



## Software Design Specification

### Z-Wave Device Class Specification

<b>Document No.:</b>	SDS10242
<b>Version:</b>	21
<b>Description:</b>	This document describes the Device and Command Classes used by Z-Wave enabled products ensuring that compliant products will be interoperable.
<b>Written By:</b>	JFR
<b>Date:</b>	2012-05-25
<b>Reviewed By:</b>	CHL;JRM;ABR;DKING;BBR
<b>Restrictions:</b>	Partners Only

#### Approved by:

Date	CET	Initials	Name	Justification
2012-05-25	14:07:52	NTJ	Niels Thybo Johansen	

This document is the property of Sigma Designs Inc. The data contained herein, in whole or in part, may not be duplicated, used or disclosed outside the recipient for any purpose. This restriction does not limit the recipient's right to use information contained in the data if it is obtained from another source without restriction.



**CONFIDENTIAL**

## REVISION RECORD

Doc. Rev	Date	By	Pages affected	Brief description of changes
1	20050224	HEH JFR	See SDS11060	Details about when the Association command class is necessary added. Described details about assigning return routes in conjunction with the Association Set command. Described how to respond on an Association Get command.
1	20050301	JFR	See SDS11060	Described the No Operation command class
1	20050405	JFR	See SDS11060	manufacturer IDs updated
1	20050409	JFR	See SDS11060	Clarified when the Proprietary command class can be used
2	20050625	JFR	See SDS11060	Multilevel Switch Do Level Change Command removed in Multilevel Switch Command Class
3	20051207	JFR	See SDS11060	Discontinued roll over bit in Multilevel Switch Start Level Change Command in Multilevel Switch Command Class
3	20051208	JFR	Section 7.1.2	Rocker switch example updated with support of the Association Command Class
3	20051208	JFR	Section 5.17 See SDS11060	Handling of a command class coming with different versions added
3	20051209	JFR	See SDS11060	Hail Command Class added
3	20051209	JFR	See SDS11060	Manufacturer Proprietary Command Class added
3	20051209	JFR	Section 5.11.2.2 Section 5.15.3.2 Section 5.4.3.3 Section 5.9.3.3 See SDS11060	Scene Device Class added
4	20060105	MVO	All	New 1 <sup>st</sup> page/header/footer contents. New Doc No
4	20060131	JFR	See SDS11060	Clarified how the reports to follow field must be interpreted.
4	20060131	JFR	See SDS11060	Manufacturer Proprietary Command Class can optional use the API call ZW-SendDataMeta.
4	20060212	JFR	See SDS11060	Node Naming and Location Command Class added
4	20060229	JFR	See SDS11060	Multilevel Switch command class version 2 added
4	20060229	JFR	See SDS11060	Interpretation of the "Ignore start level" bit updated
4	20060229	JFR	Section 5.17	Devices supporting older versions of a command class
4	20060229	JFR	Section 7.1.1.2.2	Updated to Multilevel Switch command class version 2 in the example
4	20060317	JFR	Section 3.3	the Optional Functionality flag added to the node information frame
5	20060425	JFR	Front page	Fixed front page formatting problems
6	20060425	JFR	Front page	Fixed front page formatting problems again
7	20060426	BGR JFR	Section 5.2  See SDS11060 Section 5.7  See SDS11060 See SDS11060	Basic command class mapping moved to the specific device class level for Binary Switch Generic Device Class. Binary Switch Command Class parameters clarified Basic command class mapping moved to the specific device class level for Multilevel Switch Generic Device Class. Multilevel Switch Command Class parameters clarified Basic Command Class parameters clarified
8	20060518	JFR	See SDS11060 See SDS11060 See SDS11060	Time Command Class added Time Parameters Command Class added Geographical Location Command Class added
9	20060710	JFR	Section 5.9.3.4	Multiposition Motor Device Class added to the Multilevel Switch Generic Device Class
9	20060825	JFR	See SDS11060 See SDS11060	Composite Command Class added Association Command Class version 2 added
9	20060825	JFR	Section 5.16.3.4 See SDS11060	AV Control Point Device Class added Simple AV Control Command Class added
10	20061011	JFR	See SDS11060	Manufacturer IDs updated
10	20061013	JFR	See SDS11060 See SDS11060 Section 5	Screen Meta Data Command Class added Screen Attributes Data Command Class added Basic command class mapping for devices clarified
11	20061218	JFR	See SDS11060 Section 2.3 Section 2.4	Manufacturers ID updated Precedence of definitions added Terms used added

11	20070208	JFR	Section 5.16.3.4 See SDS11060 See SDS11060 See SDS11060 See SDS11060	Setback Schedule Thermostat Specific Device Class added Wake Up Command Class Version 2 added Multi Command Command Class added Climate Control Schedule Command Class added Multi Instance Association Command Class added
11	20070302	JFR	Section 5.11.2.2 & 5.11.2.2.1  Section 5.15.3.2, 5.15.3.2.1 & 5.15.3.2.2	Scene Activation Command Class was erroneously listed as a mandatory command class to support for a portable scene controller, which typically is battery-operated. Scene Activation Command Class should not be listed as a mandatory command class but recommended to handle scene controllers which control a load locally.
12	20070509	JFR	5.16.3.4.1.1  See SDS11060	Basic Cmd Class mapping clarified for a Setback Schedule Thermostat device. Motor controls behaviour clarified in the Multilevel Switch Set command.
12	20070715	JFR	See SDS11060	Added support for Windows Vista Media Center and Media Center 2005 remote controls in Simple AV Control command class
12	20070730	SBD	Section 4 Section 5	Basic Command Class requirements and mappings partially updated. SUC and SIS criteria added for controllers. Network maintenance functionality added for slaves and controllers. Thermostat General version 2 Specific Device Class added. Satellite Receiver version 2 Specific Device Class added.
12	20070813	JFR	See SDS11060	Manufacturers ID updated
12	20070816	JFR	See SDS11060 Section 5.7 See SDS11060 See SDS11060 Section 5.6.4.3 See SDS11060 See SDS11060 See SDS11060 See SDS11060 See SDS11060 See SDS11060 See SDS11060	Multilevel Sensor Command Class Version 2 added Meter Generic Device Class added Meter Command Class added IP Configuration Command Class added Advanced Door Lock Specific Device Class added Door Lock Command Class added User Code Command Class added Grouping Name Command Class added Remote Association Configuration Command Class added Remote Association Activate Command Class added Typo fixed in End Point Mask addressing
12	20070913	JFR	See SDS11060 See SDS11060 See SDS11060 See SDS11060 See SDS11060	Screen Meta Data Command Class Version 2 added. Screen Attributes Command Class Version 2 added. Protection Command Class Version 2 added. Firmware Update Meta Data Command Class added. Table with Z-Wave Protocol version listed for a given Developer's Kit version added.
13	20070920	JFR	Section removed	All Command Classes moved to document SDS11060
13	20071023	SBD	Section 5.6.4.3	Advanced door lock specification device class clarification added.
13	20071227	JFR	Section 5.1	Alarm Sensor Device Class added Basic Report mapping clarified
13	20071227	JFR	Section 5.16.3.5	Setback Thermostat Specific Device Class added
13	20080102	JFR	Section 5.16.3.6	Setpoint Thermostat Specific Device Class added
14	20080519	SBD	Section 0 Section 3.2	References updated. Added note about selecting Device and Command Classes for devices.
14	20080812	JFR	Section 5 Section 5	Z/IP Gateway Generic Device Class added Z/IP Node Generic Device Class added
14	20080813	JFR	Section 5.11.2.3 Section 5.15.3.3	Portable Installer Tool Specific Device Class added Static Installer Tool Specific Device Class added
14	20080814	JFR	Section 5.5	Display Generic Device Class added
15	20090217	JFR	Section 5.2.3.4	Doorbell Specific Device Class added
16	20090218	JFR	Section 5.2.3 Section 5.6.4.4	Doorbell Specific Device Class identifier added Secure Keypad Door Lock Specific Device Class added
17	20090405	JFR	Section 5.17	Ventilation Generic Device Class added
17	20090430	JFR	Section 5.9.3.5, 5.9.3.6 & 5.9.3.7 Section 5.9.3.4	Multilevel Switch Generic Device Class extended with Motor Control A, B & C specific device classes. Multiposition Motor specific device class not recommended anymore.
18	20090907	JFR	Section 5.7.3.3 Section 7.2	Advanced Energy Control Specific Device Class added. Advanced Energy Control applications added.
18	20091028	JFR	Section 0	Basic Command Class implementation when controlled clarified
19	20110627	JFR	Section 5.9.3.8 Section 5.4.3.4 Section 5 Section 5	Multilevel Tunable Color Light Specific Device Class added Binary Tunable Color Light Specific Device Class added Z/IP Gateway Generic Device Class discontinued Z/IP Node Generic Device Class discontinued
20	20120221	JFR	Section 5.15.3.4 Section 2.4	Gateway Specific Device Class added Clarification of requirement keywords
21	20120327	JFR	Section 3.1.1.1	Clarification of controller device functionality wrt. interoperability

22	20120328	JFR	Section 3.1.1.2 Section 3.1.1.1 Section 3.1.1.3	Minimal control functionality w rt. mandatory command class implementations Adjusted according to minimal control functionality. Clarification of controller device functionality wrt. associations
----	----------	-----	---	---

# Table of Contents

<b>1</b>	<b>ABBREVIATIONS .....</b>	<b>1</b>
<b>2</b>	<b>INTRODUCTION .....</b>	<b>2</b>
2.1	Purpose.....	2
2.2	Audience and Prerequisites .....	2
2.3	Precedence of definitions .....	2
2.4	Terms used in this document .....	2
<b>3</b>	<b>OVERVIEW .....</b>	<b>3</b>
3.1	Device Classes .....	6
3.1.1	Controller Device Class Functionality .....	8
3.1.1.1	Interoperability .....	8
3.1.1.2	Minimal Control Functionality .....	9
3.1.1.3	Associations Handling .....	12
3.2	Command Classes .....	14
3.3	Node Information Frame .....	15
3.3.1	Z-Wave Protocol Specific Part .....	15
3.3.2	Application Specific Part .....	15
<b>4</b>	<b>BASIC DEVICE CLASSES .....</b>	<b>18</b>
4.1	Portable Controller .....	19
4.2	Static Controller .....	20
4.3	Slave .....	21
4.4	Routing Slave .....	22
<b>5</b>	<b>GENERIC AND SPECIFIC DEVICE CLASSES .....</b>	<b>23</b>
5.1	Alarm Sensor Generic Device Class .....	28
5.1.1	Mandatory Command Classes to Support .....	28
5.1.1.1	Basic Command Class Implementation .....	28
5.1.2	Mandatory Command Classes to Control .....	28
5.1.3	Specific Device Classes .....	29
5.1.3.1	No Specific Device Class defined .....	30
5.1.3.2	Basic Routing Alarm Sensor Specific Device Class .....	31
5.1.3.3	Routing Alarm Sensor Specific Device Class .....	33
5.1.3.4	Basic Zensor Net Alarm Sensor Specific Device Class .....	35
5.1.3.5	Zensor Net Alarm Sensor Specific Device Class .....	37
5.1.3.6	Advanced Zensor Net Alarm Sensor Specific Device Class .....	39
5.1.3.7	Basic Routing Smoke Sensor Specific Device Class .....	41
5.1.3.8	Routing Smoke Sensor Specific Device Class .....	43
5.1.3.9	Basic Zensor Net Smoke Sensor Specific Device Class .....	45
5.1.3.10	Zensor Net Smoke Sensor Specific Device Class .....	47
5.1.3.11	Advanced Zensor Net Smoke Sensor Specific Device Class .....	49
5.2	AV Control Point Generic Device Class .....	51
5.2.1	Mandatory Command Classes to Support .....	51
5.2.1.1	Basic Command Class Implementation .....	51
5.2.2	Mandatory Command Classes to Control .....	51
5.2.3	Specific Device Classes .....	52
5.2.3.1	No Specific Device Class defined .....	53
5.2.3.2	Satellite Receiver Specific Device Class (Not recommended) .....	54
5.2.3.3	Satellite Receiver V2 Specific Device Class .....	55
5.2.3.4	Doorbell Specific Device Class .....	56
5.3	Binary Sensor Generic Device Class .....	58
5.3.1	Mandatory Command Classes to Support .....	58
5.3.1.1	Basic Command Class Implementation .....	58

5.3.2	Mandatory Command Classes to Control .....	58
5.3.3	Specific Device Classes .....	59
5.3.3.1	Routing Binary Sensor Specific Device Class .....	60
5.4	Binary Switch Generic Device Class .....	62
5.4.1	Mandatory Command Classes to Support .....	62
5.4.2	Mandatory Command Classes to Control .....	62
5.4.3	Specific Device Classes .....	63
5.4.3.1	No Specific Device Class defined .....	64
5.4.3.2	Binary Power Switch Specific Device Class .....	65
5.4.3.3	Binary Scene Switch Specific Device Class .....	67
5.4.3.4	Binary Tunable Color Light Specific Device Class .....	69
5.5	Display Generic Device Class .....	70
5.5.1	Mandatory Command Classes to Support .....	70
5.5.1.1	Basic Command Class Implementation .....	70
5.5.2	Mandatory Command Classes to Control .....	70
5.5.3	Specific Device Classes .....	71
5.5.3.1	No Specific Device Class defined .....	72
5.5.3.2	Simple Display Specific Device Class .....	73
5.6	Entry Control Generic Device Class .....	75
5.6.1	Mandatory Command Classes to Support .....	75
5.6.1.1	Basic Command Class Implementation .....	75
5.6.2	Recommended Command Classes to Support .....	75
5.6.3	Mandatory Command Classes to Control .....	75
5.6.4	Specific Device Classes .....	76
5.6.4.1	Specific Device Class Not Used .....	77
5.6.4.2	Door Lock Specific Device Class .....	78
5.6.4.3	Advanced Door Lock Specific Device Class .....	79
5.6.4.4	Secure Keypad Door Lock Specific Device Class .....	80
5.7	Meter Generic Device Class .....	82
5.7.1	Mandatory Command Classes to Support .....	82
5.7.1.1	Basic Command Class Implementation .....	82
5.7.2	Mandatory Command Classes to Control .....	82
5.7.3	Specific Device Classes .....	83
5.7.3.1	No Specific Device Class defined .....	84
5.7.3.2	Simple Meter Specific Device Class .....	85
5.7.3.3	Advanced Energy Control Specific Device Class .....	86
5.8	Multilevel Sensor Generic Device Class .....	88
5.8.1	Mandatory Command Classes to Support .....	88
5.8.1.1	Basic Command Class Implementation .....	88
5.8.2	Mandatory Command Classes to Control .....	88
5.8.3	Specific Device Classes .....	89
5.8.3.1	Routing Multilevel Sensor Specific Device Class .....	90
5.9	Multilevel Switch Generic Device Class .....	92
5.9.1	Mandatory Command Classes to Support .....	92
5.9.2	Mandatory Command Classes to Control .....	92
5.9.3	Specific Device Classes .....	93
5.9.3.1	No Specific Device Class defined .....	94
5.9.3.2	Multilevel Power Switch Specific Device Class .....	95
5.9.3.3	Multilevel Scene Switch Specific Device Class .....	96
5.9.3.4	Multiposition Motor Specific Device Class (Not recommended) .....	98
5.9.3.5	Motor Control Class A Specific Device Class .....	99
5.9.3.6	Motor Control Class B Specific Device Class .....	101
5.9.3.7	Motor Control Class C Specific Device Class .....	103
5.9.3.8	Multilevel Tunable Color Light Specific Device Class .....	105
5.10	Pulse Meter Generic Device Class .....	106
5.10.1	Mandatory Command Classes to Support .....	106
5.10.1.1	Basic Command Class Implementation .....	106

5.10.2	Mandatory Command Classes to Control .....	106
5.10.3	Specific Device Classes .....	107
5.11	Remote Controller Generic Device Class .....	108
5.11.1	Mandatory Command Classes to Control .....	108
5.11.2	Specific Device Classes .....	109
5.11.2.1	Portable Remote Controller Specific Device Class .....	110
5.11.2.2	Portable Scene Controller Specific Device Class .....	111
5.11.2.3	Portable Installer Tool Specific Device Class .....	113
5.12	Remote Switch Generic Device Class .....	114
5.12.1	Mandatory Command Classes to Support .....	114
5.12.2	Mandatory Command Classes to Control .....	114
5.12.3	Specific Device Classes .....	115
5.12.3.1	Binary Remote Switch Specific Device Class .....	116
5.12.3.2	Multilevel Remote Switch Specific Device Class .....	116
5.12.3.3	Binary Toggle Remote Switch Specific Device Class (Not recommended) .....	117
5.12.3.4	Multilevel Toggle Remote Switch Specific Device Class (Not recommended) .....	117
5.13	Repeater Slave Generic Device Class .....	118
5.13.1	Mandatory Command Classes to Support .....	118
5.13.1.1	Basic Command Class Implementation .....	118
5.13.2	Mandatory Command Classes to Control .....	118
5.13.3	Specific Device Classes .....	119
5.13.3.1	Basic Repeater Slave Specific Device Class .....	120
5.14	Semi Interoperable Generic Device Class .....	121
5.14.1	Mandatory Command Classes to Support .....	121
5.14.1.1	Basic Command Class Implementation .....	122
5.14.2	Mandatory Command Classes to Control .....	122
5.14.3	Specific Device Classes .....	123
5.14.3.1	Energy Production Specific Device Class .....	124
5.15	Static Controller Generic Device Class .....	125
5.15.1	Recommended Command Classes to Support .....	125
5.15.1.1	Basic Command Class Implementation .....	125
5.15.2	Mandatory Command Classes to Control .....	125
5.15.3	Specific Device Classes .....	126
5.15.3.1	PC Controller Specific Device Class .....	127
5.15.3.2	Scene Controller Specific Device Class .....	128
5.15.3.3	Static Installer Tool Specific Device Class .....	130
5.15.3.4	Gateway Specific Device Class .....	131
5.16	Thermostat Generic Device Class .....	133
5.16.1	Mandatory Command Classes to Support .....	133
5.16.1.1	Basic Command Class Implementation .....	133
5.16.2	Mandatory Command Classes to Control .....	133
5.16.3	Specific Device Classes .....	133
5.16.3.1	Thermostat Heating Specific Device Class (Not permitted) .....	134
5.16.3.2	Thermostat General Specific Device Class (Not permitted) .....	135
5.16.3.3	Thermostat General V2 Specific Device Class .....	136
5.16.3.4	Setback Schedule Thermostat Specific Device Class .....	137
5.16.3.5	Setback Thermostat Specific Device Class .....	144
5.16.3.6	Setpoint Thermostat Specific Device Class .....	146
5.17	Toggle Switch Generic Device Class (Not recommended) .....	149
5.17.1	Mandatory Command Classes to Support .....	149
5.17.1.1	Basic Command Class Implementation .....	149
5.17.2	Mandatory Command Classes to Control .....	149
5.17.3	Specific Device Classes .....	150
5.17.3.1	Binary Toggle Switch Specific Device Class (Not recommended) .....	151
5.17.3.2	Multilevel Toggle Switch Specific Device Class (Not recommended) .....	151
5.18	Ventilation Generic Device Class .....	153
5.18.1	Mandatory Command Classes to Support .....	153

5.18.2	Mandatory Command Classes to Control .....	153
5.18.3	Specific Device Classes .....	153
5.18.3.1	No Specific Device Class defined .....	154
5.18.3.2	Residential Heat Recovery Ventilation Specific Device Class .....	155
5.19	Window Covering Generic Device Class (Not recommended) .....	156
5.19.1	Mandatory Command Classes to Support .....	156
5.19.1.1	Basic Command Class Implementation .....	156
5.19.2	Mandatory Command Classes to Control .....	156
5.19.3	Specific Device Classes .....	156
5.19.3.1	Simple Window Covering Control Specific Device Class (Not recommended) .....	157
<b>6</b>	<b>COMMAND CLASSES .....</b>	<b>158</b>
<b>7</b>	<b>DEVICE EXAMPLES .....</b>	<b>159</b>
7.1	Lighting Control Applications .....	159
7.1.1	Outlet adapter with dimming capability .....	159
7.1.1.1	Node Information .....	160
7.1.1.2	Switch Functionality .....	160
7.1.2	Battery-powered rocker switch .....	165
7.1.2.1	Node Information .....	166
7.1.2.2	Rocker Switch Functionality .....	166
7.2	Advanced Energy Control Applications .....	173
7.2.1	Deployment Scenarios .....	173
7.2.2	The Advanced Energy Control Architecture .....	174
7.2.2.1	Energy control approach .....	174
7.2.2.2	Advanced Energy Control Logical Device Model .....	175
7.2.2.3	Z-Wave Advanced Energy Control Metering Data Model .....	177
7.2.3	Security .....	180
7.2.3.1	Z-WaveSec .....	181
7.2.3.2	Z-WaveSecIP .....	181
7.2.3.3	Z-WaveSecSmartCard .....	182
7.2.4	Examples .....	183
7.2.4.1	Simple Meter with support for accumulated consumption .....	183
7.2.4.2	Advanced prepayment meter .....	185
7.2.4.3	Advanced Import / Export Meter .....	187
7.2.4.4	Advanced Import / Export Meter with rates .....	189
	<b>REFERENCES .....</b>	<b>191</b>
	<b>INDEX .....</b>	<b>192</b>

## List of Figures

Figure 1.	Z-Wave Home Control Network .....	3
Figure 2.	Z-Wave enabled application and underlying device structure .....	4
Figure 3.	Generic and Specific Device Class Structure .....	6
Figure 4.	Command class and related commands .....	14
Figure 5.	Node Information frame format .....	15
Figure 6.	Support/control command class list format .....	17
Figure 7.	Sequence diagram of operation after wake up .....	141
Figure 8.	Sequence diagram of operation for getting schedules .....	143
Figure 9.	Sequence diagram of operation after wake up .....	148
Figure 10.	Outlet adapter with dimming capabilities .....	159
Figure 11.	Battery-powered rocker switch .....	165
Figure 12.	Logical device model of the Advanced Energy Control profile .....	175
Figure 13.	Data model used in the Advanced Energy Control framework .....	177



Figure 14. Z-WaveSecIP solutions with respect to IP and Z/IP nodes .....	182
Figure 15. Data model for Simple Meter .....	183
Figure 16. Use case for Simple Meter .....	184
Figure 17. Data model for Adv. Prepayment Meter .....	185
Figure 18. Use case for Adv. Prepayment Meter .....	186
Figure 19. Data model for Adv. Import / Export Meter .....	187
Figure 20. Use case for Adv. Import / Export Meter .....	188
Figure 21. Data model for Adv. Import / Export Meter with rates .....	189
Figure 22. Use case for Adv. Import / Export Meter with rates .....	190

## List of Tables

Table 1. Basic Device Class identifiers .....	18
Table 2. Valid combinations of Generic and Specific Device Classes .....	25
Table 3. Generic Device Class identifiers .....	26
Table 4. Specific Device Class identifiers for the Binary Sensor Generic Device Class .....	59
Table 5. Specific Device Class identifiers for the Binary Switch Generic Device Class .....	63
Table 6. Specific Device Class identifiers for the Entry Control Generic Device Class .....	76
Table 7. Specific Device Class identifiers for the Multilevel Sensor Generic Device Class .....	89
Table 8. Specific Device Class identifiers for Multilevel Switch Generic Device Class .....	93
Table 9. Specific Device Class identifiers for the Pulse Meter Generic Device Class .....	107
Table 10. Specific Device Class identifiers for the Remote Controller Generic Device Class .....	109
Table 11. Specific Device Class identifiers for the Remote Switch Generic Device Class .....	115
Table 12. Specific Device Class identifiers for the Repeater Slave Generic Device Class .....	119
Table 13. Specific Device Class identifiers for the Semi Interoperable Generic Device Class .....	123
Table 14. Specific Device Class identifiers for the Static Controller Generic Device Class .....	126
Table 15. Specific Device Class identifiers for the Thermostat Generic Device Class .....	133
Table 16. Specific Device Class identifiers for the Toggle Switch Generic Device Class .....	150
Table 17. Specific Device Class identifiers for the Window Covering Generic Device Class .....	156

## 1 ABBREVIATIONS

Abbreviation	Explanation
AEC	Advanced Energy Control
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
DCP	Demand Control Plan
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 byte
MSB	Most significant byte
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 <a href="http://www.unicode.org/">http://www.unicode.org/</a>
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

## 2 INTRODUCTION

This document describes the device classes that **MUST** be used when designing and implementing Z-Wave™ products. The document also contains a description of the command classes and associated commands that **MUST** be used by Z-Wave products when communicating with each other. A subset of command classes is typically mandatory for a given type of device. All commands are handled by the application layer of the Z-Wave protocol.

A certification program based partly on self-certification and partly on 3<sup>rd</sup> party certification is put in place to enable manufacturers to get their product Z-Wave certified in order to ensure interoperability and get permission to carry the Z-Wave logo on the product. Interoperability is the successful interworking of multiple products from multiple manufacturers, for multiple applications, that **MAY** be based on multiple versions of Z-Wave. For details about the certification program refer to [1].

In case the current device and command classes described in this document do not sufficiently support the product to be developed, please contact Sigma Designs to initiate the required development process. For details about the device/command class development process refer to [2].

### 2.1 Purpose

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

### 2.2 Audience and Prerequisites

The audience of this document is Z-Wave partners and Sigma Designs.

### 2.3 Precedence of definitions

In terms of reviewing products for Z-Wave Compliance, definitions in this document have precedence over the header file ZW\_classcmd.h distributed as part of the Z-Wave Developer's Kit. However, the assignment of all device and command class hex identifiers can only be found in the header file ZW\_classcmd.h.

Device and Command Class Specifications approved as final version (ver. 1.00) during the device/command class development process have precedence over this document temporarily until integrated into this document.

### 2.4 Terms used in this document

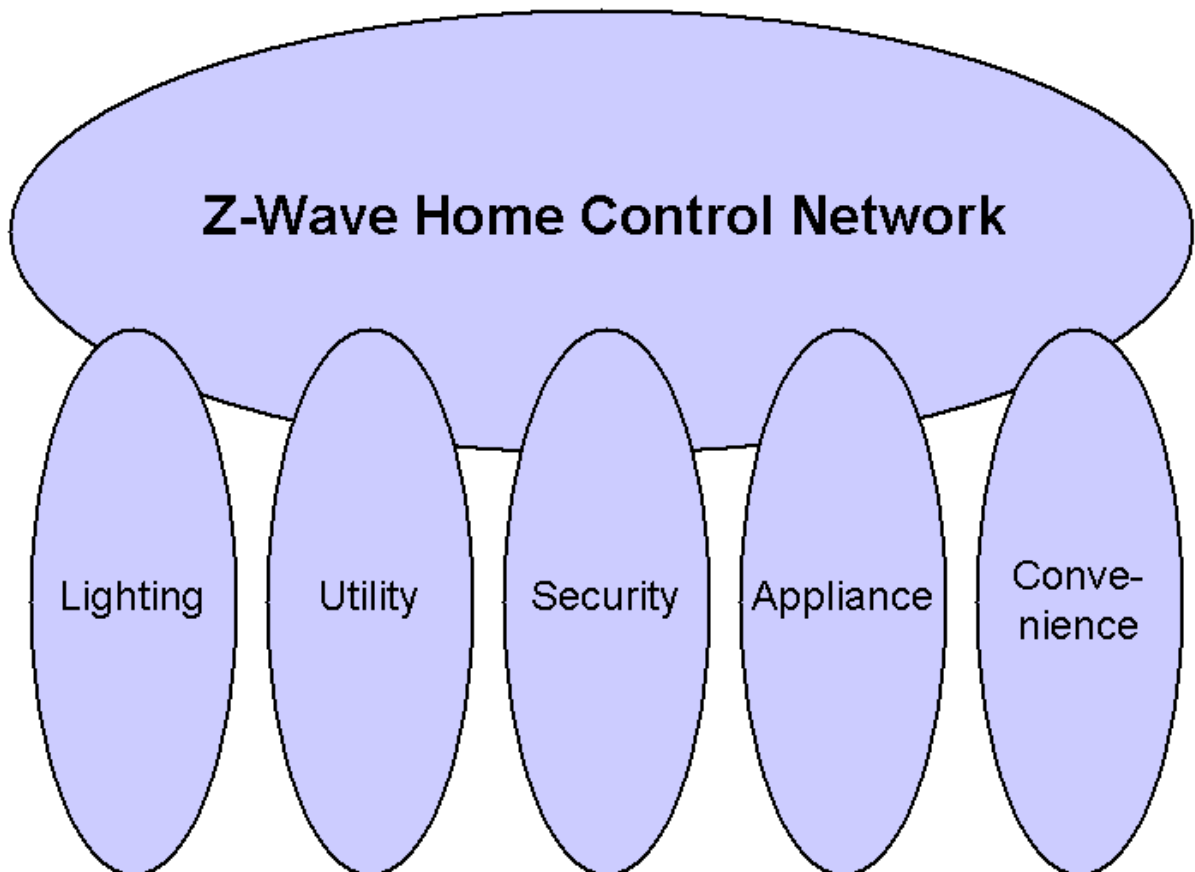
This document describes mandatory and optional aspects of the required compliance of a Z-Wave product to the Z-Wave standard.

The guidelines outlined in IETF RFC 2119 [9] with respect to key words used to indicate requirement levels are followed. Essentially, the key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119.

Products that are in violation any such statement are considered to be **not** Z-Wave compliant.

### 3 OVERVIEW

In order to achieve multi-vendor operation within and across subsystems in a home control system a standardized way to identify devices and interaction between devices **MUST** be defined. This allows a remote control from one vendor to control an outlet with dimming capabilities from another vendor within a lighting subsystem. Further, when a burglar subsystem is armed it can ask a lighting subsystem to start home simulation in order to exploit interoperability across different subsystems.

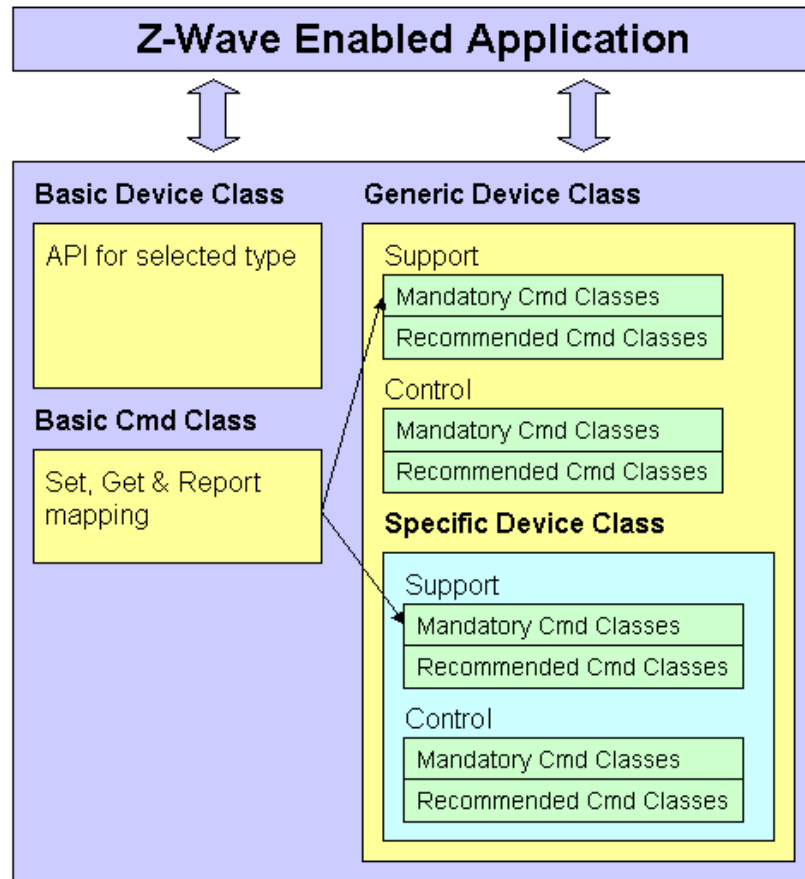


**Figure 1. Z-Wave Home Control Network**

The Z-Wave technology use basic, generic and specific device classes to identify a device and its role in the home control network. The basic device class determines the available functionality of the Z-Wave protocol depending on the library used. The Generic Device Class defines on a very high level the main functionality of a device. Generic device classes only defines the absolute minimum of functionality of a given type of device, so it is rare that any real products will be sufficiently described based on the Generic Device Class solely. The generic device class is therefore typically extended with a specific device class to define a set of additional mandatory and RECOMMENDED command classes that a given product **MUST** support to obtain the wanted functionality.

Communication between devices is carried out by a number of commands organized into a range of command classes. Command classes are the most fundamental grouping of commands, and these include all the necessary commands **REQUIRED** to implement a given functionality in a device. A device typically contains a number of different functionalities and consist therefore of a logic grouping of the necessary command classes into one device. The wanted functionality of a device is achieved by selecting the appropriate command classes in addition to the mandatory command classes for the

selected generic/specific device. This enables each vendor to provide devices with features that differentiate their product in the marketplace and at the same time achieve a high degree of interoperability.



**Figure 2. Z-Wave enabled application and underlying device structure**

The device application is based upon the selected combination of the basic and generic/specific device class defining a number of mandatory command classes. Additional command classes can be selected to extend the functionality of the device.

The command classes are divided into two different groups. One group of command classes are supported by the device itself, e.g. a temperature sensor will support a command class allowing other devices to read the temperature. Another group of command classes are used to control functionality in other devices, e.g. a battery operated switch controlling a ceiling outlet to a lamp. All combinations of generic/specific device class support the Basic Command Class as default and specify how it is mapped into one of the mandatory command classes supported by the device itself.

**NOTICE:** Compliance in controlling via Basic Command Class means the controller **MUST** implement one or more of the following features:

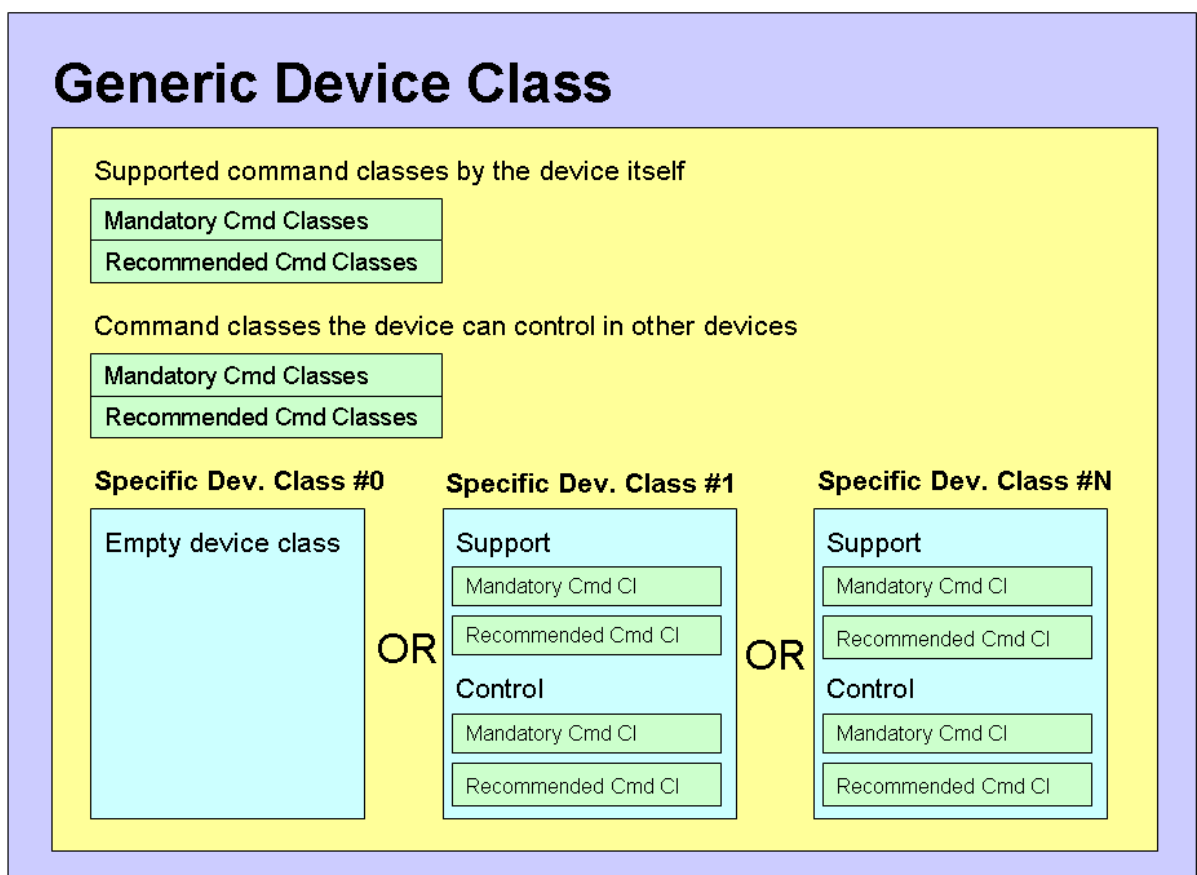
1. It controls a device by using fixed Basic Set Command values, typically 0x00 and 0xFF. Depending on the generic/specific device class used, this feature is either mandatory or **OPTIONAL**.
2. It supports configuration of all Basic Set Command values.
3. It supports learn capability of all the values by inquiring the device in question by a Basic Get Command.

**NOTICE:** It is mandatory to use the specified device and command classes in the customer applications. If further device and command classes are needed to implement the appropriate functionality in the application, please contact Sigma Designs to obtain the necessary classes.

### 3.1 Device Classes

A device class groups devices with the same functionality together.

The Basic Device Class provides the device with a certain role/functionality in the Z-Wave network based on the type of library used. A detailed description of all available Basic Device Classes is given in Chapter 4. In conjunction with the Basic Device Class a Generic/Specific Device Class is selected to achieve the wanted functionality. Generic Device Classes contain Command Classes that are mandatory for all devices that are within this device class.



**Figure 3. Generic and Specific Device Class Structure**

Specific Device Classes are a more detailed description/definition of a device based on a Generic Device Class. The Specific Device Class inherits all the mandatory commands from the Generic Device Class. An extra set of mandatory Command Classes can be specified for each Specific Device Class. The Specific Device Classes can also contain recommended Command Classes.

A Specific Device Class is therefore always a specialization of a Generic Device Class and the Specific Device Class always inherits all the commands that are specified for the Generic Device Class.

Devices **MUST** comply with one of the predefined Device Classes to ensure interoperability. Device Class compliance is checked by the Z-Wave Certification Program.

The strategy of building Device Classes allows all devices originating from the same Generic Device Class to be controlled by the same controller. Some more specialized commands can however only be used by specialized controllers, that knows how to control the Specific Device Class.

All the Generic and Specific Device Classes that have been defined to date are described in Chapter 5.



### 3.1.1 Controller Device Class Functionality

A controller plays an important role in a Z-Wave network because this device hosts important functionality to create, maintain and configure the home automation application. The following sections describe important rules to ensure that a controller-based device is capable of fulfilling this important role.

#### 3.1.1.1 Interoperability

To ensure interoperability a controller based device **MUST** comply with the following requirements:

1. It is not acceptable to block interoperability.
2. It is not acceptable to prevent inclusion of certified devices into a system or force exclusion of non-preferred devices after inclusion.
3. It is not required to control all mandatory command classes for a given device class (see section 3.1.1.2 regarding minimal controller functionality).
4. Devices from non-preferred manufacturers may be placed in a special section of the user interface; this section should be referred to as "Additional Z-Wave Ecosystem Devices". Additionally, it is acceptable to inform the user, upon inclusion of non-preferred devices that the device being included is not part of the vendors preferred ecosystem, and that control and support of the device by the vendor may be limited.
  - a. It is acceptable at this point to have the user select if they wish to continue with inclusion of the non-preferred device or reverse the action. This warning is only permitted to be shown once during each inclusion.
  - b. It is not permitted to display additional pop-ups, ask for pin codes or implement any other blocking or discouraging behavior for inclusion or control of non-preferred devices.
  - c. The Z-Wave Alliance recommends wording as follows. "You are about to include a Z-Wave compatible device that is not promoted by '*service provider name*' for use in this application. While the device should work as expected the device may or may not support all of the features of the '*service provider name*' recommended device."

### 3.1.1.2 Minimal Control Functionality

Minimal control functionality with respect to mandatory command class implementation takes into consideration consumer/user expectations and varies by generic/specific device class.

If the controller in question controls and/or supports a type of device then it must provide at least minimal functionality for all certified devices of that same type, regardless of manufacturer.

It is not acceptable to block interoperability although it is acceptable to differentiate between preferred and non-preferred devices of the same type as long as the controller provides at least minimal control of both.

The table of products below shows the minimal control, which a controller **MUST** provide. Please note that the minimum control requirements listed apply only if the functionality has been implemented for a given device.

Product Type/ Generic Device Class	Specific Device Class (SPECIFIC_TYPE...)	Minimal Control
<b>Door Lock</b> GENERIC_TYPE_ENTRY_CONTROL	...SECURE_KEYPAD_DOOR_LOCK	Lock, Unlock and Status (Locked/unlocked) using Door Lock Operation CC with security.
<b>Key fob</b> GENERIC_TYPE_GENERIC_CONTROLLER	...NOT_USED, ...PORTABLE_REMOTE_CONTROLLER, ...PORTABLE_SCENE_CONTROLLER, ...PORTABLE_INSTALLER_TOOL	None required.
<b>Keypad</b> GENERIC_TYPE_GENERIC_CONTROLLER	...NOT_USED, ...PORTABLE_REMOTE_CONTROLLER, ...PORTABLE_SCENE_CONTROLLER, ...PORTABLE_INSTALLER_TOOL	None required.
<b>Remote controller</b> GENERIC_TYPE_GENERIC_CONTROLLER	...NOT_USED, ...PORTABLE_REMOTE_CONTROLLER, ...PORTABLE_SCENE_CONTROLLER, ...PORTABLE_INSTALLER_TOOL	None required.
<b>Table top Controller</b> GENERIC_TYPE_GENERIC_CONTROLLER	...NOT_USED, ...PORTABLE_REMOTE_CONTROLLER, ...PORTABLE_SCENE_CONTROLLER, ...PORTABLE_INSTALLER_TOOL	None required.
<b>Whole house Energy meter</b> GENERIC_TYPE_METER	...NOT_USED, ...SIMPLE_METER	KWh using Meter CC. At least Accumulative values needs to be exposed to user.
<b>Meter Pulse type</b> GENERIC_TYPE_METER_PULSE	...NOT_USED	Count/KWh using Pulse Meter CC. At least Accumulative values.
<b>Door/window sensor</b> GENERIC_TYPE_SENSOR_BINARY	...ROUTING_SENSOR_BINARY	Application needs to be able to receive and interpret reports using Basic, Binary, or Alarm CC. Wake Up CC, must support Association but it does not have to be exposed to end user control.
<b>Freeze Alarm</b> GENERIC_TYPE_SENSOR_BINARY	...ROUTING_SENSOR_BINARY	Application needs to be able to receive and interpret reports using Basic, Binary, or Alarm CC. Wake Up CC, must support Association but it does not have to be exposed to end user control.
<b>PIR/Occupancy Sensor</b> GENERIC_TYPE_SENSOR_BINARY	...ROUTING_SENSOR_BINARY	Application needs to be able to receive and interpret reports using Basic, Binary, or Alarm CC. Wake Up CC, must support Association but it does not have to be exposed to end user control.
<b>Water leak Sensor/Alarm</b>	...ROUTING_SENSOR_BINARY	Application needs to be able to receive and interpret

Product Type/ Generic Device Class	Specific Device Class (SPECIFIC_TYPE...)	Minimal Control
GENERIC_TYPE_SENSOR_BINARY		reports using Basic, Binary, or Alarm CC. Wake Up CC, must support Association but it does not have to be exposed to end user control.
<b>Light Sensor</b> GENERIC_TYPE_SENSOR_MULTILEVEL	...ROUTING_SENSOR_MULTILEVEL	Light Level using Multilevel Sensor CC.
<b>Temperature Sensor</b> GENERIC_TYPE_SENSOR_MULTILEVEL	...ROUTING_SENSOR_MULTILEVEL	Application needs to be able to receive and interpret reports using Multilevel Sensor CC. Wake Up CC, must support Association but it does not have to be exposed to end user control.
<b>Gateway/Bridge/CentralCtrl</b> GENERIC_TYPE_STATIC_CONTROLLER	...NOT_USED, ...PC_CONTROLLER, ...STATIC_INSTALLER_TOOL	None required.
<b>In Home Display</b> GENERIC_TYPE_STATIC_CONTROLLER	...NOT_USED, ...PC_CONTROLLER, ...STATIC_INSTALLER_TOOL	None required.
<b>Scene controller</b> GENERIC_TYPE_STATIC_CONTROLLER	...NOT_USED, ...PC_CONTROLLER, ...STATIC_INSTALLER_TOOL	None required.
<b>Security Keypad</b> GENERIC_TYPE_STATIC_CONTROLLER	...NOT_USED, ...PC_CONTROLLER, ...STATIC_INSTALLER_TOOL	Application needs to be able to receive and interpret reports using Alarm CC. Wake Up CC, must support Association but it does not have to be exposed to end user control.
<b>Security Panel</b> GENERIC_TYPE_STATIC_CONTROLLER	...NOT_USED, ...PC_CONTROLLER, ...STATIC_INSTALLER_TOOL	None required.
<b>USB stick</b> GENERIC_TYPE_STATIC_CONTROLLER	...NOT_USED, ...PC_CONTROLLER, ...STATIC_INSTALLER_TOOL	None required.
<b>Appliance Module</b> GENERIC_TYPE_SWITCH_BINARY	...NOT_USED, ...POWER_SWITCH_BINARY, ...SCENE_SWITCH_BINARY	ON, OFF and Status of device using Basic or Binary CC.
<b>Appliance Module w. metering</b> GENERIC_TYPE_SWITCH_BINARY	...NOT_USED, ...POWER_SWITCH_BINARY, ...SCENE_SWITCH_BINARY	ON, OFF, Status of device and kWh using Basic or Binary CC and Meter CC. At least accumulative values needs to be exposed to user.
<b>On/OFF Switch</b> GENERIC_TYPE_SWITCH_BINARY	...NOT_USED, ...POWER_SWITCH_BINARY, ...SCENE_SWITCH_BINARY	ON, OFF (through Basic or Binary CC) using Basic or Binary CC.
<b>Receptacle</b> GENERIC_TYPE_SWITCH_BINARY	...NOT_USED, ...POWER_SWITCH_BINARY, ...SCENE_SWITCH_BINARY	ON, OFF and Status of device using Basic or Binary CC.
<b>Switch Plug-in</b> GENERIC_TYPE_SWITCH_BINARY	...NOT_USED, ...POWER_SWITCH_BINARY, ...SCENE_SWITCH_BINARY	ON, OFF and Status of device using Basic or Binary CC.
<b>Dimmer</b> GENERIC_TYPE_SWITCH_MULTILEVEL	...NOT_USED, ...POWER_SWITCH_MULTILEVEL, ...SCENE_SWITCH_MULTILEVEL	ON, OFF and Level (through Basic or Multilevel Switch CC) using Basic or Multilevel Switch CC.
<b>Dimmer Plug-in</b> GENERIC_TYPE_SWITCH_MULTILEVEL	...NOT_USED, ...POWER_SWITCH_MULTILEVEL, ...SCENE_SWITCH_MULTILEVEL	ON, OFF, Level and Status of device using Basic or Multilevel Switch CC.
<b>Fan control</b> GENERIC_TYPE_SWITCH_MULTILEVEL	...NOT_USED, ...POWER_SWITCH_MULTILEVEL, ...SCENE_SWITCH_MULTILEVEL	ON, OFF and Status of device using Basic or Multilevel Switch CC.

Product Type/ Generic Device Class	Specific Device Class (SPECIFIC_TYPE...)	Minimal Control
<b>Shade/Shutter controller</b> GENERIC_TYPE_SWITCH_MULTILEVEL	...CLASS_A_MOTOR_CONTROL, ...CLASS_B_MOTOR_CONTROL, ...CLASS_C_MOTOR_CONTROL	ON, OFF, Level and Status of device using Basic or Multilevel Switch CC. Level only if supported by device.
<b>Remote Dimmer</b> GENERIC_TYPE_SWITCH_REMOTE	...NOT_USED, ...SWITCH_REMOTE_MULTILEVEL	None required
<b>Remote Switch</b> GENERIC_TYPE_SWITCH_REMOTE	...NOT_USED, ...SWITCH_REMOTE_BINARY	None required
<b>Thermostat</b> GENERIC_TYPE_THERMOSTAT	...THERMOSTAT_GENERAL, ...THERMOSTAT_GENERAL_V2, ...THERMOSTAT_HEATING	Adjust Mode and Setpoints using Thermostat Mode and Thermostat Setpoint CC. At least Heat and Cool, and OFF modes if supported by device.
<b>Thermostat Setback</b> GENERIC_TYPE_THERMOSTAT	...NOT_USED, ...SETBACK_THERMOSTAT, ...SETPOINT_THERMOSTAT	ON, OFF and Status of device using Basic CC. It will switch between Normal/Comfort and Energy Saving.

### 3.1.1.3 Associations Handling

General recommendation to ensure correct command transmissions during the inclusion process the controller should keep track of whether the node in the network supports Association Command Class or Multi Channel Association Command Class and what versions of these classes are supported. If the controller is being included into a network, it should interview the nodes in the network for such information by means of retrieving the Node Information Frame.

#### Creating association links from the Z-Wave controller device without GUI

The Z-Wave controller without any LCD/GUI<sup>1</sup> is referred to as the “simple” controller and can only transmit association set commands based on a received Node Information Frame (NIF) and/or Multi Channel Capability Report (MCCR) to determine what node shall receive the association commands containing what node in the network it shall be associated to. The **first** received NIF/MCCR must be referred to as the Association Destination Node (ADN) of the association link, while the **second** received NIF/MCCR must be referred to as the Association Source Node (ASN), i.e. the ASN is the device that receives the Association Set or Multi Channel Association Set command.

**NOTE!** This controller can only setup association links to Group Id=1, unless the ASN supports Association Command Class V2 or higher.

Depending on what class the Z-Wave controller has implemented control for, below procedure must apply accordingly:

1. Controlled Command Classes (CC): Association CC
  - a. The controller can only accept reception of NIF. Received MCCR frames during association-mode<sup>2</sup> must be ignored, since the controller has not implemented Multi Channel Association CC; hence not capable of understanding these commands.
  - b. The **first** received NIF must be interpreted as the ADN and the **second** received NIF must be interpreted as the ASN for the association link.
  - c. What Group Id to use? The Group Id used for the association link must be determined based on below rules:
    - i. If the ASN supports only Association CC V1, the controller must use Group Id=1.
    - ii. If the controller has implemented control for Association CC V2 **AND** the ASN supports Association CC V2, the controller must first interview what Group Id to use by means of Association Specific Group Get command. The ASN will then respond with an Association Specific Group Report command containing the Group Id in which the controller must use for the association link.
      1. If the Association Specific Group Report contains Group Id=0 **OR** no Association Specific Group Report is received, the controller must use Group Id=1 as default.
      2. Else the controller must use the Group Id retrieved from the Association Specific Group Report command.
  - d. Transmitting Association Set: The Association Set command must be transmitted with the assembled data (ADN Id, ASN Id & Group Id) from above steps a-c.
  - e. Assigning Return Routes when using slaves: Subsequently after Association Set command, the controller must assign the return routes for the ADN Id in question into the ASN.

---

<sup>1</sup> UI as in the category of providing an interface that can display to the user a selection of devices in the network it can choose for association link configurations.

<sup>2</sup> The Z-Wave controller must integrate a function that allows the user to set the controller into association-mode, where it will expect the incoming frames according to the specified association processes.

2. Controlled CC's: Association, Multi Channel Association and Multi Channel
  - a. The controller can accept both NIF and MCCR frames.
  - b. Depending on the order of received frames the controller must follow below steps:
    - i. **First received frame = NIF, second frame = NIF**  
The Z-Wave controller must perform the same steps as described in 1.c. - 1.e.
    - ii. **First received frame = NIF, second frame = MCCR**  
The Z-Wave controller must perform the same steps as described in 1.c. - 1.e., however a Multi Channel Encapsulation must be transmitted to address the End Point of the ASN.  
The encapsulated command must be an Association Set.
    - iii. **First received frame = MCCR, second frame = NIF**  
The Z-Wave controller must perform the same steps as described in 1.c. - 1.e., however depending on what CC is supported by the ASN the following rules must apply:
      1. If ASN supports Multi Channel Association CC, a Multi Channel Association Set command with the End Point captured from the received MCCR frame must be transmitted to the ASN.
      2. Else an Association Set command with the Node Id of the ADN must be transmitted to the ASN.
    - iv. **First received frame = MCCR, second frame = MCCR**  
The Z-Wave controller must perform the same steps as described in 1.c. - 1.e., however a Multi Channel Encapsulation must be transmitted to address the End Point of the ASN.  
The encapsulated command must be a Multi Channel Association Set command with the End Point captured from the first received MCCR frame.

#### Creating association links from a Z-Wave controller device with GUI

The Z-Wave controller with LCD/GUI is referred to as the "advanced" controller. This controller can transmit Association commands without the prior steps of waiting for received NIF/MCCR frames to determine ADN, ASN and Group Id's, since the UI can provide a list of devices for the user to select for configuration of association links.

### 3.2 Command Classes

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 choose to implement control of selected commands within this command class depending on the requirements with respect to control.

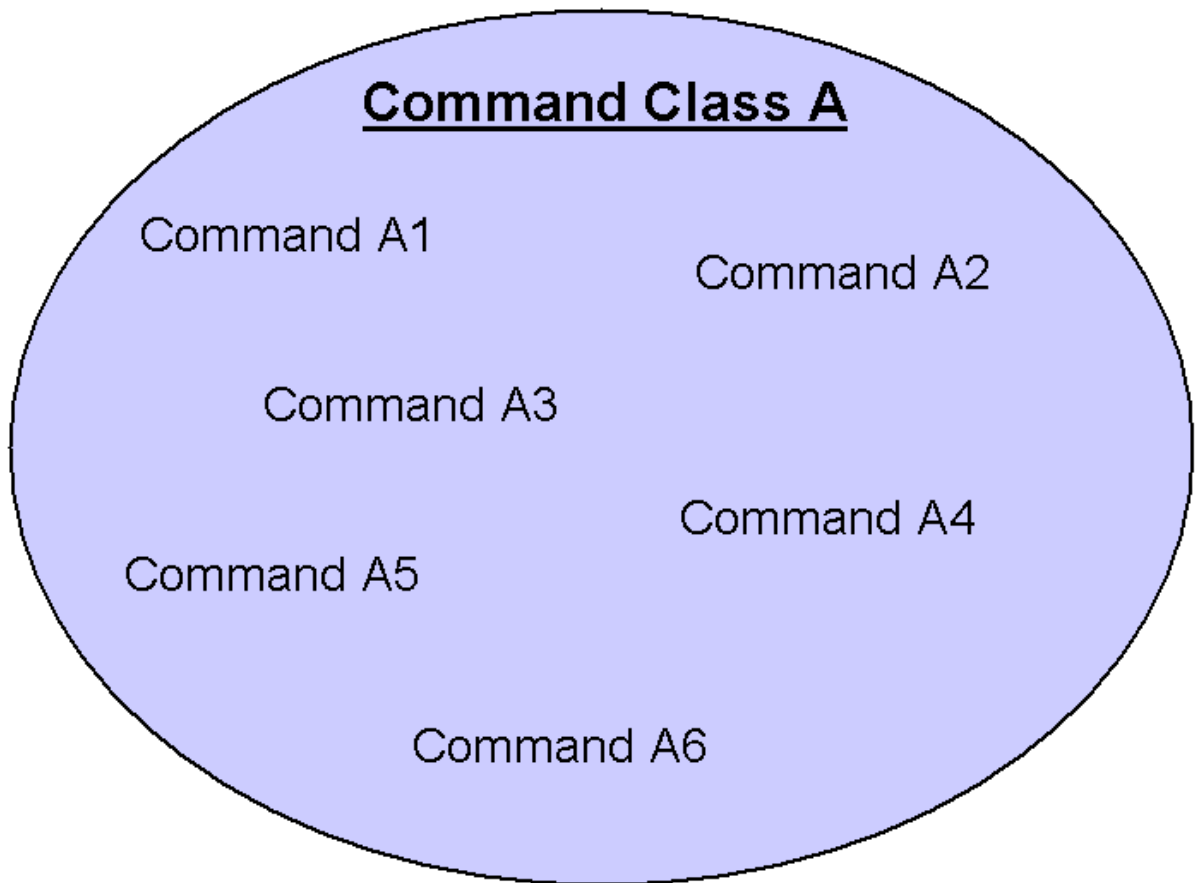


Figure 4. Command class and related commands

A detailed description of all the available command classes are given in [8]

**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”.

### 3.3 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							

**Figure 5. 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 is for future use. The Z-Wave protocol zero these fields and applications SHALL make no assumptions on the values of these fields nor perform processing based on their content.

#### 3.3.1 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 Chapter 4.

#### 3.3.2 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 also included in the routing table to assist as router in the network. The routing table is static during normal operation and this is the reason an always listening node **MUST NOT** be moved around in the network. In case the Listening flag is cleared then the node is non-listening and this is typically used for battery operated nodes being asleep when the protocol is idle to prolong battery lifetime. The battery operated node is not included in the routing table and is not as a router in the network, but 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 inform that this node supports other command classes than the mandatory for the selected generic/specific device class and that it needs to look at the supported command classes to fully control this device. The application **MUST** set the Optional Functionality flag in the packet.

## 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 Chapter 5.

## 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 Chapter 5.

## Command Class

The Command Class field specifies a list of Command Class identifiers and is set by the application. There can be from 1 to n different Command Classes included. Be aware of that n can differ depending on the library version used.

The list starts with Command Classes that describe functionality the device support itself. The mandatory Command Classes the device support itself **MUST** be shown in the list. The Basic Command Class is implicitly supported by all generic devices and is therefore not listed in the Node Information frame.

OPTIONAL the list of Command Classes can be appended with a list of the functionality the device can control in other devices. Even though mandatory Command Classes are defined describing functionality the device **MUST** control in other devices, it is **OPTIONAL** to include them in the list. To distinguish between the two lists of command classes the mark `COMMAND_CLASS_MARK` is used. The figure below shows how both lists are contained in 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 6. 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. A detailed description of all available Command Classes is given in Chapter 5.17.

## 4 BASIC DEVICE CLASSES

This chapter describes the Basic Device Classes used to identify the Z-Wave library used by the application for a given device. The valid Basic Device Classes are:

Basic Device Class	ZW_classcmd.h
Node is a portable controller	BASIC_TYPE_CONTROLLER
Node is a static controller	BASIC_TYPE_STATIC_CONTROLLER
Node is a slave	BASIC_TYPE_SLAVE
Node is a slave with routing capabilities	BASIC_TYPE_ROUTING_SLAVE

**Table 1. Basic Device Class identifiers**

Refer to ZW\_classcmd.h source code file for the assigned Basic Device Class identifiers and a detailed description of the capabilities of the Basic Device's can be found in [6].

## 4.1 Portable Controller

The portable controller can be used to control other nodes in a Z-Wave network. The portable controller can freely be moved around but it SHOULD of course always remain within direct range of minimum one node in the Z-Wave network. An important feature is the ability to include other devices locally into the Z-Wave network. A typical portable controller application could be a remote control.

The Basic Device Class identifier for a portable controller is BASIC\_TYPE\_CONTROLLER.

It is mandatory that the application on a portable controller support the following functionalities:

- Node address assignment
- Full routing support
- Controller replication of protocol related data

The mandatory supported functionalities are described in the following:

### Node address assignment

The portable controller has the ability to include/exclude other controllers and slaves in the Z-Wave network. This functionality is application level initiated. The capability to include/exclude other devices requires it is a primary controller. Controllers included by a primary controller are named secondary controllers, or inclusion controllers in case the primary controller also has the SIS role in the network.

### Full routing support

The portable controller SHALL support the full routing part of the Z-Wave protocol. This includes collecting Node Information, maintaining a routing table, creating routing lists and using routing lists for data transmissions. The controller will not assume a certain position in the network, but will try to discover which nodes can be reached directly and used for routing further out into the network.

If the controller is a primary or inclusions controller it is RECOMMENDED to initiate a rediscovery when noticing routing problems in the networks. If a SUC / SIS is present in the network a device is RECOMMENDED to call ZW\_RequestNetworkUpdate once per day and to call ZW\_RequestNetworkUpdate before configuring associations.

### Controller replication of protocol related data

The portable controller MUST be able to copy node information, routing information and other protocol related data to another controller in the Z-Wave network. Both replication send and replication receive on protocol level SHOULD be supported.

### SUC / SIS Capabilities

Inclusion controllers SHALL always try to enable an included controller as SIS, alternatively as SUC.

## 4.2 Static Controller

The static controller is tailored to controllers that are in a fixed position in the Z-Wave network. The static controller can serve as a receiver for e.g. sensors or other battery-operated devices that needs to send unsolicited reports to a controller. It can also be used in a system where the controller needs to know the status of each controlled device in the network. A system like this could be an internet gateway, which can be accessed remotely. By being able to learn and store the best route to all nodes in the system it can also significantly reduce the latency in larger systems. A static controller will typically be a secondary controller in a Z-Wave network, and can also take on the roles of SUC and SIS in the network.

The Basic Device Class identifier for a static controller is `BASIC_TYPE_STATIC_CONTROLLER`.

It is mandatory that the application on a static controller support the following functionalities:

- Node address assignment
- Full routing support
- Router support
- Controller replication of protocol related data

The mandatory supported functionalities are described in the following:

### Node address assignment

The portable controller has the ability to include/exclude other controllers and slaves in the Z-Wave network. This functionality is application level initiated. The capability to include/exclude other devices requires it is a primary controller. Controllers included by a primary controller are named secondary controllers, or inclusion controllers in case the primary controller also has the SIS role in the network.

### Full routing support

The portable controller SHALL support the full routing part of the Z-Wave protocol. This includes collecting Node Information, maintaining a routing table, creating routing lists and using routing lists for data transmissions. The controller will not assume a certain position in the network, but will try to discover which nodes can be reached directly and used for routing further out into the network.

If the controller is a primary or inclusions controller it is RECOMMENDED to initiate a rediscovery when noticing routing problems in the networks. If a SUC / SIS is present in the network a device is RECOMMENDED to call `ZW_RequestNetworkUpdate` once per day and to call `ZW_RequestNetworkUpdate` before configuring associations.

### Router support

The portable controller has the ability to act as router when a source node tries to reach a destination node out of direct range.

### Controller replication of protocol related data

The portable controller MUST be able to copy node information, routing information and other protocol related data to another controller in the Z-Wave network. Both replication send and replication receive on protocol level SHOULD be supported.

### SUC / SIS Capabilities

Every suitable static controller SHALL be capable to operate as a SUC. Inclusion controllers SHALL always try to enable an included controller as SIS, alternatively as SUC.

### 4.3 Slave

The slave is the simplest node available in a Z-Wave network. The slave node is unable to initiate transmission of data to other nodes in a Z-Wave network unless it is as a response to a request. A slave can only be mains powered. A typical slave application could be a light dimmer or router.

The Basic Device Class identifier for a slave is BASIC\_TYPE\_SLAVE.

It is mandatory that the application on a slave support following functionalities:

- Node add/remove
- Router support

The mandatory supported functionalities at application level are described in the following:

#### **Node add/remove**

The slave **MUST** have the ability to be added/removed in a Z-Wave network by a primary or inclusion controller.

All devices **SHALL** operate at normal transmit power when joining an existing network. All devices **SHALL** support reduce transmit power during adding a node when asked by the inclusion controller.

#### **Router support**

The slave has the ability to act as router when a source node tries to reach a destination node out of direct range.

It is **RECOMMENDED** that slave devices **SHOULD** ask to be rediscovered when multiple communication attempts have failed, the number of attempts can be application depended.

If a SUC / SIS is present in the network a device is **RECOMMENDED** to call `ZW_RequestNetworkUpdate` once per day and to call `ZW_RequestNetworkUpdate` before configuring associations.

#### 4.4 Routing Slave

The routing slave has the same Z-Wave protocol functionality as a slave, but in addition the node can initiate transmission of data to a limited number of other nodes in the Z-Wave network. A routing slave can both be mains or battery powered. A typical routing slave application could be a movement detector.

An enhanced slave has the same Z-Wave protocol functionality as a routing slave, but in addition, the enhanced slave is configured with an external EEPROM for storing application data. An enhanced and routing slave can be mains or battery powered. A typical enhanced slave application could be a push button, motion detector (PIR), and temperature sensor.

The Basic Device Class identifier for a slave with routing capabilities is BASIC\_TYPE\_ROUTING\_SLAVE.

It is mandatory that the routing slave support following functionalities:

- Node add/remove
- Router support

The mandatory supported functionalities are described in the following:

##### **Node add/remove**

The routing slave has the ability to be added/removed in a Z-Wave network using a primary or inclusion controller.

All devices SHALL operate at normal transmit power when joining a existing network. All devices SHALL support reduce transmit power during adding a node when asked by the inclusion controller.

##### **Router support**

The mains powered routing slave has the ability to act as router when a source node tries to reach a destination node out of direct range. A battery powered routing slave will not act as router.

It is RECOMMENDED that slave devices SHOULD ask to be rediscovered when multiple communication attempts have failed, the number of attempts can be application depended.

If a SUC / SIS is present in the network a device is RECOMMENDED to call ZW\_RequestNetworkUpdate once per day and to call ZW\_RequestNetworkUpdate before configuring associations.

## 5 GENERIC AND SPECIFIC DEVICE CLASSES

This chapter describes the Generic Device Classes together with the accompanying Specific Device Classes. The Generic Device Classes forms the main functionality for a given Z-Wave devices. Each Generic Device Class has a number of Specific Device Classes to define different variants of the given Z-Wave device.

The description of each Generic Device Class contains a description of the main functionality. The Generic Device Classes contain a set of mandatory Command Classes. If a device needs to implement other commands than described in the Generic Device Classes, a Specific Device Class is needed. Each Generic Device Class has an identifier assigned to it.



The table below shows the current list of valid combinations of Generic and Specific Device Classes:

Generic Device Classes	Specific Device Classes
Alarm Sensor	Basic Routing Alarm Sensor
	Routing Alarm Sensor
	Basic Zensor Net Alarm Sensor
	Zensor Net Alarm Sensor
	Advanced Zensor Net Alarm Sensor
	Basic Routing Smoke Sensor
	Routing Smoke Sensor
	Basic Zensor Net Smoke Sensor
	Zensor Net Smoke Sensor
	Advanced Zensor Net Smoke Sensor
AV Control Point	Specific Device Class not used
	Satellite Receiver
	Satellite Receiver V2
	Doorbell
Binary Sensor	Specific Device Class not used
	Routing Binary Sensor
Binary Switch	Specific Device Class not used
	Binary Power Switch
	Binary Scene Switch
	Binary Tunable Color Light
Display	Specific Device Class not used
	Simple Display
Entry Control	Specific Device Class not used
	Door Lock
	Advanced Door Lock
	Secure Keypad Door Lock
Meter	Specific Device Class not used
	Simple Meter
	Advanced Energy Control
Multilevel Sensor	Specific Device Class not used
	Routing Multilevel Sensor
Multilevel Switch	Specific Device Class not used
	Multilevel Power Switch
	Multilevel Scene Switch
	Multiposition Motor
	Motor Control A
	Motor Control B
	Motor Control C

Generic Device Classes	Specific Device Classes
	Multilevel Tunable Color Light
Non interoperable	
Pulse Meter	Specific Device Class not used
Remote Controller	Specific Device Class not used
	Portable Remote Controller
	Portable Scene Controller
	Portable Installer Tool
Remote Switch	Specific Device Class not used
	Binary Remote Switch
	Multilevel Remote Switch
	Binary Toggle Remote Switch
	Multilevel Toggle Remote Switch
Repeater Slave	Specific Device Class not used
	Basic Repeater Slave
Semi Interoperable	Specific Device Class not used
	Energy Production
Static Controller	Specific Device Class not used
	PC Controller
	Scene Controller
	Static Installer Tool
Thermostat	Specific Device Class not used
	Heating Thermostat
	General Thermostat
	General Thermostat V2
	Setback Schedule Thermostat
	Setback Thermostat
	Setpoint Thermostat
Toggle Switch	Specific Device Class not used
	Binary Toggle Switch
	Multilevel Toggle Switch
Ventilation	Specific Device Class not used
	Residential Heat Recovery Ventilation
Window Covering	Specific Device Class not used
	Simple Window Covering Control

**Table 2. Valid combinations of Generic and Specific Device Classes**

A given combination of a Generic/Specific Device Class can OPTIONAL support all command classes. However, the RECOMMENDED command classes specified in the specific device classes are the most obvious for such a device class.

Although a number of functionalities are OPTIONAL, it SHOULD be noted that if a function is supported, then all the commands of the corresponding Command Class MUST be supported.

The Generic Device Class identifiers are show in the table below:

Generic Device Classes	ZW_classcmd.h
Alarm Sensor	GENERIC_TYPE_SENSOR_ALARM
AV Control Point	GENERIC_TYPE_AV_CONTROL_POINT
Binary Sensor	GENERIC_TYPE_SENSOR_BINARY
Binary Switch	GENERIC_TYPE_SWITCH_BINARY
Display	GENERIC_TYPE_DISPLAY
Entry Control	GENERIC_TYPE_ENTRY_CONTROL
Meter	GENERIC_TYPE_METER
Multilevel Sensor	GENERIC_TYPE_SENSOR_MULTILEVEL
Multilevel Switch	GENERIC_TYPE_SWITCH_MULTILEVEL
Non interoperable	GENERIC_TYPE_NON_INTEROPERABLE
Pulse Meter	GENERIC_TYPE_METER_PULSE
Remote Controller	GENERIC_TYPE_GENERIC_CONTROLLER
Remote Switch	GENERIC_TYPE_SWITCH_REMOTE
Repeater Slave	GENERIC_TYPE_REPEATER_SLAVE
Semi Interoperable	GENERIC_TYPE_SEMI_INTEROPERABLE
Static Controller	GENERIC_TYPE_STATIC_CONTROLLER
Thermostat	GENERIC_TYPE_THERMOSTAT
Toggle Switch	GENERIC_TYPE_SWITCH_TOGGLE
Ventilation	GENERIC_TYPE_VENTILATION
Window Covering	GENERIC_TYPE_WINDOW_COVERING

**Table 3. Generic Device Class identifiers**

Refer to ZW\_classcmd.h source code file for the assigned Generic Device Class identifiers. In ZW\_classcmd.h additional constants and variables are defined to support the implementation.

Regarding the Specific Device Class identifiers refer to the following sections describing the valid combinations of Generic and Specific Device Classes.

**NOTICE:** The following device classes are considered legacy device classes that are superseded by newer Z-Wave specifications:

- Toggle Switch Generic Device Class
- Binary Toggle Remote Switch Specific Device Class
- Multilevel Toggle Remote Switch Specific Device Class
- Window Covering Generic Device Class
- Chimney Fan Device Class [4]
- Thermostat Heating Specific Device Class [5]
- General Thermostat Specific Device Class
- Satellite Receiver Specific Device Class

New Z-Wave enabled devices SHOULD NOT be based on these device classes. See in the corresponding sections below for further details.

## **5.1 Alarm Sensor Generic Device Class**

The Alarm Sensor Generic Device Class supports a range of alarm sensors, such as smoke, CO, CO<sub>2</sub>, flood, heat, etc.

The Alarm Sensor Generic Device Class identifier is equal to `GENERIC_TYPE_SENSOR_ALARM` and will typically be based on the Basic Device Class identifier `BASIC_TYPE_ROUTING_SLAVE`.

### **5.1.1 Mandatory Command Classes to Support**

The generic device **MUST** support the following command classes:

- Basic Command Class

#### **5.1.1.1 Basic Command Class Implementation**

Refer to the specific device classes.

### **5.1.2 Mandatory Command Classes to Control**

Nothing mandatory with respect to command classes the device can control in other devices.

### 5.1.3 Specific Device Classes

The table below shows the current list of defined Specific Device Classes for the Alarm Sensor Generic Device Class:

**Table 1, Specific Device Class identifiers for the Alarm Sensor Generic Device Class**

<b>Specific Device Class</b>	<b>ZW_classcmd.h</b>
Specific Device Class not used	SPECIFIC_TYPE_NOT_USED
Basic Routing Alarm Sensor	SPECIFIC_TYPE_BASIC_ROUTING_ALARM_SENSOR
Routing Alarm Sensor	SPECIFIC_TYPE_ROUTING_ALARM_SENSOR
Basic Zensor Net Alarm Sensor	SPECIFIC_TYPE_BASIC_ZENSOR_NET_ALARM_SENSOR
Zensor Net Alarm Sensor	SPECIFIC_TYPE_ZENSOR_NET_ALARM_SENSOR
Advanced Zensor Net Alarm Sensor	SPECIFIC_TYPE_ADV_ZENSOR_NET_ALARM_SENSOR
Basic Routing Smoke Sensor	SPECIFIC_TYPE_BASIC_ROUTING_SMOKE_SENSOR
Routing Smoke Sensor	SPECIFIC_TYPE_ROUTING_SMOKE_SENSOR
Basic Zensor Net Smoke Sensor	SPECIFIC_TYPE_BASIC_ZENSOR_NET_SMOKE_SENSOR
Zensor Net Smoke Sensor	SPECIFIC_TYPE_ZENSOR_NET_SMOKE_SENSOR
Advanced Zensor Net Smoke Sensor	SPECIFIC_TYPE_ADV_ZENSOR_NET_SMOKE_SENSOR

Refer to ZW\_classcmd.h source code file for the assigned Specific Device Class identifiers.

### 5.1.3.1 No Specific Device Class defined

This option is selected in case the generic level is appropriate for defining the device.

#### 5.1.3.1.1 Mandatory Command Classes to Support

No mandatory command classes are specified to access functionality in the specific device itself.

##### 5.1.3.1.1.1 Basic Command Class Implementation

The basic command class MUST be implemented in the following way:

Basic Set and Basic Get = The Basic Set and basic Get can freely be mapped to another command class supported by the device. The mapping MUST be documented in the User's Manual. The Basic Set and Get MUST be mapped within the same command class. In case the manufacture deems that no relevant commands are available for mapping the command a device receiving these commands MUST ignore them.

Basic Report = The Basic Report can be mapped to another command class supported by the device. The mapping MUST be documented in the User's Manual. The Basic Report MUST be mapped within the same command class as the Basic Get and MUST be in accordance with the mapped Get. In case the manufacture deems that no relevant commands are available for mapping the Basic Get command a device is not allowed to respond with a Basic Report under any circumstances.

### 5.1.3.2 Basic Routing Alarm Sensor Specific Device Class

A Basic Routing Alarm Sensor Specific Device Class inherits all the commands from the Alarm Sensor Generic Device Class.

The Basic Routing Alarm Sensor Specific Device Class supports simple battery operated alarm sensor devices based on typically a routing slave or static controller. The device supports the following functionalities:

1. Alarm Sensor (mandatory)
2. Association (mandatory)
3. Manufacturer Specific (mandatory)
4. Version (mandatory)

#### Alarm Sensor

The alarm sensor is used to signal an event in case the detector detects an alarm.

#### Association

The alarm sensor sends the detected alarm to the defined associations. The alarm can alternatively be broadcasted in case no associations are defined. In both cases are the Alarm Sensor Command Class used for this purpose.

#### 5.1.3.2.1 Mandatory Command Classes to Support

In addition to the inherited command classes from the generic device the following MUST also be supported:

- Alarm Sensor Command Class
- Association Command Class
- Manufacturer Specific Command Class
- Version Command Class

##### 5.1.3.2.1.1 Basic Command Class Implementation

The basic command class MUST be implemented in the following way:

Basic Set	=	Cannot be mapped because the mapped command class doesn't have a Set command. A device receiving this command MUST simply ignore it.
Basic Get	=	Alarm Sensor Get
Basic Report	=	Alarm Sensor Report (Sensor State field is mapped into the Basic Reports Value field)

#### 5.1.3.2.2 Recommended Command Classes to Support

No RECOMMENDED command classes supported by the device.

#### 5.1.3.2.3 Mandatory Command Classes to Control

The following command classes to control with are defined on the specific device class level:

- Alarm Sensor Command Class



#### 5.1.3.2.4 Recommended Command Classes to Control

Nothing RECOMMENDED with respect to command classes the device can control in other devices.

### 5.1.3.3 Routing Alarm Sensor Specific Device Class

A Routing Alarm Sensor Specific Device Class inherits all the commands from the Alarm Sensor Generic Device Class.

The Routing Alarm Sensor Specific Device Class supports simple battery operated alarm sensor devices based on typically a routing slave or static controller. The device supports the following functionalities:

5. Alarm Sensor (mandatory)
6. Association (mandatory)
7. Battery (mandatory)
8. Manufacturer Specific (mandatory)
9. Version (mandatory)

#### Alarm Sensor

The alarm sensor is used to signal an event in case the detector detects an alarm.

#### Association

The alarm sensor sends the detected alarm to the defined associations. The alarm can alternatively be broadcasted in case no associations are defined. In both cases are the Alarm Sensor Command Class used for this purpose.

#### 5.1.3.3.1 Mandatory Command Classes to Support

In addition to the inherited command classes from the generic device the following MUST also be supported:

- Alarm Sensor Command Class
- Association Command Class
- Battery Command Class
- Manufacturer Specific Command Class
- Version Command Class

##### 5.1.3.3.1.1 Basic Command Class Implementation

The basic command class MUST be implemented in the following way:

Basic Set	=	Cannot be mapped because the mapped command class doesn't have a Set command. A device receiving this command MUST simply ignore it.
Basic Get	=	Alarm Sensor Get
Basic Report	=	Alarm Sensor Report (Sensor State field is mapped into the Basic Reports Value field)

#### 5.1.3.3.2 Recommended Command Classes to Support

The Silence Alarm command class is RECOMMENDED in case the sensor alarm needs to be silenced.

#### 5.1.3.3.3 Mandatory Command Classes to Control

The following command classes to control with are defined on the specific device class level:

- Alarm Sensor Command Class

#### 5.1.3.3.4 Recommended Command Classes to Control

Nothing RECOMMENDED with respect to command classes the device can control in other devices.

#### 5.1.3.4 Basic Zensor Net Alarm Sensor Specific Device Class

A Basic Zensor Net Alarm Sensor Specific Device Class inherits all the commands from the Alarm Sensor Generic Device Class.

The Basic Zensor Net Alarm Sensor Specific Device Class supports battery operated Alarm Sensor devices in a Zensor Net. The devices can not be configured as Frequently Listening Routing Slaves (FLIRS). Notice that Zensor Net nodes don't support the node information frame because the broadcast forwarding mechanism is the only transport mechanism available. The device supports the following functionalities:

10. Alarm Sensor (mandatory)
11. Manufacturer Specific (mandatory)
12. Version (mandatory)

#### Alarm Sensor

The alarm sensor is used to signal an event in case the detector detects an alarm. The alarm is distributed to the other nodes in the Zensor Net using broadcast forwarding.

##### 5.1.3.4.1 Mandatory Command Classes to Support

In addition to the inherited command classes from the generic device the following **MUST** also be supported:

- Alarm Sensor Command Class
- Manufacturer Specific Command Class
- Version Command Class

##### 5.1.3.4.1.1 Basic Command Class Implementation

The basic command class **MUST** be implemented in the following way:

Basic Set	=	Cannot be mapped because the mapped command class doesn't have a Set command. A device receiving this command <b>MUST</b> simply ignore it.
Basic Get	=	Alarm Sensor Get
Basic Report	=	Alarm Sensor Report (Sensor State field is mapped into the Basic Reports Value field)

##### 5.1.3.4.2 Recommended Command Classes to Support

The Silence Alarm command class is **RECOMMENDED** in case the sensor alarm needs to be silenced.

##### 5.1.3.4.3 Mandatory Command Classes to Control

The following command classes to control with are defined on the specific device class level:

- Alarm Sensor Command Class

#### 5.1.3.4.4 Recommended Command Classes to Control

Nothing RECOMMENDED with respect to command classes the device can control in other devices.

#### 5.1.3.4.5 Recommended Command Classes to Control

Nothing RECOMMENDED with respect to command classes the device can control in other devices.

### 5.1.3.5 Zensor Net Alarm Sensor Specific Device Class

A Zensor Net Alarm Sensor Specific Device Class inherits all the commands from the Alarm Sensor Generic Device Class.

The Zensor Net Alarm Sensor Specific Device Class supports battery operated Alarm Sensor devices in a Zensor Net. The devices can not be configured as Frequently Listening Routing Slaves (FLiRS). Notice that Zensor Net nodes don't support the node information frame because the broadcast forwarding mechanism is the only transport mechanism available. The device supports the following functionalities:

13. Alarm Sensor (mandatory)
14. Battery (mandatory)
15. Manufacturer Specific (mandatory)
16. Version (mandatory)

### Alarm Sensor

The alarm sensor is used to signal an event in case the detector detects an alarm. The alarm is distributed to the other nodes in the Zensor Net using broadcast forwarding.

#### 5.1.3.5.1 Mandatory Command Classes to Support

In addition to the inherited command classes from the generic device the following **MUST** also be supported:

- Alarm Sensor Command Class
- Battery Command Class
- Manufacturer Specific Command Class
- Version Command Class

##### 5.1.3.5.1.1 Basic Command Class Implementation

The basic command class **MUST** be implemented in the following way:

Basic Set	=	Cannot be mapped because the mapped command class doesn't have a Set command. A device receiving this command <b>MUST</b> simply ignore it.
Basic Get	=	Alarm Sensor Get
Basic Report	=	Alarm Sensor Report (Sensor State field is mapped into the Basic Reports Value field)

#### 5.1.3.5.2 Recommended Command Classes to Support

The Silence Alarm command class is **RECOMMENDED** in case the sensor alarm needs to be silenced.

#### 5.1.3.5.3 Mandatory Command Classes to Control

The following command classes to control with are defined on the specific device class level:

- Alarm Sensor Command Class

#### 5.1.3.5.4 Recommended Command Classes to Control

Nothing RECOMMENDED with respect to command classes the device can control in other devices.

### 5.1.3.6 Advanced Zensor Net Alarm Sensor Specific Device Class

An Advanced Zensor Net Alarm Sensor Specific Device Class inherits all the commands from the Alarm Sensor Generic Device Class.

The Advanced Zensor Net Alarm Sensor Specific Device Class supports battery operated Alarm Sensor devices in a Zensor Net. The device is also configured as Frequently Listening Routing Slaves (FLiRS) enabling communication with a classic Z-Wave network. Notice that the FLiRS functionality enables support of the node information frame. The device supports the following functionalities:

17. Alarm Sensor (mandatory)
18. Association (mandatory)
19. Battery (mandatory)
20. Manufacturer Specific (mandatory)
21. Version (mandatory)

#### Alarm Sensor

The alarm sensor is used to signal an event in case the detector detects an alarm. The alarm is distributed to the other nodes in the Zensor Net using broadcast forwarding.

#### Association

The alarm sensor sends the detected alarm etc. to the defined associations. The alarm can alternatively be broadcasted in case no associations are defined. In both cases are the Alarm Sensor Command Class used for this purpose.

#### 5.1.3.6.1 Mandatory Command Classes to Support

In addition to the inherited command classes from the generic device the following **MUST** also be supported:

- Association Command Class
- Alarm Sensor Command Class
- Battery Command Class
- Manufacturer Specific Command Class
- Version Command Class

##### 5.1.3.6.1.1 Basic Command Class Implementation

The basic command class **MUST** be implemented in the following way:

Basic Set	=	Cannot be mapped because the mapped command class doesn't have a Set command. A device receiving this command <b>MUST</b> simply ignore it.
Basic Get	=	Alarm Sensor Get
Basic Report	=	Alarm Sensor Report (Sensor State field is mapped into the Basic Reports Value field)

#### 5.1.3.6.2 Recommended Command Classes to Support

The Silence Alarm command class is **RECOMMENDED** in case the sensor alarm needs to be silenced.



#### 5.1.3.6.3 Mandatory Command Classes to Control

The following command classes to control with are defined on the specific device class level:

- Alarm Sensor Command Class

#### 5.1.3.6.4 Recommended Command Classes to Control

Nothing RECOMMENDED with respect to command classes the device can control in other devices.

### 5.1.3.7 Basic Routing Smoke Sensor Specific Device Class

A Basic Routing Smoke Sensor Specific Device Class inherits all the commands from the Alarm Sensor Generic Device Class.

The Basic Routing Smoke Sensor Specific Device Class supports simple battery operated smoke sensor devices based on typically a routing slave or static controller. The device supports the following functionalities:

- 22. Alarm Sensor (mandatory)
- 23. Association (mandatory)
- 24. Manufacturer Specific (mandatory)
- 25. Version (mandatory)

#### Alarm Sensor

The smoke sensor is used to signal an event in case the detector detects smoke. Only Sensor Type equal to 'Smoke Sensor=0x01' is allowed.

#### Association

The smoke sensor sends the detected alarm to the defined associations. The alarm can alternatively be broadcasted in case no associations are defined. In both cases are the Alarm Sensor Command Class used for this purpose.

#### 5.1.3.7.1 Mandatory Command Classes to Support

In addition to the inherited command classes from the generic device the following MUST also be supported:

- Alarm Sensor Command Class
- Association Command Class
- Manufacturer Specific Command Class
- Version Command Class

##### 5.1.3.7.1.1 Basic Command Class Implementation

The basic command class MUST be implemented in the following way:

Basic Set	=	Cannot be mapped because the mapped command class doesn't have a Set command. A device receiving this command MUST simply ignore it.
Basic Get	=	Alarm Sensor Get
Basic Report	=	Alarm Sensor Report (Sensor State field is mapped into the Basic Reports Value field)

#### 5.1.3.7.2 Recommended Command Classes to Support

No RECOMMENDED command classes supported by the device.

#### 5.1.3.7.3 Mandatory Command Classes to Control

The following command classes to control with are defined on the specific device class level:

- Alarm Sensor Command Class

#### 5.1.3.7.4 Recommended Command Classes to Control

Nothing RECOMMENDED with respect to command classes the device can control in other devices.

### 5.1.3.8 Routing Smoke Sensor Specific Device Class

A Routing Smoke Sensor Specific Device Class inherits all the commands from the Alarm Sensor Generic Device Class.

The Simple Meter Specific Device Class supports simple battery operated smoke sensor devices based on typically a routing slave or static controller. The device supports the following functionalities:

- 26. Alarm Sensor (mandatory)
- 27. Association (mandatory)
- 28. Battery (mandatory)
- 29. Manufacturer Specific (mandatory)
- 30. Version (mandatory)

#### Alarm Sensor

The smoke sensor is used to signal an event in case the detector detects smoke. Only Sensor Type equal to 'Smoke Sensor=0x01' is allowed.

#### Association

The smoke sensor sends the detected alarm to the defined associations. The alarm can alternatively be broadcasted in case no associations are defined. In both cases are the Alarm Sensor Command Class used for this purpose.

#### 5.1.3.8.1 Mandatory Command Classes to Support

In addition to the inherited command classes from the generic device the following MUST also be supported:

- Alarm Sensor Command Class
- Association Command Class
- Battery Command Class
- Manufacturer Specific Command Class
- Version Command Class

##### 5.1.3.8.1.1 Basic Command Class Implementation

The basic command class MUST be implemented in the following way:

Basic Set	=	Cannot be mapped because the mapped command class doesn't have a Set command. A device receiving this command MUST simply ignore it.
Basic Get	=	Alarm Sensor Get
Basic Report	=	Alarm Sensor Report (Sensor State field is mapped into the Basic Reports Value field)

#### 5.1.3.8.2 Recommended Command Classes to Support

The Silence Alarm command class is RECOMMENDED in case the sensor alarm needs to be silenced.

#### 5.1.3.8.3 Mandatory Command Classes to Control

The following command classes to control with are defined on the specific device class level:

- Alarm Sensor Command Class

#### 5.1.3.8.4 Recommended Command Classes to Control

Nothing RECOMMENDED with respect to command classes the device can control in other devices.

### 5.1.3.9 Basic Zensor Net Smoke Sensor Specific Device Class

A Basic Zensor Net Smoke Sensor Specific Device Class inherits all the commands from the Alarm Sensor Generic Device Class.

The Basic Zensor Net Smoke Sensor Specific Device Class supports battery operated Smoke Sensor devices in a Zensor Net. The devices can not be configured as Frequently Listening Routing Slaves (FLIRS). Notice that Zensor Net nodes don't support the node information frame because the broadcast forwarding mechanism is the only transport mechanism available. The device supports the following functionalities:

31. Alarm Sensor (mandatory)
32. Manufacturer Specific (mandatory)
33. Version (mandatory)

#### Alarm Sensor

The smoke sensor is used to signal an event in case the detector detects smoke. Only Sensor Type equal to 'Smoke Sensor=0x01' is allowed. The alarm is distributed to the other nodes in the Zensor Net using broadcast forwarding.

#### 5.1.3.9.1 Mandatory Command Classes to Support

In addition to the inherited command classes from the generic device the following MUST also be supported:

- Alarm Sensor Command Class
- Manufacturer Specific Command Class
- Version Command Class

##### 5.1.3.9.1.1 Basic Command Class Implementation

The basic command class MUST be implemented in the following way:

Basic Set	=	Cannot be mapped because the mapped command class doesn't have a Set command. A device receiving this command MUST simply ignore it.
Basic Get	=	Alarm Sensor Get
Basic Report	=	Alarm Sensor Report (Sensor State field is mapped into the Basic Reports Value field)

#### 5.1.3.9.2 Recommended Command Classes to Support

The Silence Alarm command class is RECOMMENDED in case the Alarm Sensor needs to be silenced.

#### 5.1.3.9.3 Mandatory Command Classes to Control

The following command classes to control with are defined on the specific device class level:

- Alarm Sensor Command Class

#### 5.1.3.9.4 Recommended Command Classes to Control

Nothing RECOMMENDED with respect to command classes the device can control in other devices.

#### 5.1.3.10 Zensor Net Smoke Sensor Specific Device Class

A Zensor Net Smoke Sensor Specific Device Class inherits all the commands from the Alarm Sensor Generic Device Class.

The Zensor Net Smoke Sensor Specific Device Class supports battery operated Smoke Sensor devices in a Zensor Net. The devices can not be configured as Frequently Listening Routing Slaves (FLiRS). Notice that Zensor Net nodes don't support the node information frame because the broadcast forwarding mechanism is the only transport mechanism available. The device supports the following functionalities:

- 34. Alarm Sensor (mandatory)
- 35. Battery (mandatory)
- 36. Manufacturer Specific (mandatory)
- 37. Version (mandatory)

#### Alarm Sensor

The smoke sensor is used to signal an event in case the detector detects smoke. Only Sensor Type equal to 'Smoke Sensor=0x01' is allowed. The alarm is distributed to the other nodes in the Zensor Net using broadcast forwarding.

##### 5.1.3.10.1 Mandatory Command Classes to Support

In addition to the inherited command classes from the generic device the following MUST also be supported:

- Alarm Sensor Command Class
- Battery Command Class
- Manufacturer Specific Command Class
- Version Command Class

##### 5.1.3.10.1.1 Basic Command Class Implementation

The basic command class MUST be implemented in the following way:

Basic Set	=	Cannot be mapped because the mapped command class doesn't have a Set command. A device receiving this command MUST simply ignore it.
Basic Get	=	Alarm Sensor Get
Basic Report	=	Alarm Sensor Report (Sensor State field is mapped into the Basic Reports Value field)

##### 5.1.3.10.2 Recommended Command Classes to Support

The Silence Alarm command class is RECOMMENDED in case the sensor alarm needs to be silenced.

##### 5.1.3.10.3 Mandatory Command Classes to Control

The following command classes to control with are defined on the specific device class level:

- Alarm Sensor Command Class



#### 5.1.3.10.4 Recommended Command Classes to Control

Nothing RECOMMENDED with respect to command classes the device can control in other devices.

#### 5.1.3.11 Advanced Zensor Net Smoke Sensor Specific Device Class

An Advanced Zensor Net Smoke Sensor Specific Device Class inherits all the commands from the Alarm Sensor Generic Device Class.

The Advanced Zensor Net Smoke Sensor Specific Device Class supports battery operated Smoke Sensor devices in a Zensor Net. The device is also configured as Frequently Listening Routing Slaves (FLIRS) enabling communication with a classic Z-Wave network. Notice that the FLIRS functionality enables support of the node information frame. The device supports the following functionalities:

- 38. Alarm Sensor (mandatory)
- 39. Association (mandatory)
- 40. Battery (mandatory)
- 41. Manufacturer Specific (mandatory)
- 42. Version (mandatory)

#### Alarm Sensor

The smoke sensor is used to signal an event in case the detector detects smoke. Only Sensor Type equal to 'Smoke Sensor=0x01' is allowed. The alarm is distributed to the other nodes in the Zensor Net using broadcast forwarding.

#### Association

The smoke sensor sends the detected alarm etc. to the defined associations. The alarm can alternatively be broadcasted in case no associations are defined. In both cases are the Alarm Sensor Command Class used for this purpose.

##### 5.1.3.11.1 Mandatory Command Classes to Support

In addition to the inherited command classes from the generic device the following **MUST** also be supported:

- Association Command Class
- Alarm Sensor Command Class
- Battery Command Class
- Manufacturer Specific Command Class
- Version Command Class

##### 5.1.3.11.1.1 Basic Command Class Implementation

The basic command class **MUST** be implemented in the following way:

Basic Set	=	Cannot be mapped because the mapped command class doesn't have a Set command. A device receiving this command <b>MUST</b> simply ignore it.
Basic Get	=	Alarm Sensor Get
Basic Report	=	Alarm Sensor Report (Sensor State field is mapped into the Basic Reports Value field)

##### 5.1.3.11.2 Recommended Command Classes to Support

The Silence Alarm command class is **RECOMMENDED** in case the sensor alarm needs to be silenced.

#### 5.1.3.11.3 Mandatory Command Classes to Control

The following command classes to control with are defined on the specific device class level:

- Alarm Sensor Command Class

#### 5.1.3.11.4 Recommended Command Classes to Control

Nothing RECOMMENDED with respect to command classes the device can control in other devices.

## **5.2 AV Control Point Generic Device Class**

The AV Control Point generic device class is used by AV equipment such as TVs, stereo systems etc.

The AV Control Point Device Class identifier is equal to GENERIC\_TYPE\_AV\_CONTROL\_POINT.

### **5.2.1 Mandatory Command Classes to Support**

The following command classes to support are defined on the generic device class level:

- Basic Command Class

#### **5.2.1.1 Basic Command Class Implementation**

Refer to the specific devices for a description

### **5.2.2 Mandatory Command Classes to Control**

Nothing mandatory with respect to command classes the device can control in other devices.

### 5.2.3 Specific Device Classes

The table below shows the current list of defined Specific Device Classes for the AV Control Point Generic Device Class:

**Table 2. Specific Device Class identifiers for the AV Control Point Generic Device Class.**

Specific Device Class	ZW_classcmd.h
Specific Device Class not used	SPECIFIC_TYPE_NOT_USED
Satellite Receiver	SPECIFIC_TYPE_SATELLITE_RECEIVER
Satellite Receiver V2	SPECIFIC_TYPE_SATELLITE_RECEIVER_V2
Doorbell	SPECIFIC_TYPE_DOORBELL

Refer to ZW\_classcmd.h source code file for the assigned Specific Device Class identifiers.

### 5.2.3.1 No Specific Device Class defined

This option is selected in case the generic level is appropriate for defining the device.

#### 5.2.3.1.1 Mandatory Command Classes to Support

No mandatory command classes to support are defined on the specific device class level.

##### 5.2.3.1.1.1 Basic Command Class Implementation

The basic command class MUST be implemented in the following way:

Basic Set and Basic Get = The Basic Set and basic Get can freely be mapped to another command class supported by the device. The mapping MUST be documented in the User's Manual. The Basic Set and Get MUST be mapped within the same command class. In case the manufacture deems that no relevant commands are available for mapping the command a device receiving these commands MUST ignore them.

Basic Report = The Basic Report can be mapped to another command class supported by the device. The mapping MUST be documented in the User's Manual. The Basic Report MUST be mapped within the same command class as the Basic Get and MUST be in accordance with the mapped Get. In case the manufacture deems that no relevant commands are available for mapping the Basic Get command a device is not allowed to respond with a Basic Report under any circumstances.

### 5.2.3.2 Satellite Receiver Specific Device Class (Not recommended)

**It is not RECOMMENDED to use this specific device class for new devices.**

**Instead it is RECOMMENDED to base devices of this category on the Satellite Receiver V2 specific device class.**

The Satellite Receiver device can be used as a satellite receiver.

#### 5.2.3.2.1 Mandatory Command Classes to Support

The following command classes to support are defined on the specific device class level:

- Simple AV Control Command Class
- Manufacturer Specific Command Class
- Version Command Class

##### 5.2.3.2.1.1 Basic Command Class Implementation

The basic command class MUST be implemented in the following way:

Basic Set and Basic Get = The Basic Set and basic Get can freely be mapped to another command class supported by the device. The mapping MUST be documented in the User's Manual. The Basic Set and Get MUST be mapped within the same command class. In case the manufacture deems that no relevant commands are available for mapping the command a device receiving these commands MUST ignore them.

Basic Report = The Basic Report can be mapped to another command class supported by the device. The mapping MUST be documented in the User's Manual. The Basic Report MUST be mapped within the same command class as the Basic Get and MUST be in accordance with the mapped Get. In case the manufacture deems that no relevant commands are available for mapping the Basic Get command a device is not allowed to respond with a Basic Report under any circumstances.

#### 5.2.3.2.2 Recommended Command Classes to Support

The Screen Meta Data command class can be used in case the device is equipped with a display.

#### 5.2.3.2.3 Mandatory Command Classes to Control

Nothing mandatory with respect to command classes the device can control in other devices.

### 5.2.3.3 Satellite Receiver V2 Specific Device Class

The Satellite Receiver device can be used as a satellite receiver.

#### 5.2.3.3.1 Mandatory Command Classes to Support

The following command classes to support are defined on the specific device class level:

- Simple AV Control Command Class
- Basic Command Class
- Manufacturer Specific Command Class
- Version Command Class

##### 5.2.3.3.1.1 Basic Command Class Implementation

The basic command class SHALL be implemented in the following way:

Basic Set (Value = 0x00) = Device = Off (standby)

Basic Set (Value = 0xFF) = Device = On / On last volume

Basic Get = Get report

Basic Report (Value = 0x00) = Device = Off (standby)

Basic Report (Value = 0xFF) = Device = On / On last volume

Note: OPTIONAL: Volume control can be implemented using Basic Set / Get values 1...99.

#### 5.2.3.3.2 Recommended Command Classes to Support

The Screen Meta Data command class can be used in case the device is equipped with a display.

#### 5.2.3.3.3 Mandatory Command Classes to Control

Nothing mandatory with respect to command classes the device can control in other devices.



#### 5.2.3.4 Doorbell Specific Device Class

A Doorbell Specific Device Class inherits all the commands from the AV Control Point Generic Device Class. The doorbell application accepts a control command that triggers the doorbell to play a sound or any multimedia function that might be attached to the doorbell.

##### 5.2.3.4.1 Mandatory Command Classes to Support

The Doorbell Specific Device Class MUST support the below command classes.

- **Binary Sensor Command Class**

The Binary Sensor Command Class is used by the doorbell to report an event when the bell has been triggered. A Binary Sensor Report (0xFF) will report a detected event for an application specific period of timeout. Once this timeout has expired and no further event has been detected the Binary Sensor Report MUST contain (0x00) to indicate an idle state.

- **Association Command Class**

The Association Command Class is be used by the doorbell to configure event subscriptions e.g. unsolicited Binary Sensor Reports to be transmitted when doorbell has been triggered.

- **Manufacture Specific Command Class**

The Manufacture Specific Command Class will advertise manufacture specific information e.g. manufacture id, product id etc.

- **Version Command Class**

The Version Command Class is used to report the library type, protocol version and application version of the device.

##### 5.2.3.4.2 Basic Command Class Implementation

The Basic Command Class MUST be implemented in the following way:

Basic Set (0x00)	=	Basic Set (0x00). If the doorbell is replaying a ringing tone or application specific sound while receiving a Basic Set (0x00), the replay will be interrupted and stopped instantly. If the doorbell is idle i.e. no replay is active, while receiving Basic Set (0x00) the command MUST be ignored.
Basic Set (0xFF)	=	Basic Set (0xFF). If the doorbell is idle while receiving Basic Set (0xFF), the doorbell will replay a ringing tone or application specific sound. If replaying is active while receiving Basic Set (0xFF) the doorbell MUST interrupt the current replay and restart the replay of the ringing tone or application specific sound from start.
Basic Get	=	Binary Sensor Get
Basic Report	=	Binary Sensor Report

#### 5.2.3.4.3 Recommended Command Classes to Support

The following Command Classes can OPTIONAL be supported:

43. **Alarm Command Class**

Command Class to announce alarm condition.

44. **Protection Command Class**

The Doorbell MAY OPTIONAL support the Protection functionality. If the protection functionality is supported, the device SHALL be able to handle protection related commands.

45. **Battery Command Class**

The Battery Command Class can report to the controller about the battery status and level.

#### 5.2.3.4.4 Mandatory Command Classes to Control

Nothing mandatory with respect to command classes the device can control in other devices .

### 5.3 Binary Sensor Generic Device Class

The Binary Sensor device class is used to realize various sensors.

The Binary Sensor Device Class identifier is equal to `GENERIC_TYPE_SENSOR_BINARY` and will typically be based on the Basic Device Class identifier `BASIC_TYPE_SLAVE` or `BASIC_TYPE_ROUTING_SLAVE`.

The device supports the following functionalities:

- Binary Sensor (mandatory)

The supported functionalities are described in the following:

#### Binary Sensor

The Binary Sensor MUST support the Binary Sensor functionality. The Binary Sensor functionality can be used by event driven sensors, i.e. sensors that only send reports when a given action occurs. This is e.g. a movement sensor.

#### 5.3.1 Mandatory Command Classes to Support

The following command classes to support are defined on the generic device class level:

- Basic Command Class
- Binary Sensor Command Class

##### 5.3.1.1 Basic Command Class Implementation

The basic command class MUST be implemented in the following way:

Basic Set = Cannot be mapped because the mapped command class doesn't have a Set command. A device receiving this command MUST simply ignore it.

Basic Get = Binary Sensor Get

Basic Report = Binary Sensor Report

#### 5.3.2 Mandatory Command Classes to Control

Nothing mandatory with respect to command classes the device can control in other devices on this generic device class level.

### 5.3.3 Specific Device Classes

The table below shows the current list of defined Specific Device Classes for the Binary Sensor Generic Device Class:

Specific Device Class	ZW_classcmd.h
Specific Device Class not used	SPECIFIC_TYPE_NOT_USED
Routing Binary Sensor	SPECIFIC_TYPE_ROUTING_SENSOR_BINARY

**Table 4. Specific Device Class identifiers for the Binary Sensor Generic Device Class**

Refer to ZW\_classcmd.h source code file for the assigned Specific Device Class identifiers.

### 5.3.3.1 Routing Binary Sensor Specific Device Class

The Routing Binary Sensor device is used to realize various sensors. The device supports the following functionalities:

- Wakeup (RECOMMENDED)
- Association (RECOMMENDED)
- Clock (RECOMMENDED)
- Battery (RECOMMENDED)

The supported functionalities are described in the following:

#### Wakeup

The Routing Binary Sensor MAY OPTIONAL support the wakeup functionality. If the wakeup functionality is supported, then the Routing Binary Sensor SHALL be able to handle wakeup related commands.

#### Association

The Routing Binary Sensor MAY OPTIONAL support the association functionality. If the association functionality is supported, then the Routing Binary Sensor SHALL be able to handle association related commands.

#### Clock

The Routing Binary Sensor MAY OPTIONAL support the clock functionality. If the clock functionality is supported, then the Routing Binary Sensor SHALL be able to handle clock related commands.

#### Battery

The Routing Binary Sensor MAY OPTIONAL support the battery functionality. If the battery functionality is supported, then the Routing Binary Sensor SHALL be able to handle battery related commands.

#### 5.3.3.1.1 Mandatory Command Classes to Support

The following command classes to support are defined on the specific device class level:

- Binary Sensor Command Class

#### 5.3.3.1.2 Recommended Command Classes to Support

The manufacturer specific and version command class is highly RECOMMENDED because they can be used to identify the device and its capabilities.

The following Command Classes can OPTIONAL be supported:

- Wakeup Command Class
- Association Command Class
- Clock Command Class
- Battery Command Class.

#### 5.3.3.1.3 Mandatory Command Classes to Control

Nothing mandatory with respect to command classes the device can control in other devices on this specific device class level.

## **5.4 Binary Switch Generic Device Class**

The Binary Switch Generic device class supports all devices that need to switch between two states e.g. turn power on/off or other “simple” devices.

The Binary Switch Device Class identifier is equal to `GENERIC_TYPE_SWITCH_BINARY` and will typically be based on the Basic Device Class identifier `BASIC_TYPE_SLAVE` or `BASIC_TYPE_ROUTING_SLAVE`.

### **5.4.1 Mandatory Command Classes to Support**

The following command classes to support are defined on the generic device class level:

- Basic Command Class
- Binary Switch Command Class

### **5.4.2 Mandatory Command Classes to Control**

Nothing mandatory with respect to command classes the device can control in other devices.

### 5.4.3 Specific Device Classes

The table below shows the current list of defined Specific Device Classes for the Binary Switch Generic Device Class:

Specific Device Class	ZW_classcmd.h
Specific Device Class not used	SPECIFIC_TYPE_NOT_USED
Binary Power Switch	SPECIFIC_TYPE_POWER_SWITCH_BINARY
Binary Scene Switch	SPECIFIC_TYPE_SCENE_SWITCH_BINARY
Binary Tunable Color Light	SPECIFIC_TYPE_COLOR_TUNABLE_BINARY

**Table 5. Specific Device Class identifiers for the Binary Switch Generic Device Class**

Refer to ZW\_classcmd.h source code file for the assigned Specific Device Class identifiers.



#### **5.4.3.1 No Specific Device Class defined**

This option is selected in case the generic level is appropriate for defining the device.

##### **5.4.3.1.1 Mandatory Command Classes to Support**

No mandatory command classes to support are defined on the specific device class level.

###### **5.4.3.1.1.1 Basic Command Class Implementation**

The basic command class MUST be implemented in the following way:

- Basic Set = Binary Switch Set
- Basic Get = Binary Switch Get
- Basic Report = Binary Switch Report

##### **5.4.3.1.2 Mandatory Command Classes to Control**

Nothing mandatory with respect to command classes the device can control in other devices.

#### 5.4.3.2 Binary Power Switch Specific Device Class

A Binary Power Switch device inherits all the commands from the Binary Switch Device Class. Apart from these Command Classes it also supports the All Switch Command Class.

The Binary Power Switch Device Class supports devices that need to turn power on/off. The device supports the following functionalities:

- All switch (mandatory)
- Protection (RECOMMENDED)
- Clock (RECOMMENDED)

The supported functionalities are described in the following:

##### All Switch

The all switch functionality is mandatory. This functionality allows the switch to take part in all on/all off operations.

##### Protection

The Binary Power Switch MAY OPTIONAL support the Protection functionality. If the protection functionality is supported, then the device SHALL be able to handle protection related commands.

##### Clock

The Binary Power Switch MAY OPTIONAL support the clock functionality. If the clock functionality is supported, then the Binary Power Switch SHALL be able to handle clock related commands.

#### 5.4.3.2.1 Mandatory Command Classes to Support

The following command classes to support are defined on the specific device class level:

- All Switch Command Class

##### 5.4.3.2.1.1 Basic Command Class Implementation

The basic command class MUST be implemented in the following way:

- Basic Set = Binary Switch Set
- Basic Get = Binary Switch Get
- Basic Report = Binary Switch Report

#### 5.4.3.2.2 Recommended Command Classes to Support

The following Command Classes can OPTIONAL be supported:

- Clock Command Class
- Protection command class

The manufacturer specific and version command class is highly RECOMMENDED because they can be used to identify the device and its capabilities.

#### 5.4.3.2.3 Mandatory Command Classes to Control

Nothing mandatory with respect to command classes the device can control in other devices .

### 5.4.3.3 Binary Scene Switch Specific Device Class

A Binary Scene Switch device inherits all the mandatory command classes from the Binary Switch Generic Device Class.

The Binary Scene Switch Specific Device Class supports devices that need to turn power on/off and take part in scene operations. The device supports the following functionalities:

- Scene Activation (mandatory)
- Scene Actuator Configuration (mandatory)
- All Switch (mandatory)
- Manufacturer Specific (mandatory)

Some of the supported functionalities are described in the following:

#### Scene Activation

Used to control scenes in a binary scene switch.

#### Scene Actuator Configuration

Used to configure scenes in a binary scene switch.

#### All Switch

This functionality allows the switch to take part in all on/all off operations.

#### 5.4.3.3.1 Mandatory Command Classes to Support

The following command classes to support are defined on the specific device class level:

- Scene Activation Command Class
- Scene Actuator Configuration Command Class
- All Switch Command Class
- Manufacturer Specific Command Class

##### 5.4.3.3.1.1 Basic Command Class Implementation

The basic command class MUST be implemented in the following way:

- Basic Set = Binary Switch Set
- Basic Get = Binary Switch Get
- Basic Report = Binary Switch Report

#### 5.4.3.3.2 Recommended Command Classes to Support

The version command class can be used to retrieve the command class revisions etc. supported by the device.

#### 5.4.3.3.3 Mandatory Command Classes to Control

Nothing mandatory with respect to command classes the device can control in other devices .

#### 5.4.3.3.4 Recommended Command Classes to Control

No RECOMMENDED command classes to control with are defined on the specific device class level.

#### **5.4.3.4 Binary Tunable Color Light Specific Device Class**

The Binary Tunable Color Light Specific Device Class inherits all the commands from the Binary Switch Generic Device Class. The Binary switch command class is used to switch between two states e.g. turn power on/off.

The Device Class supports devices that can be controlled both in terms of light intensity and color. The device supports the following functionalities:

##### **5.4.3.4.1 Mandatory Command Classes to Support**

- 46. All Switch Command Class
- 47. Manufacture Specific Command Class
- 48. Basic Command Class
- 49. Color Control Command Class

All Switch, Manufacture Specific, Basic command class, please refer to [2]

##### **5.4.3.4.2 Recommended Command Classes to Support**

- 50. Node naming and Location Command class
  - Association

Node Naming and Location, Association command class, please refer to [2]

##### **5.4.3.4.3 Basic Command Class Implementation**

The basic command class MUST be implemented in the following way:

Basic Set = Binary Switch Set

Basic Get = Binary Switch Get

Basic Report = Binary Switch Report

##### **5.4.3.4.4 Mandatory Command Classes to Control**

No mandatory command classes to control functionality in other devices from the generic device.

## 5.5 Display Generic Device Class

The Display Generic Device Class supports a range of displays showing various data such as the current power consumption (electricity), or the number of emails in your in-box, or the outside temperature, or the weather forecast etc.

The Display Generic Device Class identifier is equal to `GENERIC_TYPE_DISPLAY` and the Basic Device Class identifier is typically `BASIC_TYPE_SLAVE`.

### 5.5.1 Mandatory Command Classes to Support

The generic device **MUST** support the following command classes:

- Basic Command Class

#### 5.5.1.1 Basic Command Class Implementation

Refer to the specific device classes.

### 5.5.2 Mandatory Command Classes to Control

Nothing mandatory with respect to command classes the device can control in other devices on this generic device class level.

### 5.5.3 Specific Device Classes

The table below shows the current list of defined Specific Device Classes for the Display Generic Device Class:

**Table 3, Specific Device Class identifiers for the Display Generic Device Class**

Specific Device Class	ZW_classcmd.h
Specific Device Class not used	SPECIFIC_TYPE_NOT_USED
Simple Display	SPECIFIC_TYPE_SIMPLE_DISPLAY

Refer to ZW\_classcmd.h source code file for the assigned Specific Device Class identifiers.



### 5.5.3.1 No Specific Device Class defined

Select this option in case the generic level is appropriate for defining the device.

#### 5.5.3.1.1 Mandatory Command Classes to Support

No mandatory command classes specified to access functionality in the specific device itself.

##### 5.5.3.1.1.1 Basic Command Class Implementation

The basic command class MUST be implemented in the following way:

- Basic Set and Basic Get = The Basic Set and basic Get can freely be mapped to another command class supported by the device. The mapping MUST be documented in the User's Manual. The Basic Set and Get MUST be mapped within the same command class. In case the manufacture deems that no relevant commands are available for mapping the command a device receiving these commands MUST ignore them.
- Basic Report = The Basic Report can be mapped to another command class supported by the device. The mapping MUST be documented in the User's Manual. The Basic Report MUST be mapped within the same command class as the Basic Get and MUST be in accordance with the mapped Get. In case the manufacture deems that no relevant commands are available for mapping the Basic Get command a device is not allowed to respond with a Basic Report under any circumstances.

### 5.5.3.2 Simple Display Specific Device Class

The Simple Display Specific Device Class supports a range of general-purpose displays showing text based information of various kinds. Typically, a Simple Display Specific Device Class devices use a slave. The device supports the following functionalities:

- 51. Screen Attributes (mandatory)
- 52. Screen Meta Data (mandatory)
- 53. Manufacturer Specific (mandatory)
- 54. Version (mandatory)

A Simple Display Specific Device Class inherits all the commands from the Display Generic Device Class.

#### Screen Attributes

Make it possible to obtain display parameters such as number of display lines, number of characters per line etc.

#### Screen Meta Data

Enables streaming of text based information to the device hosting the display.

#### 5.5.3.2.1 Mandatory Command Classes to Support

In addition to the inherited command classes from the generic device the device itself **MUST** also support the following:

- Screen Attributes Command Class
- Screen Meta Data Command Class
- Manufacturer Specific Command Class
- Version Command Class

##### 5.5.3.2.1.1 Basic Command Class Implementation

The basic command class **MUST** be implemented in the following way:

Basic Set and Basic Get = The Basic Set and basic Get can freely be mapped to another command class supported by the device. The mapping **MUST** be documented in the User's Manual. The Basic Set and Get **MUST** be mapped within the same command class. In case the manufacture deems that no relevant commands are available for mapping the command a device receiving these commands **MUST** ignore them.

Basic Report = The Basic Report can be mapped to another command class supported by the device. The mapping **MUST** be documented in the User's Manual. The Basic Report **MUST** be mapped within the same command class as the Basic Get and **MUST** be in accordance with the mapped Get. In case the manufacture deems that no relevant commands are available for mapping the Basic Get command a device is not allowed to respond with a Basic Report under any circumstances.

#### 5.5.3.2.2 Recommended Command Classes to Support

Nothing RECOMMENDED with respect to command classes supported by the device itself.

#### 5.5.3.2.3 Mandatory Command Classes to Control

Nothing mandatory with respect to command classes the device can control in other devices on this specific device class level.

#### 5.5.3.2.4 Recommended Command Classes to Control

Nothing RECOMMENDED with respect to command classes the device can control in other devices.

## **5.6 Entry Control Generic Device Class**

The Entry Control Generic Device Class is used to implement various forms of entry control devices such as door locks.

The Entry Control Generic Device Class identifier is equal to `GENERIC_TYPE_ENTRY_CONTROL` and will typically be based on the Basic Device Class identifier `BASIC_TYPE_ROUTING_SLAVE`.

It is RECOMMENDED to secure the communication for this type of device in order to provide message integrity and confidentiality with regards to access control.

### **5.6.1 Mandatory Command Classes to Support**

The generic device MUST support the following Command Classes:

- 55. Basic Command Class

#### **5.6.1.1 Basic Command Class Implementation**

Refer to the specific devices for a description.

### **5.6.2 Recommended Command Classes to Support**

No RECOMMENDED command classes to support are defined on the generic device class level.

### **5.6.3 Mandatory Command Classes to Control**

Nothing mandatory with respect to command classes the device can control in other devices.

#### 5.6.4 Specific Device Classes

The table below shows the current list of defined Specific Device Classes for the Entry Control Generic Device Class:

Specific Device Class	ZW_classcmd.h
Specific Device Class Not Used	SPECIFIC_TYPE_NOT_USED
Door Lock	SPECIFIC_TYPE_DOOR_LOCK
Advanced Door Lock	SPECIFIC_TYPE_ADVANCED_DOOR_LOCK
Secure Keypad Door Lock	SPECIFIC_TYPE_SECURE_KEYPAD_DOOR_LOCK

**Table 6. Specific Device Class identifiers for the Entry Control Generic Device Class**

Refer to ZW\_classcmd.h source code file for the assigned Specific Device Class identifiers.

#### 5.6.4.1 Specific Device Class Not Used

This option is selected in case the generic level is appropriate for defining the device.

##### 5.6.4.1.1 Mandatory Command Classes to Support

No mandatory command classes to support are defined on the specific device class level..

###### 5.6.4.1.1.1 Basic Command Class Implementation

The basic command class MUST be implemented in the following way:

- Basic Set and Basic Get = The Basic Set and basic Get can freely be mapped to another command class supported by the device. The mapping MUST be documented in the User's Manual. The Basic Set and Get MUST be mapped within the same command class. In case the manufacture deems that no relevant commands are available for mapping the command a device receiving these commands MUST ignore them.
- Basic Report = The Basic Report can be mapped to another command class supported by the device. The mapping MUST be documented in the User's Manual. The Basic Report MUST be mapped within the same command class as the Basic Get and MUST be in accordance with the mapped Get. In case the manufacture deems that no relevant commands are available for mapping the Basic Get command a device is not allowed to respond with a Basic Report under any circumstances.

#### 5.6.4.2 Door Lock Specific Device Class

An Door Lock Specific Device Class inherits all the commands from the Entry Control Generic Device Class.

The Door Lock Device Class supports devices that can be used to lock or unlock a door. The device supports the following functionalities:

##### 5.6.4.2.1 Mandatory Command Classes to Support

The Door Lock Specific Device MUST support the following Command Classes:

- 56. Lock Command Class  
Simple Command Class to control the Door Lock

##### 5.6.4.2.1.1 Basic Command Class Implementation

The basic command class MUST be implemented in the following way:

Basic Set = MUST be ignored to avoid unintentional operation. Cannot be mapped to another command class.

Basic Get = Lock Get

Basic Report = Lock Report

##### 5.6.4.2.2 Recommended Command Classes to Support

The manufacturer specific and version command class is highly RECOMMENDED because they can be used to identify the device and its capabilities

- 57. Manufacture Specific Command Class  
Command Class to advertise manufacture specific information

- 58. Version Command Class  
The Version Command Class can be used to report the library type, protocol version and application version from a device

##### 5.6.4.2.3 Mandatory Command Classes to Control

Nothing mandatory with respect to command classes the device can control in other devices.

### 5.6.4.3 Advanced Door Lock Specific Device Class

An Advanced Door Lock Specific Device Class inherits all the commands from the Entry Control Generic Device Class.

The Advanced Door Lock Device Class supports devices that can be used to lock or unlock a door. The device supports the following functionalities:

#### 5.6.4.3.1 Mandatory Command Classes to Support

The Door Lock Command Class is used to control the Advanced Door Lock Device and to monitor the status.

- 59. Door Lock Command Class  
Command Class to control the Door Lock
- 60. Manufacture Specific Command Class  
Command Class to advertise manufacture specific information
- 61. Version Command Class  
The Version Command Class can be used to report the library type, protocol version and application version from a device

##### 5.6.4.3.1.1 Basic Command Class Implementation

The Basic Command Class MUST be implemented in the following way:

- Basic Set = MUST be ignored to avoid unintentional operation. Cannot be mapped to another command class.
- Basic Get = Door Lock Operation Get
- Basic Report = Door Lock Operation Report

#### 5.6.4.3.2 Recommended Command Classes to Support

The following Command Classes can OPTIONAL be supported:

- 62. User Code Command Class  
The User Code Command Class can be used by the application to configure the user codes that can unlock the door.
- 63. Alarm Command Class  
Command Class to announce alarm condition e.g. wrong code entered or door is open but lock status is closed.
- 64. Protection Command Class  
The Door Lock MAY OPTIONAL support the Protection functionality. If the protection functionality is supported, the device SHALL be able to handle protection related commands
- 65. Battery Command Class  
The Battery Command Class can report to the controller about the battery status and level.

#### 5.6.4.3.3 Mandatory Command Classes to Control

Nothing mandatory with respect to command classes the device can control in other devices.



#### 5.6.4.4 Secure Keypad Door Lock Specific Device Class

The Secure Keypad Door Lock Specific Device Class inherits all the commands from the Entry Control Generic Device Class.

The Secure Keypad Door Lock Specific Device Class supports devices that can be used to lock or unlock a door both remotely and locally using keypad.

Please note that Security Command class is REQUIRED for these devices, and that is mandatory to support some of the command classes only encapsulated into the Security Command Class.

##### 5.6.4.4.1 Mandatory Command Classes to Support

The Secure Keypad Door Lock Specific Device MUST support the following Command Classes:

- Security Command Class  
Command Class to encrypt messages using AES-128
- Basic Command Class (secured)  
The basic command class can be used to obtain the operation mode of the lock, the mapping is given in Section 5.6.4.4.1.1.
- Door Lock Command Class (secured)  
Command Class to control the Door Lock
- User Code Command Class (secured)  
Command Class to set and retrieve user codes
- Manufacture Specific Command Class (unsecure)  
Command Class to advertise manufacture specific information
- Version Command Class (unsecure)  
The Version Command Class can be used to report the library type, protocol version and application version from a device

##### 5.6.4.4.1.1 Basic Command Class Implementation

The Basic Command Class MUST be implemented in the following way:

Basic Set	=	MUST be ignored to avoid unintentional operation. Cannot be mapped to another command class.
Basic Get	=	Door Lock Operation Get
Basic Report	=	Door Lock Operation Report

Due to Security concerns non-secure Basic Command Class commands will be ignored by device.

##### 5.6.4.4.2 Recommended Command Classes to Support

The following Command Classes can OPTIONAL be supported:

- Alarm Command Class (secure or not secure)  
Command Class to announce alarm condition e.g. wrong code entered or door is open but lock status is closed.
- Protection Command Class (secure or not secure)

The Door Lock MAY OPTIONAL support the Protection functionality. If the protection functionality is supported, the device SHALL be able to handle protection related commands

- Battery Command Class (secure or not secure)  
The Battery Command Class can report to the controller about the battery status and level.

#### 5.6.4.4.3 Mandatory Command Classes to Control

Nothing mandatory with respect to command classes the device can control in other devices.

## **5.7 Meter Generic Device Class**

The Meter Generic Device Class supports all water meters or energy metering devices (gas, electric etc.) which enable readout of accumulated values and thereby realizing 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.

The Meter Generic Device Class identifier is equal to `GENERIC_TYPE_METER` and will typically be based on the Basic Device Class identifier `BASIC_TYPE_SLAVE` or `BASIC_TYPE_ROUTING_SLAVE`.

### **5.7.1 Mandatory Command Classes to Support**

The generic device **MUST** support the following command classes:

- Basic Command Class

#### **5.7.1.1 Basic Command Class Implementation**

Refer to the specific device classes.

### **5.7.2 Mandatory Command Classes to Control**

Nothing mandatory with respect to command classes the device can control in other devices.

### 5.7.3 Specific Device Classes

The table below shows the current list of defined Specific Device Classes for the Meter Generic Device Class:

**Table 4, Specific Device Class identifiers for the Meter Generic Device Class**

Specific Device Class	ZW_classcmd.h
Specific Device Class not used	SPECIFIC_TYPE_NOT_USED
Simple Meter	SPECIFIC_TYPE_SIMPLE_METER
Advanced Energy Control	SPECIFIC_TYPE_ADV_ENERGY_CONTROL

Refer to ZW\_classcmd.h source code file for the assigned Specific Device Class identifiers.

### 5.7.3.1 No Specific Device Class defined

This option is selected in case the generic level is appropriate for defining the device.

#### 5.7.3.1.1 Mandatory Command Classes to Support

No mandatory command classes are specified to access functionality in the specific device itself.

##### 5.7.3.1.1.1 Basic Command Class Implementation

The basic command class MUST be implemented in the following way:

Basic Set and Basic Get = The Basic Set and basic Get can freely be mapped to another command class supported by the device. The mapping MUST be documented in the User's Manual. The Basic Set and Get MUST be mapped within the same command class. In case the manufacture deems that no relevant commands are available for mapping the command a device receiving these commands MUST ignore them.

Basic Report = The Basic Report can be mapped to another command class supported by the device. The mapping MUST be documented in the User's Manual. The Basic Report MUST be mapped within the same command class as the Basic Get and MUST be in accordance with the mapped Get. In case the manufacture deems that no relevant commands are available for mapping the Basic Get command a device is not allowed to respond with a Basic Report under any circumstances.

### 5.7.3.2 Simple Meter Specific Device Class

A Simple Meter Specific Device Class inherits all the commands from the Meter Generic Device Class.

The Simple Meter Specific Device Class supports readout of accumulated values in physical units from water meters and energy metering devices. The device supports the following functionalities:

- 66. Meter (mandatory)
- 67. Manufacturer Specific (mandatory)
- 68. Version (mandatory)

#### 5.7.3.2.1 Mandatory Command Classes to Support

In addition to the inherited command classes from the generic device the following **MUST** also be supported:

- Meter Command Class
- Manufacturer Specific Command Class
- Version Command Class

##### 5.7.3.2.1.1 Basic Command Class Implementation

The basic command class **MUST** be implemented in the following way:

Basic Set	=	Cannot be mapped because the mapped command class doesn't have a Set command. A device receiving this command <b>MUST</b> simply ignore it.
Basic Get	=	Meter Get
Basic Report	=	Meter Report

#### 5.7.3.2.2 Recommended Command Classes to Support

No **RECOMMENDED** command classes supported by the device.

#### 5.7.3.2.3 Mandatory Command Classes to Control

Nothing mandatory with respect to command classes the device can control in other devices.

#### 5.7.3.2.4 Recommended Command Classes to Control

Nothing **RECOMMENDED** with respect to command classes the device can control in other devices.

### 5.7.3.3 Advanced Energy Control Specific Device Class

The Advanced Energy Control Specific Device Class inherits all the commands from the Meter Generic Device Class.

The Advanced Energy Control Specific Device Class supports readout of accumulated and historical values in physical units from water meters and energy metering devices. The device supports the following functionalities:

- 69. Meter Table Monitor (mandatory)
- 70. Meter Table Configuration (mandatory)
- 71. Manufacturer Specific (mandatory)
- 72. Version (mandatory)

#### 5.7.3.3.1 Mandatory Command Classes to Support

In addition to the inherited command classes from the generic device the following **MUST** also be supported:

- Meter Table Monitor Command Class
- Meter Table Configuration Command Class
- Manufacturer Specific Command Class
- Version Command Class

##### 5.7.3.3.1.1 Basic Command Class Implementation

The basic command class **MUST** be implemented in the following way:

Basic Set and Basic Get = The Basic Set and basic Get can freely be mapped to another command class supported by the device. The mapping **MUST** be documented in the User's Manual. The Basic Set and Get **MUST** be mapped within the same command class. In case the manufacture deems that no relevant commands are available for mapping the command a device receiving these commands **MUST** ignore them.

Basic Report = The Basic Report can be mapped to another command class supported by the device. The mapping **MUST** be documented in the User's Manual. The Basic Report **MUST** be mapped within the same command class as the Basic Get and **MUST** be in accordance with the mapped Get. In case the manufacture deems that no relevant commands are available for mapping the Basic Get command a device is not allowed to respond with a Basic Report under any circumstances.

#### 5.7.3.3.2 Recommended Command Classes to Support

The following command classes can **OPTIONAL** be supported:

- Rate Table Configuration Command Class
- Rate Table Monitor Command Class
- Tariff Table Configuration Command Class
- Tariff table Monitor Command Class
- Demand Control Plan Configuration Command Class
- Demand Control Plan Monitor Command Class

No RECOMMENDED command classes supported by the device.

#### 5.7.3.3.3 Mandatory Command Classes to Control

Nothing mandatory with respect to command classes the device can control in other devices.

#### 5.7.3.3.4 Recommended Command Classes to Control

Nothing RECOMMENDED with respect to command classes the device can control in other devices.



## 5.8 Multilevel Sensor Generic Device Class

The Multilevel Sensor device class is used to realize various sensors.

The Multilevel Sensor Device Class identifier is equal to `GENERIC_TYPE_SENSOR_MULTILEVEL` and will typically be based on the Basic Device Class identifier `BASIC_TYPE_SLAVE` or `BASIC_TYPE_ROUTING_SLAVE`.

The device supports the following functionalities:

- Multilevel Sensor (mandatory)

The supported functionalities are described in the following:

### Multilevel Sensor

The device **MUST** support the Multilevel Sensor functionality. This functionality allows the Multilevel Sensor to report various values back to the controller.

#### 5.8.1 Mandatory Command Classes to Support

The following command classes to support are defined on the generic device class level:

- Basic Command Class
- Multilevel Sensor Command Class

##### 5.8.1.1 Basic Command Class Implementation

The basic command class **MUST** be implemented in the following way:

Basic Set = Cannot be mapped because the mapped command class doesn't have a Set command. A device receiving this command **MUST** simply ignore it.

Basic Get = Multilevel Sensor Get

Basic Report = Multilevel Sensor Report

#### 5.8.2 Mandatory Command Classes to Control

Nothing mandatory with respect to command classes the device can control in other devices on this generic device class level.

### 5.8.3 Specific Device Classes

The table below shows the current list of defined Specific Device Classes for the Multilevel Sensor Generic Device Class:

Specific Device Class	ZW_classcmd.h
Specific Device Class not used	SPECIFIC_TYPE_NOT_USED
Routing Multilevel Sensor	SPECIFIC_TYPE_ROUTING_SENSOR_MULTILEVEL

**Table 7. Specific Device Class identifiers for the Multilevel Sensor Generic Device Class**

Refer to ZW\_classcmd.h source code file for the assigned Specific Device Class identifiers.

### 5.8.3.1 Routing Multilevel Sensor Specific Device Class

The Routing Multilevel Sensor device is used to realize various sensors. The device supports the following functionalities:

- Wakeup (RECOMMENDED)
- Association (RECOMMENDED)
- Clock (RECOMMENDED)
- Battery (RECOMMENDED)

The supported functionalities are described in the following:

#### Wakeup

The Routing Multilevel Sensor MAY OPTIONAL support the wakeup functionality. If the wakeup functionality is supported, then the Routing Binary Sensor SHALL be able to handle wakeup related commands.

#### Association

The Routing Multilevel Sensor MAY OPTIONAL support the association functionality. If the association functionality is supported, then the Routing Binary Sensor SHALL be able to handle association related commands.

#### Clock

The sensor MAY OPTIONAL support the clock functionality. If the clock functionality is supported, then the sensor SHALL be able to handle clock related commands.

#### Battery

The sensor MAY OPTIONAL support the battery functionality. If the battery functionality is supported, then the sensor SHALL be able to handle battery related commands.

#### 5.8.3.1.1 Mandatory Command Classes to Support

The following command classes to support are defined on the specific device class level:

- Multilevel Sensor Command Class

#### 5.8.3.1.2 Recommended Command Classes to Support

The manufacturer specific and version command class is highly RECOMMENDED because they can be used to identify the device and its capabilities.

The following Command Classes can OPTIONAL be supported:

- Wakeup Command Class
- Association Command Class
- Clock Command Class
- Battery Command Class

#### 5.8.3.1.3 Mandatory Command Classes to Control

Nothing mandatory with respect to command classes the device can control in other devices on this specific device class level.

## 5.9 Multilevel Switch Generic Device Class

The Multilevel Switch Generic device class supports all devices that need to switch between several (more than two) states. This Device Class supports for example light dimmers.

The Multilevel Switch Device Class identifier is equal to `GENERIC_TYPE_SWITCH_MULTILEVEL` and will typically be based on the Basic Device Class identifier `BASIC_TYPE_SLAVE` or `BASIC_TYPE_ROUTING_SLAVE`.

### 5.9.1 Mandatory Command Classes to Support

The generic device **MUST** support the following Command Classes:

- Basic Command Class
- Multilevel Switch Command Class

### 5.9.2 Mandatory Command Classes to Control

Nothing mandatory with respect to command classes the device can control in other devices on this generic device class level.

### 5.9.3 Specific Device Classes

The table below shows the current list of defined Specific Device Classes for the Multilevel Switch Generic Device Class:

Specific Device Class	ZW_classcmd.h
Specific Device Class not used	SPECIFIC_TYPE_NOT_USED
Multilevel Power Switch	SPECIFIC_TYPE_POWER_SWITCH_MULTILEVEL
Multilevel Scene Switch	SPECIFIC_TYPE_SCENE_SWITCH_MULTILEVEL
Multiposition Motor	SPECIFIC_TYPE_MOTOR_MULTIPosition
Motor Control Class A	SPECIFIC_TYPE_CLASS_A_MOTOR_CONTROL
Motor Control Class B	SPECIFIC_TYPE_CLASS_B_MOTOR_CONTROL
Motor Control Class C	SPECIFIC_TYPE_CLASS_C_MOTOR_CONTROL
Multilevel Tunable Color Light	SPECIFIC_TYPE_COLOR_TUNABLE_MULTILEVEL

**Table 8. Specific Device Class identifiers for Multilevel Switch Generic Device Class**

Refer to ZW\_classcmd.h source code file for the assigned Specific Device Class identifiers.

### **5.9.3.1 No Specific Device Class defined**

This option is selected in case the generic level is appropriate for defining the device.

#### **5.9.3.1.1 Mandatory Command Classes to Support**

No mandatory command classes to support are defined on the specific device class level.

##### ***5.9.3.1.1.1 Basic Command Class Implementation***

The basic command class MUST be implemented in the following way:

Basic Set	=	Multilevel Switch Set
Basic Get	=	Multilevel Switch Get
Basic Report	=	Multilevel Switch Report

### 5.9.3.2 Multilevel Power Switch Specific Device Class

A Multilevel Power Switch device inherits all the commands from the Multilevel Switch Device Class. Apart from these Command Classes it also supports the All Switch Command Class.

The Multilevel Power Switch Device Class supports devices that need to turn power on/off and also be able to dim. The device supports the following functionalities:

- All switch (mandatory)
- Protection (RECOMMENDED)
- Clock (RECOMMENDED)

The supported functionalities are described in the following:

#### All Switch

The All Switch functionality is mandatory. This functionality allows the switch to take part in all on/all off operations.

#### Protection

The Multilevel Power Switch MAY OPTIONAL support the Protection functionality. If the protection functionality is supported, then the device SHALL be able to handle protection related commands.

#### Clock

The Multilevel Power Switch MAY OPTIONAL support the clock functionality. If the clock functionality is supported, then the Multilevel Power Switch SHALL be able to handle clock related commands.

#### 5.9.3.2.1 Mandatory Command Classes to Support

The following command classes to support are defined on the specific device class level:

- All Switch Command Class

##### 5.9.3.2.1.1 Basic Command Class Implementation

The basic command class MUST be implemented in the following way:

- Basic Set = Multilevel Switch Set
- Basic Get = Multilevel Switch Get
- Basic Report = Multilevel Switch Report

#### 5.9.3.2.2 Recommended Command Classes to Support

The manufacturer specific and version command class is highly RECOMMENDED because they can be used to identify the device and its capabilities.

The following Command Classes can OPTIONAL be supported:

- Clock Command Class
- Protection command class

#### 5.9.3.2.3 Mandatory Command Classes to Control

Nothing mandatory with respect to command classes the device can control in other devices on this specific device class level.



### 5.9.3.3 Multilevel Scene Switch Specific Device Class

A Multilevel Scene Switch device inherits all the mandatory command classes from the Multilevel Switch Generic Device Class.

The Multilevel Scene Switch Device Class supports devices that need to turn power on/off and also are able to dim. Furthermore the device can take part in scene operations. The device supports the following functionalities:

- Scene Activation (mandatory)
- Scene Actuator Configuration (mandatory)
- All Switch (mandatory)
- Manufacturer Specific (mandatory)

Some of the supported functionalities are described in the following:

#### Scene Activation

Is used to control scenes in the multilevel switch.

#### Scene Actuator Configuration

Is used to configure scenes in the multilevel switch.

#### All Switch

This functionality allows the switch to take part in all on/all off operations.

#### 5.9.3.3.1 Mandatory Command Classes to Support

The following command classes to support are defined on the specific device class level:

- Scene Activation Command Class
- Scene Actuator Configuration Command Class
- All Switch Command Class
- Manufacturer Specific Command Class

##### 5.9.3.3.1.1 Basic Command Class Implementation

The basic command class MUST be implemented in the following way:

Basic Set	=	Multilevel Switch Set
Basic Get	=	Multilevel Switch Get
Basic Report	=	Multilevel Switch Report

#### 5.9.3.3.2 Recommended Command Classes to Support

The version command class can be used to retrieve the command class revisions etc. supported by the device.

#### 5.9.3.3.3 Mandatory Command Classes to Control

Nothing mandatory with respect to command classes the device can control in other devices on this specific device class level.

#### 5.9.3.3.4 Recommended Command Classes to Control

No RECOMMENDED command classes to control with are defined on the specific device class level.

#### 5.9.3.4 Multiposition Motor Specific Device Class (Not recommended)

**It is not RECOMMENDED to use the Multiposition Motor Specific Device Class for new products.**

**Instead it is RECOMMENDED to base product development of this category on the Motor Control Class A/B/C Specific Device Class.**

A Multiposition Motor device inherits all the mandatory command classes from the Multilevel Switch Generic Device Class.

The Multiposition Motor Device Class supports simple motorized devices such as blinds and shades and other multiposition/bidirectional motors. Enable the movement to absolute positions and to start / stop relative movements using the same controller keys as for dimmers. Expand the view from window coverings to other types of motorized devices that share the same requirements and capabilities. The device supports the following functionalities:

- Manufacturer Specific (mandatory)
- Version (mandatory)

##### 5.9.3.4.1 Mandatory Command Classes to Support

In addition to the inherited command classes from the generic device the following **MUST** also be supported:

- Manufacturer Specific Command Class
- Version Command Class

No mandatory command classes to support are defined on the specific device class level.

##### 5.9.3.4.1.1 Basic Command Class Implementation

The basic command class **MUST** be implemented in the following way:

Basic Set = Multilevel Switch Set.

Basic Get = Multilevel Switch Get

Basic Report = Multilevel Switch Report

##### 5.9.3.4.2 Recommended Command Classes to Support

No RECOMMENDED command classes to support are defined on the specific device class level.

##### 5.9.3.4.3 Mandatory Command Classes to Control

Nothing mandatory with respect to command classes the device can control in other devices on this specific device class level.

##### 5.9.3.4.4 Recommended Command Classes to Control

No RECOMMENDED command classes to control with are defined on the specific device class level.

### 5.9.3.5 Motor Control Class A Specific Device Class

A Motor Control Class A Specific Device Class inherits all the commands from the Multilevel Switch Generic Device Class.

The Motor Control Class A is used for applications with no indication of position or endpoints hence it is not possible for this type of device to report its position at endpoints or position between endpoints. It can be ordered to move in directions and to stop the movement. Typical application implementing Motor Control Class A is e.g. a ceiling fan.

#### 5.9.3.5.1 Mandatory Command Classes to Support

The Motor Control Class A Specific Device Class MUST support and implement the below command classes accordingly.

- **Binary Switch Command Class**

The Binary Switch Command Class support simple controller devices requiring “stop-on-reverse” functionality.

**Note!** It is allowed for the Motor Control Class A Specific Device Class to report 0xFE as an unknown state/position.

Binary Switch Set (0x00) = If motor is stopped then start motion in direction A.  
If motor is already running in direction A, then ignore.  
If motor is running in direction B then stop motion.

Binary Switch Set (0x01 – 0xFE) = MUST be ignored.

Binary Switch Set (0xFF) = If motor is stopped then start motion in direction B.  
If motor is already running in direction B, then ignore.  
If motor is running in direction A then stop motion.

Binary Switch Get = Respond with Binary Switch Report. Valid values:  
0x00 = last moving direction A  
0xFE = unknown  
0xFF = last moving direction B

- **Multilevel Switch Command Class (Version 3)**

The Multilevel Switch Command Class support controller devices requiring absolute deterministic control of the Motor Control device.

**Note!** “dimming duration”, “start level” and “ignore start level” fields from the Multilevel Switch Set and Multilevel Switch Start Level Change commands MUST be ignored and in addition it is allowed for the Motor Control Class A Specific Device Class to report 0xFE as an unknown state/position.

Multilevel Switch Set (0x00) = Start motion in direction A.

Multilevel Switch Set (0x01-0x62 – 0x64-0xFE) = MUST be ignored.

Multilevel Switch Set (0x63 or 0xFF) = Start motion in direction B.

Multilevel Switch Get	=	Respond with Multilevel Switch Report. Valid values: 0x00 = last moving direction A 0xFE = unknown 0x63/0xFF = last moving direction B
Multilevel Switch Start Level Change	=	Command parameter Up/Down controls motion in direction A/B respectively and parameters Increment/Decrement and Step Size controls the general increment and decrement functions. All other command parameters MUST be ignored.
Multilevel Switch Stop Level Change	=	Stop motion.

- **Manufacture Specific Command Class**

The Manufacture Specific Command Class will advertise manufacture specific information e.g. manufacture id, product id etc.

- **Version Command Class**

The Version Command Class is used to report the library type, protocol version and application version of the device.

#### 5.9.3.5.2 Basic Command Class Implementation

The Basic Command Class MUST be implemented in the following way:

Basic Set	=	Multilevel Switch Set (see Mandatory Command Class to support)
Basic Get	=	Multilevel Switch Get (see Mandatory Command Class to support)
Basic Report	=	Multilevel Switch Report (see Mandatory Command Class to support)

#### 5.9.3.5.3 Recommended Command Classes to Support

No RECOMMENDED command classes to support.

#### 5.9.3.5.4 Mandatory Command Classes to Control

Nothing mandatory with respect to command classes the device can control in other devices on this specific device class level.

### 5.9.3.6 Motor Control Class B Specific Device Class

A Motor Control Class B Specific Device Class inherits all the commands from the Multilevel Switch Generic Device Class.

The Motor Control Class B is used for applications with knowledge of endpoints but imprecise position. It can be instructed to go to endpoints and to stop motion. This motor class can report its position at endpoints and also an imprecise position between endpoints. Typical application implementing Motor Control Class B is e.g. up/down motion of window shades where the position is measured over time.

#### 5.9.3.6.1 Mandatory Command Classes to Support

The Motor Control Class B Specific Device Class MUST support and implement the below command classes accordingly.

- **Binary Switch Command Class**

The Binary Switch Command Class support simple controller devices requiring “stop-on-reverse” functionality.

**Note!** It is allowed for the Motor Control Class B Specific Device Class to report 0xFE as an unknown state/position.

Binary Switch Set (0x00)	=	If motor is stopped then go to endpoint A. If motor is already running towards endpoint A then ignore. If motor is running towards endpoint B then stop motion.
Binary Switch Set (0x01 – 0xFE)	=	MUST be ignored.
Binary Switch Set (0xFF)	=	If motor is stopped then go to endpoint B. If motor is already running towards endpoint B then ignore. If motor is running towards endpoint A then stop motion.
Binary Switch Get	=	Respond with Binary Switch Report. Valid values: 0x00 = at endpoint A 0xFE = unknown 0xFF = at endpoint B

- **Multilevel Switch Command Class (Version 3)**

The Multilevel Switch Command Class support controller devices requiring absolute deterministic control of the Motor Control device.

**Note!** “dimming duration”, “start level” and “ignore start level” fields from the Multilevel Switch Set and Multilevel Switch Start Level Change commands MUST be ignored and in addition it is allowed for the Motor Control Class B Specific Device Class to report 0xFE as an unknown state/position.

Multilevel Switch Set (0x00)	=	Go to endpoint A.
Multilevel Switch Set (0x01-0x62 – 0x64-0xFE)	=	MUST be ignored.
Multilevel Switch Set (0x63 or 0xFF)	=	Go to endpoint B.

Multilevel Switch Get	=	Respond with Multilevel Switch Report. Valid values: 0x00 = at endpoint A 0x01-0x62 = approximate position between endpoints 0xFE = unknown 0x63/0xFF = at endpoint B
Multilevel Switch Start Level Change	=	Command parameter Up/Down controls motion in direction A/B respectively and parameters Increment/Decrement and Step Size controls the general increment and decrement functions. All other command parameters MUST be ignored.
Multilevel Switch Stop Level Change	=	Stop motion.

- **Manufacture Specific Command Class**

The Manufacture Specific Command Class will advertise manufacture specific information e.g. manufacture id, product id etc.

- **Version Command Class**

The Version Command Class is used to report the library type, protocol version and application version of the device.

### 5.9.3.6.2 Basic Command Class Implementation

The Basic Command Class MUST be implemented in the following way:

Basic Set	=	Multilevel Switch Set (see Mandatory Command Class to support)
Basic Get	=	Multilevel Switch Get (see Mandatory Command Class to support)
Basic Report	=	Multilevel Switch Report (see Mandatory Command Class to support)

### 5.9.3.6.3 Recommended Command Classes to Support

No RECOMMENDED command class to support.

### 5.9.3.6.4 Mandatory Command Classes to Control

Nothing mandatory with respect to command classes the device can control in other devices on this specific device class level.

### 5.9.3.7 Motor Control Class C Specific Device Class

A Motor Control Class C Specific Device Class inherits all the commands from the Multilevel Switch Generic Device Class.

The Motor Control Class C is used for application with knowledge of precise position i.e. the motor component provides feedback. This class can be instructed to go to as well as report back its exact position and endpoints. Typical application implementing Motor Control Class C is motor controllers having feedback mechanism.

#### 5.9.3.7.1 Mandatory Command Classes to Support

The Motor Control Class C Specific Device Class MUST support the below command classes.

- **Binary Switch Command Class**

The Binary Switch Command Class support simple controller devices requiring “stop-on-reverse” functionality.

**Note!** It is allowed for the Motor Control Class C Specific Device Class to report 0xFE as an unknown state/position.

Binary Switch Set (0x00) = If motor is stopped then go to endpoint A.  
If motor is already running towards endpoint A, then ignore.  
If motor is running towards endpoint B then stop motion.

Binary Switch Set (0x01 – 0xFE) = MUST be ignored.

Binary Switch Set (0xFF) = If motor is stopped then go to endpoint B.  
If motor is already running towards endpoint B, then ignore.  
If motor is running towards endpoint A then stop motion.

Binary Switch Get = Respond with Binary Switch Report. Valid values:  
0x00 = at endpoint A  
0xFE = unknown  
0xFF = at endpoint B

- **Multilevel Switch Command Class (Version 3)**

The Multilevel Switch Command Class support controller devices requiring absolute deterministic control of the Motor Control device.

**Note!** “dimming duration”, “start level” and “ignore start level” fields from the Multilevel Switch Set and Multilevel Switch Start Level Change commands MUST be ignored and in addition it is allowed for the Motor Control Class C Specific Device Class to report 0xFE as an unknown state/position.

Multilevel Switch Set (0x00) = Go to endpoint A.

Multilevel Switch Set (0x01-0x62) = Go to the exact instructed position.

Multilevel Switch Set (0x64-0xFE) = MUST be ignored.



Multilevel Switch Set (0x63 or 0xFF)	=	Go to endpoint B.
Multilevel Switch Get	=	Respond with Multilevel Switch Report. Valid values: 0x00 = at endpoint A 0x01-0x62 = <u>exact</u> position between endpoints 0x63/0xFF = at endpoint B
Multilevel Switch Start Level Change	=	Command parameter Up/Down controls motion in direction A/B respectively and parameters Increment/Decrement and Step Size controls the general increment and decrement functions. All other command parameters MUST be ignored.
Multilevel Switch Stop Level Change	=	Stop motion.

- **Manufacture Specific Command Class**

The Manufacture Specific Command Class will advertise manufacture specific information e.g. manufacture id, product id etc.

- **Version Command Class**

The Version Command Class is used to report the library type, protocol version and application version of the device.

#### 5.9.3.7.2 Basic Command Class Implementation

The Basic Command Class MUST be implemented in the following way:

Basic Set	=	Multilevel Switch Set (see Mandatory Command Class to support)
Basic Get	=	Multilevel Switch Get (see Mandatory Command Class to support)
Basic Report	=	Multilevel Switch Report (see Mandatory Command Class to support)

#### 5.9.3.7.3 Recommended Command Classes to Support

No RECOMMENDED command classes to support.

#### 5.9.3.7.4 Mandatory Command Classes to Control

Nothing mandatory with respect to command classes the device can control in other devices on this specific device class level.

### **5.9.3.8 Multilevel Tunable Color Light Specific Device Class**

The Multilevel Tunable Color Light Specific Device Class inherits all the commands from the Multilevel Switch Generic Device Class. The Multilevel switch command class is used to control the light intensity.

The Device Class supports devices that can be controlled both in terms of light intensity and color. The device supports the following functionalities:

#### **5.9.3.8.1 Mandatory Command Classes to Support**

- All Switch Command Class
- Manufacture Specific Command Class
- Basic Command Class
- Color Control Command Class

All Switch, Manufacture Specific, Basic command class, please refer to [2]

#### **5.9.3.8.2 Recommended Command Classes to Support**

- Node naming and Location Command Class
- Association Command Class

Node Naming and Location, Association command class, please refer to [2]

#### **5.9.3.8.3 Basic Command Class Implementation**

The basic command class MUST be implemented in the following way:

Basic Set = Multilevel Switch Set

Basic Get = Multilevel Switch Get

Basic Report = Multilevel Switch Report

#### **5.9.3.8.4 Mandatory Command Classes to Control**

No mandatory command classes to control functionality in other devices from the generic device.

## 5.10 Pulse Meter Generic Device Class

The Pulse Meter device class is used to realize various meters, such as gas, water and electricity meters. A meter typically reports a number of pulses, which have a specific meaning for a given meter.

The Pulse Meter Device Class identifier is equal to `GENERIC_TYPE_METER_PULSE` and will typically be based on the Basic Device Class identifier `BASIC_TYPE_SLAVE` or `BASIC_TYPE_ROUTING_SLAVE`.

The device supports the following functionalities:

- Pulse Meter (mandatory)

The supported functionalities are described in the following:

### Pulse Meter

The device **MUST** support the Pulse Meter functionality.

#### 5.10.1 Mandatory Command Classes to Support

The following command classes to support are defined on the generic device class level:

- Basic Command Class
- Pulse Meter Command Class

##### 5.10.1.1 Basic Command Class Implementation

The basic command class **MUST** be implemented in the following way:

Basic Set = Cannot be mapped because the mapped command class doesn't have a Set command. A device receiving this command **MUST** simply ignore it.

Basic Get = Pulse Meter Get

Basic Report = Pulse Meter Report

#### 5.10.2 Mandatory Command Classes to Control

Nothing mandatory with respect to command classes the device can control in other devices on this generic device class level.

### 5.10.3 Specific Device Classes

The table below shows the current list of defined Specific Device Classes for the Pulse Meter Generic Device Class:

Specific Device Class	ZW_classcmd.h
Specific Device Class not used	SPECIFIC_TYPE_NOT_USED

**Table 9. Specific Device Class identifiers for the Pulse Meter Generic Device Class**

Refer to ZW\_classcmd.h source code file for the assigned Specific Device Class identifiers.

## 5.11 Remote Controller Generic Device Class

The Remote Controller Generic Device Class is a device that can be used to control other nodes in a Z-Wave network. There are no restrictions on the placement of the controller i.e. it can be moved around.

Remote Controller Device Class identifier is equal to `GENERIC_TYPE_GENERIC_CONTROLLER` and will typically be based on the Basic Device Class identifier `BASIC_TYPE_CONTROLLER`.

This Device Class can be used to realize remote controllers. Very simple remote controllers can be implemented using the Basic Command Class to control other devices. This Device Class is tailored to controllers that MAY be moved around but it SHOULD of course always remain within range of minimum one device in the Z-Wave network.

### 5.11.1 Mandatory Command Classes to Control

The following command classes to control with are defined on the generic device class level:

- Basic Command Class

In case a remote controller device is configured to control an unknown device it SHALL use basic command to control this device.

### 5.11.2 Specific Device Classes

The table below shows the current list of defined Specific Device Classes for the Remote Controller Generic Device Class:

Specific Device Class	ZW_classcmd.h
Specific Device Class not used	SPECIFIC_TYPE_NOT_USED
Portable Remote Controller	SPECIFIC_TYPE_PORTABLE_REMOTE_CONTROLLER
Portable Scene Controller	SPECIFIC_TYPE_PORTABLE_SCENE_CONTROLLER
Portable Installer Tool	SPECIFIC_TYPE_PORTABLE_INSTALLER_TOOL

**Table 10. Specific Device Class identifiers for the Remote Controller Generic Device Class**

Refer to ZW\_classcmd.h source code file for the assigned Specific Device Class identifiers.

### 5.11.2.1 Portable Remote Controller Specific Device Class

The Portable Remote Controller Specific Device Class can be used to control other nodes in a Z-Wave network. There are no restrictions on the placement of the controller i.e. it can be moved around. The Portable Controller supports the following functionalities:

- Battery (RECOMMENDED)
- Clock (RECOMMENDED)

This Device Class can be used to realize e.g. remote controllers. This Device Class is tailored to controllers that MAY be moved around but it SHOULD of course always remain within range of minimum one device in the network.

The supported functionalities are described in the following:

#### Battery

The Portable Controller MAY OPTIONAL support the battery functionality. If the battery functionality is supported, then the Portable Controller SHALL be able to handle battery related commands.

#### Clock

The Portable Controller MAY OPTIONAL support the clock functionality. If the clock functionality is supported, then the Portable Controller SHALL be able to handle clock related commands.

#### 5.11.2.1.1 Mandatory Command Classes to Support

No mandatory command classes to support are defined on the specific device class level.

#### 5.11.2.1.2 Recommended Command Classes to Support

The manufacturer specific and version command class is highly RECOMMENDED because they can be used to identify the device and its capabilities.

The following Command Classes can OPTIONAL be supported:

- Clock Command Class
- Battery Command Class.

#### 5.11.2.1.3 Mandatory Command Classes to Control

Nothing mandatory with respect to command classes the device can control in other devices.

#### 5.11.2.1.4 Recommended Command Classes to Control

No RECOMMENDED command classes to control with are defined on the specific device class level.

### 5.11.2.2 Portable Scene Controller Specific Device Class

The Portable Scene Controller Specific Device Class can be used to configure and control scenes in other nodes in a Z-Wave network. There are no restrictions on the placement of the Portable Scene Controller i.e. it can be moved around.

The Portable Scene Controller supports the following functionality

- Association (mandatory - support)
- Scene Activation (mandatory - control)
- Scene Controller Configuration (mandatory - support)
- Manufacturer Specific (mandatory - support)
- Basic (mandatory – control)
- Scene Actuator Configuration (OPTIONAL - control)
- Controller replication of the application part (RECOMMENDED)

Some of the supported functionalities are described in the following:

#### Associations

Associations is used to organize nodes in different groups allowing the device to identify the nodes by a group identifier. The groups can also be copied to other devices.

#### Scene Activation

Is used to activate scenes in other devices and indicate the active scene on the portable scene-controlling device. The Basic command class to ensure backwards compatibility MUST control nodes in a group not supporting scene activation.

#### Scene Controller Configuration

Scene controller configuration is used to bind a physical identifier e.g. assigning a push button to a group of associations and the wanted scene settings.

#### Controller replication of the application part

The Portable Scene Controller MUST respond with ZW\_ReplicationReceiveComplete in case it receives the replication command class otherwise will the replication of protocol data fail. OPTIONAL can the Portable Scene Controller interpret the data transferred by the replication command class or simply skip the data.

#### 5.11.2.2.1 Mandatory Command Classes to Support

The following command classes to support are defined on the specific device class level:

- Association Command Class
- Scene Controller Configuration Command Class
- Manufacturer Specific Command Class

#### 5.11.2.2.2 Recommended Command Classes to Support

The version command class can be used to retrieve the command class revisions etc. supported by the device.



#### 5.11.2.2.3 Mandatory Command Classes to Control

The following command classes to control with are defined on the specific device class level:

- Scene Activation Command Class

#### 5.11.2.2.4 Recommended Command Classes to Control

A number of command classes are RECOMMENDED to enable a portable scene controller to exchange configuration information. The following command classes can OPTIONAL be used to control other devices:

- Association Command Class
- Scene Controller Configuration Command Class
- Scene Actuator Configuration Command Class
- All Switch Command Class

### 5.11.2.3 Portable Installer Tool Specific Device Class

The Portable Installer Tool Specific Device Class could be used to realize a controller which is primarily used to set-up a network and configure all nodes in the network. The installer tool is not envisioned to be used as a day-to-day controller. The controller is portable it SHOULD therefore be noticed that the SUC/SIS functionality MUST NOT be enable for this device. The device MUST however be able to appoint the SUC/SIS role to another node in the network.

#### 5.11.2.3.1 Mandatory Command Classes to Support

The following command classes to support are defined on the specific device class level:

- Controller Replication Command Class
- Multi Command Command Class
- Version Command Class
- Manufacturer Specific Command Class

#### 5.11.2.3.2 Recommended Command Classes to Support

The following command classes are RECOMMENDED to be supported:

- Clock Command Class
- Battery Command Class

#### 5.11.2.3.3 Mandatory Command Classes to Control

The following command classes to control with are defined on the specific device class level:

- Multi Channel Command Class - Version 2
- Controller Replication Command Class
- Association Command Class
- Multi Channel Association Command Class - Version 2
- Wake Up Command Class
- Configuration Command Class
- Manufacturer Specific Command Class
- Version Command Class

#### 5.11.2.3.4 Recommended Command Classes to Control

A number of command classes are RECOMMENDED in order to retrieve information and configure advanced devices:

- Binary Switch Command Class
- Multilevel Switch Command Class
- Battery Command Class
- Scene Controller Configuration Command Class
- Scene Actuator Configuration Command Class
- All Switch Command Class
- Node Naming and Location Command Class
- Grouping Name Command Class
- Protection Command Class
- Powerlevel Command Class

## 5.12 Remote Switch Generic Device Class

A Remote Switch device class provides a user interface to control the operation of one or more devices via the Z-Wave network.

The Remote Switch Device Class identifier is equal to `GENERIC_TYPE_SWITCH_REMOTE` and will typically be based on the Basic Device Class identifier `BASIC_TYPE_STATIC_CONTROLLER` or `BASIC_TYPE_ROUTING_SLAVE`.

### 5.12.1 Mandatory Command Classes to Support

No mandatory command classes to support are defined on the generic device class level.

### 5.12.2 Mandatory Command Classes to Control

The following command classes to control with are defined on the generic device class level:

- Basic Command Class

In case a remote switch device is configured to control an unknown device it SHALL use basic command to control this device.

### 5.12.3 Specific Device Classes

The table below shows the current list of defined Specific Device Classes for the Remote Switch Generic Device Class:

Specific Device Class	ZW_classcmd.h
Specific Device Class not used	SPECIFIC_TYPE_NOT_USED
Binary Remote Switch	SPECIFIC_TYPE_SWITCH_REMOTE_BINARY
Multilevel Remote Switch	SPECIFIC_TYPE_SWITCH_REMOTE_MULTILEVEL
Binary Toggle Remote Switch	SPECIFIC_TYPE_SWITCH_REMOTE_TOGGLE_BINARY
Multilevel Toggle Remote Switch	SPECIFIC_TYPE_SWITCH_REMOTE_TOGGLE_MULTILEVEL

**Table 11. Specific Device Class identifiers for the Remote Switch Generic Device Class**

Refer to ZW\_classcmd.h source code file for the assigned Specific Device Class identifiers.

### **5.12.3.1 Binary Remote Switch Specific Device Class**

A Binary Remote Switch device supports for example a switch there can control a Binary Switch Generic Device Class via the Z-Wave network.

#### **5.12.3.1.1 Mandatory Command Classes to Support**

No mandatory command classes to support are defined on the specific device class level.

#### **5.12.3.1.2 Recommended Command Classes to Support**

The manufacturer specific and version command class is highly RECOMMENDED because they can be used to identify the device and its capabilities..

#### **5.12.3.1.3 Mandatory Command Classes to Control**

The following command classes to control with are defined on the specific device class level:

- Binary Switch Command Class

### **5.12.3.2 Multilevel Remote Switch Specific Device Class**

A Multilevel Remote Switch device supports for example a switch there can control a Multilevel Switch Generic Device Class via the Z-Wave network.

#### **5.12.3.2.1 Mandatory Command Classes to Support**

No mandatory command classes to support are defined on the specific device class level.

#### **5.12.3.2.2 Recommended Command Classes to Support**

The manufacturer specific and version command class is highly RECOMMENDED because they can be used to identify the device and its capabilities.

#### **5.12.3.2.3 Mandatory Command Classes to Control**

The following command classes to control with are defined on the specific device class level:

- Multilevel Switch Command Class

### 5.12.3.3 Binary Toggle Remote Switch Specific Device Class (Not recommended)

**It is not RECOMMENDED to use this specific device class for new devices.**

**Instead it is RECOMMENDED to base devices of this category on the Binary Remote Switch specific device class.**

A Binary Toggle Remote Switch device supports for example a one-button switch there can control a Binary Toggle Switch Generic Device Class via the Z-Wave network.

#### 5.12.3.3.1 Mandatory Command Classes to Support

No mandatory command classes to support are defined on the specific device class level.

#### 5.12.3.3.2 Recommended Command Classes to Support

The manufacturer specific and version command class is highly RECOMMENDED because they can be used to identify the device and its capabilities.

#### 5.12.3.3.3 Mandatory Command Classes to Control

The following command classes to control with are defined on the specific device class level:

- Binary Toggle Switch Command Class

### 5.12.3.4 Multilevel Toggle Remote Switch Specific Device Class (Not recommended)

**It is not RECOMMENDED to use this specific device class for new devices.**

**Instead it is RECOMMENDED to base devices of this category on the Multilevel Remote Switch specific device class.**

A Multilevel Toggle Remote Switch device supports for example a one-button switch there can control a Multilevel Toggle Switch Generic Device Class via the Z-Wave network.

#### 5.12.3.4.1 Mandatory Command Classes to Support

No mandatory command classes to support are defined on the specific device class level.

#### 5.12.3.4.2 Recommended Command Classes to Support

The manufacturer specific and version command class is highly RECOMMENDED because they can be used to identify the device and its capabilities.

#### 5.12.3.4.3 Mandatory Command Classes to Control

The following command classes to control with are defined on the specific device class level:

- Multilevel Toggle Switch Command Class

### 5.13 Repeater Slave Generic Device Class

A Repeater Slave device class is a node that doesn't have any application functionality that can be controlled, but only exist in the network to assist other nodes to reach each other.

The Repeater Slave Device Class identifier is equal to `GENERIC_TYPE_REPEATER_SLAVE` and will typically be based on the Basic Device Class identifier `BASIC_TYPE_SLAVE` or `BASIC_TYPE_ROUTING_SLAVE`.

#### 5.13.1 Mandatory Command Classes to Support

The following command classes to support are defined on the generic device class level:

- Basic Command Class

##### 5.13.1.1 Basic Command Class Implementation

The basic command class MUST be implemented in the following way:

- |                         |   |   |
|-------------------------|---|---|
| Basic Set and Basic Get | = | The Basic Set and basic Get can freely be mapped to another command class supported by the device. The mapping MUST be documented in the User's Manual. The Basic Set and Get MUST be mapped within the same command class. In case the manufacture deems that no relevant commands are available for mapping the command a device receiving these commands MUST ignore them.   |
| Basic Report            | = | The Basic Report can be mapped to another command class supported by the device. The mapping MUST be documented in the User's Manual. The Basic Report MUST be mapped within the same command class as the Basic Get and MUST be in accordance with the mapped Get. In case the manufacture deems that no relevant commands are available for mapping the Basic Get command a device is not allowed to respond with a Basic Report under any circumstances. |

#### 5.13.2 Mandatory Command Classes to Control

Nothing mandatory with respect to command classes the device can control in other devices on this generic device class level.

### 5.13.3 Specific Device Classes

The table below shows the current list of defined Specific Device Classes for the Repeater Slave Generic Device Class:

Specific Device Class	ZW_classcmd.h
Specific Device Class not used	SPECIFIC_TYPE_NOT_USED
Basic Repeater Slave	SPECIFIC_TYPE_REPEATER_SLAVE

**Table 12. Specific Device Class identifiers for the Repeater Slave Generic Device Class**

Refer to ZW\_classcmd.h source code file for the assigned Specific Device Class identifiers.



### 5.13.3.1 Basic Repeater Slave Specific Device Class

The Basic Repeater Slave device can be used to repeat frames using several hops to extend range.

The Basic Repeater Slave supports the following functionality

- Powerlevel test (RECOMMENDED)
- Association (RECOMMENDED)

The supported functionalities are described in the following:

#### **Powerlevel test**

Used to perform a RF link test during installation.

#### **Association**

Can be used to configure a RF link test during installation.

#### 5.13.3.1.1 Mandatory Command Classes to Support

No mandatory command classes to support are defined on the specific device class level.

#### 5.13.3.1.2 Recommended Command Classes to Support

The manufacturer specific and version command class is highly RECOMMENDED because they can be used to identify the device and its capabilities.

#### 5.13.3.1.3 Mandatory Command Classes to Control

Nothing mandatory with respect to command classes the device can control in other devices on this specific device class level.

## 5.14 Semi Interoperable Generic Device Class

The Semi Interoperable Generic Device Class is used to realize devices comprising additional functionality compared to the typically home automation application.

The Semi Interoperable Device Class identifier is equal to `GENERIC_TYPE_SEMI_INTEROPERABLE` and will typically be based on the Basic Device Class identifier `BASIC_TYPE_SLAVE` or `BASIC_TYPE_ROUTING_SLAVE`.

The Semi Interoperable Device supports the following functionalities:

- Device Information (mandatory)
- Proprietary Support (mandatory)

The supported functionalities are described in the following:

### Device Information

The device **MUST** support various command classes to obtain detailed information regarding manufacturer, versions etc. This information is important to transfer the vendor specific parameters correct.

### Proprietary Support

The device **MUST** be able to transfer the vendor specific parameters using a command class without pre-defined contents of the data fields.

#### 5.14.1 Mandatory Command Classes to Support

The following command classes to support are defined on the generic device class level:

- Basic Command Class
- Manufacturer Specific Command Class
- Version Command Class
- Proprietary Command Class

The use of proprietary commands is always subject to approval.

Support and/or control of other command classes **MAY** be made a condition when approving the use of proprietary commands.

#### **5.14.1.1 Basic Command Class Implementation**

Refer to the specific device classes.

#### **5.14.2 Mandatory Command Classes to Control**

Nothing mandatory with respect to command classes the device can control in other devices.

### 5.14.3 Specific Device Classes

The table below shows the current list of defined Specific Device Classes for the Semi Interoperable Generic Device Class:

Specific Device Class	ZW_classcmd.h
Specific Device Class not used	SPECIFIC_TYPE_NOT_USED
Energy Production	SPECIFIC_TYPE_ENERGY_PRODUCTION

**Table 13. Specific Device Class identifiers for the Semi Interoperable Generic Device Class**

Refer to ZW\_classcmd.h source code file for the assigned Specific Device Class identifiers.

### 5.14.3.1 Energy Production Specific Device Class

An Energy Production device supports various equipment for energy production. The device supports the following functionalities:

- Energy Production (mandatory)

The supported functionalities are described in the following:

#### Energy Production

The energy production command class is used to read various production parameters.

##### 5.14.3.1.1 Mandatory Command Classes to Support

The following command classes to support are defined on the specific device class level:

- Energy Production Command Class

##### 5.14.3.1.1.1 Basic Command Class Implementation

The basic command class MUST be implemented in the following way:

Basic Set and Basic Get = The Basic Set and basic Get can freely be mapped to another command class supported by the device. The mapping MUST be documented in the User's Manual. The Basic Set and Get MUST be mapped within the same command class. In case the manufacture deems that no relevant commands are available for mapping the command a device receiving these commands MUST ignore them.

Basic Report = The Basic Report can be mapped to another command class supported by the device. The mapping MUST be documented in the User's Manual. The Basic Report MUST be mapped within the same command class as the Basic Get and MUST be in accordance with the mapped Get. In case the manufacture deems that no relevant commands are available for mapping the Basic Get command a device is not allowed to respond with a Basic Report under any circumstances.

##### 5.14.3.1.2 Mandatory Command Classes to Control

Nothing mandatory with respect to command classes the device can control in other devices.

## 5.15 Static Controller Generic Device Class

The Static Controller Generic Device Class contains much of the same functions as the Remote Control, but has some restrictions and some extra functionality. The main limitation is that the controller is expected to be stationary. Based on the routing table it is therefore able to tell slave nodes how it can be reached.

The Static Controller Device Class identifier is equal to `GENERIC_TYPE_STATIC_CONTROLLER` and will typically be based on the Basic Device Class identifier `BASIC_TYPE_STATIC_CONTROLLER`.

This Device Class can be used to realize Static Controllers. This Device Class is tailored to controllers that are in a fixed position in the network. The Static Controller can serve as a receiver for e.g. sensors or other battery-operated devices that needs to send unsolicited reports to a controller. It can also be used in a system where the controller needs to know the status of each controlled device in the network. A system like this could be a residential gateway, which can be accessed remotely. By being able to learn and store the best route to all nodes in the system it can also significantly reduce the latency in larger systems.

### 5.15.1 Recommended Command Classes to Support

The following command classes can OPTIONAL be supported:

- Basic Command Class

#### 5.15.1.1 Basic Command Class Implementation

The basic command class MUST be implemented in the following way:

Basic Set and Basic Get = The Basic Set and basic Get can freely be mapped to another command class supported by the device. The mapping MUST be documented in the User's Manual. The Basic Set and Get MUST be mapped within the same command class. In case the manufacture deems that no relevant commands are available for mapping the command a device receiving these commands MUST ignore them.

Basic Report = The Basic Report can be mapped to another command class supported by the device. The mapping MUST be documented in the User's Manual. The Basic Report MUST be mapped within the same command class as the Basic Get and MUST be in accordance with the mapped Get. In case the manufacture deems that no relevant commands are available for mapping the Basic Get command a device is not allowed to respond with a Basic Report under any circumstances.

### 5.15.2 Mandatory Command Classes to Control

The following command classes to control with are defined on the generic device class level:

- Basic Command Class

In case a static controller device is configured to control an unknown device it SHALL use basic command to control this device.

### 5.15.3 Specific Device Classes

The table below shows the current list of defined Specific Device Classes for the Static Controller Generic Device Class:

Specific Device Class	ZW_classcmd.h
Specific Device Class not used	SPECIFIC_TYPE_NOT_USED
PC Controller	SPECIFIC_TYPE_PC_CONTROLLER
Scene Controller	SPECIFIC_TYPE_SCENE_CONTROLLER
Static Installer Tool	SPECIFIC_TYPE_STATIC_INSTALLER_TOOL
Gateway	SPECIFIC_TYPE_GATEWAY

**Table 14. Specific Device Class identifiers for the Static Controller Generic Device Class**

Refer to ZW\_classcmd.h source code file for the assigned Specific Device Class identifiers.

### 5.15.3.1 PC Controller Specific Device Class

The PC Controller Specific Device Class can be used to control and monitor other nodes in a Z-Wave network. The PC controller is a controller based on a PC with a graphical user interface. The controller is a static controller.

The PC controller supports the following functionality

- Static Update Controller (RECOMMENDED)
- Controller replication of the application part (RECOMMENDED)
- Clock (RECOMMENDED)

This Device Class is tailored to controllers that have a fixed position and is always powered. It can take advantage of the possibilities that a powerful PC platform offers in terms of processing power and graphics. Only the controller needs to be powered to fulfill the SUC role in the network, so the PC can be turned off when not used.

The supported functionalities are described in the following:

#### Static Update Controller (SUC)

The PC controller MAY OPTIONAL support the Static Update Controller functionality. The controller MUST be enabled to accept an assignment of the SUC role in the Z-Wave network. Finally, if the SUC functionality is supported, the controller MUST accept being assigned as the SUC of a Z-Wave network.

#### Controller replication of the application part

The PC Controller MUST respond with `ZW_ReplicationReceiveComplete` in case it receives the replication command class; otherwise, the replication of protocol data will fail. OPTIONAL: can the PC Controller interpret the data transferred by the replication command class or simply skip the data.

#### Clock

The PC Controller MAY OPTIONAL support the clock functionality.

#### 5.15.3.1.1 Mandatory Command Classes to Support

No mandatory command classes to support are defined on the specific device class level.

#### 5.15.3.1.2 Recommended Command Classes to Support

The manufacturer specific and version command class is highly RECOMMENDED because they can be used to identify the device and its capabilities.

#### 5.15.3.1.3 Mandatory Command Classes to Control

Nothing mandatory with respect to command classes the device can control in other devices on this specific device class level.



### 5.15.3.2 Scene Controller Specific Device Class

The Scene Controller Specific Device Class can be used to configure and control scenes in other nodes in a Z-Wave network.

The Scene controller supports the following functionality

- Association (mandatory - support)
- Scene Activation (mandatory - control)
- Scene Controller Configuration (mandatory - support)
- Manufacturer Specific (mandatory - support)
- Scene Actuator Configuration (OPTIONAL - control)
- Static update controller (RECOMMENDED)
- Controller replication of the application part (RECOMMENDED)

Some of the supported functionalities are described in the following:

#### Associations

Is used to organize nodes in different groups allowing the device to identify the nodes by a group identifier. The groups can also be copied to other devices.

#### Scene Activation

Is used to activate scenes in other devices and indicate the active scene on the scene-controlling device. The Basic command class to ensure backwards compatibility MUST control nodes in a group not supporting scene activation.

#### Scene Controller Configuration

Is used to bind a physical identifier e.g. a push button to a group of associations and the wanted scene settings.

#### Static Update Controller (SUC) or SUC ID Server (SIS)

The Scene controller MAY OPTIONAL support the SUC/SIS functionality. In case this functionality is supported the controller MUST accept being assigned the role as SUC/SIS in the Z-Wave network.

#### Controller replication of the application part

The Scene Controller MUST respond with ZW\_ReplicationReceiveComplete in case it receives the replication command class otherwise will the replication of protocol data fail. OPTIONAL can the Scene Controller interpret the data transferred by the replication command class or simply skip the data.

#### Combo devices

A Scene Controller could OPTIONAL be extended with dimmer functionality. The combo device MUST then be able to differ between which functionalities (load or LED indicator) it SHOULD react on when receiving a Scene Activation Set command.

The combo device does this by checking its own configuration to interpret how the received command SHOULD be carried out. When a combo device receives Scene Activation Set command then it SHOULD check if any of the controller's buttons has the received scene ID associated with it. If NO, the command is clearly send for the dimmer/switch and SHOULD be implemented. If YES, the device SHOULD check the association list for the controller's button bind to the received scene. If the list has device's own ID in it then the scene command send for both identities of the device: dimmer and controller and so SHOULD be implemented. If the own ID is not in the association list for button then this scene ID was send just for display and dimmer/switch SHOULD NOT respond to it.

#### 5.15.3.2.1 Mandatory Command Classes to Support

The following command classes to support are defined on the specific device class level:

- Association Command Class
- Scene Controller Configuration Command Class
- Basic Command Class
- Manufacturer Specific Command Class

#### 5.15.3.2.2 Recommended Command Classes to Support

The version command class can be used to retrieve the command class revisions etc. supported by the device.

The scene activation command class can be used in case the scene controller which in addition to being scene controller also control a load locally.

#### 5.15.3.2.3 Mandatory Command Classes to Control

The following command classes to control with are defined on the specific device class level:

- Scene Activation Command Class

#### 5.15.3.2.4 Recommended Command Classes to Control

The following command classes can OPTIONAL be supported to allow scene controllers exchange configurations:

- Association Command Class
- Scene Controller Configuration Command Class
- Scene Actuator Configuration Command Class
- All Switch Command Class

### 5.15.3.3 Static Installer Tool Specific Device Class

The Static Installer Tool Specific Device Class is envisioned to be used to set-up and configure a Z-Wave network and for configuration of the various nodes in a network. The controller is a static controller but is not envisioned to be a part of the network after the initial installation and configuration is performed. For this reason it is not RECOMMENDED to enable SUC /SIS functionality in the controller, but the controller MUST be able to appoint the SUC/SIS role to another node in the network.

#### 5.15.3.3.1 Mandatory Command Classes to Support

The following command classes to support are defined on the specific device class level:

- Controller Replication Command Class
- Multi Command Command Class
- Version Command Class
- Manufacturer Specific Command Class

#### 5.15.3.3.2 Recommended Command Classes to Support

The following command classes are RECOMMENDED to be supported:

- Clock Command Class

#### 5.15.3.3.3 Mandatory Command Classes to Control

The following command classes to control with are defined on the specific device class level:

- Multi Channel Command Class - Version 2
- Controller Replication Command Class
- Association Command Class
- Multi Channel Association Command Class - Version 2
- Wake Up Command Class
- Configuration Command Class
- Manufacturer Specific Command Class
- Version Command Class

#### 5.15.3.3.4 Recommended Command Classes to Control

A number of command classes are RECOMMENDED in order to retrieve information and configure more advanced devices:

- Binary Switch Command Class
- Multilevel Switch Command Class
- Battery Command Class
- Scene Controller Configuration Command Class
- Scene Actuator Configuration Command Class
- All Switch Command Class
- Node Naming and Location Command Class
- Grouping Name Command Class
- Protection Command Class
- Powerlevel Command Class

### 5.15.3.4 Gateway Specific Device Class

The Gateway Specific Device Class supports integration of classic Z-Wave nodes to other technology platforms.

#### 5.15.3.4.1 Multi-Channel Gateway Specific Device Class

The Multi-Channel Gateway Specific Device Class supports control of multi-channel and/or security enabled Z-Wave nodes.

##### 5.15.3.4.1.1 *Mandatory Command Classes to Support*

In addition to the inherited command classes from the generic device level the following **MUST** also be supported:

- Version Command Class
- Manufacturer Specific Command Class
- Security Command Class

##### 5.15.3.4.1.2 *Recommended Command Classes to Support*

RECOMMENDED that the device support the following command classes:

- Clock Command Class

##### 5.15.3.4.1.3 *Mandatory Command Classes to Control*

The following command classes to control with are defined on the specific device class level:

- Security Command Class
- Multi Channel Command Class version 3 or later

#### 5.15.3.4.2 Z/IP Gateway Specific Device Class

The Z/IP Gateway Specific Device Class supports IP integration of classic Z-Wave nodes through the Gateway Framework.

- **IP application control of Z-Wave devices**  
An IP application **MAY** send Z-Wave commands in Z/IP encapsulated UDP packets
- **Forwarding of Z-Wave commands to IP applications**  
The gateway emulates a Z-Wave device, so a Z-Wave device **MAY** send Z-Wave commands to an IP application by sending Z-Wave commands to the gateway.
- **Respond to ICMP requests on behalf of Z-Wave nodes**

Support of Z/IP Command Class indicates use of Z/IP extension to Gateway Specific Device Class

##### 5.15.3.4.2.1 *Mandatory Command Classes to support*

In addition to the inherited command classes from the generic device the following **MUST** also be supported:

- Version Command Class
- Manufacturer Specific Command Class
- Z/IP Command Class
- Z/IP-ND Command Class
- Network Management Proxy Command Class
- Network Management Basic Node Command Class
- Network Management Inclusion Command Class
- Network Management Primary Command Class

- Security Command Class
- Transport Service Command Class

#### *5.15.3.4.2.2 Recommended Command Classes to support*

RECOMMENDED that the device support the following command classes:

- Clock Command Class

#### *5.15.3.4.2.3 Mandatory Command Classes to control*

The following command classes to control with are defined on the specific device class level:

- Z/IP Command Class
- Z/IP-ND Command Class
- Security Command Class
- Transport Service Command Class
- Application Capability Command Class
- Multi Channel Command Class version 3 or later

#### *5.15.3.4.2.4 Recommended Command Classes to control*

Nothing RECOMMENDED with respect to command classes the device can control in other devices.

## 5.16 Thermostat Generic Device Class

A Thermostat device that enables the user to set a comfort setpoint temperature is maintained.

The Thermostat Device Class identifier is equal to `GENERIC_TYPE_THERMOSTAT` and will typically be based on the Basic Device Class identifier `BASIC_TYPE_STATIC_CONTROLLER` or `BASIC_TYPE_ROUTING_SLAVE`.

### 5.16.1 Mandatory Command Classes to Support

The basic command class **MUST** be implemented for all permanently listening devices. Battery-operated devices **MAY** implement the basic command class.

#### 5.16.1.1 Basic Command Class Implementation

Refer to the specific devices for a description

### 5.16.2 Mandatory Command Classes to Control

Nothing mandatory with respect to command classes the device can control in other devices.

### 5.16.3 Specific Device Classes

The table below shows the current list of defined Specific Device Classes for the Thermostat Generic Device Class:

Specific Device Class	ZW_classcmd.h
Specific Device Class not used	<code>SPECIFIC_TYPE_NOT_USED</code>
Thermostat Heating	<code>SPECIFIC_TYPE_THERMOSTAT_HEATING</code>
Thermostat General	<code>SPECIFIC_TYPE_THERMOSTAT_GENERAL</code>
Thermostat General V2	<code>SPECIFIC_TYPE_THERMOSTAT_GENERAL_V2</code>
Setback Schedule Thermostat	<code>SPECIFIC_TYPE_SETBACK_SCHEDULE_THERMOSTAT</code>
Setback Thermostat	<code>SPECIFIC_TYPE_SETBACK_THERMOSTAT</code>
Setpoint Thermostat	<code>SPECIFIC_TYPE_SETPOINT_THERMOSTAT</code>

**Table 15. Specific Device Class identifiers for the Thermostat Generic Device Class**

Refer to `ZW_classcmd.h` source code file for the assigned Specific Device Class identifiers.

### 5.16.3.1 Thermostat Heating Specific Device Class (Not permitted)

**It is not permitted to use this specific device class for new devices.**

Instead it is RECOMMENDED to base devices of this category on the General Thermostat V2 specific device class or other specific devices classes defined.

The Thermostat Heating device is defined in [5].

#### 5.16.3.1.1 Basic Command Class Implementation

The basic command class MUST be implemented in the following way:

- Basic Set and Basic Get = The Basic Set and basic Get can freely be mapped to another command class supported by the device. The mapping MUST be documented in the User's Manual. The Basic Set and Get MUST be mapped within the same command class. In case the manufacture deems that no relevant commands are available for mapping the command a device receiving these commands MUST ignore them.
- Basic Report = The Basic Report can be mapped to another command class supported by the device. The mapping MUST be documented in the User's Manual. The Basic Report MUST be mapped within the same command class as the Basic Get and MUST be in accordance with the mapped Get. In case the manufacture deems that no relevant commands are available for mapping the Basic Get command a device is not allowed to respond with a Basic Report under any circumstances.

### 5.16.3.2 Thermostat General Specific Device Class (Not permitted)

**It is not permitted to use this specific device class for new devices.**

Instead it is RECOMMENDED to base devices of this category on the General Thermostat V2 specific device class or other specific devices classes defined.

#### 5.16.3.2.1 Mandatory Command Classes to Support

The following command classes to support are defined on the specific device class level:

- Manufacturer Specific Command Class
- Thermostat Mode Command Class
- Thermostat Setpoint Command Class

##### 5.16.3.2.1.1 Basic Command Class Implementation

The basic command class MUST be implemented in the following way:

Basic Set = Not supported to avoid potential liability issues where a user inadvertently turned his heating off. A device receiving this command MUST simply ignore it.

Basic Get = Thermostat Mode Get

Basic Report = Thermostat Mode Report

#### 5.16.3.2.2 Recommended Command Classes to Support

The version command class is highly RECOMMENDED because they can be used to retrieve versions of the command classes etc. supported by the device.

The multilevel sensor command class can be used to read the current temperature. Remember also to consider the relevance of the thermostat operating state, thermostat fan mode and thermostat fan state command classes when designing a thermostat. Finally the multi instance command class can be used in case a secondary temperature sensor (outside or water temperature) is present.

#### 5.16.3.2.3 Mandatory Command Classes to Control

Nothing mandatory with respect to command classes the device can control in other devices.



### 5.16.3.3 Thermostat General V2 Specific Device Class

A Thermostat General device supports a general thermostat.

#### 5.16.3.3.1 Mandatory Command Classes to Support

The following command classes to support are defined on the specific device class level:

- Basic Command Class
- Manufacturer Specific Command Class
- Version Command Class
- Thermostat Mode Command Class
- Thermostat Setpoint Command Class

##### 5.16.3.3.1.1 Basic Command Class Implementation

The basic command class SHALL be implemented in the following way:

Basic Set (Value = 0x00) = Set Energy Saving Mode

Basic Set (Value = 0xFF) = Set Comfort Mode

Basic Get = Get report

Basic Report (Value = 0x00) = Report Energy Saving Mode

Basic Report (Value = 0xFF) = Report Comfort Mode

Note: The implementation of Energy Saving Mode is manufacturer specific, and MUST be documented in the User's Manual.

#### 5.16.3.3.2 Recommended Command Classes to Support

The multilevel sensor command class can be used to read the current temperature. Remember also to consider the relevance of the thermostat operating state, thermostat fan mode and thermostat fan state command classes when designing a thermostat. Finally the multi instance command class can be used in case a secondary temperature sensor (outside or water temperature) is present.

#### 5.16.3.3.3 Mandatory Command Classes to Control

Nothing mandatory with respect to command classes the device can control in other devices.

#### 5.16.3.4 Setback Schedule Thermostat Specific Device Class

Setback Schedule Thermostats control the temperature in one or multiple rooms. The thermostat has a setpoint, which is used for normal operation. But to save additional energy the thermostat has the ability to go into energy saving mode controlled by a schedule.

The schedule is downloaded to the thermostat from the controller once the system is set up and every time the user changes the schedule in the controller. The thermostat has a schedule for one week. After one week the sequence in the schedule is repeated.

The schedule can be overridden if the homeowner returns to his home earlier, is away on vacation for longer time or is taking a day off at home.

Note: A device MUST both support and control the Multi Command Command Class including relevant encapsulated command classes to be able to operate a Setback Schedule Thermostat Specific Device.

##### 5.16.3.4.1 Mandatory Command Classes to Support

The following command classes to support are defined on the specific device class level:

- Climate Control Schedule Command Class
- Manufacturer Specific Command Class
- Multi Command Command Class
- Version Command Class

A battery-operated Setback Schedule Thermostat device MUST additionally implement as supported the following command classes:

- Battery Command Class
- Wake Up Command Class Version 2

The Setback Schedule Thermostat SHALL understand supported commands that are encapsulated by a Multi Command.

Note: In order to prolong the battery-life of battery-operated devices it is RECOMMENDED to always send as many commands as possible encapsulated in a Multi Command.

#### 5.16.3.4.1.1 *Basic Command Class Implementation*

The basic command class SHALL be implemented in the following way:

Basic Set (Value = 0x00) =	Schedule Override Set (Temporary Override, Energy Saving Mode)
Basic Set (Value = 0xFF) =	Schedule Override Set (Temporary Override, 0 degree setback)
Basic Get =	Schedule Override Get
Basic Report (Value = 0x00) =	Schedule Override Report (Temporary Override, Energy Saving Mode)
Basic Report (Value = 0xFF) =	Schedule Override Report (Temporary Override, 0 degree setback) including the remaining options

Note: The implementation of Energy Saving Mode is manufacturer specific.

#### 5.16.3.4.2 Recommended Command Classes to Support

The battery command class is RECOMMENDED in case the Setback Schedule Thermostat device is battery-operated.

#### 5.16.3.4.3 Mandatory Command Classes to Control

The following command classes to control with are defined on the specific device class level:

- Climate Control Schedule Command Class
- Multi Command Command Class
- Clock Command Class

Note: In order to prolong the battery-life of battery-operated devices it is RECOMMENDED to always send as many commands as possible encapsulated in a Multi Command.

#### 5.16.3.4.4 Mandatory Operation

**After Addition:**

A Setback Schedule Thermostat has to know the Node ID of the device which contains the schedules.

A Setback Schedule Thermostat SHALL perform the following steps after addition:

1. Wait for a WAKE\_UP\_INTERVAL\_SET
2. Wait for a WAKE\_UP\_NO\_MORE\_INFORMATION

The Setback Schedule Thermostat MUST be able to accept other supported commands while performing these steps.

The Node which has the schedule ('The Controller') SHALL perform the following steps after it has been enabled to accept new Setback Schedule Thermostats.

1. Wait for a Node Information Frame
2. Send the relevant WAKE\_UP\_INTERVAL\_SET

The Controller MUST be able to accept other supported commands while performing these steps.

Battery-operated Setback Schedule Thermostats devices MAY implement a timeout to allow the device to enter a special mode which preserves battery-life in the case that the wake up functionality is not configured immediately after addition. It MUST require a physical activation of the device to exit the special mode.

Note: The implementation of exiting the special mode is manufacturer specific.

When exiting the special mode the following steps MUST be performed:

1. Send a Node Information Frame
2. Perform the steps from the previous list for the Setback Schedule Thermostat

If timeout occurs again, the device MAY re-enter the special mode.

**Wake up:**

A battery-operated Setback Schedule Thermostat SHALL perform the following steps on wake up:

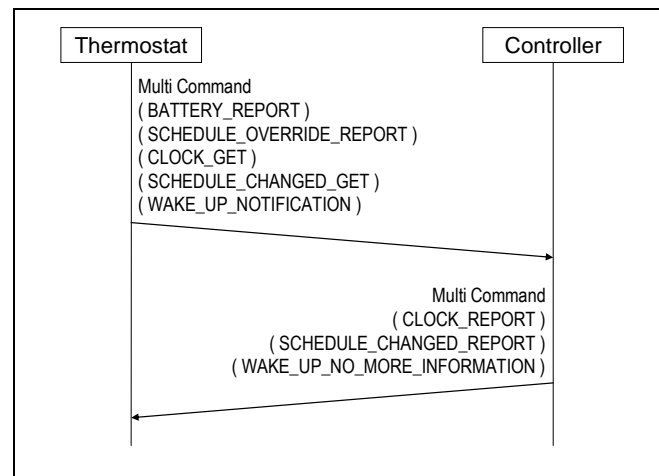
1. Send a BATTERY\_REPORT
2. Send a SCHEDULE\_OVERRIDE\_REPORT
3. Send a CLOCK\_GET
4. Send a SCHEDULE\_CHANGED\_GET
5. Send a WAKE\_UP\_NOTIFICATION

Step 1 - 3 has special considerations:

- Step 1 MUST be performed on first wake up and MAY be left out if no new battery measurements were made.
- Step 2 MUST be performed if the override state of the device has changed and otherwise it MAY be left out.
- Step 3 MUST be performed on first wake up and MAY be left out if an update of time is not needed. It is RECOMMENDED to synchronize time depending on the needed accuracy.

Note: In order to prolong the battery-life of battery-operated devices it is RECOMMENDED to always send as many commands as possible encapsulated in a Multi Command. At wake up is it allowed to send commands encapsulated in a Multi Command without checking that the destination supports relevant command classes.

The operation is visualized using Multi Commands in the following sequence diagram:



**Figure 7. Sequence diagram of operation after wake up**

**Note:** If either device, when using Multi Commands, has the need to send more commands they **SHOULD** try to integrate them into the two Multi Commands exchanged after a wake up as visualized in the above diagram.

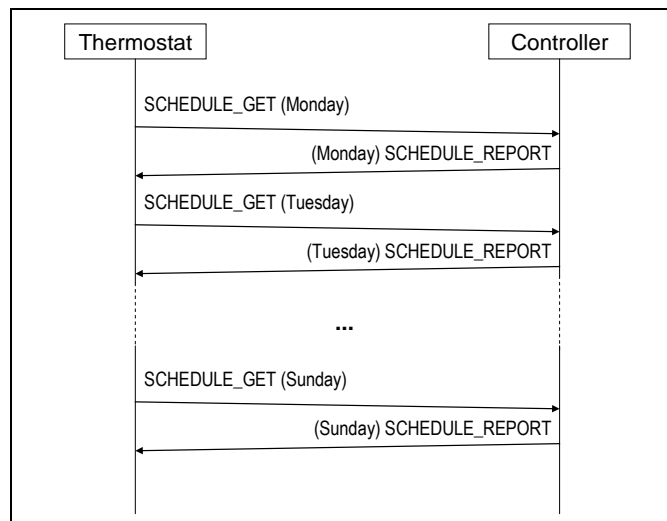
**Example:** A user has pressed a button on the thermostat which makes it wake up. The thermostat then performs the wake up operation but also adds an unsolicited report informing of the user interaction in a Multi Command.

***Schedule is not up-to-date:***

A battery-operated Setback Schedule Thermostat SHALL perform the following steps when its schedule is not up-to-date. **These steps SHOULD NOT be performed using Multi Commands as there can only be one schedule in the Multi Command report:**

1. Send a SCHEDULE\_GET (Monday)
2. Wait for the SCHEDULE\_REPORT (Monday)
3. Send a SCHEDULE\_GET (Tuesday)
4. Wait for the SCHEDULE\_REPORT (Tuesday)
5. Send a SCHEDULE\_GET (Wednesday)
6. Wait for the SCHEDULE\_REPORT (Wednesday)
7. Send a SCHEDULE\_GET (Thursday)
8. Wait for the SCHEDULE\_REPORT (Thursday)
9. Send a SCHEDULE\_GET (Friday)
10. Wait for the SCHEDULE\_REPORT (Friday)
11. Send a SCHEDULE\_GET (Saturday)
12. Wait for the SCHEDULE\_REPORT (Saturday)
13. Send a SCHEDULE\_GET (Sunday)
14. Wait for the SCHEDULE\_REPORT (Sunday)

The operation is visualized in the following sequence diagram:



**Figure 8. Sequence diagram of operation for getting schedules**



### 5.16.3.5 Setback Thermostat Specific Device Class

The Setback Thermostat device controls the temperature in a room. The thermostat has a setpoint, which is used for normal operation. To save energy it supports the ability to go into setback mode.

The Setback Thermostat device does not support a schedule; the user or another device **MUST** initiate a state change.

#### 5.16.3.5.1 Mandatory Command Classes to Support

The following command classes to support are defined on the specific device class level:

- Thermostat Setpoint Command Class
- Thermostat Setback Command Class
- Thermostat Mode Command Class
- Manufacturer Specific Command Class
- Version Command Class

A battery-operated Setback Thermostat device **MUST** additionally implement as supported the following command classes:

- Battery Command Class
- Wake Up Command Class Version 2
- Multi Command Command Class

**Note:** In order to prolong the battery-life of battery-operated devices it is **RECOMMENDED** to always send as many commands as possible encapsulated in a Multi Command.

##### 5.16.3.5.1.1 Basic Command Class Implementation

The basic command class **SHALL** be implemented in the following way:

Basic Set (Value = 0x00) =	Set Energy Saving Mode
Basic Set (Value = 0xFF) =	Set Comfort Mode
Basic Get =	Get report
Basic Report (Value = 0x00) =	Report Energy Saving Mode
Basic Report (Value = 0xFF) =	Report Comfort Mode

**Note:** The implementation of Energy Saving Mode is manufacturer specific, and **MUST** be documented in the User's Manual. Performing a set using basic commands is always a temporary override; refer to description under Thermostat Setback Command Class.

#### 5.16.3.5.2 Recommended Command Classes to Support

The following command classes are RECOMMENDED to be supported by the Setback Thermostat:

- Multilevel Sensor (Temperature)
- Thermostat Operation State
- Thermostat Fan
- Thermostat Fan State
- Indicator
- Protection
- Application State
- Association

#### 5.16.3.5.3 Mandatory Command Classes to Control

Nothing mandatory with respect to command classes the device can control in other devices.

### 5.16.3.6 Setpoint Thermostat Specific Device Class

Setpoint Thermostats control the temperature in one or multiple rooms. It uses a setpoint which defines the desired temperature.

#### 5.16.3.6.1 Mandatory Command Classes to Support

The Setpoint Thermostat device SHALL support the following command classes:

- Manufacturer Specific Command Class
- Multi Command Command Class
- Thermostat Setpoint
- Version Command Class

A battery-operated Setpoint Thermostat device SHALL additionally support the following command classes:

- Battery Command Class
- Wake Up Command Class, Version 2

The Setpoint Thermostat SHALL understand supported commands that are encapsulated by a Multi Command.

Note: In order to prolong the battery-life of battery-operated devices it is RECOMMENDED to always send as many commands as possible encapsulated in a Multi Command.

Note: The Wake Up Command Class Version 2 allows a controller to learn what wake up intervals a battery-operated device supports.

#### 5.16.3.6.1.1 Basic Command Class Implementation

The basic command class SHALL be implemented for all permanently listening devices.

The basic command class SHALL be implemented in the following way:

Basic Set	=	Thermostat Setpoint Set
Basic Get	=	Thermostat Setpoint Get
Basic Report	=	Thermostat Setpoint Report

#### 5.16.3.6.2 Recommended Command Classes to Support

It is RECOMMENDED for a Setpoint Thermostat device to support the following command classes:

- Association
- Multilevel Sensor

#### 5.16.3.6.3 Mandatory Command Classes to Control

The Setpoint Thermostat SHALL control the following command classes:

- Multi Command Command Class
- Thermostat Setpoint Command Class

Note: In order to prolong the battery-life of battery-operated devices it is RECOMMENDED to always send as many commands as possible encapsulated in a Multi Command.

#### 5.16.3.6.4 Mandatory Operation

##### ***After Addition:***

A Setpoint Thermostat has to know the Node ID of the device, which contains the schedules.

A Setpoint Thermostat SHALL perform the following steps after addition:

3. Wait for a WAKE\_UP\_INTERVAL\_SET
4. Wait for a WAKE\_UP\_NO\_MORE\_INFORMATION

The Setpoint Thermostat MUST be able to accept other supported commands while performing these steps.

Battery-operated devices SHALL implement a timeout to allow the device to enter a special mode, which preserves battery-life. It MUST be possible to exit the special mode.

Note: The implementation of exiting the special mode is manufacturer specific.

When exiting the special mode the following steps MUST be performed:

3. Send a Node Information Frame
4. Perform the steps from the previous list

If timeout occurs again, the device MUST enter the special mode.

**Wake up:**

A battery-operated Setpoint Thermostat SHALL perform the following steps on wake up:

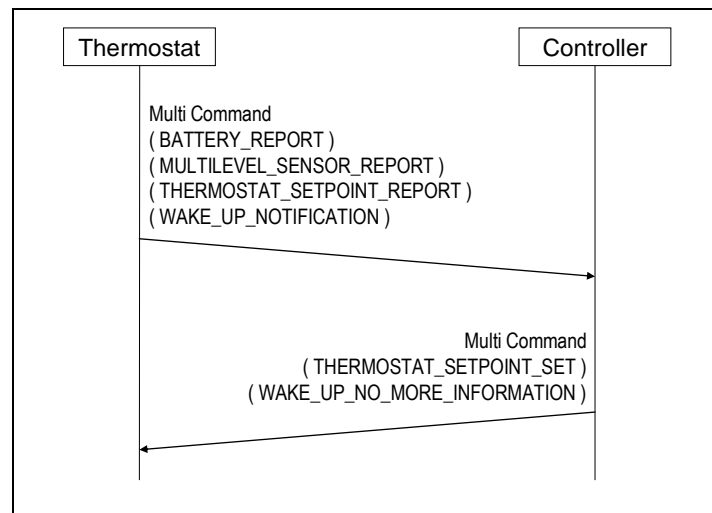
6. Send a BATTERY\_REPORT
7. Send a MULTI\_LEVEL\_SENSOR\_REPORT (if implemented)
8. Send a THERMOSTAT\_SETPOINT\_REPORT
9. Send a WAKE\_UP\_NOTIFICATION

Step 1 has special considerations:

- Step 1 MUST be performed on first wake up and MAY be left out if no new battery measurements were made.

Note: In order to prolong the battery-life of battery-operated devices it is RECOMMENDED to always send as many commands as possible encapsulated in a Multi Command.

The operation is visualized using Multi Commands in the following sequence diagram:



**Figure 9. Sequence diagram of operation after wake up**

Note: If either device, when using Multi Commands, has the need to send more commands they SHOULD try to integrate them into the two Multi Commands exchanged after a wake up as visualized in the above diagram.

Example: A user has pressed a button on the thermostat which makes it wake up. The thermostat then performs the wake up operation but also adds an unsolicited report informing of the user interaction to the Multi Command the thermostat sends to the controller.

## 5.17 Toggle Switch Generic Device Class (Not recommended)

**Not RECOMMENDED to use this generic device class in new devices.**

**Instead, it is RECOMMENDED to base devices of this category on the Binary / Multilevel Switch generic device class.**

A Toggle Switch device class supports all one-button devices that need to toggle between two states e.g. turn power on/off. Further the device can be extended with dimming capabilities.

The Toggle Switch Device Class identifier is equal to `GENERIC_TYPE_SWITCH_TOGGLE` and will typically be based on the Basic Device Class identifier `BASIC_TYPE_STATIC_CONTROLLER`, `BASIC_TYPE_SLAVE` or `BASIC_TYPE_ROUTING_SLAVE`.

### 5.17.1 Mandatory Command Classes to Support

The following command classes to support are defined on the generic device class level:

- Basic Command Class

#### 5.17.1.1 Basic Command Class Implementation

Refer to the specific device classes.

### 5.17.2 Mandatory Command Classes to Control

Nothing mandatory with respect to command classes the device can control in other devices on this generic device class level.

### 5.17.3 Specific Device Classes

The table below shows the current list of defined Specific Device Classes for the Toggle Switch Generic Device Class:

Specific Device Class	ZW_classcmd.h
Specific Device Class not used	SPECIFIC_TYPE_NOT_USED
Binary Toggle Switch	SPECIFIC_TYPE_SWITCH_TOGGLE_BINARY
Multilevel Toggle Switch	SPECIFIC_TYPE_SWITCH_TOGGLE_MULTILEVEL

**Table 16. Specific Device Class identifiers for the Toggle Switch Generic Device Class**

Refer to ZW\_classcmd.h source code file for the assigned Specific Device Class identifiers.

### **5.17.3.1 Binary Toggle Switch Specific Device Class (Not recommended)**

A Binary Toggle Switch device supports all one-button devices that need to toggle between two states e.g. turn power on/off.

#### **5.17.3.1.1 Mandatory Command Classes to Support**

The following command classes to support are defined on the specific device class level:

- Binary Switch Command Class
- Binary Toggle Switch Command Class

##### ***5.17.3.1.1.1 Basic Command Class Implementation***

The basic command class MUST be implemented in the following way:

Basic Set = Binary Toggle Switch Set

Basic Get = Binary Toggle Switch Get

Basic Report = Binary Toggle Switch Report

#### **5.17.3.1.2 Recommended Command Classes to Support**

The manufacturer specific and version command class is highly RECOMMENDED because they can be used to identify the device and its capabilities.

#### **5.17.3.1.3 Mandatory Command Classes to Control**

Nothing mandatory with respect to command classes the device can control in other devices on this specific device class level.

### **5.17.3.2 Multilevel Toggle Switch Specific Device Class (Not recommended)**

A Multilevel Toggle Switch device supports all one-button devices that both need to toggle between two states and have dimmable capabilities.

#### **5.17.3.2.1 Mandatory Command Classes to Support**

The following command classes to support are defined on the specific device class level:

- Multilevel Switch Command Class
- Multilevel Toggle Switch Command Class

##### ***5.17.3.2.1.1 Basic Command Class Implementation***



The basic command class MUST be implemented in the following way:

Basic Set = Multilevel Toggle Switch Set

Basic Get = Multilevel Toggle Switch Get

Basic Report = Multilevel Toggle Switch Report

#### **5.17.3.2.2 Recommended Command Classes to Support**

The manufacturer specific and version command class is highly RECOMMENDED because they can be used to identify the device and its capabilities.

#### **5.17.3.2.3 Mandatory Command Classes to Control**

Nothing mandatory with respect to command classes the device can control in other devices on this specific device class level.

## 5.18 Ventilation Generic Device Class

The Ventilation Generic Device Class SHOULD be used for Z-Wave Enabled ventilation systems; it will provide specific device classes to further differentiate between types of ventilation systems.

The ventilation generic device class is equal to GENERIC\_TYPE\_VENTILATION.

### 5.18.1 Mandatory Command Classes to Support

The following command classes to support are defined on the generic device class level:

- Basic Command Class

### 5.18.2 Mandatory Command Classes to Control

Nothing mandatory with respect to command classes the device can control in other devices.

### 5.18.3 Specific Device Classes

The table below shows the current list of defined specific device classes for the Ventilation Generic Device Class:

**Table 5 - Specific Device Class identifiers for the Ventilation Generic Device Class.**

Specific Device Class	ZW_classcmd.h
Specific Device Class not used	SPECIFIC_TYPE_NOT_USED
Residential Heat Recovery Ventilation	SPECIFIC_TYPE_RESIDENTIAL_HRV

### 5.18.3.1 No Specific Device Class defined

This option is selected in case the generic level is appropriate for defining the device.

#### 5.18.3.1.1 Mandatory Command Classes to Support

No mandatory command classes to support are defined on the specific device class level.

##### 5.18.3.1.1.1 Basic Command Class implementation

The basic command class MUST be implemented in the following way:

- |                         |   |
|-------------------------|---|
| Basic Set and Basic Get | = The Basic Set and basic Get can freely be mapped to another command class supported by the device. The mapping MUST be documented in the User's Manual. The Basic Set and Get MUST be mapped within the same command class. In case the manufacture deems that no relevant commands are available for mapping the command a device receiving these commands MUST ignore them.   |
| Basic Report            | = The Basic Report can be mapped to another command class supported by the device. The mapping MUST be documented in the User's Manual. The Basic Report MUST be mapped within the same command class as the Basic Get and MUST be in accordance with the mapped Get. In case the manufacture deems that no relevant commands are available for mapping the Basic Get command a device is not allowed to respond with a Basic Report under any circumstances. |

### 5.18.3.2 Residential Heat Recovery Ventilation Specific Device Class

The Residential Heat Recovery Ventilation Specific Device Class SHOULD be used for residential ventilation systems that include a heat exchanger.

The device class will mandate the command classes needed to control the ventilation system, and to read out parameters related to the ventilation system.

#### 5.18.3.2.1 Mandatory Command Classes to Support

The following command classes are mandatory to be supported by a residential heat recovery ventilation system.

- HRV Status Command Class, Version 1
  - The HRV Status Command Class allows the ventilation system to report system parameters.
- HRV Control Command Class, Version 1
  - The HRV Control Command Class Version 1 introduces commands for controlling ventilation systems.
- Version Command Class
- Manufacturer Specific Command Class

##### 5.18.3.2.1.1 Basic Command Class implementation

The basic command class MUST be implemented in the following way:

Basic Set (0x00)	=	Activate energy savings mode
Basic Set (0xFF)	=	Deactivate energy savings mode
Basic Get	=	Get report
Basic Report (0x00)	=	Energy savings mode activated
Basic Report (0xFF)	=	Energy savings mode deactivated

Note: The implementation of the energy saving mode and demand / automatic mode corresponds to the implementations in the HRV Mode Command Class.

#### 5.18.3.2.2 Mandatory Command Classes to Control

Nothing mandatory with respect to command classes the device can control in other devices.

## 5.19 Window Covering Generic Device Class (Not recommended)

**It is not RECOMMENDED to use this generic device class for new devices.**

**Instead, it is RECOMMENDED to base devices of this category on the Multi-level switch generic device class with the Multi-position Motor specific device class.**

The Window Covering device is used to control motorized drapes, shades and blinds etc.

Window Covering Device Class identifier is equal to `GENERIC_TYPE_WINDOW_COVERING` and will typically be based on the Basic Device Class identifier `BASIC_TYPE_STATIC_CONTROLLER` or `BASIC_TYPE_ROUTING_SLAVE`.

### 5.19.1 Mandatory Command Classes to Support

No mandatory command classes to support are defined on the generic device class level.

#### 5.19.1.1 Basic Command Class Implementation

Refer to the specific devices for a description

### 5.19.2 Mandatory Command Classes to Control

Nothing mandatory with respect to command classes the device can control in other devices.

### 5.19.3 Specific Device Classes

The table below shows the current list of defined Specific Device Classes for the Window Covering Generic Device Class:

Specific Device Class	ZW_classcmd.h
Specific Device Class not used	<code>SPECIFIC_TYPE_NOT_USED</code>
Simple Window Covering Control	<code>SPECIFIC_TYPE_SIMPLE_WINDOW_COVERING</code>

**Table 17. Specific Device Class identifiers for the Window Covering Generic Device Class**

Refer to `ZW_classcmd.h` source code file for the assigned Specific Device Class identifiers.

### 5.19.3.1 Simple Window Covering Control Specific Device Class (Not recommended)

The Simple Window Covering Control device can be used to control motorized drapes, shades, blinds etc. The simple control comprises of commands to start open/close and stop window coverings.

#### 5.19.3.1.1 Mandatory Command Classes to Support

The following command classes to support are defined on the specific device class level:

- Basic Window Covering

##### 5.19.3.1.1.1 Basic Command Class Implementation

The basic command class MUST be implemented in the following way:

Basic Set (Value = 0x00) = Basic Window Covering Start Level Change (Open).

Basic Set (Value = 0xFF) = Basic Window Covering Start Level Change (Close).

Basic Get and Report = Cannot be mapped because the mapped command class doesn't have a Get command. A device receiving this command MUST simply ignore it.

Basic Report = Cannot be mapped because the mapped command class doesn't have a Report command. A device is not allowed to respond with a Basic Report under any circumstances.

If the shade is stopped, a Basic Set command will start motion in the specified direction. If the shade is moving, a Basic Set command in the same direction will be ignored. If the shade is moving, a Basic Set command in the opposite direction will stop motion.

#### 5.19.3.1.2 Recommended Command Classes to Support

The manufacturer specific and version command class is highly RECOMMENDED because they can be used to identify the device and its capabilities. The multi instance command class can be used for window coverings using two motion controls. To access various settings use the configuration command class.

#### 5.19.3.1.3 Mandatory Command Classes to Control

Nothing mandatory with respect to command classes the device can control in other devices.

## 6 COMMAND CLASSES

For a detailed description of the command classes and associated commands refer to [8].

## 7 DEVICE EXAMPLES

The following sections describe application examples using the Z-Wave device and command classes.

### 7.1 Lighting Control Applications

#### 7.1.1 Outlet adapter with dimming capability

This example deals with an outlet adapter with dimming capability as shown on the figure.



**Figure 10. Outlet adapter with dimming capabilities**

A slave library is selected as the basic device class because initiating transmission to other devices is not necessary. The listening flag is set because the slave is mains powered allowing it to respond immediately on requests from other devices in the Z-Wave network. To allow the device to turn power on/off and dim it MUST comply with the Multilevel Power Switch Device Class that are a Specific Device Class created on the Generic Multilevel Switch Device Class. The selected device class supports the mandatory command classes Multilevel Switch and All Switch. The Version command class is also mandatory because Version 2 of the Multilevel Switch command class is used to allow dimming commands specifying the dimming duration/rate. No RECOMMENDED command classes are selected.



### 7.1.1.1 Node Information

The controller will assign a Node ID to the device during the registration phase. When the node is being told to register it will issue a Node Information frame. The registration is part of the protocol and will be handled automatically. The Node Information frame that is sent will have the layout shown below.

7	6	5	4	3	2	1	0
Listening = YES	Protocol Specific Part						
Protocol Specific Part							
Protocol Specific Part							
Basic Device Class = BASIC_TYPE_SLAVE (Protocol Specific Part)							
Multilevel Switch Device Class = GENERIC_TYPE_SWITCH_MULTILEVEL							
Multilevel Power Switch Device Class = SPECIFIC_TYPE_POWER_SWITCH_MULTILEVEL							
Multilevel Switch Command Class = COMMAND_CLASS_SWITCH_MULTILEVEL							
All Switch Command Class = COMMAND_CLASS_SWITCH_ALL							
Version Command Class = COMMAND_CLASS_VERSION							

### 7.1.1.2 Switch Functionality

When the device is registered to the network, the controller can begin to control the switch. The controller will use the Multilevel Switch commands and the All Switch commands to control the device. During the registration to the network the node will be assigned a Node ID. The controller uses this Node ID when it wants to address the node.

#### 7.1.1.2.1 Switching On and Off

The following describes how the controller can turn the switch on and off. The controller uses the Multilevel Switch commands.

#### Switching on

The controller can switch the device on by using the Multilevel Switch Set command with the following layout.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_MULTILEVEL							
Command = SWITCH_MULTILEVEL_SET							
Value = SWITCHED_ON							

### Switching off

The controller can switch the device off by using the Multilevel Switch Set command with the following layout.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_MULTILEVEL							
Command = SWITCH_MULTILEVEL_SET							
Value = SWITCHED_OFF							

### Checking on and off

If the controller requests the switch whether it is turned on or off, it can use the Multilevel Switch Get command with the following layout.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_MULTILEVEL							
Command = SWITCH_MULTILEVEL_GET							

The switch device will then answer with a Multilevel Switch Report command with the following layout (assuming the device is switched off).

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_MULTILEVEL							
Command = SWITCH_MULTILEVEL_REPORT							
Value = SWITCHED_OFF							

#### 7.1.1.2.2 Dimming

The following describes how the controller can set a specific dim level or dim up and down. The controller uses the Multilevel Switch commands.

##### Setting dim level

The controller can set a specific dim level on the device by using the Multilevel Switch Set command with the following layout.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_MULTILEVEL							
Command = SWITCH_MULTILEVEL_SET							
Value (0x24)							
Dimming Duration (0x02)							

In the example above the dim level is set to 36 (0x24) and it takes 2 seconds (0x02) before the specified level is reached.

##### Checking dim level

The controller can request the dim level from the switch by using the Multilevel Switch Get command with the following layout.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_MULTILEVEL							
Command = SWITCH_MULTILEVEL_GET							

The switch device will then answer with a Multilevel Switch Report command with the following layout (assuming the device is at dim level 78 (0x4E)).

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_MULTILEVEL							
Command = SWITCH_MULTILEVEL_REPORT							
Value (0x4E)							

## Dimming

The controller can tell the switch to start dimming the power down by using the Multilevel Switch Start Level Change command with the following layout.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_MULTILEVEL							
Command = SWITCH_MULTILEVEL_START_LEVEL_CHANGE							
Re- ser- ved (0b)	Up/ Down (0b)	Ignore Start Level (1b)	Reserved (00000b)				
Start Level (0x00)							
Dimming Duration (0x05)							

The switch device will then start decreasing the dim level starting from the actual level. This will continue until the controller sends a command that tells the device to stop or until the device reaches the level 0. The dimming duration specify that it will take 5 seconds (0x05) to dim from level 0 to 99. The dimming rate is therefore equal to  $99 \text{ levels} / 5 \text{ seconds} = 19.8 \text{ levels per second}$ .

The controller stops dimming by the Multilevel Switch Stop Level Change command with the following layout.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_SWITCH_MULTILEVEL							
Command = SWITCH_MULTILEVEL_STOP_LEVEL_CHANGE							

**7.1.1.2.3 All On/All Off**

The switch can participate in all on or all off actions.

The controller can request all switches to be turned on by using the All Switch On command with the following layout.

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

The controller can also request all switches to be turned off by using the All Switch Off command with the following layout.

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

A switch can be excluded from the all on/all off functionality by using the All Switch Set command with the following layout.

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

### 7.1.2 Battery-powered rocker switch

This example illustrates a battery-powered rocker switch for lighting control as shown on the figure.



**Figure 11. Battery-powered rocker switch**

The battery-powered rocker switch can be placed directly on the wall without any additional wiring. A routing slave library is selected as the basic device class because the device **MUST** be capable of initiating transmission to a limited number of other devices. The battery-powered rocker switch **MUST** have assigned return routes to all the controllable destinations to obtain reliable RF communication. The listening flag is cleared because the slave is battery-powered. To allow the device to turn power on/off and dim light in other devices it **MUST** comply with the Multilevel Remote Switch Device Class that are a Specific Device Class created on the Remote Switch Generic Device Class. When the ON button is pressed on the switch it will turn the light on in the devices it has been associated with and off when the OFF button is pressed. When the ON button is held down on the switch it will start dimming up and the OFF button result in dimming down. The selected device class has the mandatory command class Multilevel Switch it can control in other devices. In addition are the Association, Battery and Wake Up command classes selected. The Association command class is used to specify the devices the rocker switch **MUST** be capable of initiating transmission to. The Battery command class is used to sent unsolicited frames in case a battery low is detected. The Wake Up command class allows the rocker switch to wake up occasionally and notify another device with respect to pending information.

### 7.1.2.1 Node Information

The controller will assign a Node ID to the device during the registration phase. When the node is being told to register it will issue a Node Information frame. The registration is part of the protocol and will be handled automatically. The Node Information frame that is sent will have the layout shown below.

7	6	5	4	3	2	1	0
Listening = NO	Protocol Specific Part						
Protocol Specific Part							
Protocol Specific Part							
Basic Device Class = BASIC_TYPE_ROUTING_SLAVE (Protocol Specific Part)							
Remote Switch Device Class = GENERIC_TYPE_SWITCH_REMOTE							
Multilevel Remote Switch Device Class = SPECIFIC_TYPE_SWITCH_REMOTE_MULTILEVEL							
Association Command Class = COMMAND_CLASS_ASSOCIATION							
Wake Up Command Class = COMMAND_CLASS_WAKE_UP							
Battery Command Class = COMMAND_CLASS_BATTERY							
Support/Control Mark = COMMAND_CLASS_MARK							
Multilevel Switch Command Class = COMMAND_CLASS_SWITCH_MULTILEVEL							

Notice the COMMAND\_CLASS\_MARK that separate command classes supported by the device itself and command classes it can control in other devices.

### 7.1.2.2 Rocker Switch Functionality

In the following sections are it shown how the command classes notified by the node information frame are used and the functionality they provide.

#### 7.1.2.2.1 Association

The associations are typically created and maintained by a portable remote. The following describes how the associations are created and maintained.

#### Groups Supported

The rocker switch can only control other nodes via the Multilevel Switch command class and therefore is only one group adequate. Groups supported are requested as follows:

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

The rocker switch will respond as follows:

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ASSOCIATION							
Command = ASSOCIATION_GROUPINGS_REPORT							
Supported Groupings (0x01)							

### Nodes in a Group Supported

Now can the number of nodes in a given group be determined. The rocker switch supports only one group so the number of nodes MUST be requested from grouping identifier one:

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ASSOCIATION							
Command = ASSOCIATION_GET							
Grouping Identifier (0x01)							

The rocker switch will respond as follows:

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ASSOCIATION							
Command = ASSOCIATION_REPORT							
Grouping Identifier (0x01)							
Max Nodes Supported (0x05)							
Reports to Follow (0x00)							

The rocker switch supports up to five destinations because it is not possible to assign return routes to more destinations in a routing slave library. Notice that no node ID's are returned because the associations have not yet been created at this stage.



### Configuring Associations

The rocker switch with node ID equal to 0x08 MUST control five outlet dimmers with the node ID's 0x02, 0x03, 0x04, 0x05 and 0x06 assigned to group one. This is done by the Association Set command as shown below:

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ASSOCIATION							
Command = ASSOCIATION_SET							
Grouping identifier (0x01)							
Node ID1 (0x02)							
Node ID2 (0x03)							
Node ID3 (0x04)							
Node ID4 (0x05)							
Node ID5 (0x06)							

In addition to the above command MUST the following API calls be executed to assign the return routes for the five outlet dimmers:

```
ZW_DeleteReturnRoute( 0x08, callback_func );

ZW_AssignReturnRoute( 0x08, 0x02, callback_func );

ZW_AssignReturnRoute( 0x08, 0x03, callback_func );

ZW_AssignReturnRoute( 0x08, 0x04, callback_func );

ZW_AssignReturnRoute( 0x08, 0x05, callback_func );

ZW_AssignReturnRoute( 0x08, 0x06, callback_func );
```

### Managing Associations

In case the outlet dimmer with node ID equal to 0x05 is to be replaced by a new with node ID equal to 0x09. Then it is necessary to be sure that 0x05 is among the current associations by sending the command:

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ASSOCIATION							
Command = ASSOCIATION_GET							
Grouping Identifier (0x01)							

The rocker switch will respond as follows:

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ASSOCIATION							
Command = ASSOCIATION_REPORT							
Grouping Identifier (0x01)							
Max Nodes Supported (0x05)							
Reports to Follow (0x00)							
Node ID1 (0x02)							
Node ID2 (0x03)							
Node ID3 (0x04)							
Node ID4 (0x05)							
Node ID5 (0x06)							

It is also necessary to know all the associations before the return routes can be assigned.

Now is the old outlet dimmer remove by the command:

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ASSOCIATION							
Command = ASSOCIATION_REMOVE							
Grouping Identifier (0x01)							
Node ID4 (0x05)							

Then is the new outlet dimmer added by the command:

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_ASSOCIATION							
Command = ASSOCIATION_SET							
Grouping identifier (0x01)							
Node ID1 (0x09)							

and finally assignment of the return routes for the five outlet dimmers:

```
ZW_DeleteReturnRoute( 0x08, callback_func );
```

```
ZW_AssignReturnRoute( 0x08, 0x02, callback_func );
```

```
ZW_AssignReturnRoute( 0x08, 0x03, callback_func );
```

```
ZW_AssignReturnRoute( 0x08, 0x04, callback_func );
```

```
ZW_AssignReturnRoute( 0x08, 0x06, callback_func );
```

```
ZW_AssignReturnRoute( 0x08, 0x09, callback_func );
```

#### 7.1.2.2.2 Wake Up

The battery-operated device is typically asleep unable to receive commands to minimize battery consumption. In the following it is shown how a battery-operated device can wake up occasionally to obtain information from another device.

#### Wake Up Configuration

A device can configure the battery-operated device with wake up interval and node ID of the device to receive I'm awake notifications by using the Wake Up Interval Set command. In case the device MUST wake up every 6 hours and notify another device with node ID equal to 0x07 then the layout is as follows.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_WAKE_UP							
Command = WAKE_UP_INTERVAL_SET							
Seconds3 (MSB) = 0x00							
Seconds2 = 0x54							
Seconds1 (LSB) = 0x60							
Node ID = 0x07							

It is also possible to request the configuration parameters by using the Wake Up Interval Get command and the battery-operated device will then return a Wake Up Interval Report command.

### Wake Up Sequence

The battery-operated device uses the Wake Up Notification command to notify another device that it is awake. When the Wake Up Notification command is received then the two devices are ready to exchange information.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_WAKE_UP							
Command = WAKE_UP_NOTIFICATION							

The rocker switch can broadcast the wake up configuration until it gets the device destination via the Wake Up Interval Set command.

The notified device uses the Wake Up No More Information command to tell the battery-operated device to go back to sleep to minimize battery consumption.

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_WAKE_UP							
Command = WAKE_UP_NO_MORE_INFORMATION							

#### 7.1.2.2.3 Battery

The wake up notified device can request the battery level from the rocker switch when awake by using the Battery Level Get command with the following layout.

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

The rocker switch device will then answer with a Battery Level Report command with the following layout (assuming the battery level is 95% (0x5F) of the full battery).

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_BATTERY							
Command = BATTERY_REPORT							
Value (0x5F)							

The rocker switch can also send an unsolicited battery low using the following format:

7	6	5	4	3	2	1	0
Command Class = COMMAND_CLASS_BATTERY							
Command = BATTERY_REPORT							
Value (0xFF)							

This allows the battery operated rocker switch to notify about the critical battery level immediately and not wait until the next Wake Up notification.

#### 7.1.2.2.4 Switching On/Off and Dimming

Refer to the previously example regarding the outlet adapter with dimming capability with respect to the functionality available in the Multilevel Switch command class. In this example does the rocker switch not support this command class, but have the capability to control the Multilevel Switch command class in other devices.

## 7.2 Advanced Energy Control Applications

The purpose of this section is to provide an overview of the most important deployment scenarios and how the Advanced Energy Control (AEC) Framework accommodates these.

### 7.2.1 Deployment Scenarios

The scenarios for deploying smart meter and advanced energy control solutions will differ widely. It is desirable and even REQUIRED to base them on a common architecture. The advantage of the AEC framework is that it can be used in a large variety of scenarios.

The framework is designed to be flexible by supporting WAN communication both via the Internet in a secure fashion, as well as over a utility / meter network. The use of both types of network can also be combined. The utility meter network can be based on TCP/IP. However, it is not REQUIRED to be based on TCP/IP.

The AEC framework is designed to allow scalability. It allows initial simple deployment scenarios with the ability of later adding more types of devices and to extend the range of control features utilized at any time.

Examples include:

- **Electricity meter + display**

A wireless, battery operated display in the home shows the actual use of energy and can furthermore exhibit trends. Obviously, also mains powered displays can be supported.

More advanced displays using the same RF communication interface will provide more advanced analysis and advisory functions.

- **Backbone communication**

The meters will typically also provide communications with backend supplier systems. Either through:

- Secure TCP/IP-based communication using the Z-Wave application protocol through a dedicated smart meter network.
- Secure TCP/IP-based communication using the Z-Wave application protocol through a general Internet router; e.g. through an ADSL access router.
- Secure or unsecured communication over practically any type of other meter reading or smart meter network technology. The AEC Framework provides a well defined service interface for this.

Examples of smart meter media types that can be integrated easily to form a part of Z- Wave AEC Framework based solutions include LON (Echelon) and other powerline communication media, short-range wireless systems, licensed RF systems, GSM / GPRS, WiMax, and any media that is suitable to support TCP/IP communications.

- **Meter collection for other types of energy / utilities**

A smart meter that is based on the AEC Framework can also collect information about other types of energy utility use in a secure fashion, examples include:

- Gas, Electrical generation, Hot water, Cold water etc...

Such utility meters can be both battery operated as well as mains powered.

- **Home energy controller**

Instead of using the electrical meter as a “smart meter” that also serves as a data collector and concentrator for other utilities and/or the gateway to a smart meter / meter reading network, the framework allow to provide such capabilities also in a separate “home energy controller”.

- **Home control integration – Device level**

One key strength of the AEC Framework is that it easily integrates with any other type of Z-Wave compliant device, providing seamless interoperability. Due to Z-Wave’s application model, even devices that are not “aware” of the AEC Framework can be controlled.

The AEC Framework enables energy applications to learn about the actual energy use of devices and also about their possible operation modes. Similarly “smart” devices are able to learn about the current objectives or even restrictions in terms of energy use in the home. This enables such device to adjust their operation in a flexible fashion.

Through this, it is possible to make granular decisions to optimize and reduce energy use. From a consumer perspective, this is much preferable to simply shedding loads. E.g. it would be better to “instruct” the HVAC system to reduce its peak and average use of electricity than to shut it off in a load shedding operation.

- **Home control integration – Controller level**

Besides integration on the home control device level it is also possible to integrate the AEC Framework with controller devices in home control solutions. Examples of such home controller devices include e.g. wall mounted controllers touch sensitive high resolution graphics, PC-based devices, set top boxes that are connected to TVs for display, or home Internet gateways where related applications can be accessed by any web browser.

- **PC integration**

PCs (or a PC technology based devices) can take any role in the AEC Framework

## **7.2.2 The Advanced Energy Control Architecture**

The purpose of this section is to provide an overview of the technical and functional architecture of the AEC framework.

### **7.2.2.1 Energy control approach**

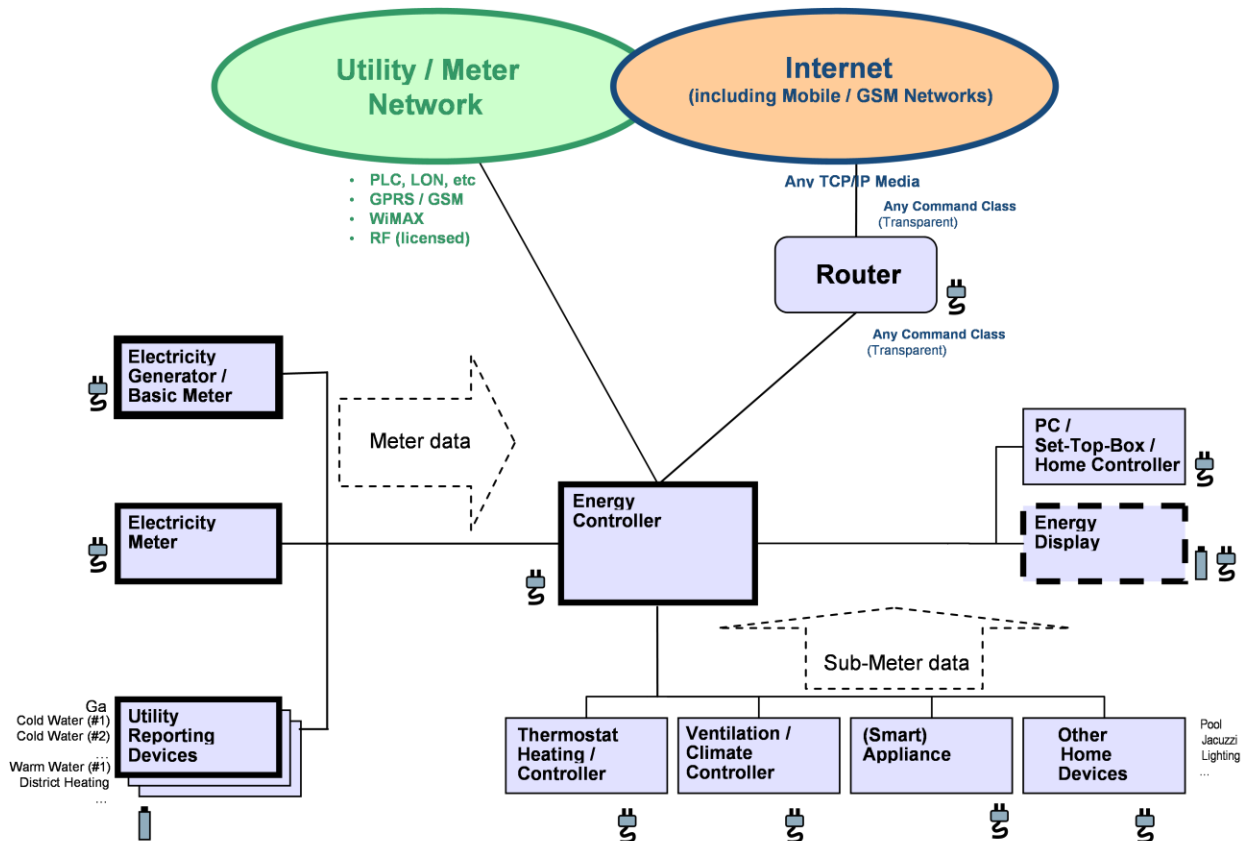
The main emphasis in the AEC framework is to enable “smart” devices and enable the consumer to control actively the use of energy to optimize and reduce cost, to respond to overall demand, and to react to emergencies in the supply. The key is to provide the information necessary to make such decisions.

Essentially, instead of forcing the air condition to turn off, the emphasis is on informing and educating the consumer with the objective to avoid emergencies where this would even be necessary. This stems from the fundamental insight that the largest and most sustainable savings in overall and peak energy use will be resulting from informed decisions made by the consumer and by informed “smart” operation of home devices.

### 7.2.2.2 Advanced Energy Control Logical Device Model

It is important to note, that logical devices shown in this section can be flexibly combined into physical products. Typically, it is not to be expected to find each of the logical devices shown in a *separate* actual physical device. Instead, multiple logical devices will often or even typically be combined into one physical device. Examples are shown below.

The diagram below illustrates the **logical device model** of the Advanced Energy Control profile.



**Figure 12. Logical device model of the Advanced Energy Control profile**

The energy controller is a central element in the Z-Wave Framework. It can serve as a collector and aggregator of meter readings, provide information to display devices, provides energy information to home control devices, run applications for active energy control, and it is handling communications to backend supplier / utility systems via smart meter networks and/or the Internet.

The types of electricity meters represented by the corresponding logical device can range from very basic single phase, single rate meters to advanced configurable smart meters. The AEC framework is also designed in a way that an electricity meter could also communicate directly with backend supplier / utility systems. This is especially important in simple deployment scenarios where the energy controller logical device function is not implemented.

For electricity generators or generally “secondary” meters, a simpler type of logical device is foreseen in the framework as shown in the diagram. Also this logical device is based on a common set of command classes.



Utility reporting devices can be either battery operated or mains powered. Their purpose is to report meter information either on an absolute or incremental basis.

Energy displays can range from very simple battery operated devices with fixed, custom LCD designs or just a few lines display to advanced control devices that conduct complex analysis and provide detailed advice to consumers.

The Framework also supports simple and low cost sub-metering functionalities implemented in other Home Appliances devices (E.g. Power plugs with built-in Watt-meter).

Thermostats; heating, HVAC, ventilation, and climate controllers, appliances, and other home control devices can learn about the energy status, energy saving objectives, and emergency situations in the energy supply and adjust their operation and thereby their use of energy accordingly. As outlined above, the AEC framework also enables “forceful” load shedding, e.g. in emergency, overload situations to protect the grid.

As explained above, backend communication with supplier / utility systems can occur either through a smart meter network or through the Internet. In case of communication through the Internet, no special functionality is REQUIRED on the Z/IP gateway that links the Z-Wave network with other TCP/IP media (such as ADSL). The Z/IP gateway is transparent for application protocols and thereby also does not need to be upgraded, updated, or exchanged when devices in the home network are being added or changed. Of course, other logical device types of the AEC Framework could be combined with the Z/IP gateway function into one single physical device.

Communication through smart meter networks can occur either based on TCP/IP or with practically any other application protocol. In case of TCP/IP based communication, the very same commands as used in the home network or that could also be used via the Internet would also be used in this case. For the integration of other protocols, the AEC Framework defines an interface

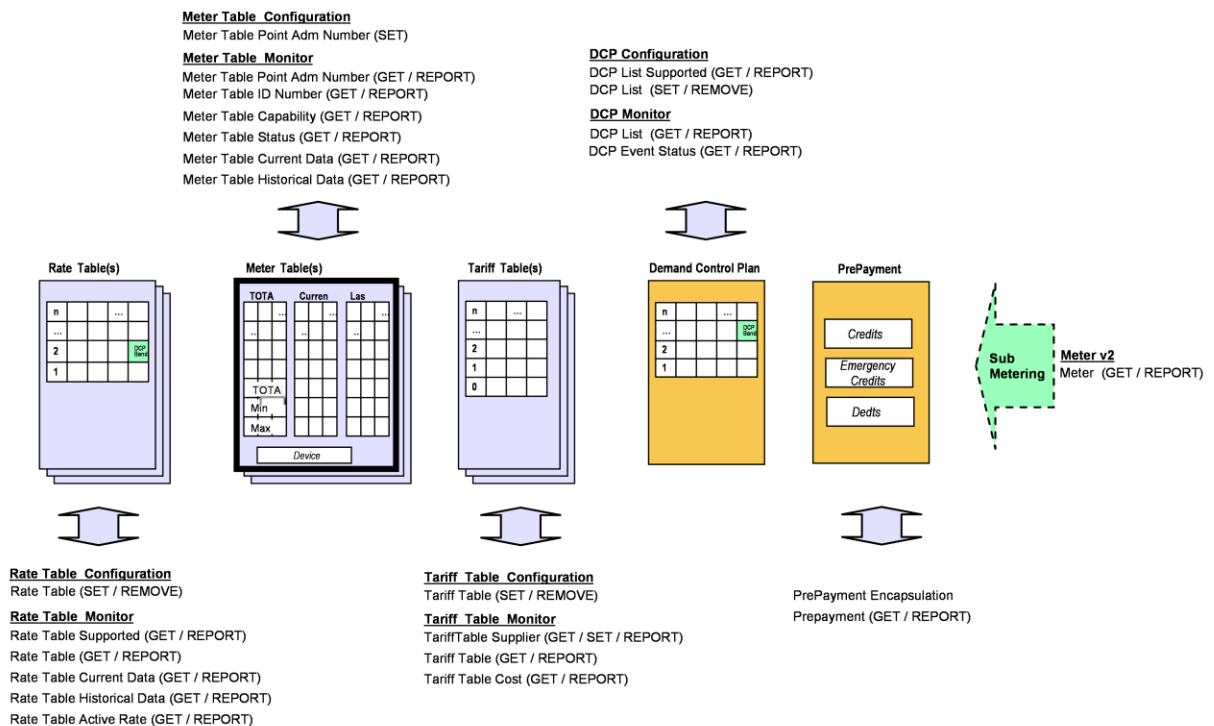
The connections shown are to illustrate the typical relationship between devices. Based on combining multiple logical devices into a single physical device, relationships could change. Furthermore, there are also more direct communication links possible between devices based on the actual command classes specified for the devices. For example, in a simpler scenario than illustrated in the diagram above, a display device can collect information directly from an electricity meter (actually without even needing to implement additional or other command classes).

### 7.2.2.3 Z-Wave Advanced Energy Control Metering Data Model

The challenge in designing a universal data model for a smart meter and advanced energy control solutions is to provide a solution that can handle very complex rate and tariff models, but that is also simple enough to be deployed cost effectively in early or “simple” solutions. Furthermore, the model needs to be flexible to fulfill future requirements and all relevant deployment scenarios without requiring entirely new definitions of the underlying command classes and commands.

The approach chosen in the Z-Wave Advanced Energy Control Framework is to model the data in a modular fashion. It will be up to the specific requirements of a product, which portions of the model are implemented. Thereby, both very simple and the most advanced meters share the same set of commands.

The following diagram illustrates the data model used in the Framework. Advanced meters MAY use all or a majority of the model.



**Figure 13. Data model used in the Advanced Energy Control framework**

#### 7.2.2.3.1 Meter Table Configuration/Monitoring

The meter table consists of a Meter header and a Meter data section. The Meter Table Header section describes the core attributes of the meter and the Meter Table Data section contains the measured values. If a physical implementation of a meter contains several meters, each meter will have its own Meter Table. The Meter Table has a very flexible format allowing efficient implementations ranging from simple meters to very advanced meters. The simplest meters only support the 'Accumulated consumption' data without bearing the burden of the more sophisticated dataset supported by more advanced meters.

The Meter Table configuration commands are separated for the Meter Table monitoring commands in the Meter Table Monitor Command Class, allowing the classes to be OPTIONAL 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).

#### 7.2.2.3.2 Rate Table Configuration/Monitoring

The Rate table consists of a list of applicable rates. Each rate is defined as a logical combination of time, Max demand, Max consumption and DCP events (see DCP table).

The Rate Table configuration commands are separated for the Rate Table monitoring commands in the rate Table Monitor Command Class, allowing the classes to be OPTIONAL supported at different Z-Wave security levels. (E.g. Rate table monitoring commands could be supported in any device, while enabling a strict and certificate based security solution for the Rate Table Configuration Command class).

##### Example A

Rate1 applicable daily from 8am to 3pm  
Rate2 applicable daily from 3pm to 8am

##### Example B

Rate1 applicable when Max consumption is between 0kWh and 2000kWh  
Rate2 applicable when Max demand is 2000kWh and 20000kWh

##### Example C

Rate1 applicable daily from 8am to 3pm and if a Demand Control Plan with DCP Rate ID=7 is active.  
Rate2 applicable daily from 8am to 3pm  
Rate3 applicable daily from 3pm to 8am

#### 7.2.2.3.3 Tariff Table Configuration/Monitoring

The Tariff table consists of a list of Tariffs, each associated with the equivalent Rate Table entry. The Tariff table consists of a Tariff table Header and a Tariff Table data section.

The Tariff Table configuration commands are separated for the Tariff Table monitoring commands in the Tariff Table Monitor Command Class, allowing the classes to be OPTIONAL supported at different Z-Wave security levels. (E.g. Tariff table monitoring commands could be supported in any device, while enabling a strict and certificate based security solution for the Tariff Table Configuration Command class).

#### 7.2.2.3.4 DCP table Configuration/Monitoring

The DCP table allows the Utility Suppliers to issues Demand control Plan (DCP) events to the end consumer requesting utility consumption reduction. A DCP event contains information regarding

criticality, products involved, requested reduction, time duration and if a certain rate (identified by the DCP Rate ID – see Rate Table) is associated with the event.

The DCP configuration commands are separated for the DCP monitoring commands in the Demand Control Plan Monitor Command Class, allowing the classes to be OPTIONAL supported at different Z-Wave security levels. (E.g. DCP monitoring commands could be supported in any device, while enabling a strict and certificate based security solution for the DCP configuration command class).

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 Association groups are either configured by the installer, the end user or Utility Supplier (remote management). 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.

#### Example A:

A utility Supplier wants to reduce a utility consumption peak. The peak is anticipated to be present from 7 Sep 2008 from 5pm to 6pm.

Through the DCP support in the AEC framework, the utility Supplier is able to transmit an event to all subscribers corresponding to the following wording:

'Sep 7 2008 from 5pm to-6pm it is requested to reduce the utility consumption to 80% of normal. For those participating in the event, the rate will be changed from the rate3 to rate4 (more attractive rate) for the duration of the event'.

#### Example B:

A utility Supplier wants to reduce a utility consumption peak. The peak is anticipated to be present from 7 Sep 2008 from 5pm to 6pm.

Through the DCP support in the AEC framework, the utility Supplier is able to transmit an event to all subscribers corresponding to the following wording:

'Sep 7 2008 from 5pm to-6pm it is mandated to reduce the utility consumption to 80% of normal.

#### Example C:

A small energy control system consisting of an Energy Controller and three home control Z-Wave devices which does not support the DCP command class: NodeId 1: Setback Thermostat device, NodeId 8: Simple Thermostat device, NodeId 4: Multilevel Power Switch device.

When the Energy Controller receives the DCP event:

'Sep 7 2008 from 4pm to-5pm it is mandated to reduce the utility consumption to 80% of normal.

It can translate the event request into the corresponding Z-Wave commands for the individual nodes:

Node1, Thermostat\_Setback\_Set(permanent override, Energy saving mode)  
 Node8, Thermostat\_Setpoint\_Set(Heating setpoint#1, 19,5°C)  
 Node4, Multilevel\_Switch:Set (Dimlevel =0x20)

#### 7.2.2.3.5 Prepayment/Prepayment Encapsulation

The Prepayment functionality allows the card reader and meter to be physically separated. Additionally it allows the Smart Card data to be shared with other devices (displays etc).

Z-Wave does not limit the prepayment protocol between the Card reader and the Meter, as Z-Wave encapsulates the communication, thereby allowing new card protocols in the future.

Z-Wave allows credit, debt and emergency credits information to be shared between devices.

The prepayment functionality has a flexible format allowing efficient implementations of very simple meters all the way to very advanced meters. Simple meters MAY therefore omit to support this functionality.

#### 7.2.2.3.6 Meter v2

The Meter v2 command class allows sub-metering in any Z-Wave device which are not a dedicated Utility meters (E.g. Power Adapter with a built-in Wattmeter). The command class combines the well know functionalities from the Energy production Command Class, Pulse Meter Command Class, Meter v1, Multilevel Sensor Command Class along with new functionality into one new class.

### 7.2.3 Security

Security is an important concern when the integrity of meter readings; meter configurations; load control of devices in the home; and also personal information of consumers are a risk.

Security is a challenging area in any wireless home control solution, since conflicting requirements are colliding especially strong. One hand, one would desire optimal security that is robust against any form of threat; both today and also in the future. On the other hand, devices need be very low cost and therefore complexity MUST be minimized. User interfaces of many home control devices are extremely limited, not allowing entering keys and/or PIN numbers into devices. Small networks need to be able to operate fully autonomously and security therefore cannot rely on always looking up central, public certificate servers. Most importantly, security MUST be extremely simple to setup and handle over the entire "lifetime" of a home control network. User MUST NOT be exposed in any way to the underlying technology and procedures.

In several Meter application areas such as "prepaid" utility usage it is advisable to separate the network and operational security from the security of e.g. accepting authorizations / payments and thereby enabling and stopping e.g. electricity supply.

Z-Wave addresses these different set of security requirements by the three Tier Z-Wave security approach:

- Z-WaveSec. – Z-Wave Security Command Class v2  
Targeted for nodes exchanging non-personal data  
Lowest Cost - High Security level – Plug & Play  
Confidentiality, Authentication, Fabrication robust – AES128 based  
Single Network Key, In-band initial symmetrical key exchange
- Z-WaveSecIP – Hybrid Security Command Class v1 and Security Link key Extension  
Targeted for nodes exchanging personal data  
Proven technology - High Security level – Plug & Play  
Confidentiality, Authentication, Fabrication robust – AES128 based  
Asymmetric key exchange, Network + Link Keys  
Certificates installed in nodes.

- **Z-WaveSecSmartCard** – Prepayment Encapsulation Command Class targeted for nodes exchanging payment data  
Highest flexibility.  
Allows Smartcard payment & Security information to be exchanged via Z-Wave

#### **7.2.3.1 Z-WaveSec**

Z-WaveSec has been designed with ease of use (Plug & Play) in mind. By employing the AES-128 block cipher technology, Z-Wave is protected against modification, fabrication, and replay attacks. Authentication is conducted based on a 128-bit authentication key with a 64-bit MAC. Confidentiality is assured through encryption with a 128-bit encryption key.

Approaches for setup as in WLAN and Bluetooth where each device needs to be setup with a network key, pass phrase, or PIN are not applicable in home control. Therefore, one major aspect of the Z-WaveSec design was to assure easy handling for installers and consumers alike and a setup that does not require any special tools or special software and no entering of any key or secret materials.

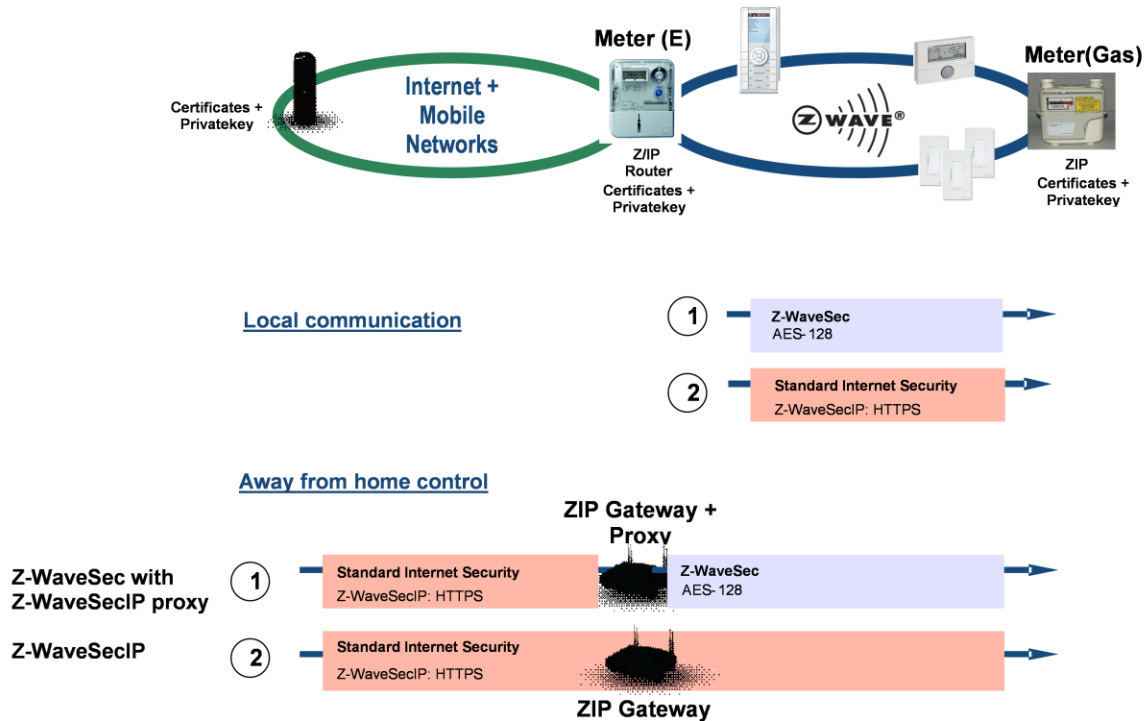
Z-WaveSec is initiated by installing devices at a short range with minimal transmitter power and to exchange the key material at this stage. Thereby installation and setup of security are entirely invisible to the user. However, the approach leaves a small vulnerability against eavesdropping at this moment of installation against intruders that would be able to intercept and analyze the communication of the joining device at that moment.

#### **7.2.3.2 Z-WaveSecIP**

For the remote home access use case, the Z-WaveSecIP based on convergence between IP and Z-Wave, adds an additional layer of security in the form of protecting the links between TCP/IP devices anywhere in the Internet to the Z-Wave nodes. HTTPS is well proven and accepted in the Internet; it is used for the vast majority of all secure web transactions today. HTTPS is also chosen as the standard method since it does not require user setup and is available in practically all TCP/IP devices; including TCP/IP stacks on JAVA enabled cell phones.

All Z-WaveSecIP enabled nodes have an installed certificate and a corresponding private key (installed by the trusted manufacturer).

As shown in the diagram below, Z-WaveSecIP security can be used both from IP nodes to the Z/IP gateway, end-to-end or locally between IP/Z-Wave nodes on the Z-Wave network.



**Figure 14. Z-WaveSecIP solutions with respect to IP and Z/IP nodes**

The Z-WaveSecIP security has a low cost Z-WaveSecIP proxy option through the Hybrid Security Command Class v1. This allows the Z-WaveSecIP to be terminated in the trusted Z-WaveSecIP proxy (located in the Z/IP Gateway). This enables public key infrastructure using the proxy in a standard Z-WaveSec Symmetric key network.

### 7.2.3.3 Z-WaveSecSmartCard

Certain applications in the Energy Control space are extremely critical to potential breaches of security. Examples include prepaid utility applications. Regardless of what security technology is chosen, for suppliers in that space it MAY not be acceptable to utilize an integrated security technology that comes with the Z-Wave and/or is standardized in the Internet.

Z-Wave is transparent for these types of solutions through the Z-WaveSecSmartCard option. Z-Wave thereby easily accommodates addition of corresponding technologies and components where REQUIRED.

## 7.2.4 Examples

### 7.2.4.1 Simple Meter with support for accumulated consumption

The following diagram illustrates how the data model would be applied to a very simple meter that provides only the accumulated consumption data.

**Meter Table Configuration**

Meter Table Point Adm Number (SET)

**Meter Table Monitor**

Meter Table Point Adm Number (GET / REPORT)

Meter Table ID Number (GET / REPORT)

Meter Table Capability (GET / REPORT)

Meter Table Status (GET / REPORT)

Meter Table Current Data (GET / REPORT)



Figure 15. Data model for Simple Meter



## Use case: Simple meter communicating with WAN and a Display

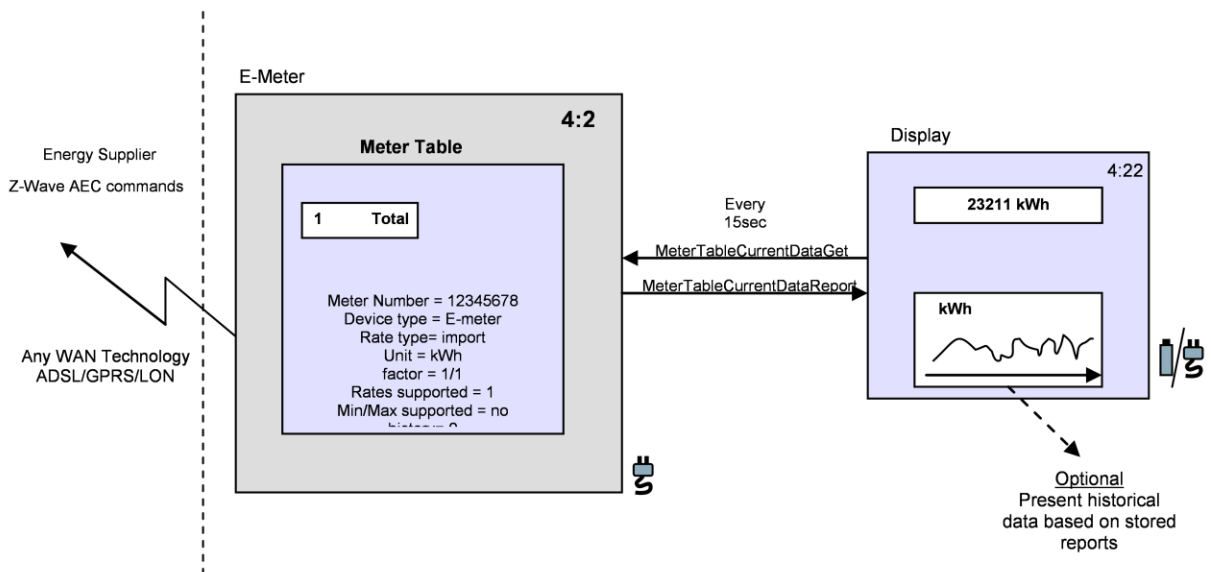


Figure 16. Use case for Simple Meter

It is important to note that in this case all command classes that are defined for the more advanced portions of the model are not **REQUIRED** to be implemented.



## Use case: Advanced Meter communicating with Display

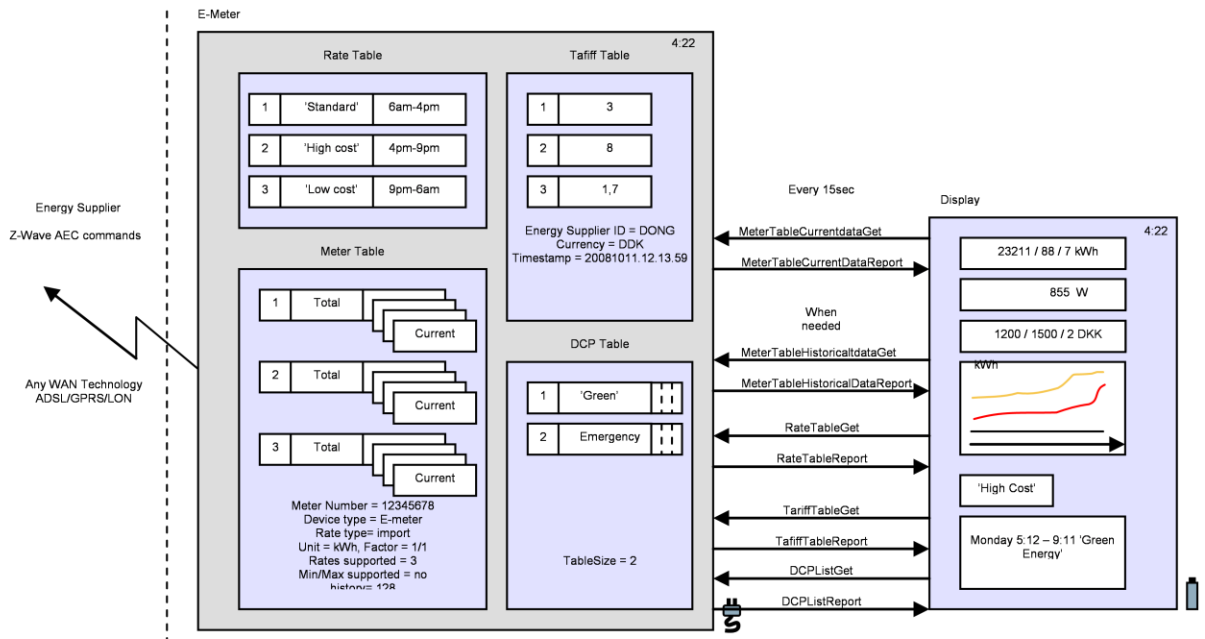


Figure 18. Use case for Adv. Prepayment Meter

### 7.2.4.3 Advanced Import / Export Meter

The following diagram and illustration will illustrate how the data model can be applied to an electricity meter that can both import and export energy, while also having an input pulse port for a gas meter.

#### Meter Table Configuration

Meter Table Point Adm Number (SET)

#### Meter Table Monitor

Meter Table Point Adm Number (GET / REPORT)

Meter Table ID Number (GET / REPORT)

Meter Table Capability (GET / REPORT)

Meter Table Status (GET / REPORT)

Meter Table Current Data (GET / REPORT)

Meter Table Historical Data (GET / REPORT)

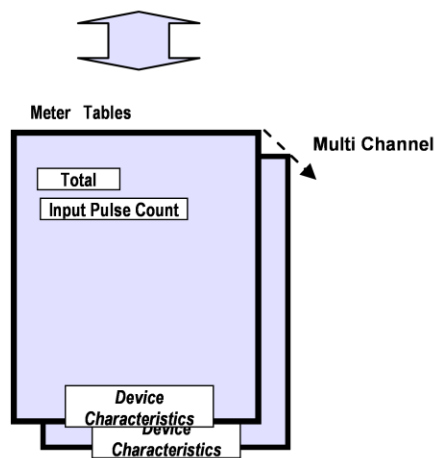


Figure 19. Data model for Adv. Import / Export Meter

Use case: Import / Export Electricity Meter with Pulse Input Port.

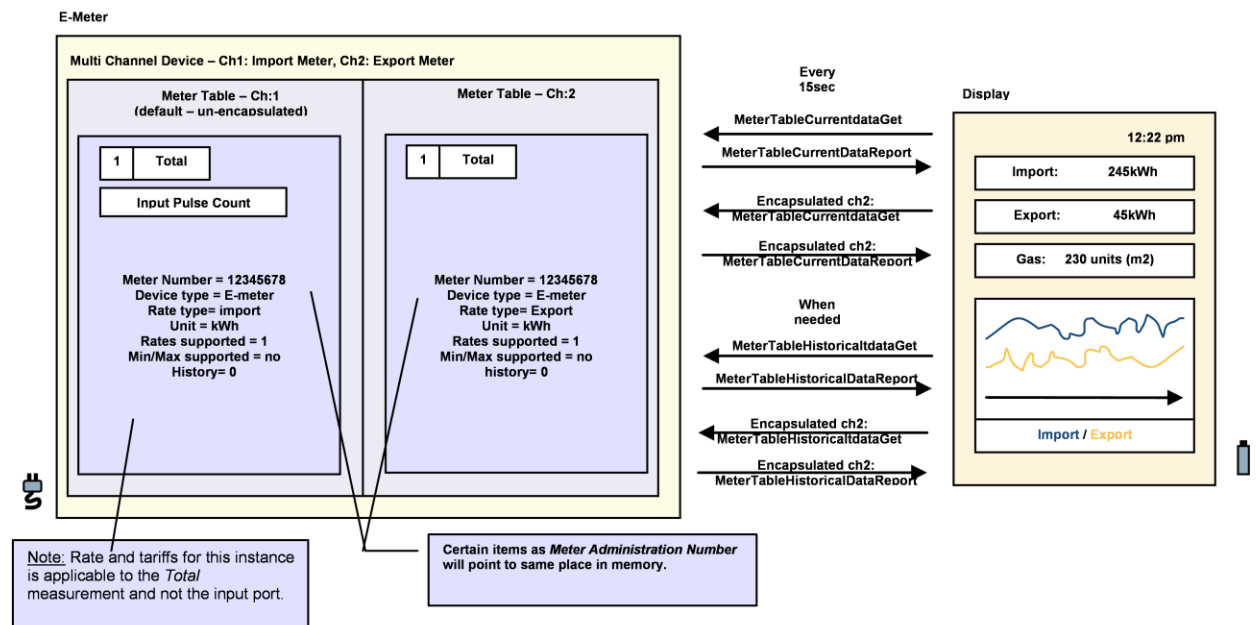


Figure 20. Use case for Adv. Import / Export Meter

#### 7.2.4.4 Advanced Import / Export Meter with rates

The following diagram and illustration will illustrate how the data model can be applied to an electricity meter that can both import and export electricity at different rates, while also having an input pulse port for a gas meter.

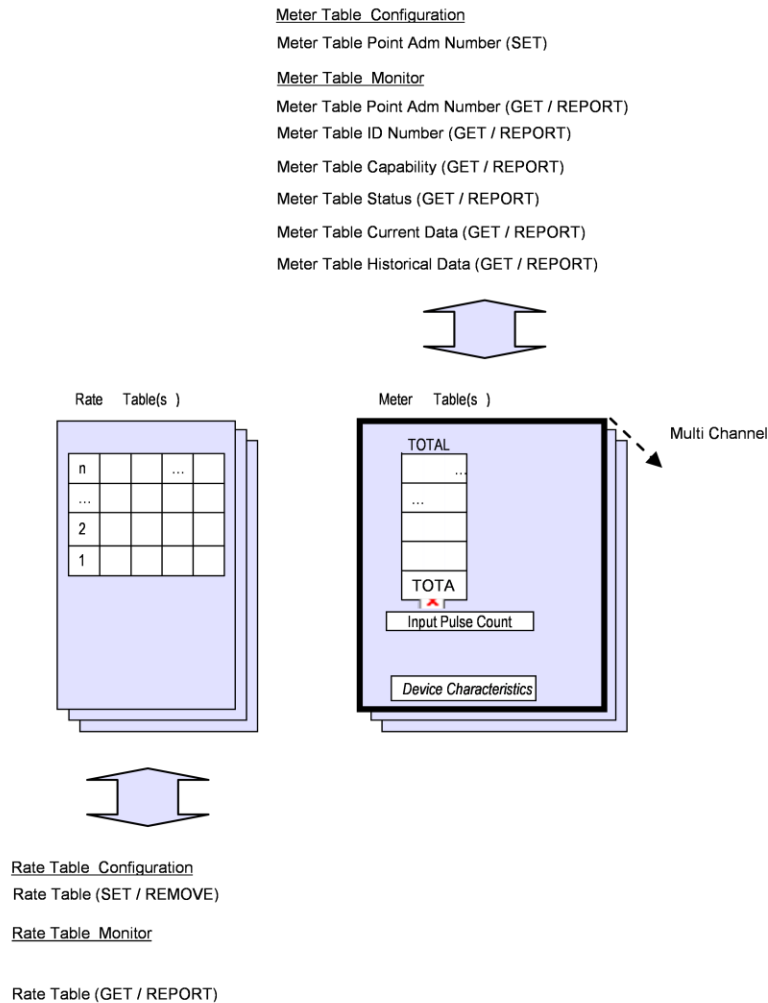


Figure 21. Data model for Adv. Import / Export Meter with rates

Use case: Rated Import / Export Electricity Meter with Pulse Input Port.

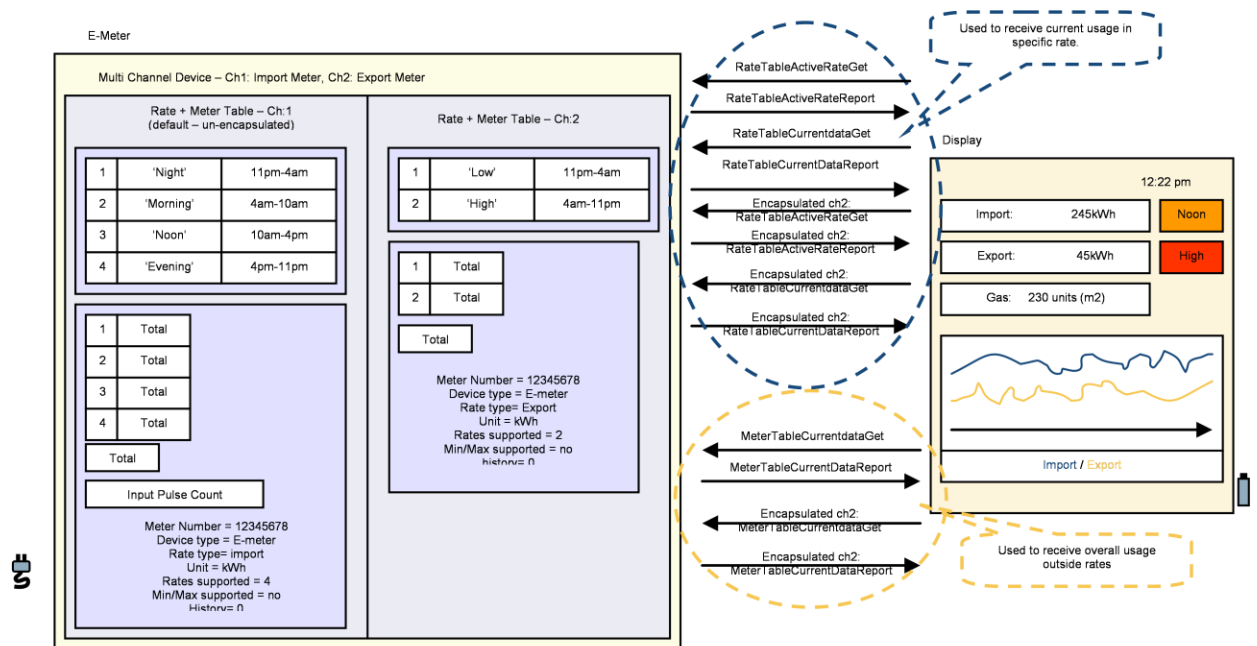


Figure 22. Use case for Adv. Import / Export Meter with rates

## REFERENCES

- [1] SD, Instruction, INS10638, Z-Wave Certification Overview.
- [2] Z-Wave Alliance, ZAD10485, Device/Command Class Development Process.
- [3] SD, Instruction, INS10247, Z-Wave ZW0102/ZW0201/ZW0301 Application Programming Guide.
- [4] SD, Software Design Specification, SDS11059, Chimney Fan Device Class.
- [5] SD, Software Design Specification, SDS11058, Thermostat Heating Device Class.
- [6] SD, Instruction, INS10244, Z-Wave Node Type Overview and Network Installation Guide.
- [7] SD, Application Note, APL10720, Programming the ZW0201 Flash from Internal MCU.
- [8] SD, Software Design Specification, SDS11060, Z-Wave Command Class Specification.
- [9] IETF RFC 2119, Key words for use in RFCs to Indicate Requirement Levels,  
<http://tools.ietf.org/pdf/rfc2119.pdf>



## INDEX

### A

Accumulated values .....	82
Advanced Door Lock Specific Device Class .....	79
Advanced Energy Control Framework .....	173
Advanced Energy Control Specific Device Class .....	86
Advanced Import / Export Meter .....	187
Advanced Import / Export Meter with rates .....	189
Advanced Prepayment Meter .....	185
Advanced Zensor Net Alarm Sensor Specific Device Class .....	39
Advanced Zensor Net Smoke Sensor Specific Device Class .....	49
Alarm Sensor Generic Device Class .....	28
All Switch Off command .....	164
All Switch On command .....	164
All Switch Set command .....	164
Application example .....	159
Automatic meter reading .....	82
AV Control Point generic device class .....	51

### B

Basic Command Class .....	16
Basic Device Class .....	15
Basic Device Classes .....	18
Basic Repeater Slave Specific Device Class .....	120
Basic Routing Alarm Sensor Specific Device Class .....	31
Basic Routing Smoke Sensor Specific Device Class .....	41
Basic Zensor Net Alarm Sensor Specific Device Class .....	35
Basic Zensor Net Smoke Sensor Specific Device Class .....	45
Battery Level Get command .....	171
Battery Level Report command .....	171
Battery-powered rocker switch .....	165
Binary Power Switch Specific Device Class .....	65
Binary Remote Switch Specific Device Class .....	116
Binary Scene Switch Specific Device Class .....	67
Binary Sensor Generic Device Class .....	58
Binary Switch Generic Device Class .....	62, 116
Binary Toggle Remote Switch Specific Device Class .....	117
Binary Toggle Switch Generic Device Class .....	117
Binary Toggle Switch Specific Device Class .....	151
Binary Tunable Color Light Specific Device Class .....	69
Blinds .....	156

### C

Certification program .....	2
Command Class field .....	16
Command classes .....	14
Commands .....	14

### D

Device classes .....	6
Device/command class development process .....	2
Display Generic Device Class .....	70

Door Lock Specific Device Class .....	78
Doorbell Specific Device Class .....	56
Drapes .....	156

**E**

Energy metering devices .....	82
Energy Production Specific Device Class .....	124
Enhanced Slave .....	22
Entry Control Generic Device Class .....	75

**G**

Gateway Specific Device Class .....	131
Generic Device Class field .....	16
Generic Device Classes .....	23

**I**

Inclusion controller .....	21, 22
Interoperability .....	2

**L**

Listening flag .....	16
----------------------	----

**M**

Mark Command Class .....	16
Meter Generic Device Class .....	82
Metering devices .....	82
Minimal control functionality .....	9
Motor Control Class A Specific Device Class .....	99
Motor Control Class B Specific Device Class .....	101
Motor Control Class C Specific Device Class .....	103
Multilevel Power Switch Specific Device Class .....	95, 159
Multilevel Remote Switch Specific Device Class .....	116, 165
Multilevel Scene Switch Specific Device Class .....	96
Multilevel Sensor Generic Device Class .....	88
Multilevel Switch Generic Device Class .....	92, 116, 159
Multilevel Switch Get command .....	161, 162
Multilevel Switch Report command .....	161, 162
Multilevel Switch Set command .....	160, 161
Multilevel Switch Start Level Change command .....	163
Multilevel Switch Stop Level Change command .....	163
Multilevel Toggle Remote Switch Specific Device Class .....	117
Multilevel Toggle Switch Generic Device Class .....	117
Multilevel Toggle Switch Specific Device Class .....	151
Multilevel Tunable Color Light Specific Device Class .....	105
Multiposition Motor Specific Device Class .....	98

**N**

NIF .....	15
Node Information Frame .....	15
Node Information Frame .....	160
Node Information Frame .....	166

**O**

Optional Functionality flag .....	16
Outlet adapter with dimming capability .....	159

**P**

PC Controller Specific Device Class.....	127
Portable controller.....	19
Portable Installer Tool Specific Device Class .....	113
Portable Remote Controller Specific Device Class .....	110
Portable Scene Controller Specific Device Class .....	111
Primary controller .....	19, 20, 21, 22
Pulse Meter Generic Device Class.....	106

**R**

Remote Controller Generic Device Class.....	108
Remote Switch Generic Device Class .....	114, 165
Repeater Slave Generic Device Class.....	118
Residential Heat Recovery Ventilation Specific Device Class.....	155
Router.....	20, 21, 22
Routing Alarm Sensor Specific Device Class.....	33
Routing Binary Sensor Specific Device Class .....	60
Routing Multilevel Sensor Specific Device Class .....	90
Routing Slave.....	22
Routing Smoke Sensor Specific Device Class.....	43

**S**

Satellite Receiver Specific Device Class.....	54, 55
Scene Controller Specific Device Class .....	128
Secondary controller.....	19, 20
Secure Keypad Door Lock Specific Device Class.....	80
Semi Interoperable Generic Device Class .....	121
Setback Thermostat Specific Device Class .....	144
Shades .....	156
Simple Display Specific Device Class.....	73
Simple Meter .....	183
Simple Meter Specific Device Class .....	85
Simple Window Covering Control Specific Device Class .....	157
Slave .....	21
Specific Device Class field .....	16
Specific Device Classes.....	23
Static controller .....	20
Static Controller Generic Device Class.....	125
Static Installer Tool Specific Device Class .....	130

**T**

Thermostat General Specific Device Class .....	136
Thermostat Generic Device Class.....	133
Toggle Switch Generic Device Class .....	149

**V**

Ventilation Generic Device Class .....	153
--	-----

**W**

Wake Up Interval Set command .....	170
Wake Up No More Information command .....	171
Wake Up Notification command .....	171
Water meters .....	82
Window Covering Generic Device Class .....	156

**Z**

Sensor Net Alarm Sensor Specific Device Class .....	37
Sensor Net Smoke Sensor Specific Device Class .....	47
ZW_classcmd.h .....	2
Z-Wave logo .....	2
Z-WaveSec .....	181
Z-WaveSecIP .....	181
Z-WaveSecSmartCard .....	182