens per metre (S·m-1)	0x00
				Reserved	0x01-0x03
Loudness (v5)	0x1E	Byte 4	Bit 5	Absolute loudness (dB)	0x00
				A-weighted decibels (dBA)	0x01

Sensor Type (8 bits)				Scale (2 bits)	
Label	Value	Supported Bitmask Byte #	Supported Bitmask Bit #	Label	Value
				Reserved	0x02-0x03
Moisture (v5)	0x1F	Byte 4	Bit 6	Percentage value	0x00
				Volume water content (m^3/m^3)	0x01
				Impedance (kΩ)	0x02
				Water activity (a_w)	0x03
Frequency (v6)	0x20	Byte 4	Bit 7	Hz - MUST be used until 4.294967295 GHz	0x00
				kHz - MUST be used after 4.294967295 GHz	0x01
				Reserved	0x02-0x03
Time (v6)	0x21	Byte 5	Bit 0	Second (s)	0x00
				Reserved	0x01-0x03
Target Temperature (v6)	0x22	Byte 5	Bit 1	Celsius (C)	0x00
				Fahrenheit (F)	0x01
				Reserved	0x02-0x03
Particulate Matter 2.5 (v7)	0x23	Byte 5	Bit 2	mol/m^3 (mole per cubic meter)	0x00
				Absolute $\mu g/m^3$ (microgram/cubic meter)	0x01
				Reserved	0x02-0x03
Formaldehyde CH_2O-level (v7)	0x24	Byte 5	Bit 3	mol/m^3 (mole per cubic meter)	0x00
				Reserved	0x01-0x03
Radon Concentration (v7)	0x25	Byte 5	Bit 4	bq/m^3 (Becquerel/cubic meter)	0x00
				pCi/L (picocuries/liter)	0x01
				Reserved	0x02-0x03
Methane Density CH_4 (v7)	0x26	Byte 5	Bit 5	mol/m^3 (mole per cubic meter)	0x00
				Reserved	0x01-0x03
Volatile Organic Compound (v7 + v10)	0x27	Byte 5	Bit 6	mol/m^3 (mole per cubic meter)	0x00
				Ppm (Parts/million) (v10)	0x01
				Reserved	0x02-0x03
Carbon Monoxide CO-level (v7 + v10)	0x28	Byte 5	Bit 7	mol/m^3 (mole per cubic meter)	0x00
				Ppm (Parts/million) (v10)	0x01
				Reserved	0x02-0x03
Soil Humidity (v7)	0x29	Byte 6	Bit 0	Percentage value	0x00
				Reserved	0x01-0x03
Soil Reactivity (v7)	0x2A	Byte 6	Bit 1	pH (acidity)	0x00
				Reserved	0x01-0x03

Sensor Type (8 bits)				**Scale (2 bits)**	
Label	**Value**	**Supported Bitmask**		**Label**	**Value**
		Byte #	**Bit #**		
Soil Salinity (v7)	0x2B	Byte 6	Bit 2	mol/m^3 (mole per cubic meter)	0x00
				Reserved	0x01-0x03
Heart Rate (v7)	0x2C	Byte 6	Bit 3	Bpm (beats/minute)	0x00
				Reserved	0x01-0x03
Blood Pressure (v7)	0x2D	Byte 6	Bit 4	Systolic mmHg (upper #)	0x00
				Diastolic mmHg (lower #)	0x01
				Reserved	0x02-0x03
Muscle Mass (v7)	0x2E	Byte 6	Bit 5	Kg	0x00
				Reserved	0x01-0x03
Fat Mass (v7)	0x2F	Byte 6	Bit 6	Kg	0x00
				Reserved	0x01-0x03
Bone Mass (v7)	0x30	Byte 6	Bit 7	Kg	0x00
				Reserved	0x01-0x03
Total Body Water, TBW (v7)	0x31	Byte 7	Bit 0	Kg	0x00
				Reserved	0x01-0x03
Basic Metabolic Rate, BMR (v7)	0x32	Byte 7	Bit 1	J (joule)	0x00
				Reserved	0x01-0x03
Body Mass Index, BMI (v7)	0x33	Byte 7	Bit 2	BMI Index	0x00
				Reserved	0x01-0x03
Acceleration, X-axis (v8)	0x34	Byte 7	Bit 3	m/s^2	0x00
				Reserved	0x01-0x03
Acceleration, Y-axis (v8)	0x35	Byte 7	Bit 4	m/s^2	0x00
				Reserved	0x01-0x03
Acceleration, Z-axis (v8)	0x36	Byte 7	Bit 5	m/s^2	0x00
				Reserved	0x01-0x03
Smoke Density (v8)	0x37	Byte 7	Bit 6	Percentage value	0x00
				Reserved	0x01-0x03
Water Flow (v9)	0x38	Byte 7	Bit 7	l/h (liter/hour)	0x00
				Reserved	0x01-0x03
Water Pressure (v9)	0x39	Byte 8	Bit 0	kPa (kilopascal)	0x00
				Reserved	0x01-0x03
RF Signal Strength (v9)	0x3A	Byte 8	Bit 1	RSSI (Percentage value)	0x00
				dBm	0x01
				Reserved	0x02-0x03
Particulate Matter 10 (v10)	0x3B	Byte 8	Bit 2	mol/m^3 (mole per cubic meter)	0x00
				Absolute $\mu g/m^3$ (microgram/cubic meter)	0x01
				Reserved	0x02-0x03

Sensor Type (8 bits)				Scale (2 bits)	
Label	Value	Supported Bitmask		Label	Value
		Byte #	Bit #		
Respiratory Rate (v10)	0x3C	Byte 8	Bit 3	Bpm (Breaths/minute)	0x00
				Reserved	0x01-0x03

Precision (3 bits)

The precision field describes what the precision of the sensor value is. The number indicates the number of decimals. The decimal value 1025 with precision 2 is therefore equal to 10.25.

Size (3 bits)

The size field indicates the number of bytes that is used for the sensor value. This field can take values from 1, 2 or 4.

Sensor Value (variable)

The Sensor Value is a signed field. The field MAY be 1, 2 or 4 bytes in size. The first byte is the most significant byte. Table 114 shows signed decimal values together with their hexadecimal equivalents.

Table 114, Multilevel Sensor Report::Sensor values encoding

Raw value (hex)	Signed 8 bit representation (decimal)
0x7F	127
0x02	2
0x01	1
0x00	0
0xFF	-1
0xFE	-2
0x80	-128

Raw value (hex)	Signed 16 bit representation (decimal)
0x7FFF	32767
0x0002	2
0x0001	1
0x0000	0
0xFFFF	-1
0xFFFE	-2
0x8000	-32768

Raw value (hex)	Signed 32 bit representation (decimal)
0x7FFFFFFF	2147483647
0x00000002	2
0x00000001	1
0x00000000	0
0xFFFFFFFF	-1
0xFFFFFFFE	-2
0x80000000	-2147483648

NOTICE: The device receiving the Multilevel Sensor Report MUST always show the sensor value as is even though the Sensor Type and/or Scale are not supported.

4.91.7.1 Detailed description: Sensor Types for Movement and Rotation

A device may report position, velocity (position change over time) or acceleration (velocity change over time). Position, velocity and acceleration may all refer to a linear scale following an axis or a polar scale circling an axis. Position may be reported in an absolute or relative fashion.

The 3D reference coordinate system outlined in Figure 37 MUST be used for reporting changes in the physical orientation.

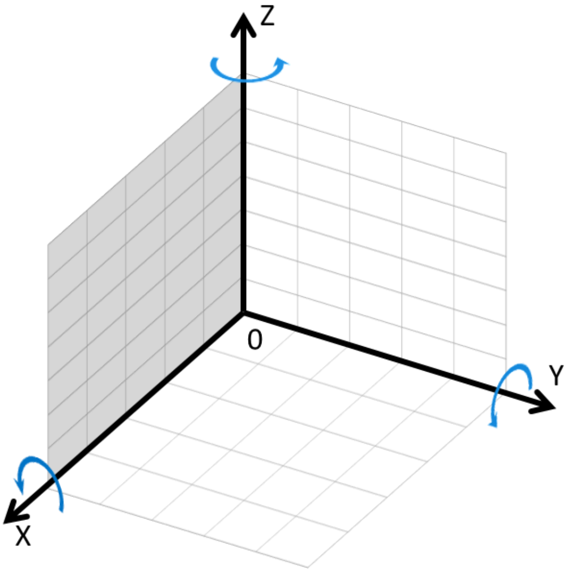


Figure 37, 3D reference coordinate system

Information relating to linear position, velocity and acceleration MUST refer to the zero position on a given axis. Thus a position change or a velocity MUST be positive if moving towards a larger position, measured from the zero position.

Information relating to an angle or change in angle MUST use the right-hand rule. This means that if one (virtually) grabs around an axis with the right hand, with the thumb in the direction of the axis, then the angle increases in the direction of the index finger.

Table 115, Definition of position, velocity or acceleration

Application	Definition
Linear position, absolute	Position with reference to 0 (e.g. 1 meter)
Linear position, relative	Position with reference to previous position (e.g. 1 meter)
Linear velocity	Position change per time unit (e.g. 1 meter/second)
Linear acceleration	Velocity change per time unit (e.g. 1 meter/second^2)
Polar position, absolute	Angle with reference to 0 (e.g. 45 degrees)
Polar position, relative	Angle with reference to previous angle (e.g. 45 degrees)
Polar velocity (rotation)	Angle change per time unit (e.g. 1 degree/second)
Polar acceleration	Velocity change per time unit (e.g. 1 degree/second^2)

Depending on the number of axes supported by a given device, changes in the physical orientation are mapped to one, two or three axes as outlined in Table 116.

Table 116, Mapping of 1D, 2D and 3D movement and rotation

Dimensions	Movement	Rotation
1	Along X axis	Around X axis
2	Along X and Y axes	Around X and Y axes
3	Along X, Y and Z axes	Around X, Y and Z axes

A number of Sensor Types allow a device to report changes in the physical orientation.

Table 117, Recommended Sensor Types for reporting movement and rotation

Application	Sensor Type	Intended usage
1D Linear position, absolute	Distance (v3)	Single axis measurement of position (m)
1D Linear position, relative	*(no support)*	Single axis measurement of position change (m)
1D Linear velocity	Velocity (v2)	Single axis measurement of velocity (m/s)
1D Linear accelleration	Acceleration, X (v8)	Single axis measurement of acceleration (m/s^2)
1D Polar position, absolute	Direction (v2)	Single axis measurement of angle (degrees)
1D Polar position, relative	*(no support)*	Single axis measurement of angle change
1D Polar velocity	Rotation (v5)	Single axis measurement of velocity (RPM)
1D Polar accelleration	*(no support)*	Single axis measurement of acceleration ($degree/s^2$)
2D Linear position, absolute	*(no support)*	Two axis measurement of position (m)
2D Linear position, relative	*(no support)*	Two axis measurement of position change (m)
2D Linear velocity	*(no support)*	Two axis measurement of velocity (m/s)
2D Linear accelleration	Acceleration, X (v8), Acceleration, Y (v8)	Two axis measurement of acceleration (m/s^2)
2D Polar position, absolute	*(no support)*	Two axis measurement of angle (degrees from North)
2D Polar position, relative	*(no support)*	Two axis measurement of angle change
2D Polar velocity	*(no support)*	Two axis measurement of velocity (RPM)
2D Polar accelleration	*(no support)*	Two axis measurement of acceleration ($degree/s^2$)

Application	Sensor Type	Intended usage
3D Linear position, absolute	*(no support)*	Three axis measurement of position (m)
3D Linear position, relative	*(no support)*	Three axis measurement of position change (m)
3D Linear velocity	*(no support)*	Three axis measurement of velocity (m/s)
3D Linear accelleration	Acceleration, X (v8), Acceleration, Y (v8), Acceleration, Z (v8)	Three axis measurement of acceleration (m/s^2)
3D Polar position, absolute	*(no support)*	Three axis measurement of angle (degrees from North)
3D Polar position, relative	*(no support)*	Three axis measurement of angle change
3D Polar velocity	*(no support)*	Three axis measurement of velocity (RPM)
3D Polar accelleration	*(no support)*	Three axis measurement of acceleration ($degree/s^2$)

4.91.7.1.1 Sensor Type = Acceleration

The sensor types “Acceleration, X”, “Acceleration, Y” and “Acceleration, Z” are used to advertise the acceleration of a device along the X, Y and Z axes, respectively.

A one-dimensional device MUST report acceleration using the “Acceleration, X” type.

A two-dimensional device MUST report acceleration using the “Acceleration, X” and “Acceleration, Y” types.

The Scale used MUST be m/s^2. An Acceleration value reported with the Scale m/s^2 may be converted by a receiving node to a “g-force” level by using the formula outlined in Figure 38.

$$1g = 9.81m/s^2$$

(g represents the unit of Earth Gravity; NOT the weight unit “gram”)

Figure 38, Converting from m/s^2 to g-force

4.91.7.2 Detailed description: Smoke Density

Figure 39 shows the principle of how a photoelectric smoke detector works. As shown the smoke sensor uses the smoke to reflect the light onto a photo cell. So when no smoke is present, the light will not be reflected onto the photo cell. When smoke is present, the light will be reflected onto the photo cell. Dependent on the smoke density, more light will be reflected onto the photo cell. So by measuring the received light strength on the photo cell, it is possible to estimate the smoke density.

It is not possible to determine an accurate unit for this, as it is estimated from the light strength on the photo cell. So the exact particle density of the smoke is not known. Also the light strength interval measured on the photo cell, may vary between sensors. So it is decided to report the smoke density in percent, where 0% is no smoke and 100% is the maximum received light strength on the photo cell.

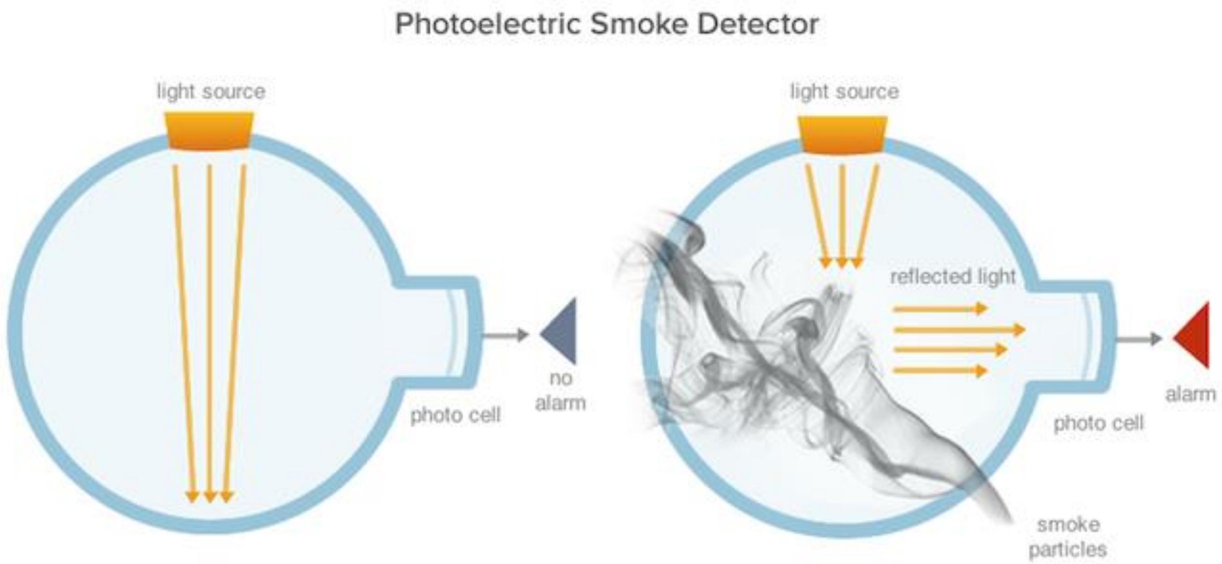


Figure 39, Photoelectric smoke detector

4.91.7.3 Detailed description: RF Signal Strength

The RF Signal Strength sensor type may report values using two different scales. While the dBm is a well-defined unit, the RSSI value represents a relative measurement where the internal sampling circuits and the actual sampling method is product specific.
The RSSI value MUST be reported in the range 0..100, where the value 100 represents the highest power level that can be measured.

4.92 Multilevel Switch Command Class, version 1

The Multilevel Switch Command Class is used to control devices with multilevel capability.

The Multilevel Switch Command Class is an actuator control command class. Refer to 3.7.

4.92.1 Multilevel Switch Set Command

The Multilevel Switch Set command, version 1 is used to set a multilevel value in a supporting device.

The device MAY apply a non-zero duration to the transition from one value to a new value.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_MULTILEVEL							
Command = SWITCH_MULTILEVEL_SET							
Value							

Value (8 bits)

The encoding of the Value field MUST be according to Table 118.

Table 118, Multilevel Switch Set :: Value

Value		Level	State
0	(0x00)	0%	Off
1..99	(0x01..0x63)	Lowest non-zero level .. 100%	On
…		*Reserved*	*Reserved*
255	(0xFF)	Restore most recent (non-zero) level.	On

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

The above mapping of the Multilevel Switch Command Class Value to hardware levels allows a controlling device to control a mixed group of Binary Switch and Multilevel Switch devices via Basic Set commands. Devices implementing the Binary Switch CC turn On or Off while devices implementing the Multilevel Switch CC sets the specified level.

The values 0x00 and 0xFF are special values which MUST be treated as state control commands indicating “Off” and “On”, respectively. A supporting device MUST restore the most recent (non-zero) value in response to the “On” state control command.

A device MAY implement up to 100 hardware levels (including 0). If a device implements less than 100 hardware levels, the hardware levels SHOULD be distributed uniformly over the entire range.
The mapping of command values to hardware levels MUST be monotonous, i.e. a higher value MUST be mapped to either the same or a higher hardware level.

Table 119, Multilevel mappings to hardware levels (example)

Multilevel Value	Hardware level	State
0	0	Off
1..33	33%	On
34..66	66%	On
67..99	100%	On

4.92.2 Multilevel Switch Get Command

The Multilevel Switch Get command, version 1 is used to request the status of a multilevel device.

The Multilevel Switch Report command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing.
The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_MULTILEVEL							
Command = SWITCH_MULTILEVEL_GET							

4.92.3 Multilevel Switch Report Command

The Multilevel Switch Report command, version 1 is used to advertise the status of a multilevel device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_MULTILEVEL							
Command = SWITCH_MULTILEVEL_REPORT							
Value							

Value (8 bits)

The encoding of the Value field MUST be according to Table 120.

Table 120, Multilevel Switch Report :: Value

Value		Hardware level	State
0	(0x00)	0%	Off
1..99	(0x01..0x63)	Lowest non-zero level .. 100%	On
...		*Reserved*	On
254	(0xFE)	Unknown	Unknown
255	(0xFF)	(100%)	On [Deprecated]

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

The value 255 (0xFF) has been deprecated as it provides no information on the actual value in the device. A sending node MUST NOT use the value 255. A receiving node MUST interpret the value 255 as 100%.

The Value field SHOULD advertise the current value of the device hardware; also while in transition to a new target value.

A controlling device MUST NOT assume that the Value is identical to a value previously issued with a Set command when a transition has ended.

4.92.4 Multilevel Switch Start Level Change Command

The Multilevel Switch Start Level Change command, version 1 is used to initiate a transition to a new level.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_SWITCH_MULTILEVEL</td></tr>
<tr><td colspan="8">Command = SWITCH_MULTILEVEL_START_LEVEL_CHANGE</td></tr>
<tr><td>Reserved</td><td>Up/ Down</td><td>Ignore Start Level</td><td colspan="5">Reserved</td></tr>
<tr><td colspan="8">Start Level</td></tr>
</table>

Up/Down (1 bit)

This field MUST specify the direction of the level change.

If the Up/Down bit is set to 0 the level change MUST be increasing.
If the Up/Down bit is set to 1 the level change MUST be decreasing.

Ignore Start Level (1 bit)

A receiving device SHOULD respect the start level if the Ignore Start Level bit is 0.
A receiving device MUST ignore the start level if the Ignore Start Level bit is 1.

A controlling device SHOULD set the Ignore Start Level bit to 1.

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Start Level (8 bits)

The Start Level field MUST specify the initial level of the level change.

4.92.5 Multilevel Switch Stop Level Change Command

The Multilevel Switch Stop Level Change command, version 1 is used to stop an ongoing transition.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_SWITCH_MULTILEVEL</td></tr>
<tr><td colspan="8">Command = SWITCH_MULTILEVEL_STOP_LEVEL_CHANGE</td></tr>
</table>

4.93 Multilevel Switch Command Class, version 2

The Multilevel Switch Command Class is used to control devices with multilevel capability.

The Multilevel Switch Command Class is an actuator control command class. Refer to 3.7.

4.93.1 Compatibility considerations

A device supporting Multilevel Switch CC, version 2 MUST support Multilevel Switch CC, version 1.

Version 2 adds a "Duration" parameter to the Multilevel Switch Set and Multilevel Switch Start/Stop Level Change commands.

Commands not described in Version 2 stays unchanged from Version 1.

4.93.2 Multilevel Switch Set Command

The Multilevel Switch Set command, version 2 is used to set a multilevel value in a supporting device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_MULTILEVEL							
Command = SWITCH_MULTILEVEL_SET							
Value							
Duration							

Value (8 bits)

Refer to Multilevel Switch Set V1 command.

Duration (8 bits)

The Duration field MUST specify the time that the transition should take from the current value to the new target value.
A supporting device SHOULD respect the specified Duration value.

The encoding of the Duration field MUST be according to Table 121:

Table 121, Duration

Duration	Description
0x00	Instantly
0x01-0x7F	1 second (0x01) to 127 seconds (0x7F) in 1-second resolution.
0x80-0xFE	1 minute (0x80) to 127 minutes (0xFE) in 1-minute resolution.
0xFF	Factory default duration.

The factory default duration SHOULD be the same as the duration used for the Multilevel Switch Set command, version 1.

4.93.3 Multilevel Switch Get Command

The Multilevel Switch Get Command version 2 is similar to version 1.

4.93.4 Multilevel Switch Report Command

The Multilevel Switch Report Command version 2 is similar to version 1.

4.93.5 Multilevel Switch Start Level Change Command

The Multilevel Switch Start Level Change command, version 2 is used to initiate a transition to a new level.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_SWITCH_MULTILEVEL</td></tr>
<tr><td colspan="8">Command = SWITCH_MULTILEVEL_START_LEVEL_CHANGE</td></tr>
<tr><td>Reserved</td><td>Up/ Down</td><td>Ignore Start Level</td><td colspan="5">Reserved</td></tr>
<tr><td colspan="8">Start Level</td></tr>
<tr><td colspan="8">Duration</td></tr>
</table>

Up/Down (1 bit)

This field MUST specify the direction of the level change.

If the Up/Down bit is set to 0 the level change MUST be increasing.
If the Up/Down bit is set to 1 the level change MUST be decreasing.

Ignore Start Level (1 bit)

A receiving device SHOULD respect the start level if the Ignore Start Level bit is 0.
A receiving device MUST ignore the start level if the Ignore Start Level bit is 1.

A controlling device SHOULD set the Ignore Start Level bit to 1.

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Start Level (8 bits)

The Start Level field MUST specify the initial level of the level change.

Duration (8 bits)

The dimming rate to use MUST be calculated to match a transition from 0 to 99 during the time specified by the Duration field.
A supporting device SHOULD respect the specified Duration value.

For encoding of the duration value refer to section 4.93.2.

4.93.6 Multilevel Switch Stop Level Change Command

The Multilevel Switch Stop Level Change Command version 2 is similar to version 1.

4.94 Multilevel Switch Command Class, version 3

The Secondary Switch Type of the Multilevel Switch Command Class, version 3 has been DEPRECATED.

The Primary Switch Type 0x00, indicating "Not supported", of the Multilevel Switch Command Class, version 3 has been OBSOLETED.

The implementation of Secondary Switch Type functionality is NOT RECOMMENDED.

While the functionality related to the Secondary Switch Type of the Multilevel Switch Command Class, version 3 is deprecated, a supporting device claiming compliance with this version MUST implement support for the Multilevel Switch Supported Get Command.

For backwards compatibility reasons, a device MAY implement Secondary Switch Type functionality. It is however RECOMMENDED that Multi Channel Command Class support is also implemented if a device provides multiple controllable resources in the same physical entity.

The Multilevel Switch Command Class is used to control devices with multilevel capability.

The Multilevel Switch Command Class is an actuator control Command Class. Refer to 3.7.

Multilevel Switch CC, version 3 adds two-dimensional resource control capabilities to the Multilevel Switch Start/Stop Level Change commands and introduces two new commands for the discovery of those capabilities.

The Secondary Switch Type is intended for devices providing two-dimensional control, e.g. Window blinds, where the Primary Switch Type is used for Up/Down control and the Secondary Switch Type is used for controlling the slat tilt angle.

The Multilevel Switch Set, Get and Report commands MUST address the primary device functionality. The Multilevel Switch Start/Stop Level Change commands MUST manipulate the Primary Switch Type functionality based on the Up/Down parameter.

The Multilevel Switch Start/Stop Level Change commands MUST manipulate the Secondary Switch Type functionality based on the Inc/Dec parameter.
If the Secondary Switch Type is 0x00 (Undefined / Not supported), the Multilevel Switch Start/Stop Level Change (Inc/Dec) command parameter MUST be ignored.

4.94.1 Compatibility considerations

A device supporting Multilevel Switch CC, version 3 MUST support Multilevel Switch CC, version 1.
A device supporting Multilevel Switch CC, version 3 MUST support Multilevel Switch CC, version 2.

A device implementing Multilevel Switch CC, Version 3 MUST implement the Primary Switch type.

Commands not described in Version 3 stays unchanged from Version 2.

4.94.2 Multilevel Switch Supported Get Command

The Multilevel Switch Supported Get command, version 3 is used to request the supported Switch Types of a supporting device.

The Multilevel Switch Supported Report command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing. The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_MULTILEVEL							
Command = SWITCH_MULTILEVEL_SUPPORTED_GET							

4.94.3 Multilevel Switch Supported Report Command

The Multilevel Switch Supported Report Command, version 3 is used to advertise the supported Switch Types implemented by a supporting device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_MULTILEVEL							
Command = SWITCH_MULTILEVEL_SUPPORTED_REPORT							
Reserved			Primary Switch Type				
Reserved			Secondary Switch Type				

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Primary Switch Type (5 bits)

The Primary Switch Type field MUST represent the primary device functionality.

<u>Primary Switch Type 0x00, indicating "Not supported", has been OBSOLETED.</u>

Previous revisions of this specification allowed that a device may (theoretically) be implemented with no primary device functionality, i.e. Primary Switch Type 0x00, and specified that such a device should indicate this situation by returning Multilevel Switch Report commands carrying the value 0xFE. The value 0xFE is incompatible with Versions 1 and 2 of the Multilevel Switch Command Class and MUST NOT be used in a Multilevel Switch Report.

The Primary Switch Type MUST comply with Table 122.

A supporting device MUST implement the Primary Switch type.
The Primary Switch Type SHOULD be 0x02 (Up/Down).
The Primary Switch Type MUST NOT be 0x00 (Undefined).

Table 122, Encoding of Primary and Secondary Switch Types

Switch Type Value	0x00 (Direction/Endpoint A)	0x63/0xFF (Direction/Endpoint B)
0x00	Undefined / Not supported (Secondary only)	
0x01	Off	On
0x02	Down	Up
0x03	Close	Open
0x04	Counter-Clockwise	Clockwise
0x05	Left	Right
0x06	Reverse	Forward
0x07	Pull	Push
0x08-0x1F	Reserved	

All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.

Secondary Switch Type (5 bits)

The Secondary Switch Type field MUST represent the secondary device functionality. The Secondary Switch Type MUST comply with Table 122.

A supporting device MAY implement the Secondary Switch type.

4.94.4 Multilevel Switch Start Level Change Command

The Multilevel Switch Start Level Change command, version 3 is used to initiate a transition to a new level.

The Multilevel Switch Command Class, Version 3 adds a “Secondary Switch Inc/Dec” and a “Secondary Switch Step Size” field to this command to support motor controlled devices featuring two-dimensional motion.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_SWITCH_MULTILEVEL</td></tr>
<tr><td colspan="8">Command = SWITCH_MULTILEVEL_START_LEVEL_CHANGE</td></tr>
<tr><td colspan="2">Primary Switch Up/Down</td><td>Ignore Start Level</td><td colspan="2">Secondary Switch Inc/Dec</td><td colspan="3">Reserved</td></tr>
<tr><td colspan="8">Primary Switch Start Level</td></tr>
<tr><td colspan="8">Duration</td></tr>
<tr><td colspan="8">Secondary Switch Step Size</td></tr>
</table>

Primary Switch Up/Down (2 bits)

The Up/Down field MUST be used for manipulating the primary device functionality.

This field MUST be encoded according to Table 123.

Table 123, Encoding of the Up/Down field

Value	Description	Details
0x00	Up	Increase level for Primary Switch Type
0x01	Down	Decrease level for Primary Switch Type
0x02	*Reserved*	
0x03	No Up/Down motion	Maintain current level for Primary Switch Type

Ignore Start Level (1 bit)

A receiving device SHOULD respect the start level if the Ignore Start Level bit is 0.
A receiving device MUST ignore the start level if the Ignore Start Level bit is 1.

Secondary Switch Inc/Dec (2 bits)

The Inc/Dec field MUST be used for controlling the secondary device functionality.

If the Secondary Switch Type is 0x00 (Undefined / Not supported), the Inc/Dec field MUST be ignored.

This field MUST be encoded according to the table below.

Table 124, Encoding of the Inc/Dec field

Value	Description	Details
0x00	Increment	Increase level for Secondary Switch Type
0x01	Decrement	Decrease level for Secondary Switch Type
0x02	*Reserved*	
0x03	No Inc/Dec	Maintain current level for Secondary Switch Type

As this field defines the value "00" of two previously reserved bits as the code "Increment", a supporting device implementing Multilevel Switch CC, version 3 MUST correctly identify the Multilevel Switch Start Level Change command to be version 3 (by the presence of the <Secondary Switch Step Size> field) before interpreting the <Secondary Switch Inc/Dec> field. Failing to do so will cause a device to start incrementing the Secondary Switch Type in response to version 1 and version 2 variants of the Multilevel Switch Start Level Change command.

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Primary Switch Start Level (8 bits)

The Start Level field MUST be used for controlling the primary device functionality.

Duration (8 bits)

This field is unchanged from version 2.

Secondary Switch Step Size (8 bits)

The Step Size field MAY be used for controlling a secondary device functionality.
If the Secondary Switch Type is 0x00 (Undefined / Not supported), the Step Size field MUST be ignored.
If the Secondary Switch Type is not 0x00, the Step Size field MUST be used for controlling the secondary device functionality.

If the Inc/Dec field is set to 3 (No Inc/Dec), the Step Size field MUST be set to 0.

This field MUST carry a value in the range {0x00..0x63, 0xFF}. All other values are reserved and MUST NOT be used by a sending node. Reserved values MUST be ignored by a receiving node.
A receiving device MUST accept any of the values in the above range. 0x01 MUST represent the lowest non-zero level and 0x63 MUST represent the highest level

A device MAY implement 100 hardware levels (including 0). If a device implements less than 100 hardware levels, then the mapping to hardware levels SHOULD be distributed equally over the entire range of 1..99 (0x01..0x63).
The mapping of command values to hardware levels MUST be monotonous, i.e. a higher value MUST be mapped to either the same or a higher hardware level.

An implementation MUST interpret the combined Inc/Dec and Step Size fields as outlined in the table below.

Table 125, Interpretation of the Inc/Dec and Step Size fields

(Secondary Switch Type)	Inc/Dec	Step Size	Interpretation
(undefined)	*(ignore)*	*(ignore)*	Secondary Switch Type is undefined ⇒ ignore fields
0x01..0x07	No Inc/Dec	*(ignore)*	No Inc/Dec ⇒ Maintain current Secondary Switch level
0x01..0x07	Increment	x	Increase Secondary Switch level by x steps
0x01..0x07	Decrement	y	Decrease Secondary Switch level by y steps

The requested level change SHOULD take the time specified by the Duration field.

4.95 Multilevel Switch Command Class, version 4

The Multilevel Switch Command Class is used to control devices with multilevel capability.

The Multilevel Switch Command Class is an actuator control Command Class. Refer to chapter 3.7.

4.95.1 Compatibility considerations

Version 4 adds reporting of target value and duration.

A device supporting Multilevel Switch CC, Version 4 MUST support Multilevel Switch CC, Version 3.

A device receiving a V1 Multilevel Set command MAY apply a factory default duration to the transition to a new value.

4.95.2 Multilevel Switch Report Command

The Multilevel Switch Report command, version 4 is used to advertise the status of a multilevel device.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_MULTILEVEL							
Command = SWITCH_MULTILEVEL_REPORT							
Current Value							
Target Value							
Duration							

Current Value (8 bits)

The encoding of the Value field MUST be according to Table 120.

The device may be queried for its current value while in transition to a new value. The response to a Get command SHOULD be the current value of the device hardware.

A controlling device MUST NOT assume that the Value is identical to a value previously issued with a Set command when a transition has ended.

Target Value (8 bits)

The Target Value field MUST advertise the target value of an ongoing transition or the most recent transition.

The encoding of the Target Value field MUST be according to Table 120.

If queried after receiving a Set command, the Target Value field MUST advertise the target value specified in the Set command. The Target Value may change at a later time due to local control or a Multilevel Switch Stop Level Change command.

If the device is in a motion controlled transition, the Target Value field MUST advertise the value 0x00 or 0x63.

Duration (8 bits)

The Duration field SHOULD advertise the time needed to reach the Target Value at the actual transition rate. The encoding of the Duration field MUST be according to Table 126.

Table 126, Multilevel Switch Report :: Duration

Duration	Description
0x00	0 seconds. Already at the Target Value.
0x01-0x7F	1 second (0x01) to 127 seconds (0x7F) in 1 second resolution.
0x80-0xFD	1 minute (0x80) to 126 minutes (0xFD) in 1 minute resolution.
0xFE	Unknown duration
0xFF	*Reserved*

All other values are reserved. Reserved values MUST NOT be used by a sending node and MUST be ignored by a receiving node.

4.96 Multilevel Toggle Switch Command Class, version 1 [DEPRECATED]

THIS COMMAND CLASS HAS BEEN DEPRECATED

A device MAY implement this command class, but it is RECOMMENDED that new implementations use the Multilevel Switch Command Class.
If implementing this command class, it is RECOMMENDED that the Multilevel Switch Command Class is also implemented.

The Multilevel Toggle Switch Command Class is used for multilevel toggle-style actuator devices.

4.96.1 Multilevel Toggle Switch Set Command

The Multilevel Toggle Switch Command may be used to set the level in a device that supports the multilevel switch functionality.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_TOGGLE_MULTILEVEL							
Command = SWITCH_TOGGLE_MULTILEVEL_SET							

4.96.2 Multilevel Toggle Switch Get Command

The Multilevel Toggle Switch Get Command may be used to request the state of the load controlled by the device.

The Multilevel Toggle Switch Report Command MUST be returned in response to this command.

This command MUST NOT be issued via multicast addressing.
A receiving node MUST NOT return a response if this command is received via multicast addressing.
The Z-Wave Multicast frame, the broadcast NodeID and the Multi Channel multi-End Point destination are all considered multicast addressing methods.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_TOGGLE_MULTILEVEL							
Command = SWITCH_TOGGLE_MULTILEVEL_GET							

4.96.3 Multilevel Toggle Switch Report Command

This command is used to advertise the level of a toggle switch.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_TOGGLE_MULTILEVEL							
Command = SWITCH_TOGGLE_MULTILEVEL_REPORT							
Value							

Value (8 bits)

The value MAY be 0x00 (off/disable) or 0xFF (on/enable).

The field MAY carry values from 1 to 99.

4.96.4 Multilevel Toggle Switch Start Level Change Command

The Multilevel Toggle Switch Start Level Change Command may be used to inform a multilevel toggle switch, that it should start changing the level. The speed that the switch increases or decreases the level with is implementation specific.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_SWITCH_TOGGLE_MULTILEVEL</td></tr>
<tr><td colspan="8">Command = SWITCH_TOGGLE_MULTILEVEL_START_LEVEL_CHANGE</td></tr>
<tr><td>Roll Over</td><td>Reser-ved</td><td>Ignore Start Level</td><td colspan="5">Reserved</td></tr>
<tr><td colspan="8">Start Level</td></tr>
</table>

Roll Over (1 bit)

If the Roll Over bit is set to 0 then the switch SHOULD stop when reaching the max or min level. If the roll over bit is set to 1 then the switch SHOULD continually increase and decrease the level until otherwise instructed.

Ignore Start Level (1 bit)

If the Ignore Start Level bit is set to 0 the switch SHOULD use the start level specified in the Command. If field is set to 1 the switch SHOULD start from the actual level in the device.

Reserved

This field MUST be set to 0 by a sending node and MUST be ignored by a receiving node.

Start Level (8 bits)

The Start Level field contains the initial level that the switch should assume when it start to change the level.

4.96.5 Multilevel Toggle Switch Stop Level Change Command

The Multilevel Toggle Switch Stop Level Change Command may be used to inform a multilevel toggle switch, that it should stop changing the level.

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Command Class = COMMAND_CLASS_SWITCH_TOGGLE_MULTILEVEL</td></tr>
<tr><td colspan="8">Command = SWITCH_TOGGLE_MULTILEVEL_STOP_LEVEL_CHANGE</td></tr>
</table>

REFERENCES

[1] Sigma Designs, SDS10242, Software Design Spec., Z-Wave Device Class Specification.
[2] Sigma Designs, SDS12652, Software Design Spec., Z-Wave Command Class Specification N-Z.
[3] IETF RFC 4861, Neighbor Discovery for IP version 6 (IPv6), http://tools.ietf.org/pdf/rfc4861.pdf
[4] IETF RFC 3122, Extensions to IPv6 Neighbor Discovery for Inverse Discovery Specification, http://tools.ietf.org/pdf/rfc3122.pdf
[5] IETF RFC 2119, Key words for use in RFCs to Indicate Requirement Levels, http://tools.ietf.org/pdf/rfc2119.pdf
[6] IETF RFC 2460, Internet Protocol, Version 6 (IPv6) Specification, http://tools.ietf.org/pdf/rfc2460.pdf
[7] IETF RFC 4291, IP Version6 Addressing Architecture, http://tools.ietf.org/pdf/rfc4291.pdf
[8] Sigma Designs, SDS11846, Software Design Spec., Z-Wave Plus Role Types Specification.
[9] Sigma Designs, SDS11847, Software Design Spec., Z-Wave Plus Device Types Specification.
[10] Graphical UI elements, http://en.wikipedia.org/wiki/Graphical_user_interface_elements
[11] Sigma Designs, SDS13425. Software Design Spec., Z-Wave Plus Assigned Manufacturer IDs.

INDEX

3

A

B

C

D

E

F

G

H

M

N

O

P

R

S

T

U

V

W