Specification

Z-Wave Host API Specification

Version:	0.7.2
Description:	Specification of the API between a Z-Wave NCP and a host processor
Written By:	CSWG
Date:	2021.09.02
Reviewed By:	Z-Wave Alliance
Restrictions:	Public

Approved by:
Z-Wave Alliance Board of Directors

Table of Contents

1 Introduction

1.1 Purpose

This document specifies the communication and commands used by host processors to interface with a module supporting a Z-Wave API.

1.2 Audience and Requirements

The audience of this document is the Z-Wave Alliance members and Z-Wave developers.

1.3 Terms

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 RFC 2119 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.

1.4 Terminology And Abbreviations

Terminology and abbreviations used in this document are listed in Table 1.1

Table 1.1: Terminology and abbreviations

Term	Abbreviation	Description
Always Listening	AL (node)	Z-Wave node that is Always Listening. Refer to [zwave_nwk_spec] for details.
Frequently Listening	FL (node)	Z-Wave node that is Frequently Listening. Refer to [zwave_nwk_spec] for details.
Least Significant Byte	LSB	Byte in the bytestream that has the lowest weight
Most Significant Byte	MSB	Byte in the bytestream that has the highest weight
Non-Listening	NL (node)	Z-Wave node that is Non-Listening. Refer to [zwave_nwk_spec] for details.
Network Wide Inclusion	NWI	Inclusion method leveraging Explore NDPU to include through repeaters. Refer to [zwave_nwk_spec] for details.
Transmitter	Tx	RF Transmitter

2 Overview

The Z-Wave Applications Programming Interface (Z-Wave API) allows a host interface to communicate with a Z-Wave chip through any kind of physical interface.

The host may be PC or a less powerful embedded host CPU, e.g., in a remote control or in a gateway device. Depending on the chip family, the Z-Wave API is typically accessed via RS-232 or USB physical interfaces.

Here are some of the applications leveraging the Z-Wave API:

- Gateway Application
- PC Controller
- Conformance Testing Tool (CTT)

In this specification, we will refer to:

- The **host application**: It is the application making use of the Z-Wave API via the physical interface.
- The **Z-Wave API Module**: It is the Z-Wave API implementation providing an API to make use of its Z-Wave capabilities.

3 Interface communication

This chapter defines the frames and communications frame flows between a Z-Wave API supporting module and a host application. Several frames types are defined to enable session-like communication.

3.1 Table Syntax

The data and command format tables contains a column with byte/bit numbering. This column specifies what byte offset the shown field is in the complete data frame. In some cases this is not a fixed number and special systax notations are used.

Table 3.1: Table Syntax

byte\bit	Description
4	This field is a 1 byte field in byte 4 of the data frame
8+N	This field is the last field of N identical fields starting in byte 8
…	This field is a number of identical fields
5/(5..6)	This field starts at byte 5 but can be both 1 or 2 bytes long
6/7	This field is a 1 byte field located either in gyte 6 or 7

3.2 Frame types

3.2.1 Data Frame

The Data frame is used to transmit a command. It can be used it both directions (Z-Wave module to host, or host to Z-Wave module). All data frames MUST be formatted according to Table 3.2.

Table 3.2: Data frame format

byte\bit	7	6	5	4	3	2	1	0
1	Frame Type = SOF (0x01)							
2	Length							
3	Type							
4	Z-Wave API Command ID							
4+1	Z-Wave API Command Payload 1							
…	…							
4+L	Z-Wave API Command Payload L							
4+L+1	Checksum							

Frame Type (8 bits)

This field is used to detect the type of frame being transmitted. For a data frame, this field MUST be set to 0x01, indicating a Start of Frame (SOF).

Length (8 bits)

The Length field is used to indicate the total length, in bytes, of the following fields:

- Length (this field)
- Type
- Z-Wave API Command ID
- Z-Wave API Command Payload

The following fields MUST NOT be included in the length calculation:

- Start of frame
- Checksum

Type (8 bits)

The Type field is used to indicate the type of Command being sent in the data frame. It MUST be encoded according to Table 3.3.

Table 3.3: Data frame format - Type encoding

Value	Description
0x00	Request frame. This type MUST be used by the host application for unsolicited new commands. Z-Wave API callbacks MUST also use the Request type.
0x01	Response frame. This type MUST be used by the Z-Wave Module to issue responses to Request frames.
0x02..0xFF	*Reserved values MUST NOT be used and MUST be ignored by a receiving interface*

Z-Wave API Command ID (8 bits)

This field is used to advertise a command identifier that enable a receiving interface to parse the payload. Commands are described in section *Z-Wave API Commands*.

Z-Wave API Command Payload (L bytes)

This field is used to indicate the payload associated with the Z-Wave API command. The payload for each command is described in the *Z-Wave API Commands* section.

Checksum (8 bits)

The Checksum field is used to validate the data received in the Data Frame. The Checksum calculation MUST include the following fields:

- Length
- Type
- Z-Wave API Command ID
- Z-Wave API Command Payload

The Checksum field MUST be calculated using XOR operations: Checksum = 0xFF (XOR) Length (XOR) Type (XOR) Z-Wave API Command ID (XOR) Z-Wave API Command Payload 1 (XOR) . . . (XOR) Z-Wave API Command Payload N

An interface receiving a non-matching checksum MUST return a *NAK Frame*. An interface receiving a matching checksum MUST return an *ACK Frame*.

3.2.2 ACK Frame

The ACK frame is used to indicate the successful reception of a *Data Frame*. It MUST be formatted according to Table 3.4

Table 3.4: ACK frame format

byte\bit	7	6	5	4	3	2	1	0
1	Frame Type = ACK (0x06)							

3.2.3 NAK Frame

The NAK frame is used to indicate an error in the reception of a *Data Frame*. It MUST be formatted according to Table 3.5.

Table 3.5: NAK frame format

byte\bit	7	6	5	4	3	2	1	0
1	Frame Type = NAK (0x15)							

3.2.4 CAN Frame

The CAN frame is used to indicate the detection of a collision during *Data Frame* transmissions.

A CAN frame is most often returned when the UART is both transmitting and receiving at the same time (a collision). This results in the receiving end receiving a frame it did not expect and thus it drops the frame and returns a CAN. The transmitting end typically is also receiving a frame which it must process and ACK and then retransmit the frame that was returned with a CAN after an appropriate backoff interval.

It MUST be formatted according to Table 3.6.

Table 3.6: CAN frame format

byte\bit	7	6	5	4	3	2	1	0
1	Frame Type = CAN (0x18)							

3.3 Command frame flows

The Z-Wave API has several possible command frame flows:

- *Unacknowledged frame*
- *Acknowledged frame*
- *Acknowledged frame with response*
- *Acknowledged frame with callback*

- *Acknowledged frame with response and callback*
- *Unsolicited frame*

In the following subsections and the rest of this specification, the frames are numbered as follow:

1. initial data frame: it is the initial (request type) data frame from the host to the Z-Wave module.
2. response data frame: it is a response type data frame returned to an initial data frame from the Z-Wave module to the host.
3. callback data frame: it is a request type data frame sent from the Z-Wave module to the host, after ithas completed an action triggered by an initial data frame.
4. unsolicited data frame: it is a request type data frame sent from the Z-Wave module to the host.

3.3.1 Unacknowledged frame

There MAY be data frames that will not be acknowledged by the destination because of hardware or software restrictions. It can for example happen if the command instructs the Z-Wave module to enter reprogramming mode or go offline.

The communication flow MUST be as shown in Figure 3.1

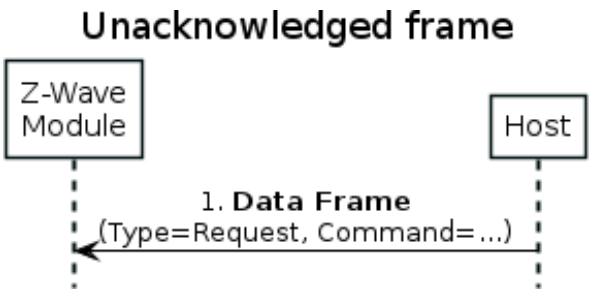

Figure 3.1: Unacknowledged frame

If a command is supposed to trigger an unacknowledged frame transmission, the host MUST NOT try to retransmit the command if no ACK frame is received. The host MUST retransmit the command if a *NAK Frame* or a *CAN Frame* is received.

3.3.2 Acknowledged frame

Acknowledged frames are frames that will not trigger any communication back from the Z-Wave module, apart from an *ACK Frame*. The communication flow MUST be as shown in Figure 3.2

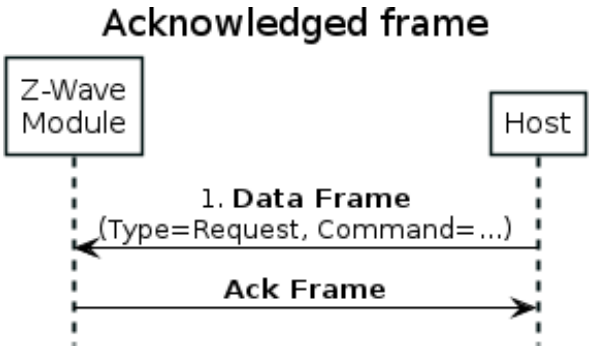

Figure 3.2: Acknowledged frame

3.3.3 Acknowledged frame with response

Acknowledged frames with response are acknowledged frames that will trigger an immediate response from the Z-Wave module. The communication flow MUST be as shown in Figure 3.3

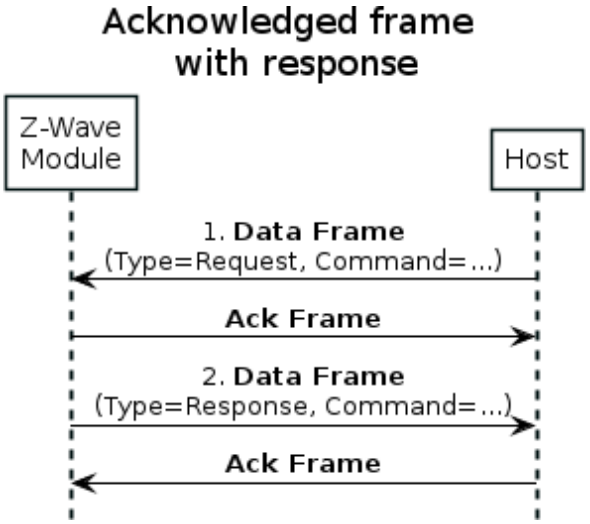

Figure 3.3: Acknowledged frame with response

3.3.4 Acknowledged frame with callback

Acknowledged frames with callback are acknowledged frames that will trigger a callback after an operation has been performed by the Z-Wave module. The communication flow MUST be as shown in Figure 3.4

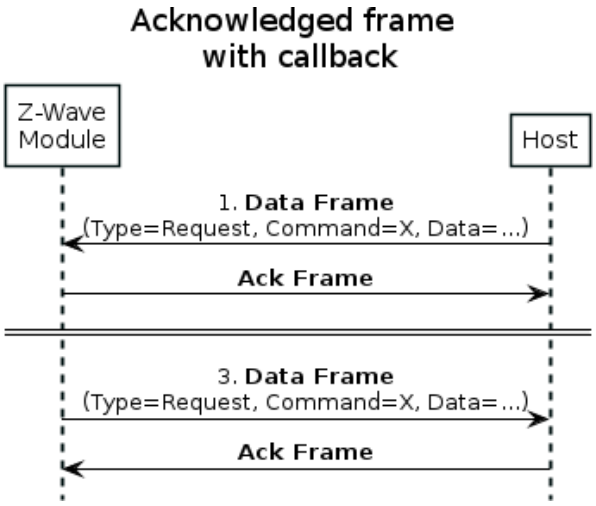

Figure 3.4: Acknowledged frame with callback

In some cases, an Initial Data Frame MAY trigger several callback frames.

3.3.5 Acknowledged frame with response and callback

Acknowledged frames with response and callback are acknowledged frames that will trigger both an immediate response and an additional callback after an operation has been performed by the Z-Wave module. The communication flow MUST be as shown in Figure 3.5

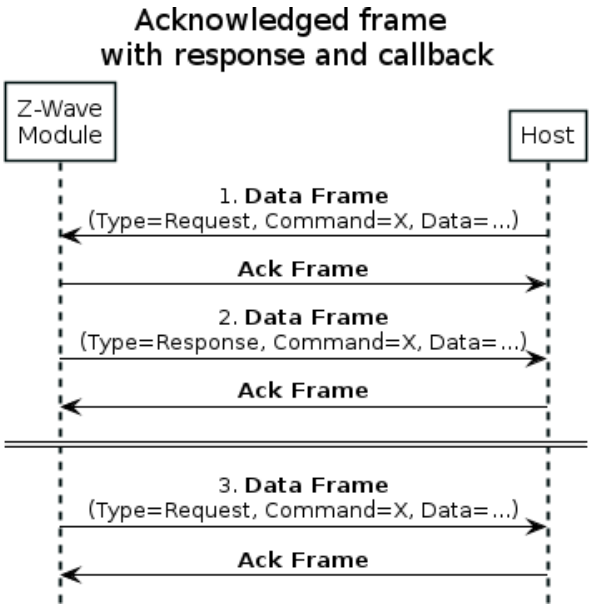

Figure 3.5: Acknowledged frame with response and callback

In some cases, an Initial Data Frame MAY trigger several callback frames.

3.3.6 Unsolicited frame

Unsolicited frames are frames that are sent from the Z-Wave module to inform the host application that an event happened. The communication flow MUST be as shown in Figure 3.6

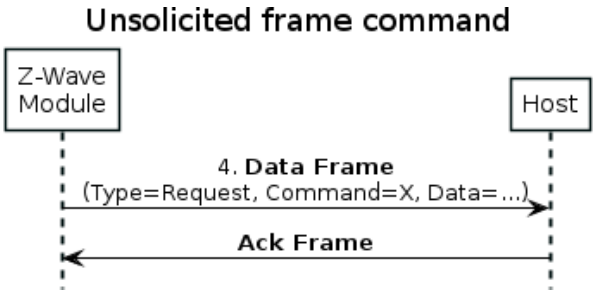

Figure 3.6: Unsolicited frame

3.4 Error handling

The following subsections show how to handle communications issues between the Z-Wave Module and the host application.

3.4.1 Retransmission timing

In general, retransmissions of a *Data Frame* MUST apply back-off timers.

The minimum back-off in milliseconds MUST be calculated according to (3.1)

$$T = 100 + \times 1000. \quad (3.1)$$

where:

- n is the retransmission number - 1. i.e. n=0 for the first retransmission.

A sending interface SHOULD add an additional random delay to the minimum back-off.

3.4.2 Missing Acknowledgment

By default, all data frames MUST be acknowledged by the receiving interface. Acknowledgement consists in sending an *ACK Frame*.

A sending interface MUST wait for 1600ms or more for an *ACK Frame* after transmitting a *Data Frame*.

In case of missing acknowledgement 1600 ms after a transmission, a transmitting interface SHOULD retransmit the unacknowledged data frame.

This recommendation MAY be adjusted based on the physical medium used for communication between the two interfaces.

A transmitting interface SHOULD make 3 retransmissions attempts. This is shown in Figure 3.7.

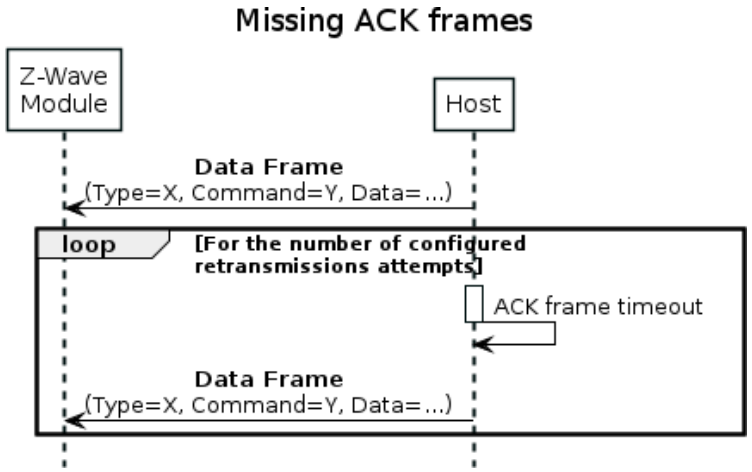

Figure 3.7: Missing acknowledgment frame

In the unlikely event that the Z-Wave Module has been unresponsive for more than 4 seconds (or 3 consecutive *Data Frame* transmission attempts), it is RECOMMENDED to issue a hard reset to the Z-Wave Module. If a hard reset is not available, a *Soft Reset Command* SHOULD be issued.

3.4.3 Collision

If the Z-Wave API module receives a *Data Frame* while it is waiting for an *ACK Frame*, it MUST return a *CAN Frame*.

A host application SHOULD NOT issue any *CAN Frame*, even if it detects a collision. A host application SHOULD initiate a back-off for its own frame, if it was not acknowledged.

When a collision occurs, the Z-Wave API Module SHOULD have priority for retransmission.

Examples are provided in Figure 3.8 and Figure 3.9

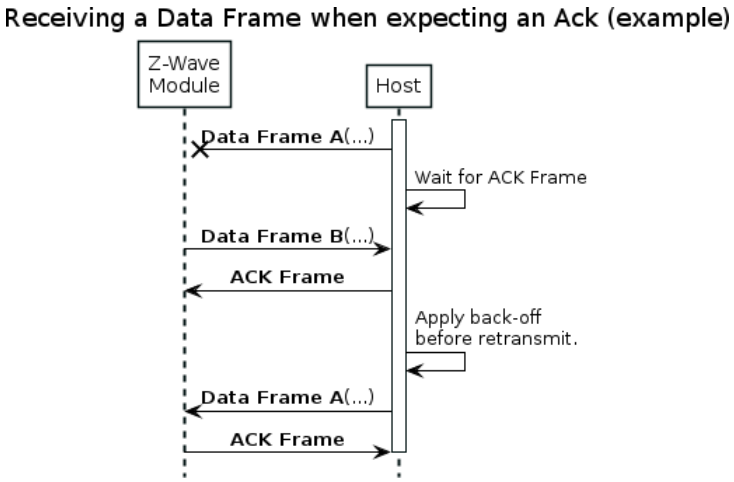

Figure 3.8: Collision detected by the host (example)

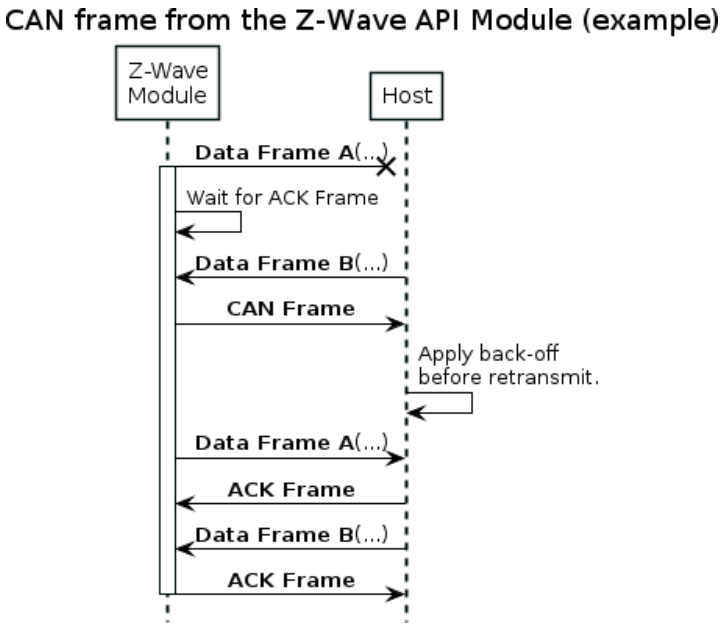

Figure 3.9: CAN frame from the Z-Wave API Module (example)

3.4.4 Frame reception timeout

A receiving interface MUST abort an ongoing reception of a *Data Frame* if the reception has lasted for more than 1500ms after the reception of the SOF byte.

When aborting the reception of a *Data Frame*, an interface MUST NOT issue a *NAK Frame*.

3.4.5 Invalid frame

A receiving interface receiving a *Data Frame* with a *Checksum* mismatch MUST return a *NAK Frame*.

This is illustrated in Figure 3.10

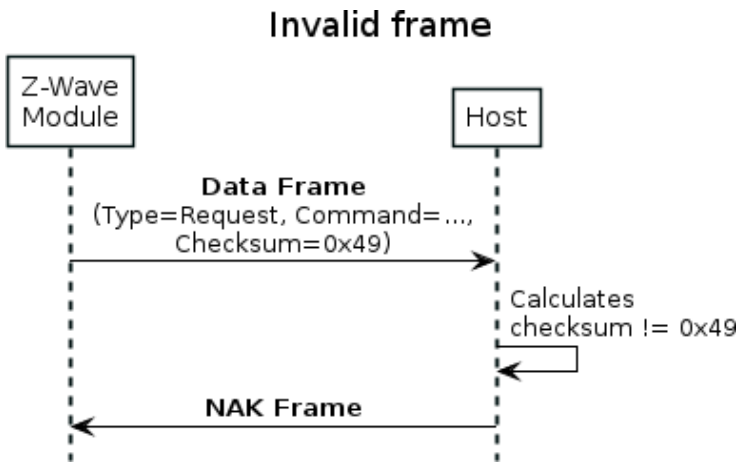

Figure 3.10: Invalid frame

If more than 3 consecutive transmission result in checksum errors, it is RECOMMENDED to issue a hard reset to the Z-Wave Module. If a hard reset is not available, a *Soft Reset Command* SHOULD be issued.

4 Z-Wave API Commands

This section lists all defined Z-Wave API commands. Note that all commands are not always supported by a Z-Wave API Module.

The *Command format* details common features and fields shared among several commands.

The subsequent sections are grouping the Z-Wave API Commands in categories.

- *Z-Wave Capability API commands*:

  This subsection groups all the Z-Wave API commands to read the Z-Wave API Module capabilities and perform initialization and setup.

- *Z-Wave API Network Management Commands*:

  This subsection groups all the Z-Wave API commands allowing to perform Network Management operations. Most of these operations are defined in [zwave_nwk_spec] for details. It is split into 3 subsubsections:

  - Commands available for all nodes: *Common Network Management Commands*
  - Commands for controller nodes only: *Controller Nodes Network Management*
  - Commands for end nodes only: *End Nodes Network Management*

- *Z-Wave API Transport Commands*:

  This subsection groups all the Z-Wave API commands that can be used to transmit application payloads.

- *Z-Wave API Firmware Update Commands*:

  This subsection groups all the Z-Wave API commands that can be used to read and write the firmware of the Z-Wave API module.

- *Z-Wave API Security Commands*:

  This subsection groups all the Z-Wave API commands related to security functionalities provided by the Z-Wave API Module.

- *Z-Wave API Memory Commands*:

  This subsection groups all the Z-Wave API commands that can be used to read data that has been saved by the Z-Wave API Module in its persistent memory.

- *Unsolicited Z-Wave API commands*:

  This subsection groups all the Z-Wave API commands that are sent as unsolicited frames (refer to *Data Frame* and *Unsolicited frame*) by the Z-Wave API Module.

- *Z-Wave API Miscellaneous Commands*:

  This subsection groups all the Z-Wave API commands that do not fit in any of the other categories.

4.1 Command format

In the following *Z-Wave API Commands* section, each command description uses a *Data Frame*, where only the *Z-Wave API Command ID* and *Z-Wave API Command Payload* fields are shown.

4.2 Generic command elements

4.2.1 Session identifier (8 bits)

Some commands contain a Session Identifier. This is a 1 byte value provided by the host application for any given command that must be used by the Z-Wave module in callback commands triggered by the initial data frame.

This Session Identifier can be used by a host application to track which command triggered the incoming callbacks or to identify associated application user data. (e.g. the host needs to retrieve data or perform actions associated to the initial command.)

4.2.2 Rx Status (8 bits)

The Rx Status field is used to indicate how a Z-Wave frame was received. This field MUST be encoded according to Table 4.1

Table 4.1: Rx Status Value encoding

Bit	Flag	Description
0	*Reserved*	This bit is reserved. Reserved bits MUST NOT be used and MUST be ignored by a receiving interface.
1	Low power	This bit indicates if the Z-Wave frame has been received with low output power. • The value 0 MUST indicate that the frame was received with normal output power. • The value 1 MUST indicate that the frame was received with low output power.
2	*Reserved*	This bit is reserved. Reserved bits MUST NOT be used and MUST be ignored by a receiving interface.
3	Broadcast addressing	This bit indicates if the Z-Wave frame has been received using broadcast addressing. • The value 0 MUST indicate that the frame was received using multicast or singlecast addressing. • The value 1 MUST indicate that the frame was received using broadcast addressing.

4	Multicast addressing	This bit indicates if the Z-Wave frame has been received using multicast addressing. • The value 0 MUST indicate that the frame was received using broadcast or singlecast addressing. • The value 1 MUST indicate that the frame was received using multicast addressing.
5	Explore NPDU	This bit indicates if the Z-Wave frame has been received using an Explore NPDU. • The value 0 MUST indicate that the frame was not received using an Explore NDPU. • The value 1 MUST indicate that the frame was received using an Explore NDPU.
6	Foreign frame	This bit indicates if the frame not addressed to the Z-Wave Module. This is useful only in promiscuous mode. • The value 0 MUST indicate that the frame was addressed to the Z-Wave Module. • The value 1 MUST indicate that the frame was not addressed to the Z-Wave Module.
7	Foreign HomeID	This bit indicates if frame was sent on another HomeID. • The value 0 MUST indicate that the frame was sent on the current HomeID. • The value 1 MUST indicate that the frame was sent on the another HomeID.

4.2.3 Tx Status (8 bits)

This field is used to advertise the outcome of a Z-Wave radio transmission attempt. This field MUST be encoded according to Table 4.2

Table 4.2: Tx Status Value encoding

Value	Flag / Description
0x00	*TRANSMIT_COMPLETE_OK* Transmission completed and successful This value is used to indicate that the transmission was successful, and acknowledged if an acknowledged transmission was requested.
0x01	*TRANSMIT_COMPLETE_NO_ACK* Transmission completed but no Acknowledgment This value is used to indicate that the transmission was completed, but no Acknowledgment has been received from the destination.
0x02	*TRANSMIT_COMPLETE_FAIL* Transmission failed. This value is used to indicate that the transmission could not be done.
0x03	*TRANSMIT_ROUTING_NOT_IDLE* Transmission failed due to routing being busy. This value is used to indicate that the transmission could not be done due to routing being locked/busy.
0x04	*TRANSMIT_COMPLETE_NOROUTE* Transmission failed due to routing resolution. This value is used to indicate that the transmission could not be done due to missing route or failed route resolution.
0x05	*TRANSMIT_COMPLETE_VERIFIED* Transmission completed and successful, including S2 resynchronization back-off This value is used to indicate that the transmission was successful, and acknowledged and that the destination has successfully decrypted the message. This status MUST be used only if the Z-Wave module performed Security encryption.

4.2.4 RSSI Measurements (8 bits)

All RSSI measurements MUST use signed representation and MUST be encoded according to Table 4.3

Table 4.3: RSSI value encoding

Hexadecima	Decimal	Description
0x80..0xFF	-128..-1	This value represents the measured RSSI in dBm
0x00..0x7C	0..124	This value represents the measured RSSI in dBm
0x7D	125	The RSSI is below sensitivity and could not be measured.
0x7E	126	The radio receiver is saturated and the RSSI could not be measured.
0x7F	127	The RSSI is not available.

4.2.5 Response status (8 bits)

Some commands Response data frame contain Response status. It is a 1 byte field value that is used indicate if the requested operation in the Initial Data Frame has been accepted and the Callback Data Frame is expected or not. This field MUST be encoded as follow:

- If this field is encoded to 0x00 and the Session Identifier is zero, the Z-Wave Module MUST not send Callback data Frame to the host.
- If this field is encoded different from 0x00 and the the Session Identifier is not zero, the Z-Wave Module MUST send Callback data Frame to the host.

4.2.6 Command Status (8 bits)

When a Z-Wave API Module receives a command, it sometimes provides the execution status of the commands using a *Command Status* field in the Response data frame.

The *Command Status* field MUST be encoded as follow:

- The value 0x00 MUST indicate that the command was not accepted or an error occurred while applying it.
- The value 0x01 MUST indicate that the command was successfully executed.
- Values in the range 0x02..0xFF MUST also be interpreted as a successful command execution.

4.2.7 Basic Device Class (8 bits)

Some commands contain Basic Device Class field which is used to identify the Z-Wave library used by the application for a given device (Refer to [device_type_spec_v2]), and such field MUST be encoded according to Table 4.4.

Table 4.4: Basic Device Class value encoding

Value	Description
0x01	BASIC_TYPE_CONTROLLER Node is a portable controller.
0x02	BASIC_TYPE_STATIC_CONTROLLER Node is a static controller.
0x03	BASIC_TYPE_END_NODE End node.
0x04	BASIC_TYPE_ROUTING_END_NODE End node with routing capabilities
0x05..0xFF	Reserved Reserved values MUST NOT be used.

4.2.8 Tx Options (8 bits)

This field is used to indicate the transmission options for sending Z-Wave frames.

This field MUST be treated as a bit mask and encoded according to Table 4.5

Table 4.5: Tx Options encoding

Bit	Description
0	*MPDU Acknowledgment request* (TRANSMIT_OPTION_ACK) This option is used to request the destination node to return an MPDU acknowledgement. This option SHOULD be used by a host application for all communication. If the destination NodeID is the broadcast NodeID, the Z-Wave Module MUST ignore this bit.
1	*Transmit with low power* (TRANSMIT_OPTION_LOW_POWER) This option is OBSOLETED. This option MUST NOT be used by a sending interface and MUST be ignored by a receiving interface.
2	*Enable Automatic Routing* (TRANSMIT_OPTION_AUTO_ROUTE) This option is used to enable automatic routing. The Z-Wave library runs on the Z-Wave Module will try transmitting the frame via repeater nodes in case destination node is out of direct range. Controller nodes MAY use this bit to enable routing via Last Working Routes, calculated routes and routes discovered via dynamic route resolution. End Nodes MAY use this bit to enable routing via return routes for the actual destination nodeID (if any exist).If the destination is the broadcast NodeID, the Z-Wave Module MUST ignore this option.
3	Reserved This option is reserved.
4	*Disable Routing* (TRANSMIT_OPTION_NO_ROUTE) This option is used to explicitly disable any routing. This option MAY be used to force the Z-Wave Module to send the frame without routing. All available routing information will be ignored. This option SHOULD NOT be specified for normal application communication. If the destination is the broadcast NodeID, the Z-Wave Module MUST ignore this option.
5	*Enable Explore NPDUs* (TRANSMIT_OPTION_EXPLORE) This option is used to enable the usage of Explore NPDUs if needed. The transmit op- tion TRANSMIT_OPTION_EXPLORE MAY be used to enable dynamic route resolution. Dynamic route resolution allows a node to discover new routes if all known routes are failing. An Explore NPDU cannot wake up FLiRS nodes. An Explore NPDU uses normal RF power level minus 6dB. This is also the power level used by a node finding its neighbors. For backwards compatibility reasons, Z-Wave Module SHOULD ignore this option if the destination NodeID does not support Explore NDPUs.
6..7	*Reserved* These options are reserved.

4.2.9 RF Region (8 bits)

This field is used to indicate the Z-Wave RF Region, defining the number of channels and center frequency on which the Z-Wave API Module operates.

This field MUST be encoded according to Table 4.6

Table 4.6: RF Region encoding

Value	Protocol	Description
0	Z-Wave	Region EU: Europe.
1	Z-Wave	Region US: USA.
2	Z-Wave	Region ANZ: Australia/New Zealand.
3	Z-Wave	Region HK: Hong Kong.
5	Z-Wave	Region IN: India.
6	Z-Wave	Region IL: Israel.
7	Z-Wave	Region RU: Russia
8	Z-Wave	Region CN: China
9	Z-Wave Long Range	Region US: USA.
32	Z-Wave	Region JP: Japan.
33	Z-Wave	Region KR: Korea
254	Any	Undefined/unknown region. This value can be used if there was an error retrieving the configured region.
255	Any	Default region. This value is used to indicate that the Z-Wave API module is running on the default region. The default region MUST be the EU Region.

4.2.10 Tx Status Report (N bytes)

When a Z-Wave transmission has been completed, the Z-Wave API Module can issue a *Tx Status Report* providing details about the transmission that was carried out.

The *Tx Status Report* is a variable length field that has grown through the revisions of the Z-Wave API. A host application MUST be resistant to unexpected length of this field (both shorter and longer).

The *Tx Status Report* field MUST be formatted according to Table 4.7

Table 4.7: Tx Status Report field structure

byte\bit	7	6	5	4	3	2	1	0
1	Transmit Ticks (MSB)							
2	Transmit Ticks (LSB)							
3	Number of repeaters							
4	Ack RSSI							
5	Measured incoming RSSI for Repeater 0							
6	Measured incoming RSSI for Repeater 1							
7	Measured incoming RSSI for Repeater 2							
8	Measured incoming RSSI for Repeater 3							
9	ACK Channel No							
10	Tx Channel No							
11	Route Scheme State							
12	Last Route Repeater 0							
13	Last Route Repeater 1							
14	Last Route Repeater 2							
15	Last Route Repeater 3							
16	Reserved	1000ms Beam	250ms Beam	Reserved		Last Route Speed		
17	Routing Attempts							
18	Last route failed link functional NodeID							
19	Last route failed link non-functional NodeID							
20	Tx Power							
21	Measured Noise Floor							
22	Destination Ack MPDU Tx Power							
23	Destination Ack MPDU measured RSSI							
24	Destination Ack MPDU measured Noise floor							

Not all values can be expected to be valid when a tranmsit fails, e.g. all fields related to acknowledge can not be expected to be valid when Tx Status is TRANSMIT_COMPLETE_NO_ACK

Transmit Ticks (16 bits)

This field is used to indicate the transmission time in multiples of 10ms. For example, the value 30 MUST indicate that the transmission took 300ms.

Number of repeaters (8 bits)

This field is used to indicate the number of repeaters used in the route to the destination.

The value 0 MUST indicate direct range communication. Values in the range 1..255 MUST indicate the number of repeaters used to reached the destination.

ACK RSSI (8 bits)

This field is used to indicate ythe RSSI value of the acknowledgement frame. This field MUST be encoded according to *RSSI Measurements (8 bits)* and Table 4.3.

Measured incoming RSSI for Repeater 0 (8 bits)

This field is used to indicate the RSSI value measured from Repeater 0 for the incoming Acknowledgement frame. This field MUST be encoded according to *RSSI Measurements (8 bits)* and Table 4.3.

Measured incoming RSSI for Repeater 1 (8 bits)

This field is used to indicate the RSSI value measured from Repeater 1 for the incoming Acknowledgement frame. This field MUST be encoded according to *RSSI Measurements (8 bits)* and Table 4.3.

Measured incoming RSSI for Repeater 2 (8 bits)

This field is used to indicate the RSSI value measured from Repeater 2 for the incoming Acknowledgement frame. This field MUST be encoded according to *RSSI Measurements (8 bits)* and Table 4.3.

Measured incoming RSSI for Repeater 3 (8 bits)

This field is used to indicate the RSSI value measured from Repeater 3 for the incoming Acknowledgement frame. This field MUST be encoded according to *RSSI Measurements (8 bits)* and Table 4.3.

ACK Channel No (8 bits)

This field is used to indicate the channel number where the ACK received from.

Tx Channel No (8 bits)

This field is used to indicate the channel number that is used to transmit the data.

Route Scheme State (8 bits)

This field is used to indicate the state of the route resolution for the transmission attempt.

The encoding of this field is implementation specific. Refer to individual manufacturer documentation for details.

Last Route Repeater 0 (8 bits)

This field is used to indicate the repeater 0 used in the route to communicate with the destination. The value 0 MUST indicate that no repeater 0 was used for this route.

Last Route Repeater 1 (8 bits)

This field is used to indicate the repeater 1 used in the route to communicate with the destination. The value 0 MUST indicate that no repeater 1 was used for this route.

Last Route Repeater 2 (8 bits)

This field is used to indicate the repeater 2 used in the route to communicate with the destination. The value 0 MUST indicate that no repeater 2 was used for this route.

Last Route Repeater 3 (8 bits)

This field is used to indicate the repeater 3 used in the route to communicate with the destination. The value 0 MUST indicate that no repeater 4 was used for this route.

1000ms Beam (1 bit)

This field is used to indicate if the destination requires a 1000ms beam (or a fragmented beam) to be reached.

250ms Beam (1 bit)

This field is used to indicate if the destination requires a 250ms beam to be reached.

Last Route Speed (3 bits)

This field is used to indicate the transmission speed used in the route to communicate with the destination.

The field MUST be encoded according to Table 4.8.

Table 4.8: Priority Route Data Rate Encoding

Value	Protocol	Description
0x01	Z-Wave	9.6 kbits/s
0x02	Z-Wave	40 kbits/s
0x03	Z-Wave	100 kbits/s
0x04	Z-Wave Long Range	100 kbits/s
0x05..0x07	*Reserved*	*These values are reserved and MUST NOT be used*

Routing Attempts (8 bits)

This field is used to indicate how many routing attempts have been made to transmit the payload to the destination NodeID.

Last route failed link functional NodeID (8 bits)

This field is used when a route failed and it indicates the last functional NodeID in the last used route.

Last route failed link non-functional NodeID (8 bits)

This field is used when a route failed and it indicates the first non-functional NodeID in the last used route.

Tx Power (8 bits)

This field is used to indicate the transmit power used for the transmission. This field MUST be encoded using the signed representation and MUST be expressed in dBm.

Values in the range -127..126 MUST indicate the transmit power. The value 127 MUST indicate that the value is not available.

Measured Noise Floor (8 bits)

This field is used to indicate the measured noise floor during the outgoing transmission. This field MUST be encoded according to *RSSI Measurements (8 bits)* and Table 4.3.

Destination Ack MPDU Tx Power (8 bits)

This field is used to advertise the Tx Power used by the destination in its Ack MPDU frame.

Values in the range -127..126 MUST indicate the transmit power. The value 127 MUST indicate that the value is not available.

Destination Ack MPDU measured RSSI (8 bits)

This field is used to indicate the measured RSSI of the acknowledgement frame received from the destination. This field MUST be encoded according to *RSSI Measurements (8 bits)* and Table 4.3.

Destination Ack MPDU measured Noise floor (8 bits)

This field is used to indicate the measured noise floor by the destination during the MDPU Ack frame transmission. This field MUST be encoded according to *RSSI Measurements (8 bits)* and Table 4.3.

4.2.11 Route Speed (8 bits)

This field is used to advertise the routed packet data rate that shall be used through the return route. The field MUST be encoded according to Table 4.9.

Table 4.9: Priority Route Data Rate Encoding

Value	Description
0x00	*Reserved* This reserved value MUST NOT be used and MUST be ignored by a receiving interface.
0x01	*ZW_PRIORITY_ROUTE_SPEED_9600* This flag indicates that the priority route MUST use a data rate of 9.6 kbits/seconds.
0x02	*ZW_PRIORITY_ROUTE_SPEED_40K* This flag indicates that the priority route MUST use a data rate of 40 kbits/seconds.
0x03	*ZW_PRIORITY_ROUTE_SPEED_100K* This flag indicates that the priority route MUST use a data rate of 100 kbits/seconds.
0x04..0xFF	*Reserved* Reserved values MUST NOT be used and MUST be ignored by a receiving interface

4.2.12 Repeater (4 bytes)

This field is used to indicate the list of repeaters that MUST be used in a return route.

Regardless of the configured NodeID basetype, each of the 4 bytes indicates the NodeID (8 bits each) of a repeater.

Refer to *Z-Wave API Setup Set NodeID Base Type Sub Command* and Table 4.64.

If the route is a direct route, the *Repeater* field MUST be set to 0x00,

4.3 Z-Wave Capability API commands

This section describes *Z-Wave API Commands* that are used to initialize and configure the Z-Wave module. It also comprises commands that are used to read the supported functionality of the Z-Wave API module.

4.3.1 Get Init Data Command

This command is used to request the initialization data and current node list in the network. The Get Init Data Command Identifier is 0x02

4.3.1.1 Frame flow

The frame flow for this command is an *Acknowledged frame with response*.

4.3.1.2 1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.10

Table 4.10: Get Init Data Command

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x02							

4.3.1.3 2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.11

Table 4.11: Get Init Data Command

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x02							
5	Z-Wave API Version							
6	Z-Wave API Capabilities							
7	Z-Wave Node List Length							
7+1	Z-Wave Node List 1							
…	…							
7+N	Z-Wave Node List N							
7+N+1	Chip Type							
7+N+2	Chip Version							

Z-Wave API Version (8 bits)

This field is used to advertise the Z-Wave API version that the Z-Wave Module is currently running.

This field MUST be encoded according to Table 4.12

Table 4.12: Get Init Data Command - Z-Wave API Version encoding

Value	Description
0..9	These values are manufacturer specific. They indicate that the Z-Wave API Module implementation follows a non-standard Z-Wave API specification.
10..255	These values indicate the Z-Wave Alliance host API specification, with an offset of 9. It means that • 10 MUST represent the version 1.0 of this document • 11 MUST represent the version 2.0 of this document • etc.

Z-Wave API Capabilities (8 bits)

This field is used to advertise the capabilities of the Z-Wave API running on the Z-Wave Module. This field MUST be encoded as a bitmask and MUST be according to Table 4.13

Table 4.13: Get Init Data Command - Z-Wave API Capabilities encoding

Bit	Flag	Description
0	End Node API	This bit indicates if the module is an End Node. • The value 0 MUST indicate that the Z-Wave module is an End Node • The value 1 MUST indicate that the Z-Wave module is a Controller Node
1	Timer functions	This bit indicates if the module supports timer functions. • The value 0 MUST indicate that the Z-Wave module does not support timer functions • The value 1 MUST indicate that the Z-Wave module supports timer functions
2	Primary Controller	This bit indicates if the module is Primary Controller in the current network. • The value 0 MUST indicate that the Z-Wave module has the Secondary Controller role in the current network. • The value 1 MUST indicate that the Z-Wave module has the Primary Controller role in the current network.
3	SIS functionality	This bit indicates if the module has the SIS functionality enabled. • The value 0 MUST indicate that the Z-Wave module does not have the SIS functionality enabled. • The value 1 MUST indicate that the Z-Wave module has SIS functionality enabled.
4..7	*Reserved*	These bits are reserved. Reserved bits MUST NOT be used and MUST be ignored by a receiving interface.

Z-Wave Node List Length (8 bits)

This field is used to indicate the length in bytes of the *Z-Wave Node List* field.

End Nodes MUST set this field to 0. Controller Nodes MUST set this field to 29.

Z-Wave Node List (N bytes)

This field is used to advertise the list of nodes present in the current network.

The length of this field, in byte, MUST be according to the *Z-Wave Node List Length* field. This field MUST be omitted if the *Z-Wave Node List Length* field is set to 0.

This field MUST encoded as a bitmask and interpreted as follow:

- bit 0 in byte 7 MUST represent NodeID 1.
- bit 1 in byte 7 MUST represent NodeID 2.
- bit 7 in byte 7 MUST represent NodeID 8.
- bit 0 in byte 8 MUST represent NodeID 9.
- etc.

Chip Type (8 bits)

This field is used to advertise the chip type of the Z-Wave Module. This value SHOULD represent the chip hardware version.

The value of this field is implementation specific. Refer to your manufacturer documentation for details.

Chip Version (8 bits)

This field is used to advertise the chip version of the Z-Wave Module.

The value of this field is implementation specific. Refer to your manufacturer documentation for details.

4.3.1.4 3. Callback data frame (Z-Wave Module → host)

None

4.3.2 Set Application Node Information Command

This command is used to generate the Node Information Frame (NIF) contents and store this information about node capabilities to the Z-Wave module. The host application may initially set up the NIF prior to starting or joining a Z-Wave network.

The Set Application Node Information Command Identifier is 0x03.

4.3.2.1 Frame flow

The frame flow for this command is an *Acknowledged frame*.

4.3.2.2 1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.14

Table 4.14: Set Application Node Information Command

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x03							
5	Device Option Mask							
6	Generic Device Type							
7	Specific Device Type							
8	Command Class List Length							
8+1	Command Class List 1							
…	…							
8+N	Command Class List N							

Device Option Mask (8 bits)

The device option mask is a bitmask where Listening and Optional functionality flags MUST be set accordingly to the nodes capabilities. This field MUST comply with the format indicated in Table 4.15.

Table 4.15: Set Application Node Information Command - Device Option Mask encoding

Bit	Flags	Description
0..6	*Reserved*	*Reserved*
7	Listening flag	This bit indicates if the node should be an Always Listening Node. (AL Node) The value 0 MUST indicate that the node MUST NOT be AL (either NL or FL) The value 1 MUST indicate that the node MUST be an AL node.

Generic Device Type (8 bits)

The Generic Device Class field contains an identifier that identifies what Generic Device Class the Z-

Wave node MUST advertise and MUST be set by the application. For a detailed description of all available Generic Device Classes, refer to [device_class_spec] for Z-Wave devices, [device_type_spec] for Z-Wave Plus devices, and [device_type_spec_v2] for Z-Wave Plus v2 devices.

Specific Device Type (8 bits)

The Specific Device Class field contains an identifier that identifies what Specific Device Class the Z-Wave node MUST advertise and MUST be set by the application. For a detailed description of all available Generic Device Classes, refer to [device_class_spec] for Z-Wave devices, [device_type_spec] for Z-Wave Plus devices, and [device_type_spec_v2] for Z-Wave Plus v2 devices.

Command Class List Length (8 bits)

This field MUST specify the length of the *Command Class List* field in bytes.

Command Class List (N bytes)

This field is used to advertise the list of supported Command Classes by the node. The length of this field MUST be according to the *Command Class List Length* field.

4.3.2.3 2. Response data frame (Z-Wave Module → host)

None

4.3.2.4 3. Callback data frame (Z-Wave Module → host)

None

4.3.3 Set Application Node Information Command Classes Command

This command is used to configure the list of supported Command Classes for each of the following inclusion states:

- Not included in a network
- Included: Non-securely supported
- Included: Securely supported

The Set Application Node Information Command Classes Command Identifier is 0x0C. This command MUST only be supported by a Z-Wave Module that employs End Node libraries (Refer to *Get Library Version Command - Library Type encoding*).

4.3.3.1 Frame flow

The frame flow for this command is an *Acknowledged frame with response*.

4.3.3.2 1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.16.

Table 4.16: Set Application Node Information Command Classes Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x0C							
5	Not Included Node Parameter Length							
6	Not Included Node Parameter 1							
..	…							
6+N	Not Included Node Parameter N							
7+N	Non-securely Included Node Parameter length							
7+N+1	Non-securely Included Node Parameter 1							
..	…							
7+N+M	Non-securely Included Node Parameter M							
8+N+M	Securely Included Node Parameter length							
8+N+M+1	Securely Included Node Parameter 1							
..	…							
8+N+M+G	Securely Included Node Parameter G							

Not Included Node Parameter Length (8 bits)

This field MUST specify the length of the *Not Included Node Parameter* field in bytes.

Not Included Node Parameter (N bytes)

This field is used to advertise the list of supported Command Classes before the node is included in a Z-Wave Network.

Non-securely Included Node Parameter Length (8 bits)

This field MUST specify the length of the *Non-securely Included Node Parameter* field in bytes.

Non-securely Included Node Parameter (M bytes)

This field is used to advertise the list of non-securely supported Command Classes after the node is included in a Z-Wave Network.

Securely Included Node Parameter Length (8 bits)

This field MUST specify the length of the *Securely Included Node Parameter* field in bytes.

Securely Included Node Parameter (G bytes)

This field is used to advertise the list of securely supported Command Classes after the node is included in a Z-Wave Network.

4.3.3.3 2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.17.

Table 4.17: Set Application Node Information Command Classes Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x0C							
5	Command Status							

Command Status (8 bits)

Refer to *Command Status (8 bits)*.

4.3.3.4 3. Callback data frame (Z-Wave Module → host)

None.

4.3.4 Get Controller Capabilities Command

This command is used to request a controller from its current network capabilities.

The Get Controller Capabilities Command Identifier is 0x05

4.3.4.1 Frame flow

The frame flow for this command is an *Acknowledged frame with response*.

4.3.4.2 1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.18

Table 4.18: Get Controller Capabilities Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x05							

4.3.4.3 2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.19

Table 4.19: Get Controller Capabilities Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x05							
5	Z-Wave API Controller Capabilities							

Z-Wave API Controller Capabilities (8 bits)

This field is used to advertise the Controller capabilities in the current network. This field MUST be treated as a bitmask and encoded according to Table 4.20

Table 4.20: Get Controller Capabilities Command - Z-Wave API Controller Capabilities encoding

Bit	Flag	Description
0	Secondary Controller	This bit indicates if the Z-Wave Module has the secondary controller role. • The value 0 MUST indicate that the Z-Wave module has the Primary Controller role. • The value 1 MUST indicate that the Z-Wave module has the Secondar Controller role.
1	Other network	This bit indicates if the module has been included on another network and did not start the current network. • The value 0 MUST indicate that the Z-Wave module has started the current network. • The value 1 MUST indicate that the Z-Wave module has not started the current network.
2	SIS is present	This bit indicates if a SIS is present in the current network. • The value 0 MUST indicate that a SIS is not present in the current network. • The value 1 MUST indicate that a SIS is present in the current network.
3	*Reserved*	This bit should be ignored.
4	SUC enabled	This bit indicates if the module provides the SUC functionality in the current network. • The value 0 MUST indicate that the Z-Wave module does not have the SUC functionality enabled in this network. • The value 1 MUST indicate that the Z-Wave module has the SUC functionality enabled in this network.
5	No nodes included	This bit indicates if the module is the only node in the network. • The value 0 MUST indicate that the Z-Wave module is the only node in the network. • The value 1 MUST indicate that the Z-Wave module is not the only node in the network.
6..7	*Reserved*	These bits are reserved. Reserved bits MUST NOT be used and MUST be ignored by a receiving interface.

4.3.4.4 3. Callback data frame (Z-Wave Module → host)

None

4.3.5 Get Capabilities Command

This command is used to request the API capabilities of a Z-Wave Module. The Get Capabilities Command identifier is 0x07.

4.3.5.1 Frame flow

The frame flow for this command is an *Acknowledged frame with response*.

4.3.5.2 1. Initial data frame (host → Z-Wave Module)

This Command MUST be formatted according to Table 4.21

Table 4.21: Get Capabilities Command

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x07							

4.3.5.3 2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.22

Table 4.22: Get Capabilities Command

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x07							
5	Z-Wave API version							
6	Z-Wave API revision							
7	Z-Wave API manufacturer ID 1 (MSB)							
8	Z-Wave API manufacturer ID 2 (LSB)							
9	Z-Wave API Product Type 1 (MSB)							
10	Z-Wave API Product Type 2 (LSB)							
11	Z-Wave API Product ID 1 (MSB)							
12	Z-Wave API Product ID 2 (LSB)							
12+1	Supported Z-Wave API Commands Bitmask 1							
…	…							
12+N	Supported Z-Wave API Commands Bitmask N							

Z-Wave API version (8 bits)

This field is used to advertise the Z-Wave API application version number.

Z-Wave API revision (8 bits)

This field is used to advertise the Z-Wave API application revision number.

Z-Wave API manufacturer ID (16 bits)

This field is used to define the Manufacturer ID for the Z-Wave Module. Refer to [zwave_manufacturer_ids] for details.

Z-Wave API Product Type (16 bits)

This field is used to advertise the Product Type of the Z-Wave Module. A host application MAY use its own Product Type in the Manufacturer Specific Command Class Refer to the Manufacturer Specific Command Class in [zwave_management_cc_spec]

Z-Wave API Product ID (16 bits)

This field is used to advertise the Product ID of the Z-Wave Module. A host application MAY use its own Product ID in the Manufacturer Specific Command Class Refer to the Manufacturer Specific Command Class in [zwave_management_cc_spec]

Supported Z-Wave API commands bitmask (N bytes)

This field is used to advertise the list of Z-Wave API commands supported by the Z-Wave Module.

This field MUST encoded as a bitmask and interpreted as follow:

- bit 0 in byte 13 MUST represent the *Z-Wave API Command ID* 1.
- bit 1 in byte 13 MUST represent the *Z-Wave API Command ID* 2.
- …
- bit 7 in byte 13 MUST represent the *Z-Wave API Command ID* 8.
- bit 0 in byte 14 MUST represent the *Z-Wave API Command ID* 9.
- etc.

Each of the bits MUST be intepreted as follow:

- A bit set to 1 MUST indicate that the corresponding Z-Wave API Command is supported.
- A bit set to 0 MUST indicate that the corresponding Z-Wave API Command is not supported.

4.3.5.4 3. Callback data frame (Z-Wave Module → host)

None

4.3.6 Get Long Range Nodes Command

This command is used to request the list of Z-Wave Long Range nodes. The Get Long Range Nodes Command Identifier is 0xDA.

There can be up to 4000 nodes a Z-Wave Long Range network. Nodes with NodeIDs smaller or equal to 255 can be retrieved using the *Get Init Data Command*. NodeIDs higher than 255 can be read using this command. The full list of NodeIDs (from 256 to 4000) can be represented using a bitmask of 3745 bits, which can be comprised in 467 bytes.

This amount may too large to be sent in a single command, and an offset mechanism is used to fetch the full list.

4.3.6.1 Frame flow

The frame flow for this command is an *Acknowledged frame with response*. This command may require multiple initial data frames and response data frames in order to read the full list. An example is shown in Figure 4.1

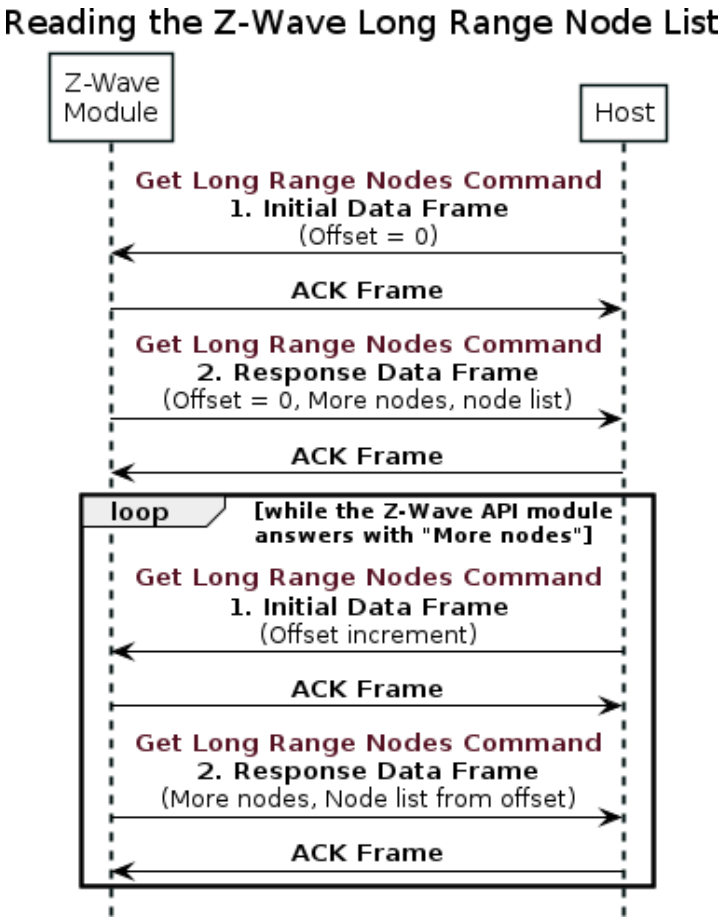

Figure 4.1: Reading the Z-Wave Long Range Node List (Example)

4.3.6.2 1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.23.

Table 4.23: Get Long Range Nodes Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0xDA							
5	Long Range Node List Start Offset							

Long Range Node List Start Offset (8 bits)

This field is used to indicate the number of bytes offset for which the Z-Wave Long Range node list must start from.

A Z-Wave API Module MUST return a Response Data frame with the same Long Range Node List Start Offset.

4.3.6.3 2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.24.

Table 4.24: Get Long Range Nodes Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0xDA							
5	More Nodes							
6	Long Range Node List Start Offset							
7	Long Range Node List Length							
7+1	Long Range Node List 1							
..	. . .							
7+N	Long Range Node List N							

More Nodes (8 bits)

This field is used to indicate if the Z-Wave API Module has advertised the last Z-Wave Long Range NodeID in the current response

This field MUST be set to 0 if the highest Z-Wave Long Range NodeID is advertised in the *Long Range Node List* field. This field MUST be set to 1 if there exist a Z-Wave Long Range node with a higher NodeID than what is advertised in the *Long Range Node List* field.

A host application receiving a response data frame with this field set to 1 SHOULD issue an initial data frame again with an increment of the last *Long Range Node List Start Offset*. Refer to Figure 4.1 for details.

Long Range Node List Start Offset (8 bits)

This field is used to indicate index where the *Long Range Node List* start from.

Each unit in this field's value represent a 128-byte offset. For instance:

- The value 1 represents 1x128 = 128 bytes.
- The value 2 represents 2x128 = 256 bytes.
- The value 3 represents 3x128 = 384 bytes
- etc.

Long Range Node List Length (8 bits)

This field is used to indicate the length in bytes of the *Long Range Node List* field.

Long Range Node List (N bytes)

This field is used to advertise the list of Long Range nodes present in the current network.

The length of this field, in byte, MUST be according to the *Long Range Node List Length* field. This field MUST be omitted if the *Long Range Node List Length* field is set to 0.

This field MUST represent NodeIDs as described in (4.1).

$$N = 255 + 8 \times J + I + 128 \times 8 \times O \quad (4.1)$$

with:

- N: The NodeID being represented.
- I: the current bit I number in the current byte (from 0 to 7).
- J: The current byte (from 0 to *Long Range Node List Length* -1)
- O: the value advertised in the *Long Range Node List Start Offset* field.

For example, with the *Long Range Node List Start Offset* field set to 0:

- Bit 0 in Byte 1 MUST represent NodeID 256
- Bit 1 in Byte 1 MUST represent NodeID 257
- …
- Bit 0 in Byte 2 MUST represent NodeID 264

with the *Long Range Node List Start Offset* field set to 1:

- Bit 0 in Byte 1 MUST represent NodeID 1280
- Bit 1 in Byte 1 MUST represent NodeID 1281
- …
- Bit 0 in Byte 2 MUST represent NodeID 1288

The value 0 for a given NodeID MUST indicate that the node is not present in the network.

The value 1 for a given NodeID MUST indicate that the node is present in the network.

4.3.6.4 3. Callback data frame (Z-Wave Module → host)

None.

4.3.7 Get Z-Wave Long Range Channel Command

This command is used to request which the radio channel is in use for Z-Wave Long Range. The Get Z-Wave Long Range Channel Command Identifier is 0xDB.

4.3.7.1 Frame flow

The frame flow for this command is an *Acknowledged frame with response*.

4.3.7.2 1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.25

Table 4.25: Get Z-Wave Long Range Channel Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0xDB							

4.3.7.3 2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.26

Table 4.26: Get Z-Wave Long Range Channel Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0xDB							
5	Z-Wave Long Range Channel							

Z-Wave Long Range Channel (8 bits)

This field is used to advertise the currently configured Z-Wave Long Range Channel at the Z-Wave API Module.

This field MUST be encoded according to Table 4.27.

Table 4.27: Get Z-Wave Long Range Channel Command - Z-Wave Long Range Channel Encoding

Value	Description
0x00	*Reserved*. This values is reserved, and reserved values MUST not be used.
0x01	Z-Wave Long Range Channel A.
0x02	Z-Wave Long Range Channel B.

0x03..0xFF	*Reserved.* These values are reserved, and reserved values MUST not be used.

4.3.7.4 3. Callback data frame (Z-Wave Module → host)

None

4.3.8 Set Z-Wave Long Range Channel Command

This command is used to configure which radio channel to use for Z-Wave Long Range. The Set Z-Wave Long Range Channel Command Identifier is 0xDC.

4.3.8.1 Frame flow

The frame flow for this command is an *Acknowledged frame*. The execution of this command SHOULD be verified by a host application by issuing a *Get Z-Wave Long Range Channel Command*.

4.3.8.2 1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.28

Table 4.28: Set Z-Wave Long Range Channel Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0xDC							
5	Z-Wave Long Range Channel							

Z-Wave Long Range Channel (8 bits)

This field is used to specify the Z-Wave Long Range Channel that the Z-Wave API Module MUST use.

This field MUST be encoded according to Table 4.29.

Table 4.29: Set Z-Wave Long Range Channel Command - Z-Wave Long Range Channel Encoding

Value	Description
0x00	*Reserved*. This values is reserved, and reserved values MUST not be used.
0x01	Z-Wave Long Range Channel A.
0x02	Z-Wave Long Range Channel B.
0x03..0xFF	*Reserved*. These values are reserved, and reserved values MUST not be used.

4.3.8.3 2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.30

Table 4.30: Set Z-Wave Long Range Channel Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0

4	Z-Wave API Command ID = 0xDC
5	Response status

Response Status (8 bits)

Refer to *Response status (8 bits)*.

4.3.8.4 3. Callback data frame (Z-Wave Module → host)

None

4.3.9 Get Protocol Version Command

This command is used to request the Z-Wave Protocol version data. The Get Protocol Version Command Identifier is 0x09.

4.3.9.1 Frame flow

The frame flow for this command is an *Acknowledged frame with response*.

4.3.9.2 1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.31

Table 4.31: Get Protocol Version Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x09							

4.3.9.3 2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.32.

A host application MUST be resistant to unexpected lengths (too short or too small) for this command.

Table 4.32: Get Protocol Version Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x09							
5	Z-Wave Protocol Type							
6	Z-Wave Protocol Major Version Number							
7	Z-Wave Protocol Minor Version Number							
8	Z-Wave Protocol Revision Version Number							
9	Z-Wave Application Framework Build Number (MSB)							
10	Z-Wave Application Framework Build Number (LSB)							
11	Git Commit hash 1							
...	...							
26	Git Commit hash 16							

Z-Wave Protocol Type (8 bits)

This field is used to indicate the protocol type. This field MUST be encoded according to Table 4.166.

Table 4.33: Z-Wave Get Protocol Version Command - Z-Wave Protocol Type Encoding

Value	Description
0x00	*Z-Wave* Protocol.
0x01	*Z-Wave AV* Protocol. This value SHOULD NOT be used by any Z-Wave API Module.
0x02	*Z-Wave for IP* Protocol This value SHOULD NOT be used by any Z-Wave API Module.
0x03..0xFF	*Reserved*. These options are reserved, and reserved values MUST not be used.

Z-Wave Protocol Major Version Number (8 bits)

This field is used to advertise the Major Version Number for the Z-Wave Protocol. This field MUST be encoded as an unsigned integer.

Z-Wave Protocol Minor Version Number (8 bits)

This field is used to advertise the Minor Version Number for the Z-Wave Protocol. This field MUST be encoded as an unsigned integer.

Z-Wave Protocol Revision Version Number (8 bits)

This field is used to advertise the Revision Version Number for the Z-Wave Protocol. This field MUST be encoded as an unsigned integer.

Z-Wave Application Framework Build Number (16 bits)

This field is used to advertise the Revision Version Number for the Z-Wave Protocol. This field MUST be encoded as an unsigned integer.

The value 0 MUST indicate that this value is not available. Values in the range 1..65535 MUST indicate the build number for the application framework.

Git commit hash (16 bytes)

This field is used to advertise the git commit hash for the Z-Wave Protocol running in the Z-Wave API Module. This field SHOULD be omitted or zeroed out if this information is not available.

4.3.9.4 3. Callback data frame (Z-Wave Module → host)

None.

4.3.10 Get Library Version Command

This command is used to request the Z-Wave library basis version that runs on a Z-Wave Module. The Get Library Command Identifier is 0x15.

4.3.10.1 Frame flow

The frame flow for this command is an *Acknowledged frame with response*.

4.3.10.2 1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.34

Table 4.34: Get Library Version Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x15							

4.3.10.3 2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.35

Table 4.35: Get Library Version Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x15							
5	Z-Wave Library Version 1							
…	…							
17	Z-Wave Library Version 12							
18	Library Type							

Z-Wave Library Version (12 bytes)

This field is used to advertise the Z-Wave API library version that runs on the Z-Wave Module using the following text format:

- Z-Wave x.y, where x and y are the major and minor library versions, respectively.

Library Type (8 bits)

This field is used to advertise the library type that runs on the Z-Wave Module.

This field MUST encoded according to Table 4.36

Table 4.36: Get Library Version Command - Library Type encoding

Value	Networking type	Library Type
0x01	Controller node	*Static Controller library* This library is intended for main home controllers, that are typically Primary controllers in a network.
0x02	Controller node	*Portable Controller library* This library is intended for small portable controllers, that are typically secondary controllers or inclusion controllers in a network.
0x03	End node	*Enhanced 232 End Node Library* This library is intended for end nodes.
0x04	End node	*End Node library* This library is intended for end nodes with more limited capabilities than the *Enhanced 232 End Node Library*. New implementations SHOULD use the *Enhanced 232 End Node Library*
0x05	Controller node	*Installer library* This library is intended for controllers nodes used for setup and monitoring of existing networks.
0x06	End node	*Routing End Node library* This library is intended for end nodes with routing capabilities. New implementations SHOULD use the *Enhanced 232 End Node Library*
0x07	Controller node	*Bridge controller library* This library is intended for controller nodes that are able to allocate more than 1 NodeID to themselves and use them for transmitting/receiving frames.
0x08..0xFF	Reserved	*Reserved* Reserved values MUST NOT be used and MUST be ignored by a receiving interface

4.3.10.4 3. Callback data frame (Z-Wave Module → host)

None

4.3.11 Get Library Command

This command is used to is used to request the Z-Wave library type that runs on a Z-Wave Module. The Get Library Command Identifier is 0xBD.

4.3.11.1 Frame flow

The frame flow for this command is an *Acknowledged frame with response*.

4.3.11.2 1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.37

Table 4.37: Get Library Type Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0xBD							

4.3.11.3 2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.38

Table 4.38: Get Library Type Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0xBD							
5	Library Type							

Library Type (8 bits)

This field is used to advertise the library type that runs on the Z-Wave Module.

This field MUST encoded according to Table 4.36

4.3.11.4 3. Callback data frame (Z-Wave Module → host)

None

4.3.12 Soft Reset Command

This command is used to request the Z-Wave Module to perform a soft reset. The Soft Reset Command Identifier is 0x08.

4.3.12.1 Frame flow

The frame flow for this command is an *Acknowledged frame*.

4.3.12.2 1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.39

Table 4.39: Soft Reset Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x08							

4.3.12.3 2. Response data frame (Z-Wave Module → host)

None

4.3.12.4 3. Callback data frame (Z-Wave Module → host)

None.

Note: A Z-Wave Module SHOULD issue a *Z-Wave API Started Command* when it has completed the reset operation.

4.3.13 Set Default Command

This command is used to set the Z-Wave API Module to its default state. It means that the Z-Wave API Module will leave its current network and erase all information related to its current Z-Wave network (topology, network keys, HomeID, etc.).

The Set Default Command Identifier is 0x42.

4.3.13.1 Frame flow

The frame flow for this command is an *Acknowledged frame with callback*.

4.3.13.2 1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.40.

Table 4.40: Set Default Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x42							
5	Session identifier							

Session identifier (8 bits)

Refer to *Session identifier (8 bits)*.

4.3.13.3 2. Response data frame (Z-Wave Module → host)

None.

4.3.13.4 3. Callback data frame (Z-Wave Module → host)

A Z-Wave module MUST issue a callback frame formatted according to Table 4.41 after the Z-Wave API Module has completed the set default operation.

Table 4.41: Set Default Command - Callback data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x42							
5	Session identifier							

Session identifier (8 bits)

Refer to *Session identifier (8 bits)*.

4.3.14 Setup Z-Wave API Command

This command is used to request and configure the Z-Wave Module and its API. The Setup Z-Wave API Command Identifier is 0x0B.

This command contains sub-commands.

4.3.14.1 Frame flow

The frame flow for this command is an *Acknowledged frame with response*.

4.3.14.2 1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.42.

Table 4.42: Setup Z-Wave API Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x0B							
5	Sub Command							
5+1	Sub Command Payload 1							
..	. . .							
5+N	Sub Command Payload N							

Sub Command (8 bits)

This field is used to advertise the Z-Wave API Setup Sub Command. The list of available Sub Commands are available in *Z-Wave API Setup sub-commands*.

Sub Command Payload (N bytes)

This field is used to indicate the data payload that corresponds to a given Z-Wave API setup Sub Command defined in Command field.

Each Sub Command payload MUST be interpreted in conjunction with the actual Sub Command. refer to *Z-Wave API Setup sub-commands*.

4.3.14.3 2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.43.

Table 4.43: Setup Z-Wave API Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x0B							
5	Sub Command							
5+1	Sub Command Payload 1							
..	. . .							

5+N	Sub Command Payload N

Command (8 bits)

This field is used to advertise the Z-Wave API Setup Sub Command. The list of available Sub Commands are available in *Z-Wave API Setup sub-commands*.

A Z-Wave API module that has received a non-supported Z-Wave API Setup Sub Command MUST return the value 0 in this field. Refer to *Z-Wave API Setup Get Supported Commands Sub Command* for the list of supported Z-Wave API setup commands.

The value 0 MUST indicate that the received Z-Wave API setup sub command in the Initial data frame is not supported. If this field is set to 0, the Sub Command Payload field MUST be 1 byte long and MUST be set to the unsupported Z-Wave API Setup Sub Command received in the Initial data frame

Sub Command Payload (N bytes)

This field is used to indicate the data payload that corresponds to a given Z-Wave API setup Sub Command defined in Command field.

Each Sub Command payload MUST be interpreted in conjunction with the actual Sub Command. refer to *Z-Wave API Setup sub-commands*.

4.3.14.4 3. Callback data frame (Z-Wave Module → host)

None.

4.3.15 Z-Wave API Setup sub-commands

This section describes subcommands of the *Setup Z-Wave API Command* that are used to configure the Z-Wave module.

4.3.15.1 Z-Wave API Setup Get Supported Commands Sub Command

This command is used to request the list of Z-Wave API Setup Sub Commands that are supported by the Z-Wave API Module.

The Z-Wave API Setup Get Supported Commands Sub Command Identifier is 0x01

Frame flow

The frame flow for this command is an *Acknowledged frame with response*.

1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.44

Table 4.44: Z-Wave API Setup Get Supported Commands Sub Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x0B							
5	Sub Command = 0x01							

Sub Command (8 bits)

This field MUST be set to 0x01 to indicate the *Z-Wave API Setup Get Supported Commands Sub Command*.

2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.45

Table 4.45: Z-Wave API Setup Get Supported Commands Sub Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x0B							
5	Sub Command = 0x01							
6	Z-Wave API Setup Supported Sub Commands flags							
6+1	Extended Z-Wave API Setup Supported Sub Commands bitmask 1							
…	…							
6+N	Extended Z-Wave API Setup Supported Sub Commands bitmask N							

Sub Command (8 bits)

This field MUST be set to 0x01 to indicate the *Z-Wave API Setup Get Supported Commands Sub Command*.

Z-Wave API Setup Supported Sub Commands flags (8 bits)

This field is used to indicate the list of supported Z-Wave API setup Sub Commands by the Z-Wave Module.

This field can only advertise support for functions that have identifiers that are powers of 2.

This field MUST be encoded as a bitmask and according to Table 4.46

- A bit set to 0 MUST indicate that the corresponding Z-Wave API Setup Sub Command is not supported.
- A bit set to 1 MUST indicate that the corresponding Z-Wave API Setup Sub Command is supported

Table 4.46: Z-Wave API Setup Get Supported Commands - Supported Sub Commands encoding

Bit number	Description
0	This bit represents the *Z-Wave API Setup Get Supported Commands Sub Command*.
1	This bit represents the *Z-Wave API Setup Set Tx Status Report Sub Command*.
2	This bit represents the *Z-Wave API Setup Set Powerlevel Sub Command*.
3	This bit represents the *Z-Wave API Setup Get Powerlevel Sub Command*.
4	This bit represents the *Z-Wave API Setup Get Maximum Payload Size Sub Command*.
5	This bit represents the *Z-Wave API Setup Get RF Region Sub Command*.
6	This bit represents the *Z-Wave API Setup Set RF Region Sub Command*.
7	This bit represents the *Z-Wave API Setup Set NodeID Base Type Sub Command*.

Extended Z-Wave API Setup Supported Sub Commands bitmask (N bytes)

This field is used to advertise the list of supported Sub Commands.

If this field is not present in the response data frame sent by a Z-Wave API Module, a host applica- tion MUST assume that only the sub commands advertised in the *Z-Wave API Setup Supported Sub Commands flags* field are supported.

This field MUST be treated as a bitmask and MUST be encoded as follow:

- Bit 0 in Byte 1 MUST represent Sub Command Identifier 1.
- Bit 1 in Byte 1 MUST represent Sub Command Identifier 2.
- Bit 2 in Byte 1 MUST represent Sub Command Identifier 3.
- . . .
- Bit 7 in Byte 1 MUST represent Sub Command Identifier 8.

- Bit 0 in Byte 2 MUST represent Sub Command Identifier 9.
- Bit 1 in Byte 2 MUST represent Sub Command Identifier 10.
- Bit 2 in Byte 2 MUST represent Sub Command Identifier 11.
- …

The list of supported Commands in the *Z-Wave API Setup Supported Sub Commands bitmask* field MUST also be advertised as supported in this field.

The length of this field MUST be set to at least the minimum length that allows to advertise all supported Z-Wave API Setup Sub Commands. The length of this field can be calculated from the total length of the response data frame.

3. Callback data frame (Z-Wave Module → host)

None

4.3.15.2 Z-Wave API Setup Set Tx Status Report Sub Command

This command is used to configure the Z-Wave API Module to return detailed Tx Status Report after sending a frame to a destination.

The Z-Wave API Setup Set Tx Status Report Sub Command Identifier is 0x02

Frame flow

The frame flow for this command is an *Acknowledged frame with response*.

1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.47

Table 4.47: Z-Wave API Setup Set Tx Status Report Sub Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x0B							
5	Sub Command = 0x02							
6	Enable Tx Status Report							

Sub Command (8 bits)

This field MUST be set to 0x02 to indicate the *Z-Wave API Setup Set Tx Status Report Sub Command* Command.

Enable Tx Status Report (8 bits)

This field is used to indicate if the Tx Status Report MUST be enabled.

- The value 0x00 MUST indicate that the Tx Status Report MUST NOT be enabled.
- All other values MUST indicate that the Tx Status Report MUST be enabled.

2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.48

Table 4.48: Z-Wave API Setup Set Tx Status Report Sub Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x0B							
5	Sub Command = 0x02							
6	Command Status							

Sub Command (8 bits)

This field MUST be set to 0x02 to indicate the *Z-Wave API Setup Set Tx Status Report Sub Command*.

Command Status (8 bits)

This field is used to indicate if the setting indicated in the inital data frame was accepted and applied.

This field MUST be encoded according to *Command Status (8 bits)*

3. Callback data frame (Z-Wave Module → host)

None

4.3.15.3 Z-Wave API Setup Set Powerlevel Sub Command

This command is used to configure the Tx Powerlevel setting of the Z-Wave API.

The power levels set by this function will first be used by the Z-Wave protocol next time the module is restarted.

The Z-Wave API Setup Set Powerlevel Sub Command Identifier is 0x04

Frame flow

The frame flow for this command is an *Acknowledged frame with response*.

1. Initial data frame (host → Z-Wave Module)

Z-Wave API version 7 and above: Table 4.21

The initial data frame MUST be formatted according to Table 4.49

Table 4.49: Z-Wave API Setup Set Powerlevel Sub Command (v7+) - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x0B							
5	Sub Command = 0x04							
6	Normal Powerlevel Setting							
7	Measured 0dBm Powerlevel Setting							

Z-Wave API version 6 and below: Table 4.21

The initial data frame MUST be formatted according to Table 4.50

Table 4.50: Z-Wave API Setup Set Powerlevel Sub Command (v6-) - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x0B							
5	Sub Command = 0x04							
6	NormalPowerCh0							
7	NormalPowerCh1							
8	NormalPowerCh2							
9	LowPowerCh0							
10	LowPowerCh1							
11	LowPowerCh2							

Sub Command (8 bits)

This field MUST be set to 0x04 to indicate the *Z-Wave API Setup Set Powerlevel Sub Command*.

Normal Powerlevel Setting (8 bits)

This field is used to indicate the requested transmit powerlevel for transmitting Z-Wave Frames.

This field MUST be expressed in deci dBm and MUST use signed encoding.

For example:

- The value 10 MUST represent 1 dBm
- The value -20 MUST represent -2 dBm

Measured 0dBm Powerlevel Setting (8 bits)

This field is used to indicate the output power measured from the antenna when the *Normal Powerlevel Setting* field is set to 0.

This field MUST be expressed in deci dBm and MUST use signed encoding.

For example:

- The value 10 MUST represent 1 dBm
- The value -20 MUST represent -2 dBm

NormalPowerChx (8 bits)

The power level used when transmitting frames at normal Tx power. This value is vendor specific and should be provided by your stack vendor.

LowPowerCh0 (8 bits)

The power level used when transmitting frames at low Tx power

This field MUST be formatted according to Table 4.51

Table 4.51: Z-Wave API Setup Set Powerlevel Sub Command - Low Tx power encoding

Value	Description
0x3F	NormalPower
0x24	NormalPower - 2dB
0x1E	NormalPower - 4dB
0x16	NormalPower - 6dB
0x11	NormalPower - 8dB
0x0E	NormalPower - 10dB
0x0B	NormalPower - 12dB
0x09	NormalPower - 14dB
0x07	NormalPower - 16dB
0x05	NormalPower - 18dB
0x04	NormalPower - 20dB
0x03	NormalPower - 22dB

2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.52

Table 4.52: Z-Wave API Setup Set Powerlevel Sub Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x0B							
5	Sub Command = 0x04							
6	Command Status							

Sub Command (8 bits)

This field MUST be set to 0x04 to indicate the *Z-Wave API Setup Set Powerlevel Sub Command*.

Command Status (8 bits)

This field is used to indicate if the setting indicated in the inital data frame was accepted and applied.

This field MUST be encoded according to *Command Status (8 bits)*

3. Callback data frame (Z-Wave Module → host)

None

4.3.15.4 Z-Wave API Setup Get Powerlevel Sub Command

This command is used to request the Powerlevel setting of the Z-Wave API.

The Z-Wave API Setup Get Powerlevel Sub Command Identifier is 0x08

Frame flow

The frame flow for this command is an *Acknowledged frame with response*.

1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.53

Table 4.53: Z-Wave API Setup Get Powerlevel Sub Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x0B							
5	Sub Command = 0x08							

Sub Command (8 bits)

This field MUST be set to 0x08 to indicate the *Z-Wave API Setup Get Powerlevel Sub Command*.

2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.54

Table 4.54: Z-Wave API Setup Get Powerlevel Sub Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x0B							
5	Sub Command = 0x08							
6	Normal Powerlevel Setting							
7	Measured 0dBm Powerlevel Setting							

Sub Command (8 bits)

This field MUST be set to 0x08 to indicate the *Z-Wave API Setup Get Powerlevel Sub Command*.

Normal Powerlevel Setting (8 bits)

This field is used to advertise the currently configured transmit powerlevel for transmitting Z-Wave Frames.

This field MUST be expressed in deci dBm and MUST use signed encoding.

For example:

- The value 10 MUST represent 1 dBm
- The value -20 MUST represent -2 dBm

Measured 0dBm Powerlevel Setting (8 bits)

This field is used to indicate the configured output power measured from the antenna when the *Normal Powerlevel Setting* field is set to 0.

This field MUST be expressed in deci dBm and MUST use signed encoding.

For example:

- The value 10 MUST represent 1 dBm
- The value -20 MUST represent -2 dBm

3. Callback data frame (Z-Wave Module → host)

None

4.3.15.5 Z-Wave API Setup Get Maximum Payload Size Sub Command

This command is used to request the maximum payload that the Z-Wave API Module can accept for transmitting Z-Wave frames. This value depends on the RF Profile.

The Z-Wave API Setup Get Maximum Payload Size Sub Command Identifier is 0x10

Frame flow

The frame flow for this command is an *Acknowledged frame with response*.

1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.55

Table 4.55: Z-Wave API Setup Get Maximum Payload Size Sub Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x0B							
5	Sub Command = 0x10							

Sub Command (8 bits)

This field MUST be set to 0x10 to indicate the *Z-Wave API Setup Get Maximum Payload Size Sub Command*.

2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.56

Table 4.56: Z-Wave API Setup Get Maximum Payload Size Sub Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x0B							
5	Sub Command = 0x10							
6	Maximum Payload Size							

Sub Command (8 bits)

This field MUST be set to 0x10 to indicate the *Z-Wave API Setup Get Maximum Payload Size Sub Command*.

Maximum Payload Size (8 bits)

This field is used to advertise the Maximum Payload Size, in bytes, supported by the Z-Wave API

Module for sending frames.

Calls to Send Data functions will be ignored if the data length is longer than the value advertised in this field.

3. Callback data frame (Z-Wave Module → host)

None

4.3.15.6 Z-Wave API Setup Get Z-Wave Long Range Maximum Payload Size Sub Command

This command is used to request the maximum payload that the Z-Wave API Module can accept for transmitting Z-Wave Long Range frames.

Z-Wave API Setup Get Z-Wave Long Range Maximum Payload Size Sub Command Identifier is 0x11

Frame flow

The frame flow for this command is an *Acknowledged frame with response*.

1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.57

Table 4.57: Z-Wave API Setup Get Z-Wave Long Range Maximum Payload Size Sub Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x0B							
5	Sub Command = 0x11							

Sub Command (8 bits)

This field MUST be set to 0x11 to indicate the *Z-Wave API Setup Get Z-Wave Long Range Maximum Payload Size Sub Command*.

2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.58

Table 4.58: Z-Wave API Setup Get Z-Wave Long Range Maximum Payload Size Sub Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x0B							
5	Sub Command = 0x11							
6	Z-Wave Long Range Maximum Payload Size							

Sub Command (8 bits)

This field MUST be set to 0x11 to indicate the *Z-Wave API Setup Get Maximum Payload Size Sub Command*.

Z-Wave Long Range Maximum Payload Size (8 bits)

This field is used to advertise the Maximum Payload Size, in bytes, supported by the Z-Wave API Module for sending frames using the Z-Wave Long Range protocol.

Calls to Send Data functions to Z-Wave Long Range NodeID destionations will be ignored if the data length is longer than the value advertised in this field.

3. Callback data frame (Z-Wave Module → host)

None

4.3.15.7 Z-Wave API Setup Get RF Region Sub Command

This command is used to request the current RF region configured at the Z-Wave API Module.

The Z-Wave API Setup Get RF Region Sub Command Identifier is 0x20

Frame flow

The frame flow for this command is an *Acknowledged frame with response*.

1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.59

Table 4.59: Z-Wave API Setup Get RF Region Sub Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x0B							
5	Sub Command = 0x20							

Sub Command (8 bits)

This field MUST be set to 0x20 to indicate the *Z-Wave API Setup Get RF Region Sub Command*.

2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.60

Table 4.60: Z-Wave API Setup Get RF Region Sub Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x0B							
5	Sub Command = 0x20							
6	RF Region							

Sub Command (8 bits)

This field MUST be set to 0x20 to indicate the *Z-Wave API Setup Get RF Region Sub Command*.

RF Region (8 bits)

This field is used to indicate the current RF Region setting.

This field MUST be encoded according to *RF Region (8 bits)* and Table 4.6

3. Callback data frame (Z-Wave Module → host)

None

4.3.15.8 Z-Wave API Setup Set RF Region Sub Command

This command is used to configure the RF region at the Z-Wave API Module.

The Z-Wave API Setup Set RF Region Sub Command Identifier is 0x40

Frame flow

The frame flow for this command is an *Acknowledged frame with response*.

1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.61

Table 4.61: Z-Wave API Setup Get RF Region Sub Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x0B							
5	Sub Command = 0x40							
6	RF Region							

Sub Command (8 bits)

This field MUST be set to 0x40 to indicate the *Z-Wave API Setup Set RF Region Sub Command*.

RF Region (8 bits)

This field is used to indicate the current RF Region setting.

This field MUST be encoded according to *RF Region (8 bits)* and Table 4.6

Note: The RF Region value will be in used by the Z-Wave API Module only after it restarted. A host application SHOULD issue a *Soft Reset Command* after configuring the RF Region.

---

2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.62

Table 4.62: Z-Wave API Setup Set RF Region Sub Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x0B							
5	Sub Command = 0x40							
6	Command Status							

Sub Command (8 bits)

This field MUST be set to 0x40 to indicate the *Z-Wave API Setup Set RF Region Sub Command*.

Command Status (8 bits)

This field is used to indicate if the RF Region setting indicated in the inital data frame was accepted.

This field MUST be encoded according to *Command Status (8 bits)*.

Note: The RF Region value will be in used by the Z-Wave API Module only after it restarted. A host application SHOULD issue a *Soft Reset Command* after configuring the RF Region.

---

3. Callback data frame (Z-Wave Module → host)

None

4.3.15.9 Z-Wave API Setup Set NodeID Base Type Sub Command

This command is used to configure the NodeID base type for the Z-Wave API.

The Z-Wave API Setup Set NodeID Base Type Sub Command Identifier is 0x80.

All Z-Wave API Commands MUST use the length defined in this Sub Command for encoding NodeID fields. The default NodeID field length MUST be 8 bits.

Frame flow

The frame flow for this command is an *Acknowledged frame with response*.

1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.63

Table 4.63: Z-Wave API Setup Set NodeID Base Type Sub Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x0B							
5	Sub Command = 0x80							
6	NodeID Base Type							

Sub Command (8 bits)

This field is used to indicate the Sub Command within the Z-Wave API Setup Command.

This field MUST be set to 0x80 for the *Z-Wave API Setup Set NodeID Base Type Sub Command*.

NodeID Base Type (8 bits)

This field is used to indicate the desired base type for NodeID fields.

This field MUST be encoded according to Table 4.64. All other values are reserved.

Table 4.64: Z-Wave API Setup Set NodeID Base Type Sub Command - NodeID Base Type encoding

Value	Description
0x01	NodeID fields MUST be 8-bit long.
0x02	NodeID fields MUST be 16-bit long.

2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.65

Table 4.65: Z-Wave API Setup Set NodeID Base Type Sub Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x0B							
5	Sub Command = 0x80							
6	Command Status							

Sub Command (8 bits)

This field is used to indicate the Sub Command within the Z-Wave API Setup Command.

This field MUST be set to 0x80 for the *Z-Wave API Setup Set NodeID Base Type Sub Command*.

Command Status (8 bits)

This field is used to indicate if the setting indicated in the inital data frame was accepted and applied.

This field MUST be encoded according to *Command Status (8 bits)*

- The value 0 MUST indicate that the requested NodeID Base Type in the initial data frame was not accepted or an error occurred. The NodeID Base Type was not applied and is set to the default length. (8-bits)
- The values 1..255 MUST indicate that the requested NodeID Base Type in the initial data frame was accepted and applied successfully.

3. Callback data frame (Z-Wave Module → host)

None

4.4 Z-Wave API Network Management Commands

This section describes *Z-Wave API Commands* that are used to perform Z-Wave Network Management.

4.4.1 Common Network Management Commands

This section describes *Z-Wave API Commands* that are used to perform Z-Wave Network Management for any nodes (both controller nodes and end nodes).

The commands described in this subsection MUST be supported by all Z-Wave API modules.

4.4.1.1 Send NOP Command

This command is used to send NOP Commands a destination to verify if it is responsive. This command SHOULD NOT be used by a host application for NL Nodes outside their Wake Up period. Refer to the [zwave_nwk_spec] for details. The Send NOP Command Identifier is 0xE9.

Frame flow

The frame flow for this command is an *Acknowledged frame with response and callback*.

1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.66

Table 4.66: Send NOP Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0xE9							
5/5..6	Destination NodeID							
6/7	Tx Options							
8/9	Session identifier							

Destination NodeID (8 bits/16 bits)

This field is used to indicate the destination NodeID to send the Z-Wave Frame to.

This field MUST be encoded according to the configured NodeID base Type. Refer to *Z-Wave API Setup Set NodeID Base Type Sub Command* and Table 4.64.

Tx Options (8 bits)

Refer to *Tx Options (8 bits)*.

Session Identifier (8 bits)

Refer to *Session identifier (8 bits)*.

2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.67

Table 4.67: Send NOP Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0xE9							
5	Response status							

Response status (8 bits)

Refer to *Response status (8 bits)*.

3. Callback data frame (Z-Wave Module → host)

A Z-Wave module MUST issue a callback frame formatted according to Table 4.68

Table 4.68: Send NOP Command - Callback data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0xE9							
5	Session Identifier							
6	Tx Status							
7	Tx Status Report 1							
…	…							
7+N	Tx Status Report N							

Session Identifier (8 bits)

Refer to *Session identifier (8 bits)*.

Tx Status (8 bits)

Refer to *Tx Status (8 bits)*.

Tx Status Report (N bytes)

This field is used to report detailed information about the Z-Wave frame transmission. This field MUST be omitted if the Z-Wave API module is not configured to enable Tx Status Reports in the *Z-Wave API Setup Set Tx Status Report Sub Command*.

For field description, refer to *Tx Status Report (N bytes)*.

4.4.1.2 Get Node Information Protocol Data Command

This command is used to request the Node Information protocol data about a NodeID to the Z-Wave API Module. The Get Node Information Protocol Data Command Identifier is 0x41.

Frame flow

The frame flow for this command is an *Acknowledged frame with response*.

1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.69

Table 4.69: Get Node Information Protocol Data Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x41							
5/5..6	NodeID							

NodeID (8/16 bits)

This field is used to indicate the NodeID for which the Node Information protocol data is requested.

This field MUST be encoded according to the configured NodeID base Type. Refer to *Z-Wave API Setup Set NodeID Base Type Sub Command* and Table 4.64.

2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.70

Table 4.70: Get Node Information Protocol Data Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x41							
5	Listening	Routing	Supported speed			Protocol version		
6	Optional Functionality	Sensor 1000ms	Sensor 250ms	Beam Capability	Routing End Node	Specific Device	Controller node	Security
7	Reserved					Speed Extension		
8	Basic Device Type							
9	Generic Device Class							
10	Specific Device Class							

Fields values in this command MUST be according to the *Node Information Frame Command* received

by the NodeID being advertised. Refer to the [zwave_nwk_spec] for details.

If the NodeID requested in the Initial Data Frame is not part of the Network or is unknown to the Z-Wave Module, the *Generic Device Class* field MUST be set to 0. All other fields SHOULD also be set to 0 and ignored by a receiving interface.

3. Callback data frame (Z-Wave Module → host)

None

4.4.1.3 Send Node Information Command

This command is used to trigger a transmission of Node Information Frame. The Send Node Information Command Command Identifier is 0x12.

Frame flow

The frame flow for this command is an *Acknowledged frame with callback*.

1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.71.

Table 4.71: Send Node Information Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x12							
5/6	Destination NodeID							
7/8	Tx Option							
8/9	Session identifier							

Destination NodeID (8/16 bits)

This field is used to indicate the destination NodeID of the node where the Node Information Frame is sent to.

Tx Option (8 bits)

Refer to *Tx Options (8 bits)*.

Session identifer (8 bits)

Refer to *Session identifier (8 bits)*.

2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.72.

Table 4.72: Send Node Information Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x12							
5	Response status							

Response status (8 bits)

Refer to *Response status (8 bits)*.

3. Callback data frame (Z-Wave Module → host)

A Z-Wave module MUST issue a callback frame formatted according to Table 4.73 after the module transmit the node information frame to target node.

Table 4.73: Send Node Information Command - Callback data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x12							
5	Session identifier							
6	Tx Status							

Session identifier(8 bits)

Refer to *Session identifier (8 bits)*.

Tx Status (8 bits)

Refer to *Rx Status (8 bits)*.

4.4.1.4 Request Node Information Command

This command is used to request the Node Information Frame from a Z-Wave Node. The Request Node Information Command Identifier is 0x60.

Frame flow

The frame flow for this command is an *Acknowledged frame with response*.

This command will trigger additional unsolicited frames from the Z-Wave API Module. Examples of the expected frame flows are shown in Figure 4.2 and Figure 4.3

If the Z-Wave API Module receives the requested *Node Information Frame Command*, it MUST issue an unsolicited *Application Update Command* with the status set to UPDATE_STATE_NODE_INFO_RECEIVED.

If the Z-Wave API Module does not receive the requested *Node Information Frame Com- mand*, it MUST issue an unsolicited *Application Update Command* with the status set to UPDATE_STATE_NODE_INFO_REQ_FAILED.

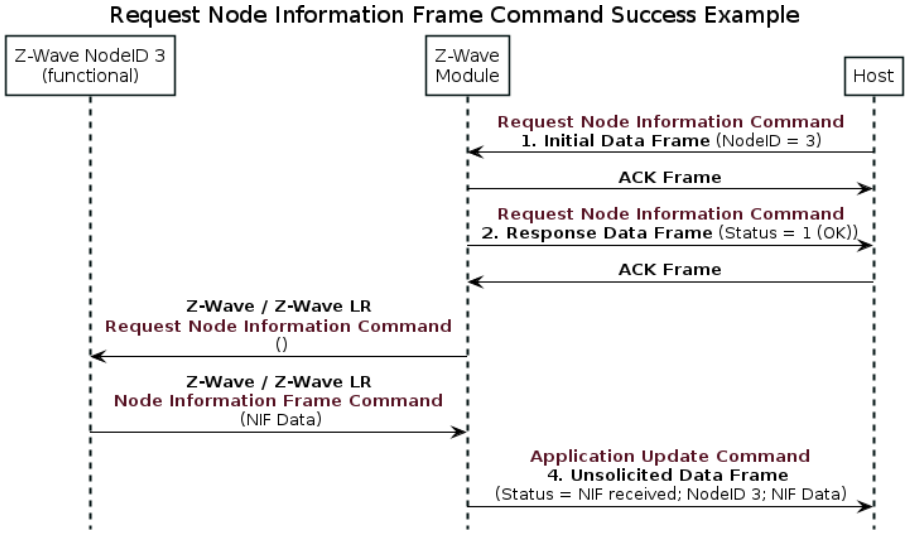

Figure 4.2: Request Node Information Command Success Example

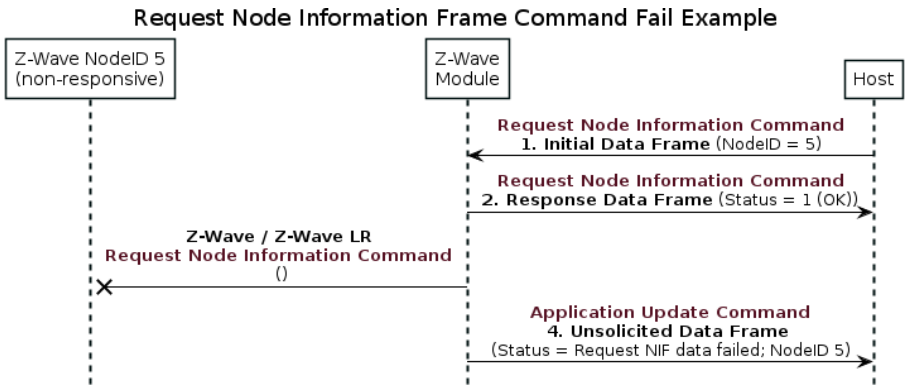

Figure 4.3: Request Node Information Command Fail Example

1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.74.

Table 4.74: Request Node Information Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x60							
5/6	NodeID							

NodeID (8/16 bits)

This field is used to indicate the NodeID for which the *Node Information Frame* must be requested.

This field MUST be encoded according to the configured NodeID base Type. Refer to *Z-Wave API Setup Set NodeID Base Type Sub Command* and Table 4.64.

2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.75.

Table 4.75: Request Node Information Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x60							
5	Command Status							

Command Status (8 bits)

Refer to *Command Status (8 bits)*.

3. Callback data frame (Z-Wave Module → host)

None.

4.4.1.5 Set Learn Mode Command

This command is used to start or stop Learn Mode The Set Learn Mode Command Identifier is 0x50.

Frame flow

The frame flow for this command is an *Acknowledged frame with response and callback*.

This command will trigger several callback frames. This is illustrated in Figure 4.4

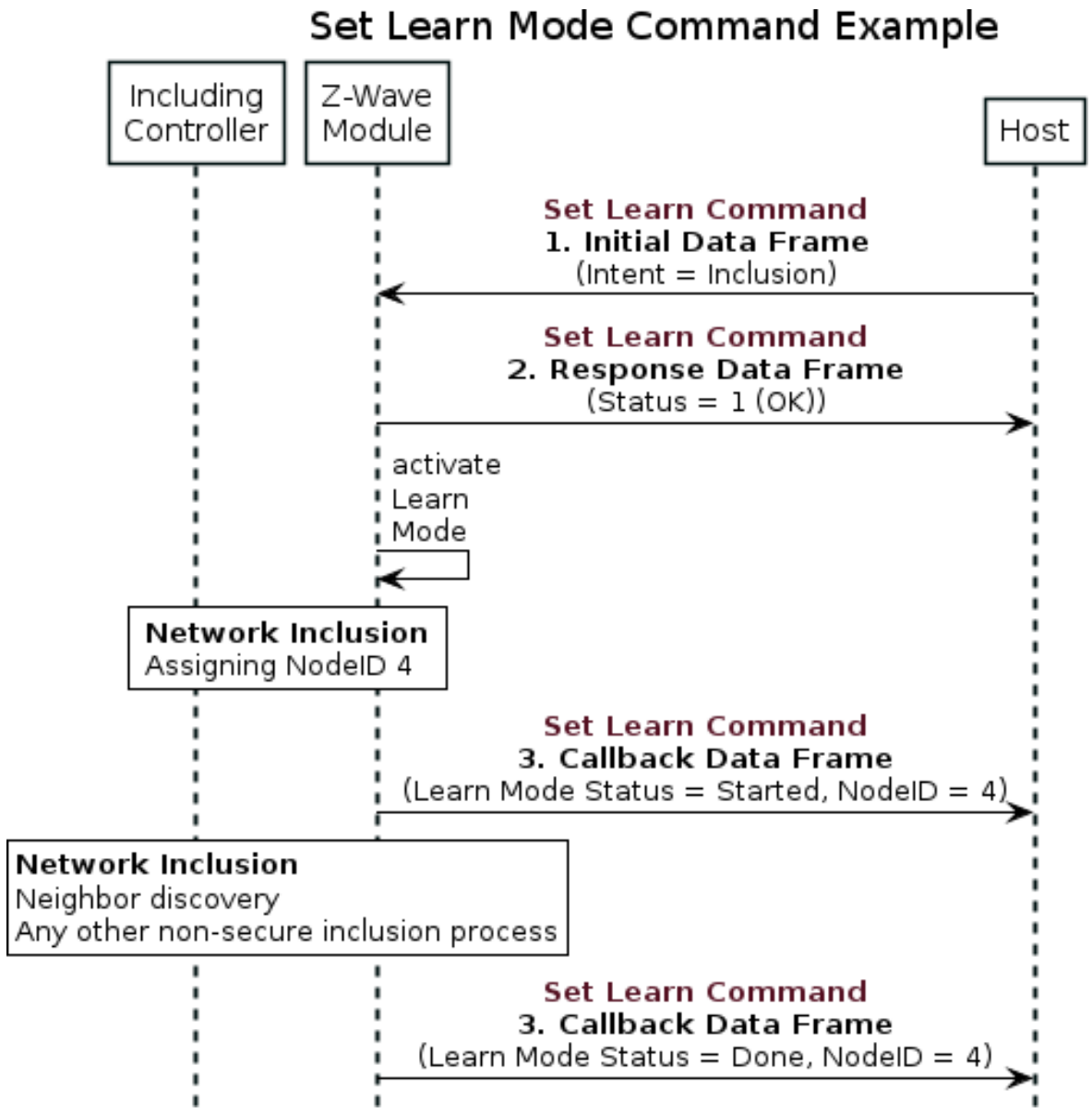

Figure 4.4: Set Learn Mode Command Example

1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.76

Table 4.76: Set Learn Mode Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x50							
5	Learn Mode Intent							
6	Session Identifier							

Learn Mode Intent (8 bits)

This field is used to indicate the Learn Mode Intent. Refer to [zwave_nwk_spec] for *Learn Mode* details.

This field MUST be encoded according to Table 4.77.

Table 4.77: Set Learn Mode Command - Learn Mode Intent encoding

Value	Description
0x00	*Disabled* This value is used to stop learn mode.
0x01	*Learn Mode Inclusion/Exclusion* *Deprecated*: It is NOT RECOMMENDED to use this value, use 0x81, 0x82 or 0x83 instead. This value is used to start learn mode and expect either a direct-range network inclusion or a direct-range network exclusion.
0x02	*Learn Mode Network Wide Inclusion* *Deprecated*: It is NOT RECOMMENDED to use this value, use 0x81 instead. This value is used to start learn mode and expect a Network-Wide Inclusion (NWI).
0x03	*Learn Mode Network Wide Exclusion* *Deprecated*: It is NOT RECOMMENDED to use this value, use 0x83 instead. This value is used to start learn mode and expect a Network Wide Exclusion (NWE).
0x81	*Learn Mode (Network Wide) Inclusion* This value is used to start learn mode and expect either a network inclusion. The Z-Wave API Module SHOULD try a direct range inclusion followed by 4 NWI attempts.
0x82	*Learn Mode Exclusion* This value is used to start learn mode and expect a direct range Network Exclusion.
0x83	*Learn Mode Network Wide Exclusion* This value is used to start learn mode and expect a Network Wide Exclusion (NWE).
0x84	*SmartStart Learn Mode* This value is used to start SmartStart learn mode.

Session Identifier (8 bits)

Refer to *Session identifier (8 bits)*.

2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.78

Table 4.78: Set Learn Mode Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x50							
5	Response status							

Response Status (8 bits)

Refer to *Response status (8 bits)*.

3. Callback data frame (Z-Wave Module → host)

A Z-Wave module MUST return issue a callback data frame formatted according to Table 4.79

Table 4.79: Set Learn Mode Command - Callback data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x50							
5	Session identifier							
6	Learn Mode Status							
7/7..8	NodeID							

Session Identifier (8 bits)

Refer to *Session identifier (8 bits)*.

Learn Mode Status (8 bits)

This field is used to indicate the current Learn Mode Status.

This field MUST be encoded according to Table 4.80.

Table 4.80: Learn Mode Status encoding

Value	Description
0x01	*Learn Mode started* This value indicates that Learn Mode has started and is now ongoing
0x06	*Learn Mode completed* This value indicates that Learn Mode has completed with a successful outcome.
0x07	*Learn Mode failed* This value indicates that Learn Mode has completed with an unsuccessful outcome.

All other values are reserved.

NodeID (8 bits/16 bits)

This field is used to indicate the NodeID currently assigned to the Z-Wave API Module.

This field MUST be encoded according to the configured NodeID base Type. Refer to *Z-Wave API Setup Set NodeID Base Type Sub Command* and Table 4.64.

4.4.1.6 Get SUC NodeID Command

This command is used to get currently registered SUC/SIS NodeID in a Z-Wave network. The Get SUC NodeID Command Identifier is 0x56.

Frame flow

The frame flow for this command is an *Acknowledged frame with response*.

1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.81.

Table 4.81: Get SUC NodeID Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x56							

2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.82.

Table 4.82: Get SUC NodeID Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x56							
5/5..6	SUC NodeID							

SUC NodeID (8 bits)

This field is used to advertise the SUC NodeID in the Z-Wave network.

The value 0x00 MUST indicate that there is no SUC NodeID in the current network. All other values MUST indicate the NodeID of the SUC in the current network.

This field MUST be encoded according to the configured NodeID base Type. Refer to *Z-Wave API Setup Set NodeID Base Type Sub Command* and Table 4.64.

3. Callback data frame (Z-Wave Module → host)

None.

4.4.1.7 Set SmartStart Inclusion Request Maximum Interval Command

This command is used to set the maximum interval between SmartStart inclusion requests. The Set Maximum SmartStart Inclusion Request Interval Command Identifier is 0xD6.

Frame flow

The frame flow for this command is an *Acknowledged frame with response*.

1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.83 to set the maximum number of ticks between SmartStart inclusion requests.

Table 4.83: Set Maximum SmartStart Inclusion Request Interval Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0xD6							
5	Requested Intervals							

Requested Intervals 1 (8 bits)

This field is used to indicate the maximum number of ticks between SmartStart inclusion requests. Each ticks MUST have 128 seconds interval.

2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.84.

Table 4.84: Set Maximum SmartStart Inclusion Request Interval Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0xD6							
5	Command Status							

Command Status (8 bits)

This field is used to advertise the status regarding the configuration of SmartStart inclusion request interval. The field value MUST be encoded according to *Command Status (8 bits)*.

3. Callback data frame (Z-Wave Module → host)

None.

4.4.1.8 Explore Request Inclusion Command

This command is used to initiate a Network-Wide Inclusion process. When the Z-Wave module receives this command, the module MUST issue an explore frame for requesting inclusion (add) to a Z-Wave network. The Explore Request Inclusion Command Identifier is 0x5E.

Frame flow

The frame flow for this command is an *Acknowledged frame with response*.

The host application MUST send *Set Learn Mode Command* with a *Learn Mode Intent* field value equals to 0x81 before sending *Explore Request Inclusion command* for requesting the Z-Wave API module to trigger a Network-Wide Inclusion process. Once a *Set Learn Mode callback data frame* (that indicates the inclusion process has started) is received, the application MUST NOT send this command to the Z-Wave API module. Figure 4.5 illustrates the usage of *Explore Request Inclusion Command*.

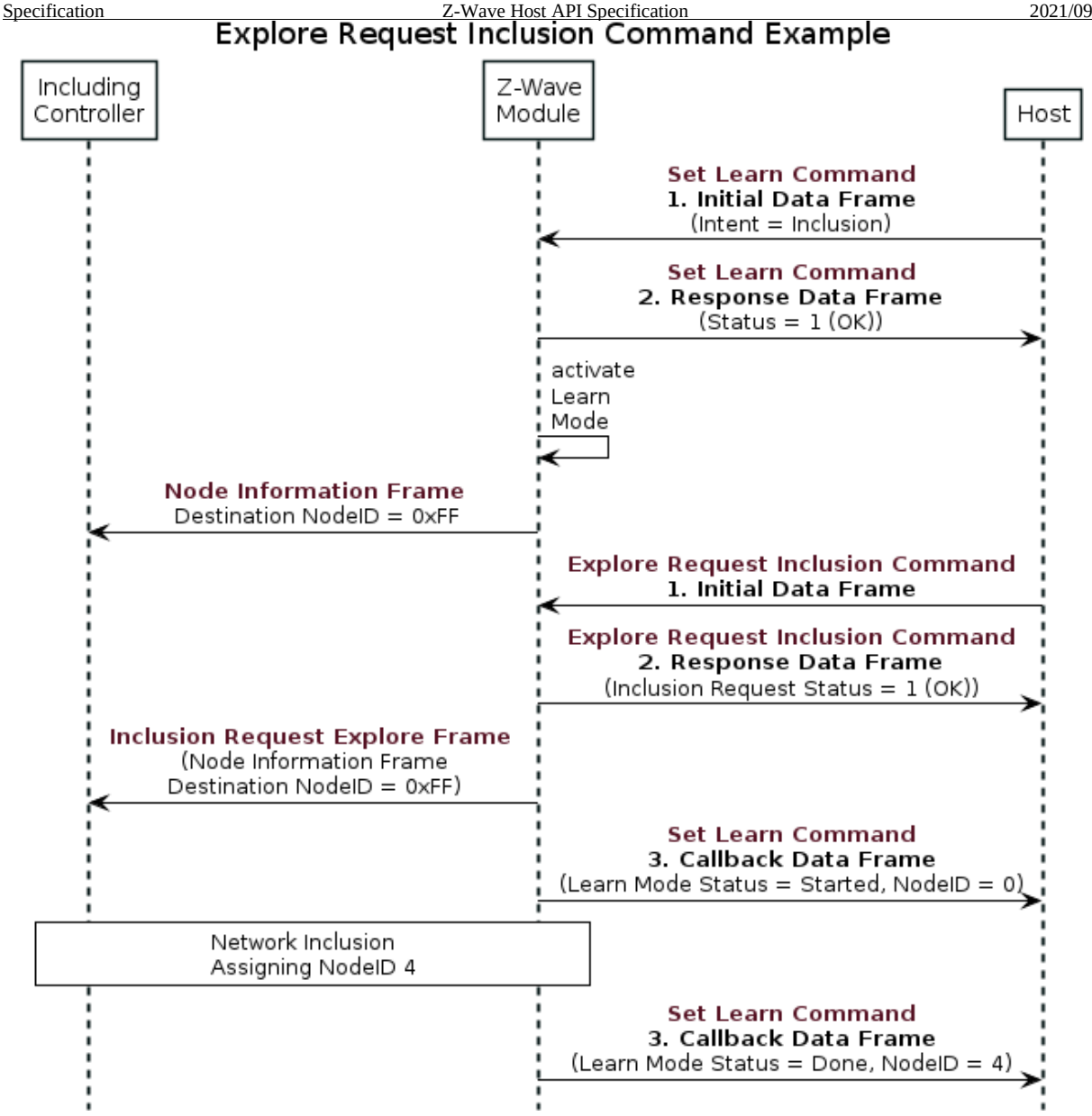

Figure 4.5: Explore Request Inclusion Command Example

It is not recommended to use this command since *Set Learn Mode Command* with a *Learn Mode Intent* field value equals to 0x81 can trigger the Inclusion process without issuing *Explore Request Inclusion Command*.

1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.85.

Table 4.85: Explore Request Inclusion Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x5E							

2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.86.

Table 4.86: Explore Request Inclusion Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x5E							
5	Inclusion Request Status							

Inclusion Request Status (8 bits)

This field is used to advertise the status regarding the acceptance of the *Explore Request Inclusion Command Initial data frame*. This field MUST be encoded as follow:

- The field value MUST set to 0x01, if the inclusion request is queued for transmission by the Z-Wave module.
- The field value MUST set to 0x00, if the Learn Mode is not set.

3. Callback data frame (Z-Wave Module → host)

None.

4.4.1.9 Explore Request Exclusion Command

This command is used to initiate a Network-Wide Exclusion process. When the Z-Wave module receives this command, the module MUST issue an explore frame for requesting exclusion (remove) from a Z-Wave network. The Request Network Wide Exclusion Command Identifier is 0x5F.

Frame flow

The frame flow for this command is an *Acknowledged frame with response*.

The host application MUST send *Set Learn Mode Command* with *Learn Mode Intent* field value equals to 0x83 before sending *Explore Request Exclusion command* to the Z-Wave module. Once a *Set Learn Mode callback data frame* (that indicates the Exclusion process has started) is received, the application MUST NOT send this command to the Z-Wave module. Figure 4.6 illustrates the usage of *Explore Request Exclusion Command*.

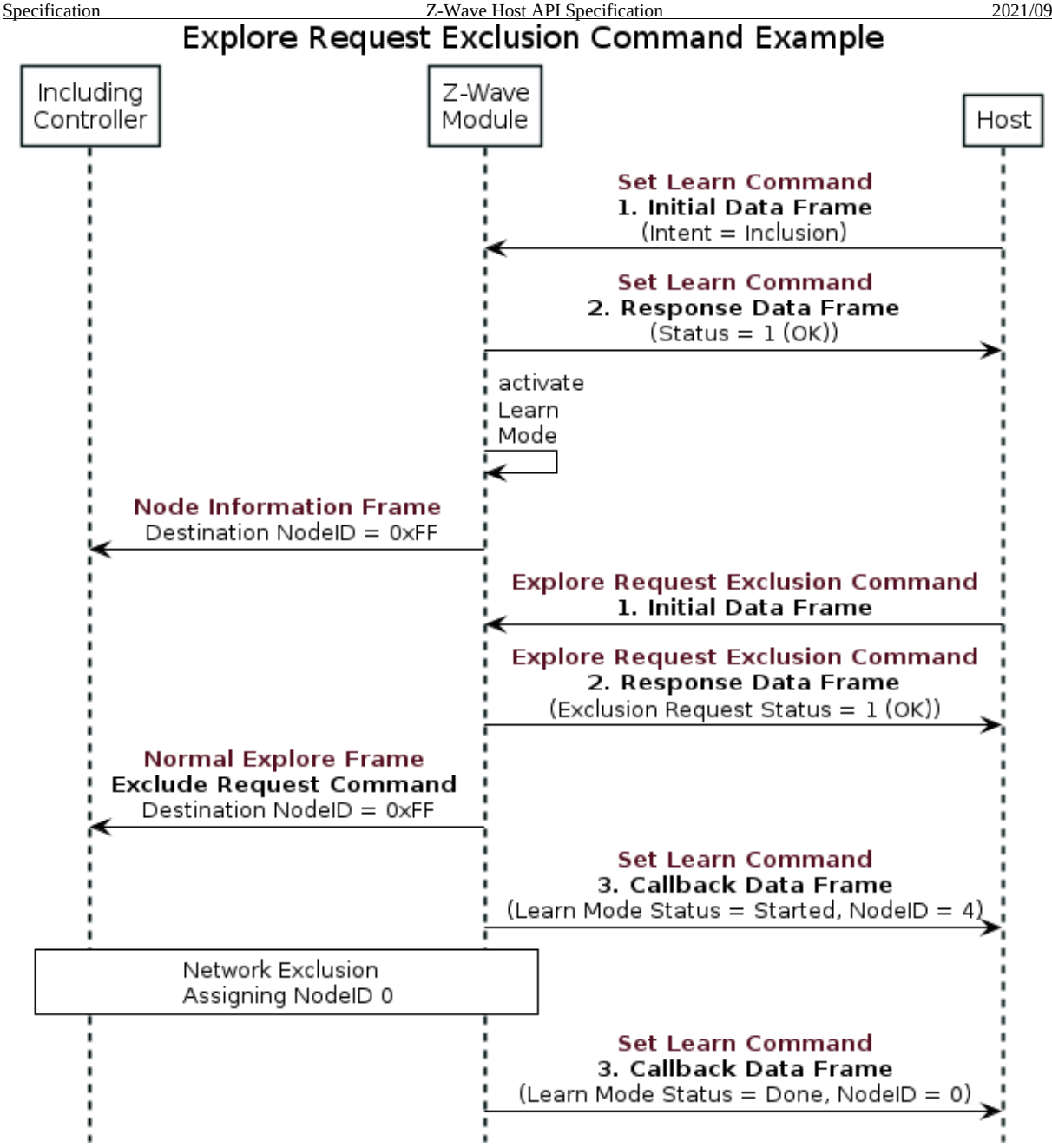

Figure 4.6: Explore Request Exclusion Command Example

It is not recommended to use this command since *Set Learn Mode Command* with a *Learn Mode Intent* field value equals to 0x83 can trigger the Exclusion process without issuing *Explore Request Exclusion Command*.

1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.87.

Table 4.87: Explore Request Exclusion Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x5F							

2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.88.

Table 4.88: Explore Request Exclusion Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x5F							
5	Exclusion Request Status							

Exclusion Request Status (8 bits)

This field is used to advertise the status regarding the acceptance of the *Explore Request Exclusion Command Initial data frame*. This field MUST be encoded as follow:

- The field value MUST set to 0x01, if the exclusion request is queued for transmission by the Z-Wave module.
- The field value MUST set to 0x00, if the Learn Mode is not set.

3. Callback data frame (Z-Wave Module → host)

None.

4.4.2 End Nodes Network Management

This section describes *Z-Wave API Commands* that are used to perform Z-Wave Network Management for End Nodes.

The commands described in this subsection MUST be supported by Z-Wave API modules implementing an *End Node* library type (refer to Table 4.36).

The commands described in this subsection MUST NOT be supported by Z-Wave API modules implementing a *Controller Node* library type (refer to Table 4.36).

4.4.2.1 Request New Route Destinations Command

This command is used to request a new route for destination nodes from SUC/SIS node. The Request New Route Destinations Command Identifier is 0x5C.

This commands MUST only be supported by a Z-Wave API module implementing a *Enhanced 232 End Node Library* or *Routing End Node library*.

Frame flow

The frame flow for this command is an *Acknowledged frame with response and callback*.

When the Z-Wave API module receives *Request New Route Destinations Command*, it will send *Static Route Request Commands* for each Destination NodeIDs. This is illustrated in Figure 4.7

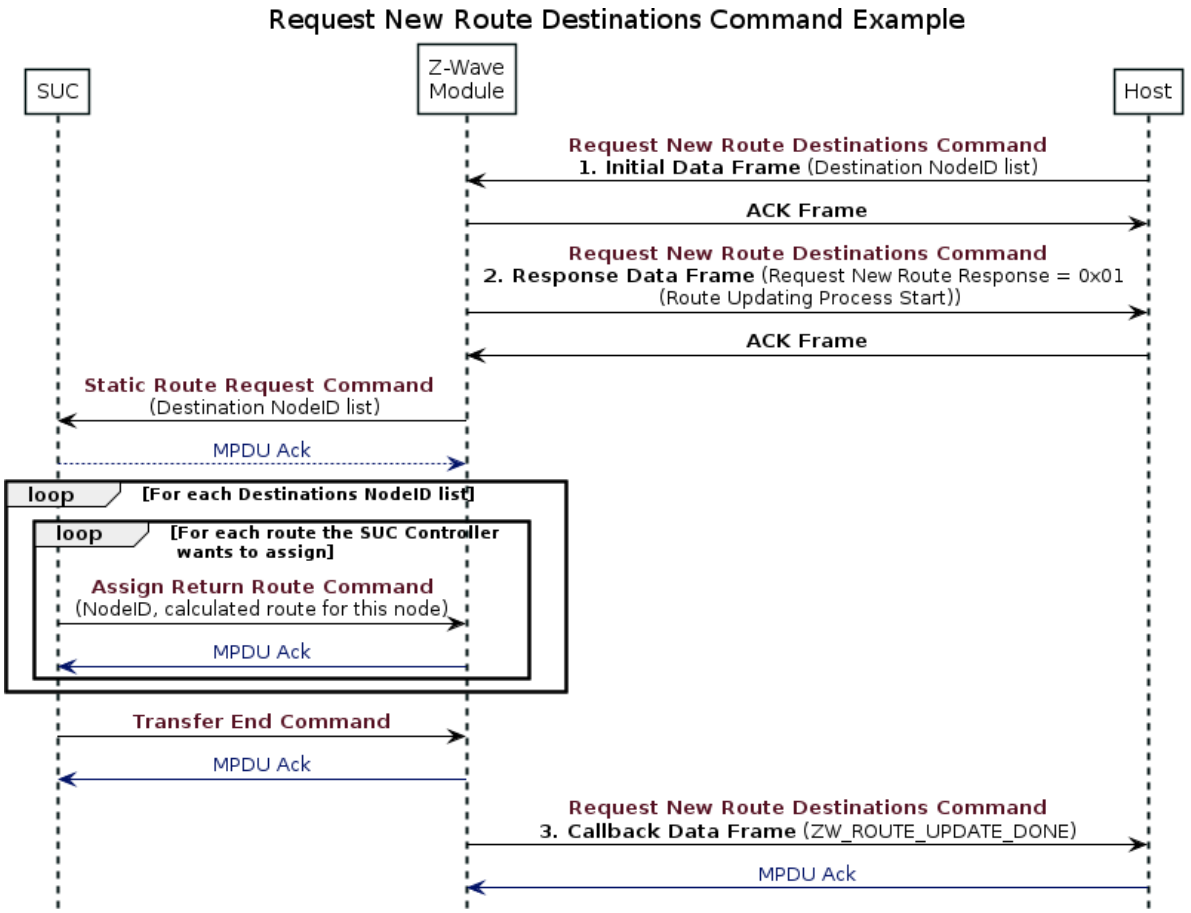

Figure 4.7: Request New Route Destinations Command Example

1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.89.

Table 4.89: Request New Route Destinations Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x5C							
5	Destinations NodeID 1							
..	. . .							
5+N	Destinations NodeID N							
6+N	Session identifier							

Destinations NodeID (N bytes)

This field is used to indicates the new destination NodeIDs for which return routes are requested.

Each byte in this field MUST represent a NodeID. All NodeIDs MUST be encoded using 8 bits regardless of the configured NodeID base Type. Refer to *Z-Wave API Setup Set NodeID Base Type Sub Command* and Table 4.64.

Session identifier (8 bits)

Refer to *Session identifier (8 bits)*.

2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.90.

Table 4.90: Request New Route Destinations Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x5C							
5	Request New Route Response							

Request New Route Response (8 bits)

This field is used to advertise the response of the Z-Wave module regarding the acceptance of the *Request New Route Destinations Command Initial data frame*. This field MUST be encoded as follow:

- If the new route updating process is started, this field value MUST be set to 0x01.
- If the protocol runs on Z-Wave module is busy or the SUC/SIS node is unknown to the protocol, this field value MUST be set to 0x00.

3. Callback data frame (Z-Wave Module → host)

A Z-Wave module MUST issue a callback frame formatted according to Table 4.91.

Table 4.91: Request New Route Destinations Command - Callback data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x5C							
5	Session identifier							
6	Request New Route Callback Status							

Session identifier (8 bits)

Refer to *Session identifier (8 bits)*.

Request New Route Callback Status (8 bits)

This field is used to notify status of the new route updating process. This field MUST be encoded according to Table 4.92.

Table 4.92: Request New Route Destinations Callback Status Value encoding

Value	Flag	Description
0x00	ZW_ROUTE_UPDATE_DONE	The new route updating process ended successfully.
0x01	ZW_ROUTE_UPDATE_ABORT	The new route updating process is aborted because of error.
0x02	ZW_ROUTE_UPDATE_WAIT	The SUC/SIS node is busy.
0x02	ZW_ROUTE_UPDATE_DISABLED	The SUC/SIS functionality is disabled.
0x04..0xFF	*Reserved*	Reserved values MUST NOT be used.

4.4.2.2 Is Node Within Direct Range Command

This command is used to check if a given NodeID is marked as a direct range node (A node that can be reached with a direct range communication from a Z-Wave API module) in any of the existing return routes. Is Node Within Direct Range Command Identifier is 0x5D.

This commands MUST only be supported by a Z-Wave module that employs *Enhanced 232 End Node Library* or *Routing End Node library*.

Frame flow

The frame flow for this command is an *Acknowledged frame with response*.

1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.93.

Table 4.93: Is Node Within Direct Range Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x5D							
5	NodeID							

NodeID (8 bits)

This field is used to indicate the NodeID which will be examined if it is stored as a direct range node in existing return routes.

This field MUST be encoded using 8 bits regardless of the configured NodeID base Type. Refer to *Z-Wave API Setup Set NodeID Base Type Sub Command* and Table 4.64.

2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.94.

Table 4.94: Is Node Within Direct Range Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x5D							
5	Direct Range Status							

Direct Range Status (8 bits)

This field is used to indicate a status regarding the node presence in existing return route as a direct range node. This field MUST be encoded as follow:

- If the node is reachable within direct range, the field value MUST be set to 0x01.
- If the node is beyond direct range or status is unknown to the Z-Wave protocol, the field value MUST be set to 0x00.

3. Callback data frame (Z-Wave Module → host)

None.

4.4.2.3 Get Network Statistics Command

This command is used to request the current Network Statistics as collected by a library runs on the Z-Wave Module. It is expected that the library will continuously update any Network Statistics counter until it reaches 65535, which then indicates that the specific counter has reached 65535 or more occurrences. The Network Statistics counters shall be cleared either on module startup, or by receiving Clear Network Statistics Command. The Get Network Statistics Command Identifier is 0x3A.

Frame flow

The frame flow for this command is an *Acknowledged frame with response*.

1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.95

Table 4.95: Get Network Statistics Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x3A							

2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.96

Table 4.96: Get Network Statistics Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x3A							
5	Tx Frames 1 (MSB)							
6	Tx Frames 2 (LSB)							
7	Tx LBT BackOffs 1 (MSB)							
8	Tx LBT BackOffs 2 (LSB)							
9	Rx Frames 1 (MSB)							
10	Rx Frames 2 (LSB)							
11	Rx Checksum Errors 1 (MSB)							
12	Rx Checksum Errors 2 (LSB)							
13	Rx CRC16 Errors 1 (MSB)							
14	Rx CRC16 Errors 2 (LSB)							
15	Rx Foreign HomeID 1 (MSB)							
16	Rx Foreign HomeID 2 (LSB)							

Tx Frames (2 bytes)

This field is used to indicate the transmitted frames. This field MUST be encoded as a 16-bits unsigned integer.

Tx LBT BackOffs (2 bytes)

This field is used to advertise the numbers of times the Tx had to wait and postpone a transmission due to a measured RSSI above the allowed LBT threshold.

This field MUST be encoded as a 16-bits unsigned integer.

Rx Frames (2 bytes)

This field is used to advertise the numner of received frames without any errors. This field MUST be encoded as a 16-bits unsigned integer.

Rx Checksum Errors (2 bytes)

This field is used to advertise the number of received frames with checksum errors. This field MUST be encoded as a 16-bits unsigned integer.

Rx CRC16 Errors (2 bytes)

This field is used to advertise the number of received frames with CRC16 checksum errors. This field MUST be encoded as a 16-bits unsigned integer.

Rx Foreign HomeID (2 bytes)

This field is used to advertise the number of valid Z-Wave frames that has been received with a HomeID not matching the HomeID of the receiving node. This field MUST be encoded as a 16-bits unsigned integer.

3. Callback data frame (Z-Wave Module → host)

None.

4.4.2.4 Clear Network Statistics Command

This command is used to clear the current Network Statistics collected by the Z-Wave API Module. The Clear Network Statistics Command Identifier is 0x39.

Frame flow

The frame flow for this command is an *Acknowledged frame with response*.

1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.97

Table 4.97: Clear Network Statistics Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x39							

2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.98

Table 4.98: Clear Network Statistics Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x39							
5	Command Status							

Command Status (8 bits)

Refer to *Command Status (8 bits)*.

3. Callback data frame (Z-Wave Module → host)

None.

4.4.3 Controller Nodes Network Management

This section describes *Z-Wave API Commands* that are used to perform Z-Wave Network Management for Controller Nodes.

The commands described in this subsection MUST NOT be supported by Z-Wave API modules implementing an *End Node* library type (refer to Table 4.36).

The commands described in this subsection MUST be supported by Z-Wave API modules implementing a *Controller Node* library type (refer to Table 4.36).

4.4.3.1 Add Node To Network Command

This command is used to trigger a node inclusion to a Z-Wave network. The Add Node To Network Command Identifier is 0x4A.

This Command MUST be supported by Controller Nodes Z-Wave API implementations. This Command MUST NOT be supported by End Nodes Z-Wave API implementations.

Frame flow

The frame flow for this command is an *Acknowledged frame with callback*.

The host application may issue several 1. initial data frames several during an inclusion and the Z-Wave API module may issue sereval 3. callback data frames during an inclusion.

Figure 4.8 shows an example of a successful network inclusion.

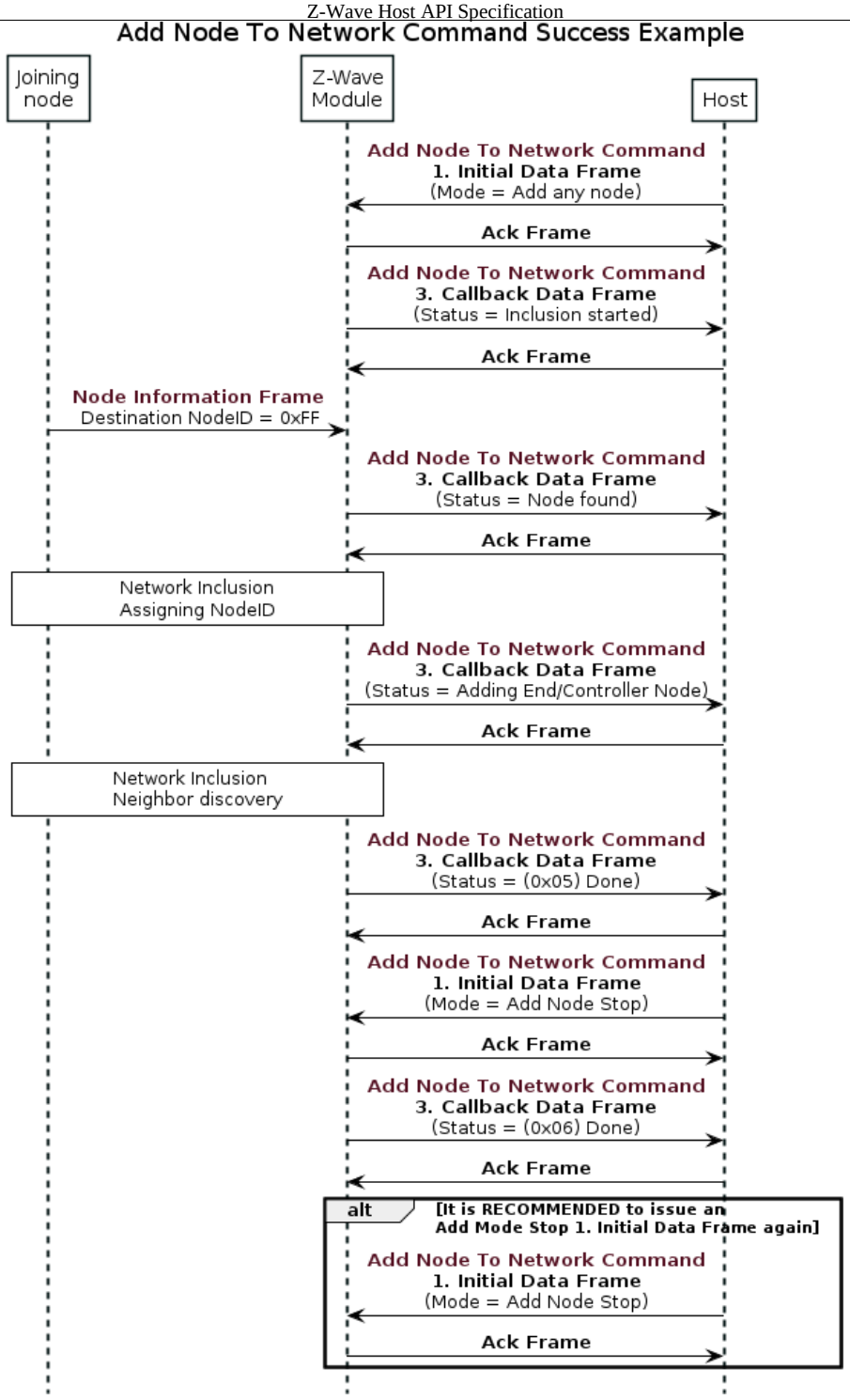

Figure 4.8: Add Node To Network Command Success Example

Figure 4.9 shows an example of a host timeout for a network inclusion.

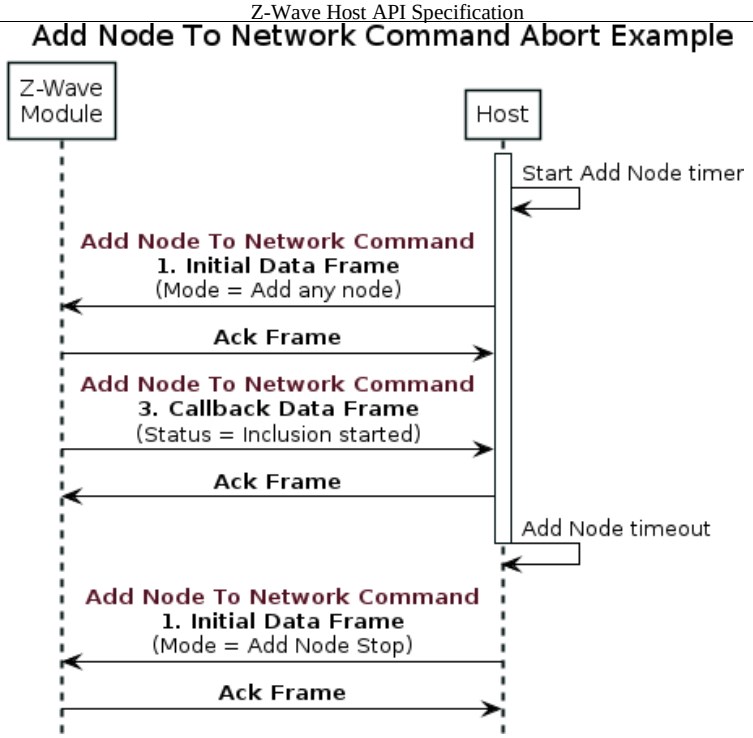

Figure 4.9: Add Node To Network Command Abort Example

Figure 4.10 shows an example of a SmartStart network inclusion.

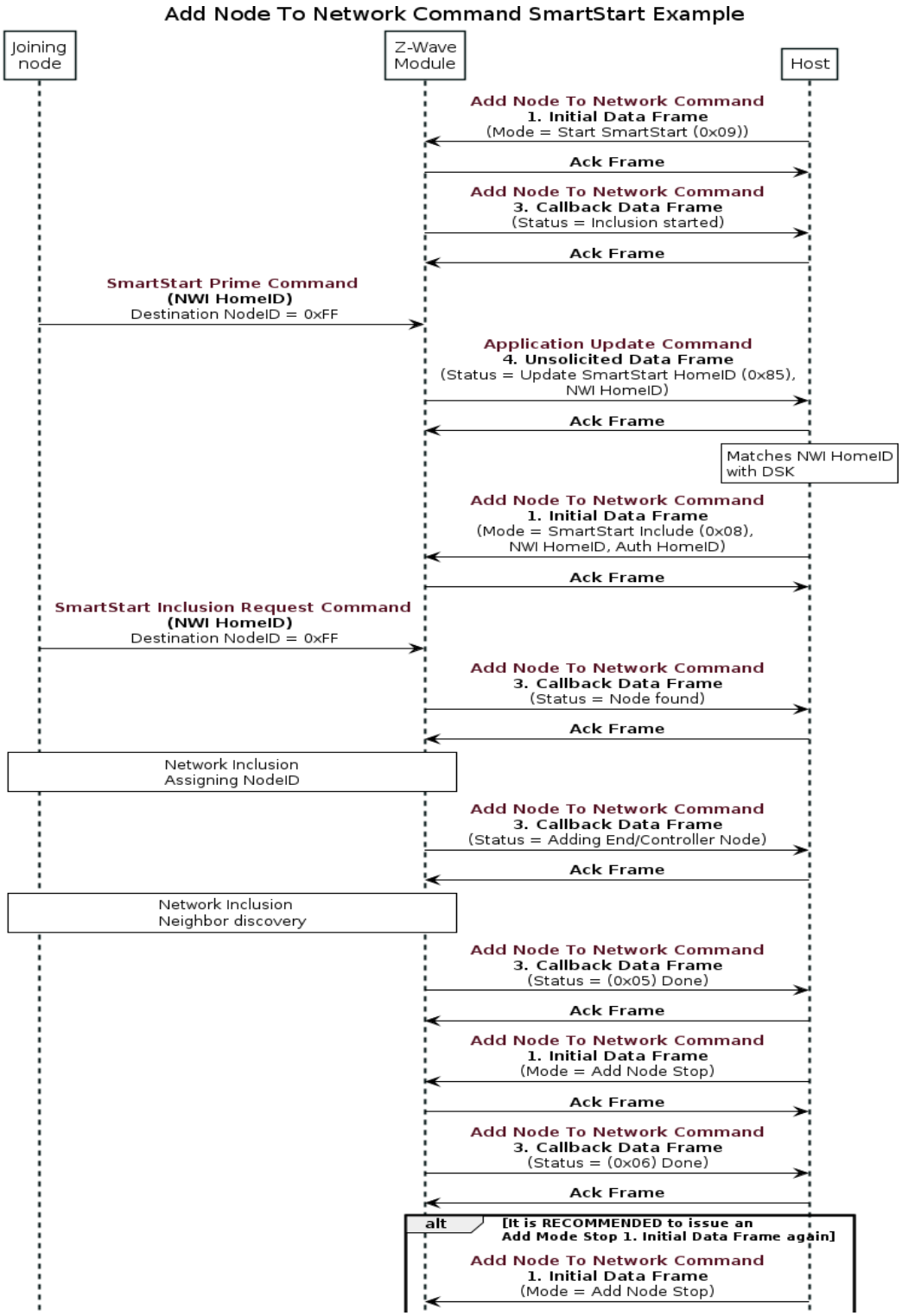

Figure 4.11 shows an example of a Z-Wave Long Range SmartStart network inclusion.

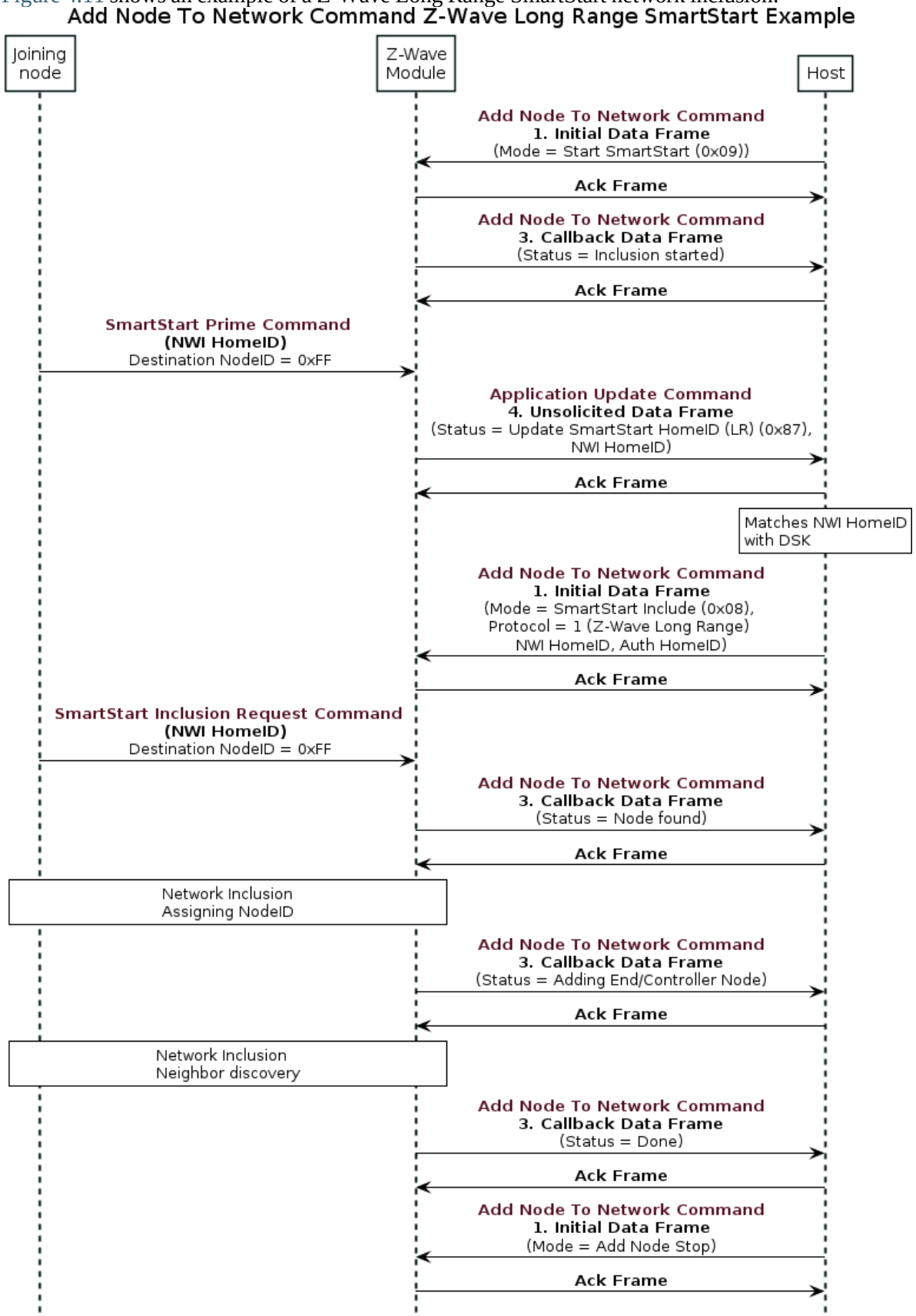

Figure 4.11: Add Node To Network Command Z-Wave Long Range SmartStart Example

1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.99

Table 4.99: Add Node To Network Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x4A							
5	Power	NWI	Protocol	Mode				
6	Session Identifier							
7	NWI HomeID 1							
8	NWI HomeID 2							
9	NWI HomeID 3							
10	NWI HomeID 4							
11	Auth HomeID 1							
12	Auth HomeID 2							
13	Auth HomeID 3							
14	Auth HomeID 4							

Power (1 bit)

This field is used to indicate which power to use for the Network Inclusion operation.

- The value 0 MUST indicate that the Z-Wave Module MUST use reduced power for the network inclusion.
- The value 1 MUST indicate that the Z-Wave Module MUST use normal power for the network inclusion.

NWI (1 bit)

This field is used to indicate if the operation must be a direct range Network Inclusion or NWI.

- The value 0 MUST indicate that the Z-Wave Module MUST use direct range network inclusion for the network inclusion.
- The value 1 MUST indicate that the Z-Wave Module MUST use NWI for the network inclusion.

Protocol (1 bit)

This field is used to indicate if the operation must be carried out using Z-Wave or Z-Wave Long Range.

- The value 0 MUST indicate that the Z-Wave Module MUST use Z-Wave for the Add mode operation.
- The value 1 MUST indicate that the Z-Wave Module MUST use Z-Wave Long Range for the Add mode operation.

Mode (5 bits)

This field is used to indicate which “Add Mode” operation the Z-Wave Module MUST carry out.

This field MUST encoded according to Table 4.100

Table 4.100: Add Node To Network Command - Mode encoding

Value	Description
0x01	*Add any node*. The Z-Wave Module MUST start network inclusion and attempt to add any type of node. The Z-Wave Module MUST keep in network inclusion mode until this command is called again with the *Mode* set to 0x05 or 0x06
0x02	*Deprecated*: Add controller node. Use *Add any node (0x01)* instead.
0x03	*Deprecated*: Add End Node. Use *Add any node (0x01)* instead.
0x04	*Deprecated*: Add existing node. Use *Add any node (0x01)* instead.
0x05	*Stop network inclusion*. The Z-Wave Module MUST stop the ongoing or completed network inclusion.
0x06	*Stop controller replication*. This value is used for stopping a network inclusion. This value SHOULD be used by a host application if a controller replication is ongoing and must be aborted.
0x07	*Reserved*. This value is reserved.
0x08	*SmartStart Include Node*. This value is used when a host application has matched a SmartStart Prime Command to a pending DSK in its provisioning list. This value indicates that the Z-Wave API module MUST initiate the SmartStart Network inclusion when the next *SmartStart Inclusion Request Command* is received from this node. When this value is used, the *NWI HomeID* and *Auth HomeID* fields MUST be set to the values from the matched DSK in the host application's provisioning list.
0x09	*Start SmartStart* The Z-Wave API Module MUST activate NWI, start listening for *SmartStart Prime Commands* and report them to the host application. No inclusion will be made yet with this option.
0x0A..0x19	*Reserved*. These values are reserved. Reserved values MUST NOT be used and MUST be ignored by a receiving interface.

Session Identifier (8 bits)

Refer to *Session identifier (8 bits)*.

NWI HomeID (4 bytes)

This field is used to indicate the NWI HomeID of the node that MUST be included using SmartStart.

If the *Mode* field is not set to 0x08, this field and the Auth HomeID field MAY be omitted. If the *Mode* field is not set to 0x08, this field SHOULD be set to 0x00000000.

If the *Mode* field is set to 0x08, this field MUST be set to the NWI HomeID of the DSK entry that MUST be included by the Z-Wave API Module.

Auth HomeID (4 bytes)

This field is used to indicate the Auth HomeID of the node that MUST be included using SmartStart.

If the *Mode* field is not set to 0x08, this field and the NWI HomeID field MAY be omitted. If the *Mode*

field is not set to 0x08, this field SHOULD be set to 0x00000000.

If the *Mode* field is set to 0x08, this field MUST be set to the Auth HomeID of the DSK entry that MUST be included by the Z-Wave API Module.

2. Response data frame (Z-Wave Module → host)

None.

3. Callback data frame (Z-Wave Module → host)

A Z-Wave module MUST issue a callback frame formatted according to Table 4.101

Table 4.101: Add Node To Network Command - Callback data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x4A							
5	Session Identifier							
6	Status							
7/7..8	Assigned NodeID							
8/9	Supported Command Class List Length							
9/10	Basic Device Type							
10/11	Generic Device Type							
11/12	Specific Device Type							
(11/12)+1	Supported Command Class List 1							
…	…							
(11/12)+N	Supported Command Class List N							

Session Identifier (8 bits)

Refer to *Session identifier (8 bits)*.

Status (8 bits)

This field is used to advertise the current status of the inclusion process.

This field MUST be according to Table 4.102

Table 4.102: Add Node To Network Command - Status encoding

Value	Description
0x01	*Network Inclusion Started*. The Z-Wave Module has initiated Network inclusion and is ready to include new nodes
0x02	*Node found*. A node requesting inclusion has been found and the network inclusion is initiated.
0x03	*Inclusion ongoing (End Node)*. The network inclusion is ongoing with an End Node.
0x04	*Inclusion ongoing (Controller Node)*. The network inclusion is ongoing with a Controller node.
0x05	*Inclusion completed (protocol part)*. The network inclusion is completed. The host application SHOULD issue a new *Add Node To Network Command - Initial data frame* with the *Mode* field set to 0x05 to stop the network inclusion.
0x06	*Inclusion completed* The network inclusion is completed. The Z-Wave Module is ready to return to idle and the host application SHOULD issue a new *Add Node To Network Command - Initial data frame* with the *Mode* field set to 0x05 to stop the network inclusion.

Assigned NodeID (8/16 bits)

This field is used to advertise the NodeID that was assigned during the inclusion process.

This field MUST be encoded according to the configured NodeID base Type. Refer to *Z-Wave API Setup Set NodeID Base Type Sub Command* and Table 4.64.

The value 0 MUST indicate that no NodeID was assigned at that stage of the inclusion process.

Supported Command Class List Length (8 bits)

This field is used to advertise the length of the *Supported Command Class List* field, in bytes.

Basic Device Type (8 bits)

This field is used to advertise the Basic Device Type reported by the node.

For a detailed description of the *Basic Device Type* field, refer to [device_class_spec] for Z-Wave nodes, [device_type_spec] for Z-Wave Plus nodes, and [device_type_spec_v2] for Z-Wave Plus v2 nodes.

Generic Device Type (8 bits)

This field is used to advertise the Generic Device Type reported by the node.

For a detailed description of the *Generic Device Type* field, refer to [device_class_spec] for Z-Wave nodes, [device_type_spec] for Z-Wave Plus nodes, and [device_type_spec_v2] for Z-Wave Plus v2 nodes.

Specific Device Type (8 bits)

This field is used to advertise the Specific Device Type reported by the node.

For a detailed description of the *Specific Device Type* field, refer to [device_class_spec] for Z-Wave nodes, [device_type_spec] for Z-Wave Plus nodes, and [device_type_spec_v2] for Z-Wave Plus v2

nodes.

Supported Command Class List (N bytes)

This field is used to advertise the list of supported Command Classes reported by the node during its inclusion.

This list represents the non-secure supported Command Classes.

The length of this field, in bytes, MUST be according to the *Supported Command Class List Length* field.

A host application SHOULD request the node's capabilities again after S0/S2 bootstrapping.

4.4.3.2 Add Controller And Assign Primary Controller Role Command

This command is used to include and give the Primary Controller Role to another controller node. The Assign Primary Controller Role Command Identifier is 0x4C.

This command MUST be used by a host application only if the Z-Wave API Module is Secondary Controller, has the SUC Role and the Primary Controller has been removed from the network.

This command MUST NOT be used in any other case.

Frame flow

The frame flow for this command is an *Acknowledged frame with callback*.

1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.103

Table 4.103: Add Controller And Assign Primary Controller Role Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x4C							
5	Power	NWI	Reserved	Mode				
6	Session Identifier							

Power (1 bit)

This field is used to indicate which power to use for the Network Inclusion operation.

- The value 0 MUST indicate that the Z-Wave Module MUST use reduced power for the network inclusion.
- The value 1 MUST indicate that the Z-Wave Module MUST use normal power for the network inclusion.

NWI (1 bit)

This field is used to indicate if the operation must be a direct range Network Inclusion or NWI.

- The value 0 MUST indicate that the Z-Wave Module MUST use direct range network inclusion for the network inclusion.
- The value 1 MUST indicate that the Z-Wave Module MUST use NWI for the network inclusion.

Mode (5 bits)

This field is used to indicate which “Add Mode” operation the Z-Wave Module MUST carry out.

This field MUST encoded according to Table 4.104

Table 4.104: Add Controller And Assign Primary Controller Role Command - Mode encoding

Value	Description
0x01	*Start Add Mode* The Z-Wave API Module MUST start network inclusion and attempt to add a controller node. The Z-Wave API Module MUST keep in network inclusion mode until this command is called again with the *Mode* set to 0x05 or 0x06
0x02..0x04	*Reserved*. These values are reserved. Reserved values MUST NOT be used and MUST be ignored by a receiving interface.
0x05	*Stop Add Mode* The Z-Wave API Module MUST stop network inclusion.
0x06	*Stop Add Mode due to a failure* The Z-Wave API Module MUST stop network inclusion and report a failure to the other controller node.
0x07..0xFF	*Reserved*. These values are reserved. Reserved values MUST NOT be used and MUST be ignored by a receiving interface.

Session Identifier (8 bits)

Refer to *Session identifier (8 bits)*.

2. Response data frame (Z-Wave Module → host)

None.

3. Callback data frame (Z-Wave Module → host)

A Z-Wave module MUST issue a callback frame formatted according to Table 4.105

Table 4.105: Add Controller And Assign Primary Controller Role Command - Callback data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x4C							
5	Session Identifier							
6	Status							
7/7..8	Assigned NodeID							
8/9	Supported Command Class List Length							
9/10	Basic Device Type							
10/11	Generic Device Type							
11/12	Specific Device Type							
12/13	Supported Command Class List 1							
…	…							
12/13+N	Supported Command Class List N							

For fields description, refer to *Add Node To Network Command*

4.4.3.3 Add Primary Controller Command

This command is used to include a controller node and assign it the Primary Controller Role. The Add Primary Controller Command Identifier is 0x4D.

Frame flow

The frame flow for this command is an *Acknowledged frame with callback*.

1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.106

Table 4.106: Add Primary Controller Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x4D							
5	Power	NWI	Reserved	Mode				
6	Session Identifier							

Power (1 bit)

This field is used to indicate which power to use for the Network Inclusion operation.

- The value 0 MUST indicate that the Z-Wave Module MUST use reduced power for the network inclusion.
- The value 1 MUST indicate that the Z-Wave Module MUST use normal power for the network inclusion.

NWI (1 bit)

This field is used to indicate if the operation must be a direct range Network Inclusion or NWI.

- The value 0 MUST indicate that the Z-Wave Module MUST use direct range network inclusion for the network inclusion.
- The value 1 MUST indicate that the Z-Wave Module MUST use NWI for the network inclusion.

Mode (5 bits)

This field is used to indicate which “Add Mode” operation the Z-Wave Module MUST carry out.

This field MUST encoded according to Table 4.107

Table 4.107: Add Primary Controller Command - Mode encoding

Value	Description

0x01	*Start Add Mode* The Z-Wave API Module MUST start network inclusion and attempt to add a controller node. The Z-Wave API Module MUST keep in network inclusion mode until this command is called again with the *Mode* set to 0x05 or 0x06.
0x02..0x04	*Reserved*. These values are reserved. Reserved values MUST NOT be used and MUST be ignored by a receiving interface.
0x05	*Stop Add Mode* The Z-Wave API Module MUST stop network inclusion.
0x06	*Stop Add Mode due to a failure* The Z-Wave API Module MUST stop network inclusion and report a failure to the other controller node.
0x07..0xFF	*Reserved*. These values are reserved. Reserved values MUST NOT be used and MUST be ignored by a receiving interface.

Session Identifier (8 bits)

Refer to *Session identifier (8 bits)*.

2. Response data frame (Z-Wave Module → host)

None.

3. Callback data frame (Z-Wave Module → host)

A Z-Wave module MUST issue a callback frame formatted according to Table 4.108

Table 4.108: Add Primary Controller Command - Callback data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x4D							
5	Session Identifier							
6	Status							
7/8	Assigned NodeID							
8/9	Supported Command Class List Length							
9/10	Basic Device Type							
10/11	Generic Device Type							
11/12	Specific Device Type							
12/13	Supported Command Class List 1							
…	…							
12/13+N	Supported Command Class List N							

For fields description, refer to *Add Node To Network Command*

4.4.3.4 Remove Node From Network Command

This command is used to trigger a node removal operation from a Z-Wave network. It is also possible to perform out-of-range removal of nodes from the network when repeater nodes are capable of forwarding the new network wide exclusion (NWE) frame. The Remove Node From Network Command Identifier is 0x4B.

Frame flow

The frame flow for this command is an *Acknowledged frame with callback*.

The host application may issue several 1. initial data frames several during an inclusion and the Z-Wave API module may issue sereval 3. callback data frames during an inclusion.

Figure 4.12 and Figure 4.13 show examples of successful network exclusion.

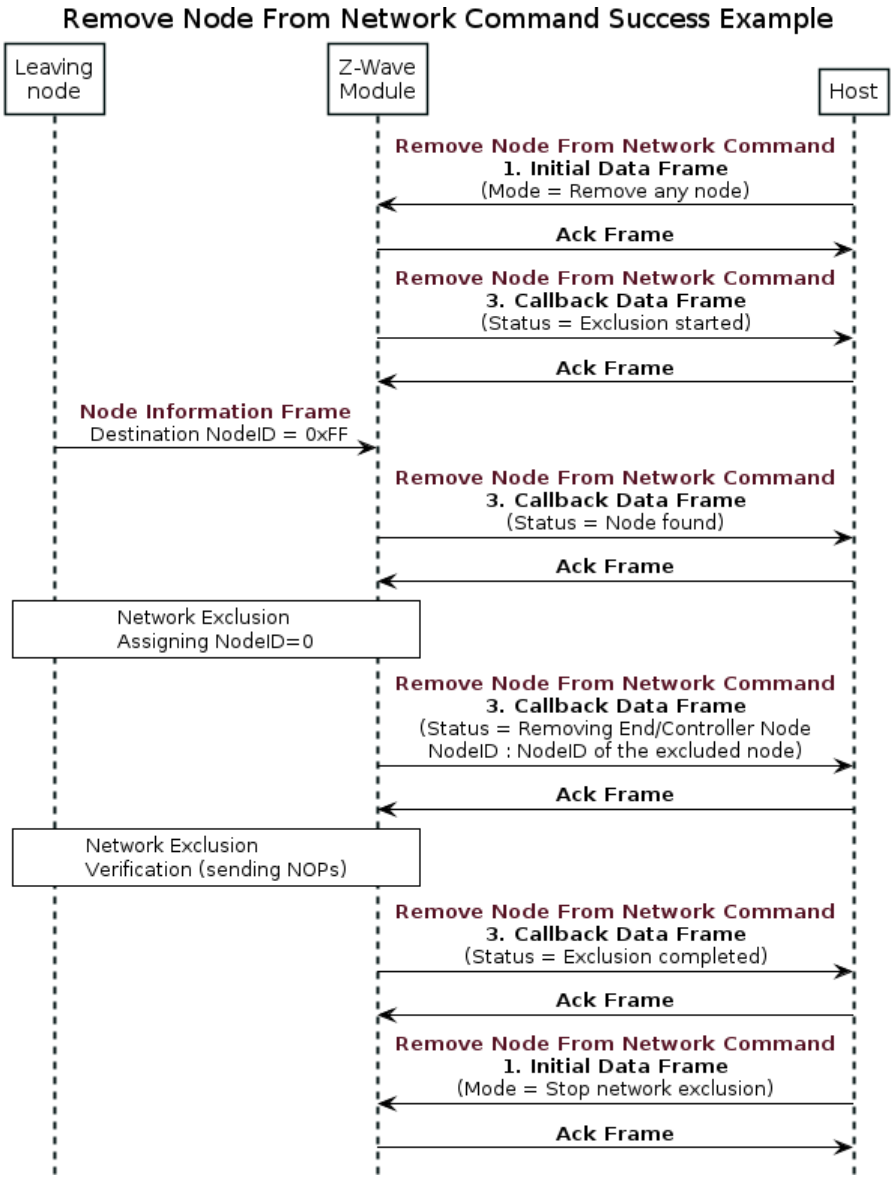

Figure 4.12: Remove Node From Network Command Success Example

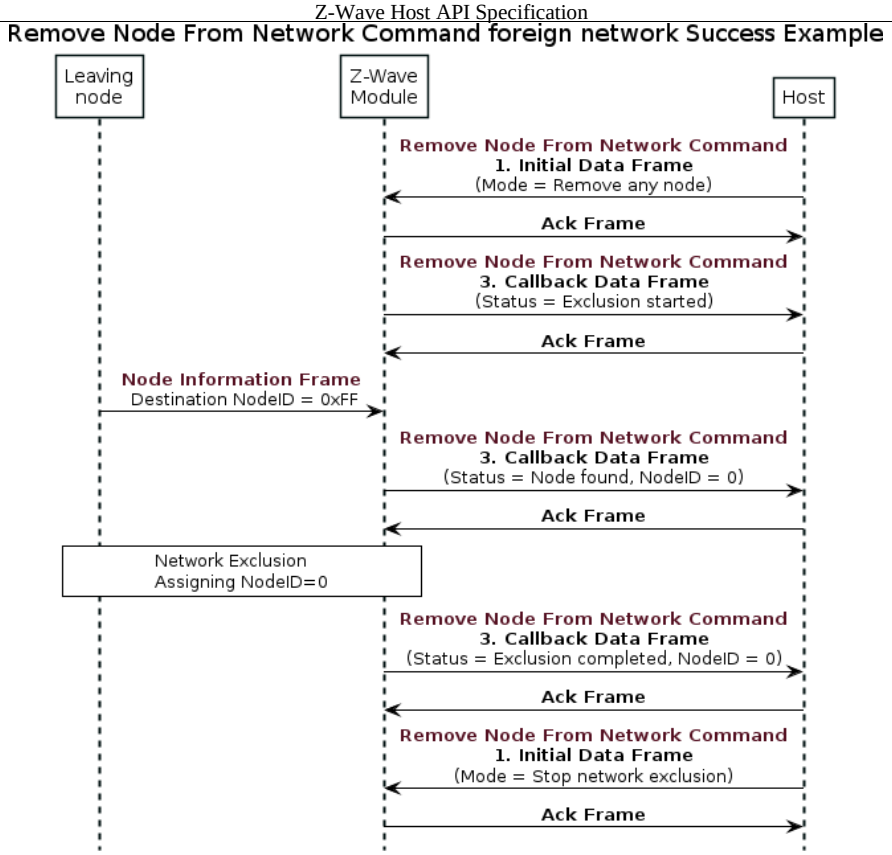

Figure 4.13: Remove Node From Network Command foreign network Success Example

1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.109

Table 4.109: Remove Node From Network Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x4B							
5	Power	NWE	Res	Mode				
6	Session Identifier							

Power (1 bit)

This field is used to indicate which power to use for the removal node operation.

- The value 0 MUST indicate that the Z-Wave Module MUST use reduced power for the removal node operation.
- The value 1 MUST indicate that the Z-Wave Module MUST use normal power for the removal node operation.

NWE (1 bit)

This field is used to indicate if direct range Network Exclusion or NWE.

- The value 0 MUST indicate that the Z-Wave Module MUST use direct range exclusion for the removal node operation.
- The value 1 MUST indicate that the Z-Wave Module MUST use NWE for the removal node operation.

Mode (4 bits)

This field is used to indicate which "Remove Mode" operation the Z-Wave Module MUST carry out.

This field MUST encoded according to Table 4.110

Table 4.110: Remove Node From Network Command - Mode encoding

Vale	Description
0x01	Remove any node. The Z-Wave Module MUST start network exclusion and attempt to remove any type of node. The Z-Wave Module MUST keep in removal process (network exclusion mode) until this command is called again with the *Mode* set to 0x05.
0x02	*Deprecated*: Remove controller node. Use *Remove any node (0x01)* instead.
0x03	*Deprecated*: Remove End Node. Use *Remove any node (0x01)* instead.
0x04	*Reserved*. This value is reserved.
0x05	Stop network exclusion. The Z-Wave Module MUST stop the ongoing or completed network exclusion.
0x06..0x3F	*Reserved*. These values are reserved. Reserved values MUST NOT be used and MUST be ignored by a receiving interface.

Session Identifier(8 bits)

Refer to *Session identifier (8 bits)*.

2. Response data frame (Z-Wave Module → host)

None.

3. Callback data frame (Z-Wave Module → host)

A Z-Wave module MUST issue a callback frame formatted according to Table 4.111

Table 4.111: Remove Node From Network Command - Callback data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x4B							
5	Session Identifier							
6	Status							
7/7..8	NodeID							
8/9	Supported Command Class List Length							
9/10	Basic Device Type							
10/11	Generic Device Type							
11/12	Specific Device Type							
12/13	Supported Command Class List 1							
…	…							
12/13+N	Supported Command Class List N							

For fields not described below, refer to *Add Node To Network Command - 3. Callback data frame*.

Session Identifier (8 bits)

Refer to *Session identifier (8 bits)*.

Status (8 bits)

This field is used to advertise the current status of the node removal process.

This field MUST be according to Table 4.112

Table 4.112: Remove Node From Network Command - Status encoding

Value	Description
0x01	*Network Exclusion Started*. The Z-Wave Module has initiated Network exclusion and is ready to remove existing nodes
0x02	*Node found*. A node requesting exclusion has been found and the node removal operation is initiated.
0x03	*Exclusion ongoing (End Node)* The network exclusion is ongoing with an End Node.
0x04	*Exclusion ongoing (controller node)* The network exclusion is ongoing with a Controller node.
0x05	*Reserved*. This value is reserved.
0x06	*Exclusion completed* Node removal operation is completed. The Z-Wave Module is ready to return to idle and the host application SHOULD issue a new *Remove Node From Network Command - Initial data frame* with the *Mode* set to 0x05 to stop the network exclusion.

0x07	*Exclusion failed* Removal node operation is failed. This indicates the node may not have been removed, and the host application SHOULD issue a new *Remove Node From Network Command - Initial data frame* with the *Mode* set to 0x05 to stop the network exclusion.
0x23	*Not Primary* The node exclusion operation cannot be performed because the Z-Wave API Module does not have the Primary Controller role and the SIS functionality is not available in the current network.

Values that are not described in table_remove_node_from_network_status_encoding are reserved and MUST NOT be used.

NodeID (8/16 bits)

This field indicates the NodeID of the node that was removed from a network.

This field MUST be encoded according to the configured NodeID base Type. Refer to *Z-Wave API Setup Set NodeID Base Type Sub Command* and Table 4.64.

The value 0 MUST indicate that no node was removed (yet) in our current network. A Z-Wave API Moduel MUST set this value to 0 if excluding a node from a foreign network.

4.4.3.5 Remove Specific Node From Network Command

This command is used to trigger a specific node removal operation from a Z-Wave network. It is also possible to perform out-of-range removal of specific node from the network when repeater nodes are capable of forwarding the new network wide exclusion (NWE) frame. The Remove Specific Node From Network Command Identifier is 0x3F.

Frame flow

The frame flow for this command is an *Acknowledged frame with callback*.

1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.113.

Table 4.113: Remove Specific Node From Network Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x3F							
5	Power	NWE	Res	Mode				
6/6..7	NodeID							
7/8	Session Identifier							

Refer to *Remove Node From Network Command Initial data frame* for fields that are not described below.

NodeID (8/16 bits)

This field is used to indicate the NodeID to be removed from a network.

Nodes with this NodeID seeking exclusion will be excluded from their network by the Z-Wave API Module. Nodes with a different NodeID seeking exclusion will not be excluded from their network by the Z-Wave API Module.

This field MUST be encoded according to the configured NodeID base Type. Refer to *Z-Wave API Setup Set NodeID Base Type Sub Command* and Table 4.64.

2. Response data frame (Z-Wave Module → host)

None.

3. Callback data frame (Z-Wave Module → host)

A Z-Wave module MUST issue a callback frame formatted according to Table 4.114 after a specific node removal operation is performed.

Table 4.114: Remove Specific Node From Network Command - Callback data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x3F							
5	Session Identifier							
6	Status							
7/7..8	NodeID							
8/9	Supported Command Class List Length							
9/10	Basic Device Type							
10/11	Generic Device Type							
11/12	Specific Device Type							
12/13	Supported Command Class List 1							
…	…							
12/13+N	Supported Command Class List N							

For fields description, refer to *Remove Node From Network Command Callback data frame*.

4.4.3.6 Is Node Failed Command

This command is used to request if a given NodeID is considered as failed by the Z-Wave API Module. Is Failed Node Command Identifier is 0x62.

Frame flow

The frame flow for this command is an *Acknowledged frame with response*.

1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.115 to check a given NodeID presence in the Z-Wave API Module failed NodeID list.

Table 4.115: Is Failed Node Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x62							
5/5..6	NodeID							

NodeID (8/16 bits)

This field is used to advertise the NodeID that will be checked if it is stored in the controller failed NodeID list.

This field MUST be encoded according to the configured NodeID base Type. Refer to *Z-Wave API Setup Set NodeID Base Type Sub Command* and Table 4.64.

2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.116.

Table 4.116: Is Failed Node Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x62							
5	FailedNodeID presence							

FailedNodeID presence (8 bits)

This field is used to advertise the presence of a given NodeID in controller failed NodeID list. This field MUST be encoded as follow:

- The field value MUST be 0x00, if the NodeID is not stored in the controller failed NodeID list.
- The field value MUST be 0x01, if the NodeID is stored in the controller failed NodeID list.

3. Callback data frame (Z-Wave Module → host)

None.

4.4.3.7 Remove Failed Node Command

This command is used to request a non-responding node removal operation from the controller routing table. When a node is non-responding, its' NodeID shall be included in the failed NodeID list. If the node responding again it shall be removed from the failed NodeID list. A failed node MUST only be removed if the NodeID is in the failed NodeID list and extra precaution shall be considered before the failed node is removed. A responding node MUST NOT be removed. The Remove Failed Node Command Identifier is 0x61.

Note that this command MUST only be used by Primary Controller and an Inclusion Controller.

Frame flow

The frame flow for this command is an *Acknowledged frame with response and callback*.

1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.117 to trigger a failed node removal from the controller routing table.

Table 4.117: Remove Failed Node Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x61							
5/5..6	NodeID							
6/7	Session identifier							

NodeID (8/16 bits)

This field is used to advertise the NodeID of a failed node that will be removed from the routing table. This field MUST be encoded according to the configured NodeID base Type. Refer to *Z-Wave API Setup Set NodeID Base Type Sub Command* and Table 4.64.

Session identifier (8 bits)

Refer to *Session identifier (8 bits)*.

2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.118.

Table 4.118: Remove Failed Node Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x61							
5	Remove Failed Node Response Status							

Remove Failed Node Response Status (8 bits)

This field is used to indicate the Z-Wave module response regarding the remove failed node request. This field MUST be encoded according to Table 4.119.

Table 4.119: Remove Failed Node Response Status value encoding

Value	Flag and description
0x00	*ZW_FAILED_NODE_REMOVE_STARTED* The failed node removing process started successfully.
0x01	*ZW_NOT_PRIMARY_CONTROLLER* The removing process was aborted because the controller is not the Primary Controller.
0x02	*ZW_NO_CALLBACK_FUNCTION* The removing process was aborted because no Session identifier is used.
0x03	*ZW_FAILED_NODE_NOT_FOUND* The requested process failed. The NodeID was not found in the controller list of failed NodeID list.
0x04	*ZW_FAILED_NODE_REMOVE_PROCESS_BUSY* The Z-Wave Protocol is busy.
0x05	*ZW_FAILED_NODE_REMOVE_FAIL* The removing process request is failed. This can happen if the controller is busy or the node is responding.
0x6..0xFF	*Reserved* Reserved values MUST NOT be used.

3. Callback data frame (Z-Wave Module → host)

A Z-Wave module MUST issue a callback frame formatted according to Table 4.120 to notify the application regarding the failed node removal operation status.

Table 4.120: Remove Failed Node Command - Callback data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x61							
5	Session identifier							
6	Remove Failed Node Operation Status							

Session identifier (8 bits)

Refer to *Response status (8 bits)*.

Remove Failed Node Operation Status (8 bits)

This field is used to notify the status of failed node removal operation. This field MUST be encoded according to Table 4.121.

Table 4.121: Remove Failed Node Operation Status value encoding

Value	Flag and description
0x00	*ZW_NODE_OK* The node is working properly and it is a responding node. The NodeID of the node MUST be removed from the failed NodeID list.
0x01	*ZW_FAILED_NODE_REMOVED* The failed node was removed from the routing table and failed nodes list.
0x02	*ZW_FAILED_NODE_NOT_REMOVED* The failed node was not removed because the removing process cannot be completed.
0x3..0xFF	*Reserved* Reserved values MUST NOT be used.

4.4.3.8 Replace Failed Node Command

This command is used to replace a non-responding node with a new node. Responding nodes MUST NOT be replaced. The Replace Failed Node Command Identifier is 0x63.

Note that this command MUST only be used by Primary Controller and an Inclusion Controller.

Frame flow

The frame flow for this command is an *Acknowledged frame with response and callback*.

Figure 4.14 shows an example of a successful failed node replacement.

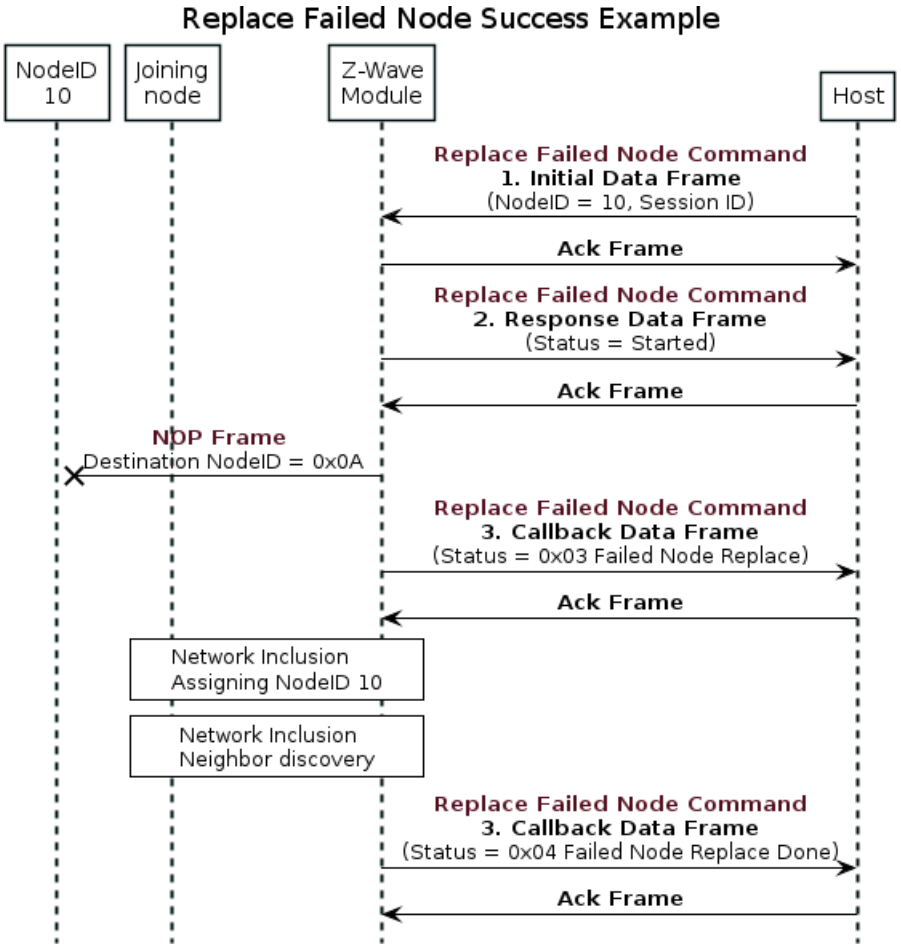

Figure 4.14: Replace Failed Node Success Example

Figure 4.15 shows an example of an unsuccessful failed node replacement.

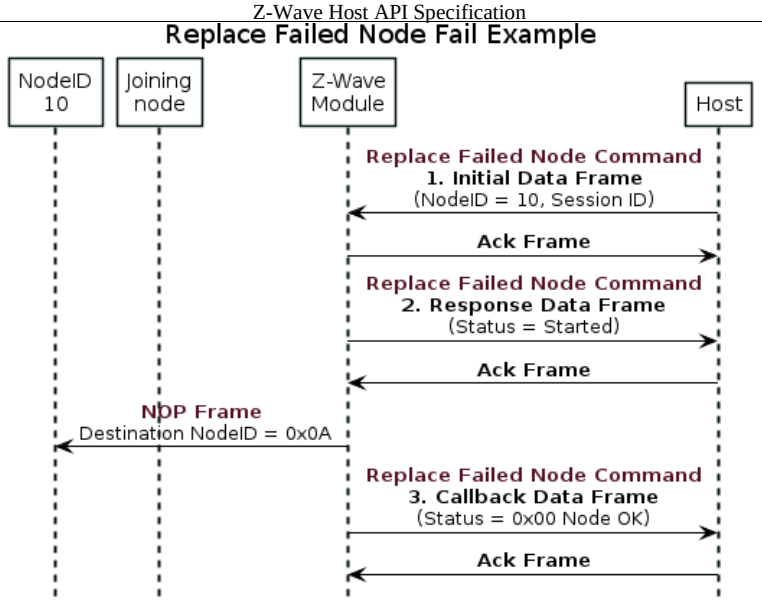

Figure 4.15: Replace Failed Node Example

1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.122 to replace a failed node with a new node.

Table 4.122: Replace Failed Node Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x63							
5/5..6	NodeID							
6	Session identifier							

NodeID (8/16 bits)

This field is used to advertise the NodeID of a failed node that will be assigned to a new node.

This field MUST be encoded according to the configured NodeID base Type. Refer to *Z-Wave API Setup Set NodeID Base Type Sub Command* and Table 4.64.

Session identifier (8 bits)

Refer to *Session identifier (8 bits)*.

2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.123.

Table 4.123: Replace Failed Node Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x63							
5	Replace Failed Node Response Status							

Replace Failed Node Response Status (8 bits)

This field is used to indicate the Z-Wave module response regarding the replace failed node request. This field MUST be encoded according to Table 4.124.

Table 4.124: Remove Failed Node Response Status value encoding

Value	Flag and description
0x00	*ZW_FAILED_NODE_REMOVE_STARTED* The failed node replacing process started successfully.
0x01	*ZW_NOT_PRIMARY_CONTROLLER* The replacing process was aborted because the controller is not the Primary Controller.
0x02	*ZW_NO_CALLBACK_FUNCTION* The replacing process was aborted because no Session identifier is used.
0x03	*ZW_FAILED_NODE_NOT_FOUND* The requested process failed. The NodeID was not found in the controller list of failed NodeID list.
0x04	*ZW_FAILED_NODE_REMOVE_PROCESS_BUSY* The Z-Wave API module is busy and cannot carry the operation.
0x05	*ZW_FAILED_NODE_REMOVE_FAIL* The replacing process request has failed. This can happen if the Z-Wave API module is busy or the node is responding.
0x6..0xFF	*Reserved* Reserved values MUST NOT be used.

3. Callback data frame (Z-Wave Module → host)

A Z-Wave module MUST issue a callback frame formatted according to Table 4.125

Table 4.125: Replace Failed Node Command - Callback data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x63							
5	Session identifier							
6	Replace Failed Node Operation Status							

Session identifier (8 bits)

Refer to *Response status (8 bits)*.

Replace Failed Node Operation Status (8 bits)

This field is used to notify the status of replace failed node operation. This field MUST be encoded according to Table 4.126.

Table 4.126: Replace Failed Node Operation Status value encoding

Value	Flag and description
0x00	*ZW_NODE_OK* The node is working properly and it is a responding node. The NodeID of the node shall be removed from the failed NodeID list. It also indicates that replace process is stopped.
0x1..0x02	*Reserved* Reserved values MUST NOT be used.
0x03	*ZW_FAILED_NODE_REPLACE* The failed node is ready to be replaced and controller is ready to add new node with the NodeID of the failed node.
0x04	*ZW_FAILED_NODE_REPLACE_DONE* The failed node has been replaced.
0x05	*ZW_FAILED_NODE_REPLACE_FAILED* The failed node has not been replaced.
0x06..0xFF	*Reserved* Reserved values MUST NOT be used.

4.4.3.9 Delete Return Route Command

This command is used to request the deletion of the static return routes. The Delete Return Route Command Identifier is 0x47.

Frame flow

The frame flow for this command is an *Acknowledged frame with response and callback*.

1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.127.

Table 4.127: Delete Return Route Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x47							
5/5..6	NodeID							
6/7	Session identifier							

NodeID (8/16 bits)

This field is used to indicates the NodeID for which the static return routes is requested to be deleted.

This field MUST be encoded according to the configured NodeID base Type. Refer to *Z-Wave API Setup Set NodeID Base Type Sub Command* and Table 4.64.

Session identifier (8 bits)

Refer to *Session identifier (8 bits)*.

2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.128.

Table 4.128: Delete Return Route Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x47							
5	Delete Return Route Response							

Delete Return Route Response (8 bits)

This field is used to advertise the response of the Z-Wave module regarding the acceptance of the *Delete Return Route Command Initial data frame*. This field MUST be encoded as follow:

- If delete return route operation is started, this field value MUST be set to 0x01.

- If an “assign/delete return route” operation is already active, this field value MUST be set to 0x00.

3. Callback data frame (Z-Wave Module → host)

A Z-Wave module MUST issue a callback frame formatted according to Table 4.129.

Table 4.129: Delete Return Route Command - Callback data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x47							
5	Session identifier							
6	Tx Status							

Session identifier (8 bits)

Refer to *Session identifier (8 bits)*.

Tx Status (8 bits)

Refer to *Tx Status (8 bits)*.

4.4.3.10 Assign Return Route Command

This command is used to assign return routes to end nodes in a network. Refer to [zwave_nwk_spec] for details. The Assign Return Route Command Identifier is 0x46.

Frame flow

The frame flow for this command is an *Acknowledged frame with response and callback*.

1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.130.

Table 4.130: Assign Return Route Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x46							
5/5..6	NodeID							
6/7..8	Destination NodeID							
7/9	Session Identifier							

NodeID (8 bits/16 bits)

This field is used to indicate the NodeID for which the return route for the destination must be assigned by the Z-Wave API Module.

This field MUST be encoded according to the configured NodeID base Type. Refer to *Z-Wave API Setup Set NodeID Base Type Sub Command* and Table 4.64.

Destination NodeID (8 bits/16 bits)

This field is used to indicate the Destination NodeID for which the return route must be assigned by the Z-Wave API Module.

This field MUST be encoded according to the configured NodeID base Type. Refer to *Z-Wave API Setup Set NodeID Base Type Sub Command* and Table 4.64.

Session identifier (8 bits)

Refer to *Session identifier (8 bits)*.

2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.131

Table 4.131: Assign Return Route Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0

4	Z-Wave API Command ID = 0x46
5	Response Status

Response status (8 bits)

Refer to *Response status (8 bits)*.

3. Callback data frame (Z-Wave Module → host)

A Z-Wave module MUST issue a callback frame formatted according to Table 4.132

Table 4.132: Assign Return Route Command - Callback data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x46							
5	Session Identifier							
6	Tx Status							

Session Identifier (8 bits)

Refer to *Session identifier (8 bits)*.

Tx Status (8 bits)

Refer to *Tx Status (8 bits)*.

4.4.3.11 Assign SUC Return Route Command

This command is used to Assign a Return Route to the SUC NodeID.

The Assign SUC Return Route Command Identifier is 0x51

Frame flow

The frame flow for this command is an *Acknowledged frame with response and callback*.

1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.133

Table 4.133: Assign SUC Return Route Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x51							
5/5..6	NodeID							
6/7	Session Identifier							

NodeID (8 bits/16 bits)

This field is used to indicate the NodeID for which the return route for the SUC must be assigned by the Z-Wave API Module.

This field MUST be encoded according to the configured NodeID base Type. Refer to *Z-Wave API Setup Set NodeID Base Type Sub Command* and Table 4.64.

Session identifier (8 bits)

Refer to *Session identifier (8 bits)*.

2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.134

Table 4.134: Assign SUC Return Route Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x51							
5/5..6	Response Status							

Response status (8 bits)

Refer to *Response status (8 bits)*.

3. Callback data frame (Z-Wave Module → host)

A Z-Wave module MUST issue a callback frame formatted according to Table 4.135

Table 4.135: Assign SUC Return Route Command - Callback data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x51							
5	Session Identifier							
6	Tx Status							

Session Identifier (8 bits)

Refer to *Session identifier (8 bits)*.

Tx Status (8 bits)

Refer to *Tx Status (8 bits)*.

4.4.3.12 Assign Priority Return Route Command

This command is used to assign priority route to end nodes. An end node MUST always use the priority route for the first transmission attempt. Refer to [zwave_nwk_spec] for details. The Assign Priority Return Route Command Identifier is 0x4F.

Frame flow

The frame flow for this command is an *Acknowledged frame with response and callback*.

1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.136.

Table 4.136: Assign Priority Return Route Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x4F							
5/5..6	NodeID							
6/7..8	Route Destination NodeID							
7/9	Repeater 0							
8/10	Repeater 1							
9/11	Repeater 2							
10/12	Repeater 3							
11/13	Route Speed							
12/14	Session Identifier							

NodeID (8/16 bits)

This field is used to advertise the NodeID of the end node which shall receive defined priority route.

This field MUST be encoded according to the configured NodeID base Type. Refer to *Z-Wave API Setup Set NodeID Base Type Sub Command* and Table 4.64.

Route Destination NodeID (8/16 bits)

This field is used to indicate the destination NodeID which the end node shall use the defined priority route while transmitting a packet to it.

This field MUST be encoded according to the configured NodeID base Type. Refer to *Z-Wave API Setup Set NodeID Base Type Sub Command* and Table 4.64.

Repeater (4 bytes)

Refer to *Repeater (4 bytes)*.

Route Speed (8 bits)

Refer to *Route Speed (8 bits)*.

Session identifier (8 bits)

Refer to *Session identifier (8 bits)*.

2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.137.

Table 4.137: Assign Priority Return Route Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x4F							
5	Assign Priority Route Response							

Assign Priority Route Response (8 bits)

This field is used to advertise the response of the Z-Wave API module regarding the acceptance of the *Assign Priority Return Route Command Initial data frame*. This field MUST be encoded as follow:

- If assign priority return route operation is started, this field value MUST be set to 0x01.
- If an "assign/delete return route" operation is already active, this field value MUST be set to 0x00.

3. Callback data frame (Z-Wave Module → host)

A Z-Wave module MUST issue a callback frame formatted according to Table 4.138.

Table 4.138: Assign Priority Return Route Command - Callback data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x4F							
5	Session Identifier							
6	Tx Status							

Session identifier (8 bits)

Refer to *Session identifier (8 bits)*.

Tx Status (8 bits)

Refer to *Tx Status (8 bits)*.

4.4.3.13 Assign Priority SUC Return Route Command

This command is used to assign a priority return route to reach the SUC NodeID. The Assign Priority SUC Return Route Command Identifier is 0x58.

Frame flow

The frame flow for this command is an *Acknowledged frame with response and callback*.

1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.139.

Table 4.139: Assign Priority SUC Return Route Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x58							
5/5..6	NodeID							
6/7	Repeater 0							
7/8	Repeater 1							
8/10	Repeater 2							
9/11	Repeater 3							
10/12	Route Speed							
11/13	Session Identifier							

NodeID (8/16 bits)

This field is used to advertise the NodeID of the end node which shall receive defined priority route to reach SUC/SIS.

This field MUST be encoded according to the configured NodeID base Type. Refer to *Z-Wave API Setup Set NodeID Base Type Sub Command* and Table 4.64.

Repeater (4 bytes)

Refer to *Repeater (4 bytes)*.

Route Speed (8 bits)

Refer to *Route Speed (8 bits)*.

Session identifier (8 bits)

Refer to *Session identifier (8 bits)*.

2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.140.

Table 4.140: Assign Priority SUC Return Route Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x58							
5	Assign Priority SUC Route Response							

Assign Priority SUC Route Response (8 bits)

This field is used to advertise the response of the Z-Wave API module regarding the acceptance of the *Assign Priority SUC Return Route Command Initial data frame*. This field MUST be encoded as follow:

- If assign priority SUC return route operation is started, this field value MUST be set to 0x01.
- If an "assign/delete return route" operation is already active, this field value MUST be set to 0x00.

3. Callback data frame (Z-Wave Module → host)

A Z-Wave module MUST issue a callback frame formatted according to Table 4.141.

Table 4.141: Assign Priority SUC Return Route Command - Callback data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x58							
5	Session Identifier							
6	Tx Status							

Session identifier (8 bits)

Refer to *Session identifier (8 bits)*.

Tx Status (8 bits)

Refer to *Tx Status (8 bits)*.

4.4.3.14 Set Priority Route Command

This command is used to set the Priority Route for a destination node. The Priority Route is the route that shall be used as the first routing attempt by the Z-Wave protocol when transmitting to a node. The Priority Route is expected to be stored in NVM of the Z-Wave module. The Set Priority Route Command Identifier is 0x93.

Frame flow

The frame flow for this command is an *Acknowledged frame with response*.

1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.142 to set the Priority Route.

Table 4.142: Set Priority Route Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x93							
5/5..6	NodeID							
6/7	Repeater 0							
7/8	Repeater 1							
8/9	Repeater 2							
9/10	Repeater 3							
10/11	Route Speed							

NodeID (8/16 bits)

This field is used to indicate the destination NodeID for which the Priority Route is set to.

This field MUST be encoded according to the configured NodeID base Type. Refer to *Z-Wave API Setup Set NodeID Base Type Sub Command* and Table 4.64.

Repeater (4 bytes)

Refer to *Repeater (4 bytes)*.

Route Speed (8 bits)

Refer to *Route Speed (8 bits)*.

2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.143.

Table 4.143: Set Priority Route Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x93							
5	Command Status							

Command Status (8 bits)

Refer to *Command Status (8 bits)*.

3. Callback data frame (Z-Wave Module → host)

None.

4.4.3.15 Get Priority Route Command

This command is used to request the priority route that is defined in the Z-Wave API module. If a route has been set to the module using *Set Priority Route Command*, the module MUST provide the priority route using *Get Priority Route Command* Response frame. If no priority route has been set in the module, the *Get Priority Route Command* Response frame MUST contain either the Last WorkingRoute (LWR) or the Next to Last Working Route (NLWR). The Get Priority Route Command Identifieris 0x92.

Frame flow

The frame flow for this command is an *Acknowledged frame with response*.

1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.144.

Table 4.144: Get Priority Route Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x92							
5/5..6	NodeID							

NodeID (8/16 bits)

This field is used to indicate the destination NodeID for which the Priority Route is requested for.

This field MUST be encoded according to the configured NodeID base Type. Refer to *Z-Wave API Setup Set NodeID Base Type Sub Command* and Table 4.64.

2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.145.

Table 4.145: Get Priority Route Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x92							
5/5..6	NodeID							
6/7	Repeater 0							
7/8	Repeater 1							
8/9	Repeater 2							
9/10	Repeater 3							
10/11	Route Speed							

NodeID (8/16 bits)

This field is used to indicate the destination NodeID for which the Priority Route corresponds to.

This field MUST be encoded according to the configured NodeID base Type. Refer to *Z-Wave API Setup Set NodeID Base Type Sub Command* and Table 4.64.

Repeater (4 bytes)

Refer to *Repeater (4 bytes)*.

Route Speed (8 bits)

Refer to *Route Speed (8 bits)*.

3. Callback data frame (Z-Wave Module → host)

None.

4.4.3.16 Lock Unlock Last Route Command

This command is used to lock or unlock all last working route. The Lock Unlock Last Route Command Identifier is 0x90.

Frame flow

The frame flow for this command is an *Acknowledged frame*.

1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.146.

Table 4.146: Lock Unlock Last Route Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x90							
5	Lock mode							

Lock mode (8 bits)

This command is used to indicate if the last working routes MUST be saved in the Z-Wave API module. This field value MUST be encoded as follows:

- The value MUST be set to 0x01, if the last working routes MUST be saved in the Z-Wave API module.
- The value MUST be set to 0x00, if the last working routes MUST NOT be saved in the Z-Wave API module.

2. Response data frame (Z-Wave Module → host)

None.

3. Callback data frame (Z-Wave Module → host)

None.

4.4.3.17 Set SUC NodeID Command

This command is used to configure a static/bridge controller to be a SUC/SIS node or not. The Primary Controller should use this function to set a static/bridge controller to be the SUC/SIS node. The Set SUC NodeID Command Identifier is 0x54.

Frame flow

The frame flow for this command is an *Acknowledged frame with response and callback*.

1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.147.

Table 4.147: Set SUC NodeID Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x54							
5/5..6	NodeID							
6/7	SUC state							
7/8	Tx Option							
8/9	Capabilities							
9/10	Session identifier							

NodeID (8/16 bits)

This field is used to advertise the NodeID of the controller node that MUST take the SUC/SIS node.

This field MUST be encoded according to the configured NodeID base Type. Refer to *Z-Wave API Setup Set NodeID Base Type Sub Command* and Table 4.64.

SUC state (8 bits)

This field is used to enable/disable the SUC/SIS functionalities. The field MUST be encoded as follow:

- If the static/bridge controller are targeted to be a SUC/SIS node, the field value MUST be set to 0x01.
- If the static/bridge controller should not be a SUC/SIS node, the field value MUST be set to 0x00.

Tx Options (8 bits)

Refer to *Tx Options (8 bits)*.

Capabilities (8 bits)

This field is used to advertise the SUC capabilities that can be enabled on the Z-Wave API module. This field MUST be encoded according to Table 4.148.

Table 4.148: Set SUC NodeID Capabilities Value encoding

Value	Description
0x00	*Reserved* This reserved value MUST NOT be used.
0x01	*ZW_SUC_FUNC_NODEID_SERVER* This flag is used to enable the NodeID server functionality to become a SIS.
0x02..0xFF	*Reserved* Reserved values MUST NOT be used.

Session identifier (8 bits)

Refer to *Session identifier (8 bits)*.

2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.149.

Table 4.149: Set SUC NodeID Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x54							
5	Command Status							

Command Status (8 bits)

Refer to *Command Status (8 bits)*.

3. Callback data frame (Z-Wave Module → host)

A Z-Wave module MUST issue a callback frame formatted according to Table 4.150.

Table 4.150: Set SUC NodeID Command - Callback data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x54							
5	Session identifier							
6	Set SUC Status							

Session identifier (8 bits)

Refer to *Session identifier (8 bits)*.

Set SUC Status (8 bits)

This field is used to indicate the statue regarding the configuration of a static/bridge controller to be SUC/SIS node. This field MUST be encoded according to Table 4.151.

Table 4.151: Set SUC NodeID Status Value encoding

Value	Flag	Description
0x00..0x04	Reserved	Reserved values MUST NOT be used.
0x05	ZW_SUC_SET_SUCCEEDED	the process of configuring the static/bridge controller is ended successfully.
0x06	ZW_SUC_SET_FAILED	the process of configuring the static/bridge controller is failed.
0x07..0xFF	Reserved	Reserved values MUST NOT be used..

4.4.3.18 Delete SUC Return Route Command

This command is used to request the deletion of the SUC/SIS return routes. The Delete Return Route Command Command Identifier is 0x55.

Frame flow

The frame flow for this command is an *Acknowledged frame with response and callback*.

1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.152.

Table 4.152: Delete SUC Return Route Command Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x55							
5/5..6	NodeID							
6/7	Session identifier							

NodeID (8/16 bits)

This field is used to indicates the NodeID for which the SUC return routes is requested to be deleted.

This field MUST be encoded according to the configured NodeID base Type. Refer to *Z-Wave API Setup Set NodeID Base Type Sub Command* and Table 4.64.

Session identifier (8 bits)

Refer to *Session identifier (8 bits)*.

2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.153.

Table 4.153: Delete SUC Return Route Command Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x55							
5	Delete SUC Return Route Response							

Delete SUC Return Route Response (8 bits)

This field is used to advertise the response of the Z-Wave module regarding the acceptance of the *Delete SUC Return Route Command Initial data frame*. This field MUST be encoded as follow:

- If delete SUC return route operation is started, this field value MUST be set to 0x01.
- If an “assign/delete return route” operation is already active, this field value MUST be set to 0x00.

Callback data frame (Z-Wave Module → host)

A Z-Wave module MUST issue a callback frame formatted according to Table 4.154.

Table 4.154: Delete SUC Return Route Command Command - Callback data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x55							
5	Session identifier							
6	Tx Status							

Session identifier (8 bits)

Refer to *Session identifier (8 bits)*.

Tx Status (8 bits)

Refer to *Tx Status (8 bits)*.

4.4.3.19 Send SUC NodeID Command

This command is used to trigger the transfer of SUC/SIS NodeID from Primary/Static controller to a given controller NodeID. The Send SUC NodeID Command Identifier is 0x57.

Frame flow

The frame flow for this command is an *Acknowledged frame with response and callback*.

1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.155.

Table 4.155: Send SUC NodeID Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x57							
5/5..6	NodeID							
6/7	Tx Options							
7/8	Session identifier							

NodeID (8/16 bits)

This field is used to advertise the NodeID of a controller that will receive the current SUC/SIS NodeID.

This field MUST be encoded according to the configured NodeID base Type. Refer to *Z-Wave API Setup Set NodeID Base Type Sub Command* and Table 4.64.

Tx Options (8 bits)

Refer to *Tx Options (8 bits)*.

Session identifier (8 bits)

Refer to *Session identifier (8 bits)*.

2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.156

Table 4.156: Send SUC NodeID Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x57							
5	Command Status							

Command Status (8 bits)

Refer to *Command Status (8 bits)*.

3. Callback data frame (Z-Wave Module → host)

A Z-Wave module MUST issue a callback frame formatted according to Table 4.157.

Table 4.157: Send SUC NodeID Command - Callback data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x57							
5	Session identifier							
6	Tx Status							

Session identifier(8 bits)

Refer to *Session identifier (8 bits)*.

Tx Status (8 bits)

Refer to *Tx Status (8 bits)*.

4.4.3.20 Request Node Neighbor Discovery Command

This command is used to request a node to perform a new neighbor discovery and receive the updated list of neighbors. The Request Node Neighbor Discovery Command Identifier is 0x48.

Frame flow

The frame flow for this command is an *Acknowledged frame with callback*.

This command may trigger several callback frames. Examples are given in Figure 4.16 and Figure 4.17

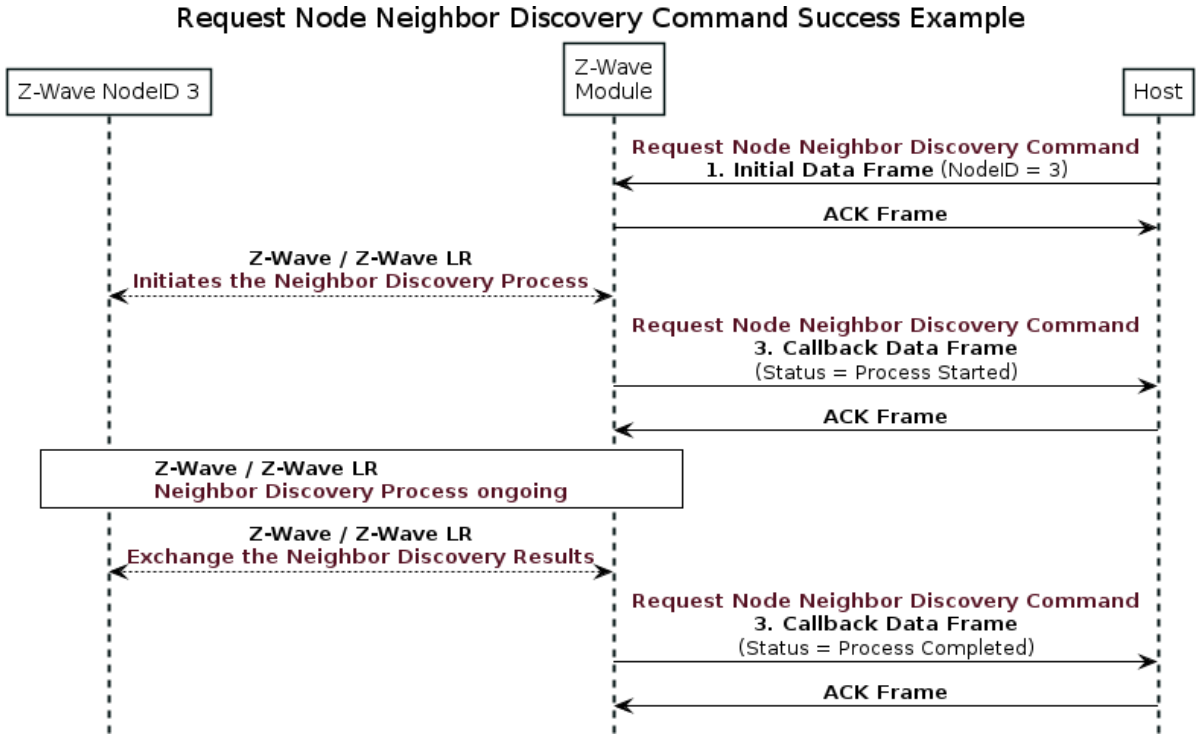

Figure 4.16: Request Node Neighbor Discovery Command Success Example

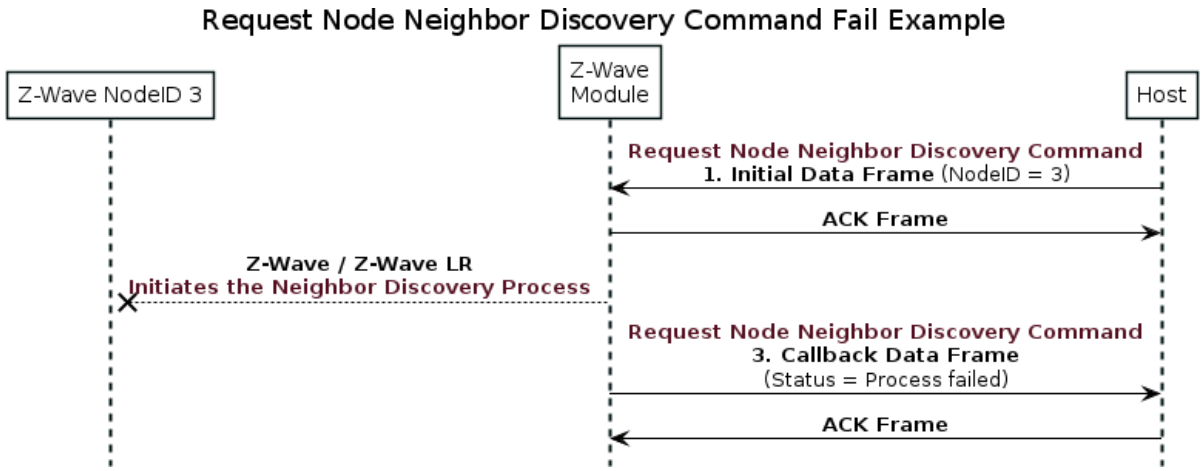

Figure 4.17: Request Node Neighbor Discovery Command Fail Example

Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.158

Table 4.158: Request Node Neighbor Discovery Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x48							
5/5..6	NodeID							
7/8	Session Identifier							

NodeID (8/16 bits)

This field indicates the NodeID that must perform a new discovery of its neighbors.

This field MUST be encoded according to the configured NodeID base Type. Refer to *Z-Wave API Setup Set NodeID Base Type Sub Command* and Table 4.64.

Session Identifier (8 bits)

Refer to *Session identifier (8 bits)*.

1. Response data frame (Z-Wave Module → host)

None

2. Callback data frame (Z-Wave Module → host)

A Z-Wave module MUST issue a callback frame formatted according to Table 4.159

Table 4.159: Request Node Neighbor Discovery Command - Callback data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x48							
5	Session Identifier							
6	Neighbor Discovery Status							

Session Identifier (8 bits)

Refer to *Session identifier (8 bits)*.

Neighbor Discovery Status (8 bits)

This field is used to indicate the current status of the Neighbor Discovery Process. Refer to [zwave_nwk_spec] for details about the Neighbor Discovery Process.

This field MUST be encoded as a bitmask and according to Table 4.160

Table 4.160: Request Node Neighbor Discovery Command - Neighbor Discovery Status encoding

Value	Description
0x21	*Neighbor Discovery Started* This value is used to indicate that the Neighbor Discovery Process has started.
0x22	*Neighbor Discovery Completed* This value is used to indicate that the Neighbor Discovery Process has been completed successfully.
0x23	*Neighbor Discovery Failed* This value is used to indicate that the Neighbor Discovery Process failed.
0xFF	*Neighbor Discovery Not Supported* This value is used to indicate that the functionality is not supported by one of the nodes.

4.4.3.21 Request Network Update Command

This command is used to instruct the Z-Wave API Module to request an Automatic Controller Update to the SUC. The Request Network Update Command Identifier is 0x53.

A host application SHOULD use this command only if there is a SUC in the current network.

A Z-Wave API Module receiving Node information updates during the Automatic Controller Update process MUST issue unsolicited *Application Update Command* to the host application.

Frame flow

The frame flow for this command is an *Acknowledged frame with response and callback*.

This command MAY trigger additional unsolicited frames from the Z-Wave API Module. An example of the expected frame flow is shown in Figure 4.18

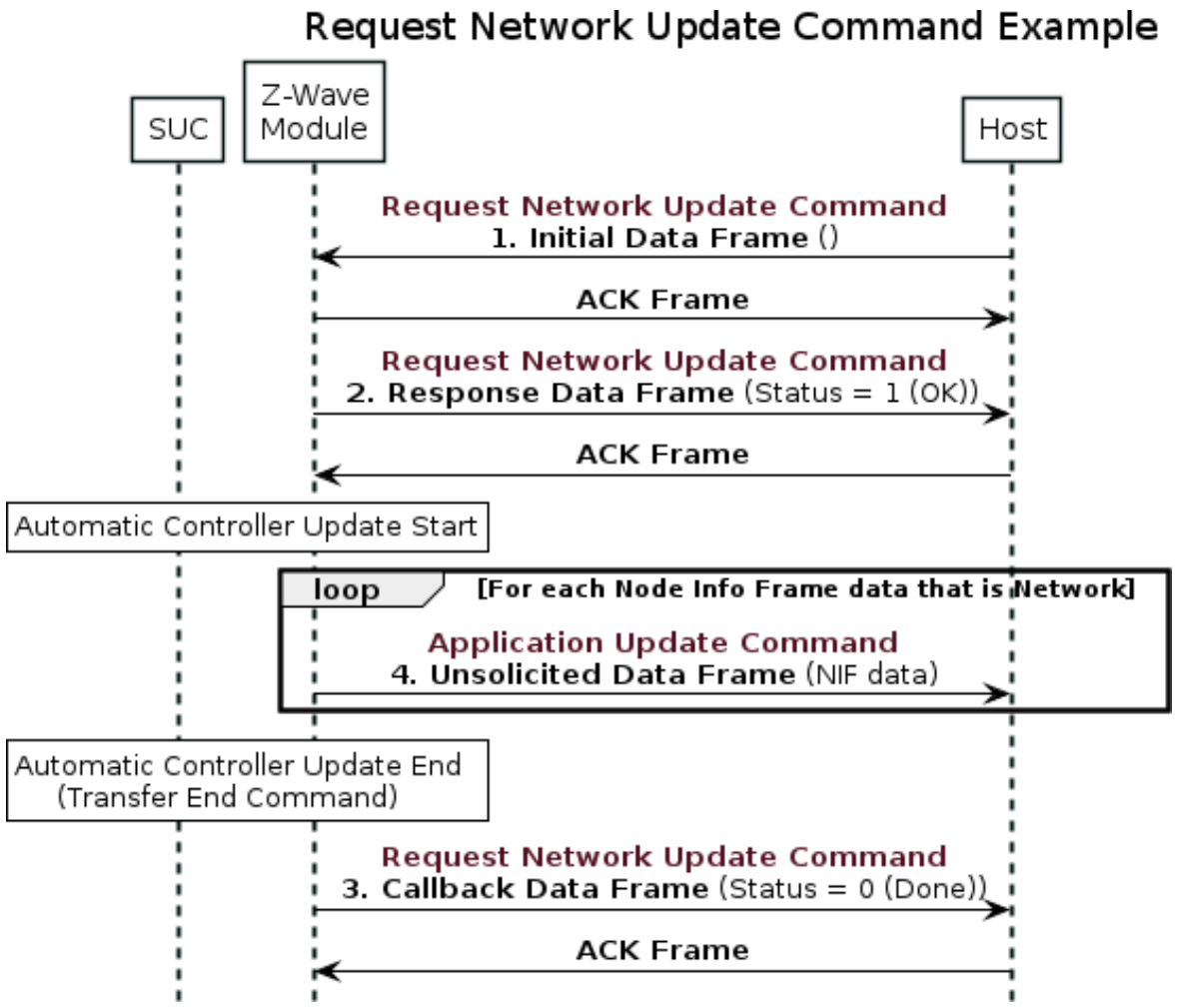

Figure 4.18: Request Network Update Command Example

1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.161.

Table 4.161: Request Network Update Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x53							
5	Session identifier							

Session identifier (8 bits)

Refer to *Session identifier (8 bits)*.

2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.162.

Table 4.162: Request Network Update Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x53							
5	Command Status							

Command Status (8 bits)

Refer to *Command Status (8 bits)*.

3. Callback data frame (Z-Wave Module → host)

A Z-Wave module MUST issue a callback frame formatted according to Table 4.163

Table 4.163: Z-Wave API Setup Set NodeID Base Type Sub Command - Callback data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x53							
5	Session identifier							
6	Network Update Status							

Session identifier (8 bits)

Refer to *Session identifier (8 bits)*.

Network Update Status (8 bits)

This field is used to advertise the outcome of the network update request. This field MUST be encoded

according to Table 4.164.

Table 4.164: Network Update Status Value encoding

Value	Flag	Description
0x00	ZW_SUC_UPDATE_DONE	The network update process succeeded.
0x01	ZW_SUC_UPDATE_ABORT	The network update process aborted because of an error.
0x02	ZW_SUC_UPDATE_WAIT	The SUC node is busy.
0x03	ZW_SUC_UPDATE_DISABLED	The SUC functionality is disabled.
0x04	ZW_SUC_UPDATE_OVERFLOW	The controller requested an update after more than 64 changes have occurred in the network. The update information is then out of date in respect to that controller. In this situation the controller have to make a replication (copy) before trying to request any new network updates.
0x05..0xFF	Reserved	These options are reserved, and the reserved values MUST NOT be used.

4.4.3.22 Set Virtual Node To Learn Mode Command

This command is used to enable or disable a virtual end node to *Learn Mode* operation that facilitates the node to be included or removed to/from a Z-Wave Network. The Set Virtual Node Learn Mode Command Identifier is 0xA4.

This command MUST only be supported by nodes implementing a Bridge Controller library type (refer to Table 4.36).

NOTE: A *Learn Mode* should only be enabled on a virtual end node when necessary, and it should always be disabled again as quickly as possible. It is recommended that the *Learn Mode* should not enabled for more than 1 second on a virtual end node.

Frame flow

The frame flow for this command is an *Acknowledged frame with callback*.

1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.165.

Table 4.165: Set Virtual Node To Learn Mode Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0xA4							
5	NodeID							
6	Mode							
7	Session identifier							

NodeID (8 bits)

This field is used to advertise the virtual NodeID that is set to *Learn Mode*.

This field MUST be encoded using 8 bits regardless of the configured NodeID base Type. Refer to *Z-Wave API Setup Set NodeID Base Type Sub Command* and Table 4.64.

Mode (8 bits)

This field is used to indicate the *Learn Mode* intent. This field MUST be encoded according to Table 4.166.

Table 4.166: Virtual End Node Learn Mode Encoding

Value	Flag and description
0x00	*VIRTUAL_END_NODE_LEARN_MODE_DISABLE*. This option is used to disable the *Learn Mode* operation on virtual end node.
0x01	*VIRTUAL_END_NODE_LEARN_MODE_ENABLE*. This option is used to enable the *Learn Mode* operation on virtual end node.

0x02	*VIRTUAL_END_NODE_LEARN_MODE_ADD*. This option is used to create locally a virtual end node and add it to the Z-Wave network (this will only be possible bridge controller node is a Primary/Inclusion Controller).
0x03	*VIRTUAL_END_NODE_LEARN_MODE_REMOVE*. This option is used to remove locally present virtual end node from the Z-Wave network (this will only be possible bridge node controller is a Primary/Inclusion Controller).
0x04..0xFF	*Reserved* These options are reserved, and reserved values MUST not be used.

Session identifier (8 bits)

Refer to *Session identifier (8 bits)*.

2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.167.

Table 4.167: Set Virtual Node To Learn Mode Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0xA4							
5	Response status							

Response status (8 bits)

Refer to *Response status (8 bits)*.

3. Callback data frame (Z-Wave Module → host)

A Z-Wave module MUST issue a callback frame formatted according to Table 4.168 to notify the status of the *Learn Mode* operation.

Table 4.168: Set Virtual Node To Learn Mode Command - Callback data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0xA4							
5	Session identifier							
6	Status							
7	Original NodeID							
8	New NodeID							

Session identifier (8 bits)

Refer to *Session identifier (8 bits)*.

Status (8 bits)

This field is used to indicate the status of the *Learn Mode* process. This field MUST be encoded according to Table 4.169.

Table 4.169: Virtual End Node Learn Mode Status Encoding

Value	Flag and description
0x00	*ASSIGN_COMPLETE* This option is used to indicate the *Learn Mode* is enabled and assignment process is done.
0x01	*ASSIGN_NODEID_DONE* This option is used to indicate the NodeID has been assigned.
0x02	*ASSIGN_RANGE_INFO_UPDATE* This option is used to indicate the node is doing neighbour discovery. The application should not attempt to send any frames during this time.
0x03..0xFF	*Reserved* These options are reserved, and reserved values MUST not be used.

Original NodeID (8 bits)

This field is used to advertise the original NodeID of the virtual end node when it was set to *Learn Mode* operation.

This field MUST be encoded using 8 bits regardless of the configured NodeID base Type. Refer to *Z-Wave API Setup Set NodeID Base Type Sub Command* and Table 4.64.

New NodeID (8 bits)

This field is used to advertise the new assigned NodeID. If the virtual end node is deleted, this field MUST be set to 0x00.

This field MUST be encoded using 8 bits regardless of the configured NodeID base Type. Refer to *Z-Wave API Setup Set NodeID Base Type Sub Command* and Table 4.64.

4.4.3.23 Virtual Node Send Node Information Command

This command is used to create and transmit a virtual end node *Node Information Frame*. The Virtual Node Send Node Information Command Identifier is 0xA2.

This command MUST only be supported by nodes implementing a Bridge Controller library type (refer to Table 4.36).

Frame flow

The frame flow for this command is an *Acknowledged frame with callback*.

1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.170.

Table 4.170: Virtual Node Send Node Information Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0xA2							
5/6	Source NodeID							
6/7	Destination NodeID							
7/8	Tx Options							
8/9	Session identifier							

Source NodeID (8/16 bits)

This field is used to indicate the virtual NodeID where the *Node Information Frame* is sent from. The field size could be 8 bits or 16 bits depends on employed physical layer module such as Z-Wave or Z-Wave Long Range, respectively.

Destination NodeID (8/16 bits)

This field is used to indicate the destination NodeID where the *Node Information Frame* will be sent to. The field size could be 8 bits or 16 bits depends on employed physical layer module such as Z-Wave or Z-Wave Long Range, respectively.

Tx Options (8 bits)

Refer to *Tx Options (8 bits)*.

Session identifier(8 bits)

Refer to *Session identifier (8 bits)*.

2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.171.

Table 4.171: Virtual Node Send Node Information Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0xA2							
5	Response status							

Response status (8 bits)

Refer to *Response status (8 bits)*.

3. Callback data frame (Z-Wave Module → host)

A Z-Wave module MUST issue a callback frame formatted according to Table 4.172 after the *Node Information Frame* transmission is performed.

Table 4.172: Virtual Node Send Node Information Command - Callback data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0xA2							
5	Session identifier							
6	Tx Status							

Session identifier(8 bits)

Refer to *Session identifier (8 bits)*.

Tx Status (8 bits)

Refer to *Tx Status (8 bits)*.

4.4.3.24 Set Virtual Nodes Application Node Information Command

This command is used to configure the Node Information Data for the Virtual nodes own by the Z-Wave API Module. The Set Virtual Node Application Node Information Command Identifier is 0xA0.

This command MUST only be supported by nodes implementing a Bridge Controller library type (refer to Table 4.36). The frame flow for this command is an *Acknowledged frame*.

1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.173

Table 4.173: Set Virtual Nodes Application Node Information Command

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0xA0							
5	Virtual NodeID							
6	Device Option Mask							
7	Generic Device Type							
8	Specific Device Type							
9	Node Parameter Length							
9+1	Node Parameter 1							
…	…							
9+N	Node Parameter N							

Virtual NodeID (8 bits)

This field is used to indicate the virtual NodeID for which the Node information frame data must be configured.

This field MUST be encoded using 8 bits regardless of the configured NodeID base Type. Refer to *Z-Wave API Setup Set NodeID Base Type Sub Command* and Table 4.64.

Device Option Mask (8 bits)

The device option mask is a bitmask where Listening and Optional functionality flags MUST be set accordingly to the nodes capabilities. This field MUST comply with the format indicated in Table 4.174.

Table 4.174: Set Application Node Information Command - Device Option Mask encoding

Bit	Flags	Description
0..6	*Reserved*	*Reserved*
7	Listening flag	This bit indicates if the node should be an Always Listening Node. (AL Node) The value 0 MUST indicate that the node MUST NOT be AL (either NL or FL) The value 1 MUST indicate that the node MUST be an AL node.

Generic Device Type (8 bits)

The Generic Device Class field contains an identifier that identifies what Generic Device Class the Z-Wave node MUST advertise and MUST be set by the application. For a detailed description of all available Generic Device Classes, refer to [device_class_spec] for Z-Wave devices, [device_type_spec] for Z-Wave Plus devices, and [device_type_spec_v2] for Z-Wave Plus v2 devices.

Specific Device Type (8 bits)

The Specific Device Class field contains an identifier that identifies what Specific Device Class the Z-Wave node MUST advertise and MUST be set by the application. For a detailed description of all available Generic Device Classes, refer to [device_class_spec] for Z-Wave devices, [device_type_spec] for Z-Wave Plus devices, and [device_type_spec_v2] for Z-Wave Plus v2 devices.

Node Parameter length (8 bits)

This field MUST specify the length of the Node Parameter field in bytes.

Node Parameter (N bytes)

This field is used to advertise the list of supported Command Classes by the node.

2. Response data frame (Z-Wave Module → host)

None

3. Callback data frame (Z-Wave Module → host)

None

4.4.3.25 Set Z-Wave Long Range Shadow NodeIDs Commmand

This command is used to enable the use of Shadow NodeIDs in the Long Range capable controller. The command will enable the controller to use more NodeIDs than its native NodeID to transmit and receive frames. The shadow NodeID is assigned outside of the normal NodeID range for Z-Wave Long Range nodes.

The Set Z-Wave Long Range Shadow NodeIDs Commmand Identifier is 0xDD.

This command MUST only be supported by Z-Wave API Module using the Controller Bridge library types (refer to Table 4.36).

Frame flow

The frame flow for this command is an *Acknowledged frame*.

1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.175

Table 4.175: Set Z-Wave Long Range Shadow NodeIDs Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0xDD							
5	Z-Wave Long Range Shadow NodeIDs bitmask							

Z-Wave Long Range Shadow NodeIDs bitmask (8 bits)

This field is used to indicate which shadow NodeIDs must be enabled by the Z-Wave API Module.

This field MUST encoded as a bitmask and interpreted as follow:

- bit 0 MUST represent NodeID 2002.
- bit 1 MUST represent NodeID 2003.
- bit 2 MUST represent NodeID 2004.
- bit 3 MUST represent NodeID 2005.
- bit 4..7 are reserved.

2. Response data frame (Z-Wave Module → host)

None

3. Callback data frame (Z-Wave Module → host)

None

4.5 Z-Wave API Memory Commands

This section describes *Z-Wave API Commands* that are used to interact with the Z-Wave Module memory or storage.

4.5.1 Get Network IDs from Memory Command

This command is used to get the HomeID and NodeID from the Z-Wave Module. The Get Network IDs from Memory Command Identifier is 0x20.

4.5.1.1 Frame flow

The frame flow for this command is an *Acknowledged frame with response*.

4.5.1.2 1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.176

Table 4.176: Get Network IDs from Memory Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x20							

4.5.1.3 2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.177

Table 4.177: Get Network IDs from Memory Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x20							
5	HomeID 1							
6	HomeID 2							
7	HomeID 3							
8	HomeID 4							
9/9..10	NodeID							

HomeID (4 bytes)

This field is used to advertise the current HomeID of the Z-Wave API Module.

NodeID (8 bits/16 bits)

This field is used to indicate the NodeID currently assigned to the Z-Wave API Module.

This field MUST be encoded according to the configured NodeID base Type. Refer to *Z-Wave API Setup Set NodeID Base Type Sub Command* and Table 4.64.

4.5.1.4 3. Callback data frame (Z-Wave Module → host)

None.

4.6 Z-Wave API Firmware Update Commands

This section describes *Z-Wave API Commands* that are used to perform firmware update operations.

4.6.1 NVM Operations Command

This command is used to read and write the firmware data of the Z-Wave API Module. The NVM Operations Command Identifier is 0x2E.

4.6.1.1 Frame flow

The frame flow for this command is an *Acknowledged frame with response*.

This command has sub-commands that allow either read or write operations.

The recommended frame flow for reading the firmware data of a Z-Wave API module is shown in Figure 4.19

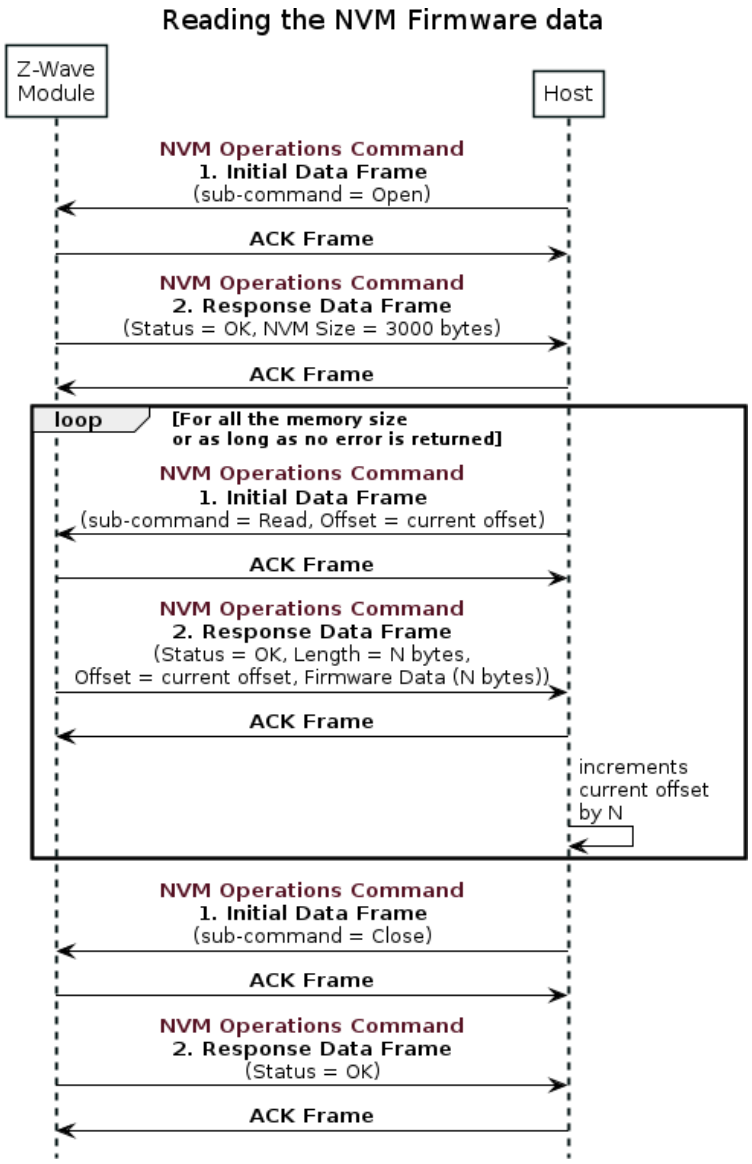

Figure 4.19: NVM Read Operation

The recommended frame flow for writing firmware data to a Z-Wave API module is shown in Figure 4.20. The host application MUST send a *Soft Reset Command* in order to activate the new firmware.

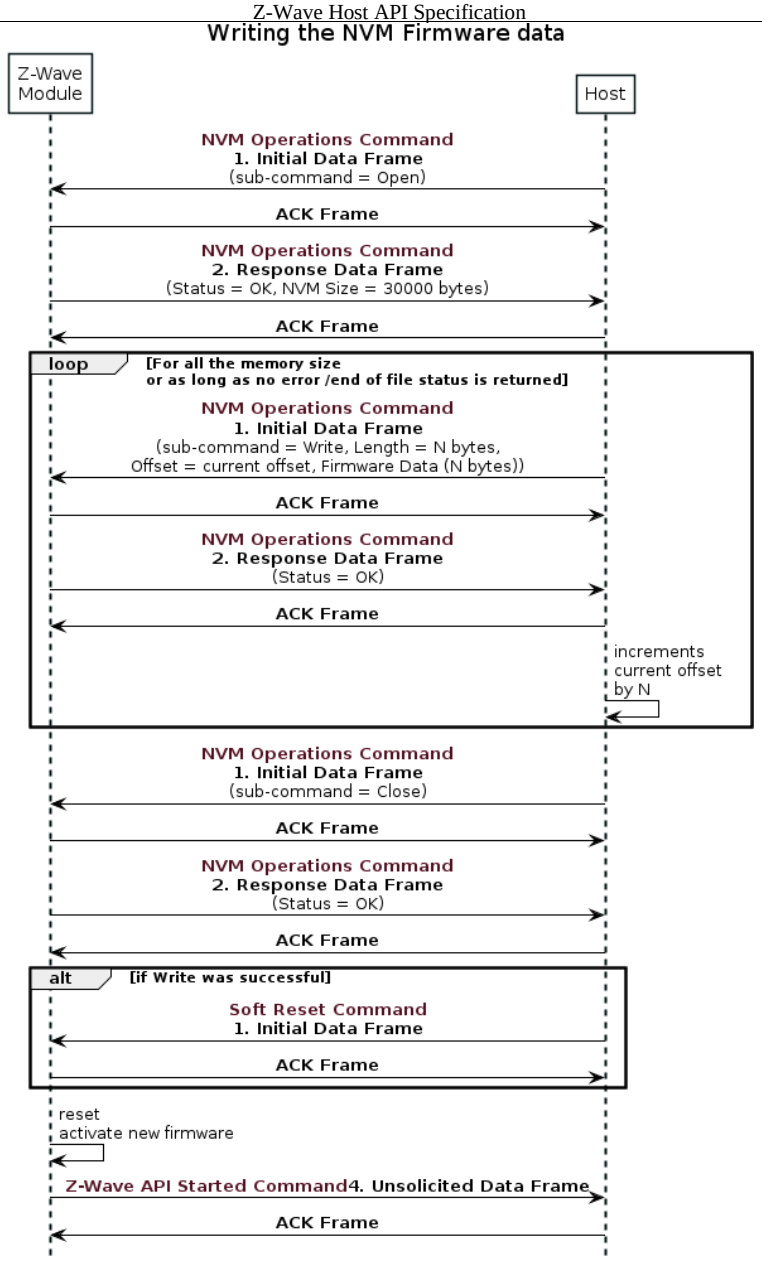

Figure 4.20: NVM Write Operation

4.6.1.2 1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.178

Table 4.178: NVM Operations Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x2E							
5	NVM Operation sub-command							
6	Firmware Data Length							

7	Address Offset (MSB)
8	Address Offset (LSB)
8+1	Firmware Data 1
…	…
8+N	Firmware Data N

NVM Operation sub-command (8 bits)

This field is used to indicate which operation to perform. This field MUST be encoded according to Table 4.179

Table 4.179: NVM Operations Command - NVM Operation sub-command encoding

Value	Description
0x00	*Open* This value is used to open the NVM for a subsequent read or write operation.
0x01	*Read* This value is used to read the NVM contents.
0x02	*Write* This value is used to write data to the NVM contents.
0x03	*Close* This value is used to close the NVM when the read/write operation is completed.

Firmware Data Length (8 bits)

This field is used to specify the length of the data that should be read/written from/to the Z-Wave API firmware data.

This field SHOULD be omitted for if the *NVM Operation sub-command* is set to Open (0x00) or Close (0x03).

If the *NVM Operation sub-command* is set to Write (0x02), this field MUST indicate the length of the *Firmware Data* field, in bytes.

Address Offset (16 bits)

This field is used to specify a memory address offset for read/write operations.

This field SHOULD be omitted for if the *NVM Operation sub-command* is set to Open (0x00) or Close (0x03).

If the *NVM Operation sub-command* is set to Read (0x01) or Write (0x02), this field MUST indicate the address offset for which the data must be read/written.

Firmware Data (N bytes)

This field is used to specify data to write to the Z-Wave API Module firmware memory.

This field SHOULD be omitted for if the *NVM Operation sub-command* is set to a different value than Write (0x02).

The Z-Wave API Module MUST write the data indicated in this field starting at the address offset indicated by the *Address Offset* field.

4.6.1.3 2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.180

Table 4.180: NVM Operations Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x2E							
5	NVM Operation sub-command status							
6	Firmware Data Length							
7	Address Offset / NVM Size (MSB)							
8	Address Offset / NVM Size (LSB)							
8+1	Firmware Data 1							
…	…							
8+N	Firmware Data N							

NVM Operation sub-command status (8 bits)

This field is used to indicate the status of the requested operation in the initial data frame. This field MUST be encoded according to Table 4.181

Table 4.181: NVM Operations Command - NVM Operation sub-command status encoding

Value	Description
0x00	*OK* This value is used to indicate that the requested operation was successful.
0x01	*Error* This value is used to indicate that an error occurred.
0x02	*Error: Operation Mismatch* This value is used to indicate that the wrong operation was requested (e.g. write in the middle of a read operation or vice-versa)
0x03	*Error: Operation interference* This value is used to indicate that the read operation cannot be performed or trusted due to another process writing in the Firmware Data.
0xFF	*End Of File* This value is used to indicate that the read/write operation has reached the end of the file/memory area. The host application SHOULD issue a Close sub-command when receiving this value.

Firmware Data Length (8 bits)

This field is used to specify the length of the data that is present in the *Firmware Data* field.

This field will be different than 0 only for Read (0x01) operation responses.

Address Offset / NVM Size (16 bits)

This field is used to specify a memory address offset for read/write operations.

This field SHOULD be ignored for if the *NVM Operation sub-command* is set to Write (0x02) or Close (0x03).

If the *NVM Operation sub-command* was set to Read (0x01) in the 1. Initial Data Frame, this field MUST indicate the address offset for which the data is being read.

If the *NVM Operation sub-command* was set to Open (0x00) in the 1. Initial Data Frame, this field MUST indicate the total size of the Firmware Memory, in bytes.

Firmware Data (N bytes)

This field is used to adverties the read data from the Z-Wave API Module firmware memory.

This field SHOULD be ignored for if the *NVM Operation sub-command* was set not to Read (0x01) in the 1. Initial Data Frame.

4.6.1.4 3. Callback data frame (Z-Wave Module → host)

None.

4.7 Unsolicited Z-Wave API commands

This section describes *Z-Wave API Commands* that are used to intialize and configure the Z-Wave module. It also comprises commands that are used to read the supported functionality of the Z-Wave API module.

4.7.1 Application Command Handler Command

This command is used by a Z-Wave module to notify a host application that a Z-Wave frame has been received. The Application Command Handler Command Identifier is 0x04

4.7.1.1 Frame flow

The frame flow for this command is an *Unsolicited frame*.

4.7.1.2 1. Initial data frame (host → Z-Wave Module)

None

4.7.1.3 2. Response data frame (Z-Wave Module → host)

None

4.7.1.4 3. Callback data frame (Z-Wave Module → host)

None

4.7.1.5 4. Unsolicited frame (Z-Wave Module → host)

A Z-Wave module MUST issue an unsolicited frame formatted according to Table 4.182

Table 4.182: Application Command Handler Command - Unsolicited data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x04							
5	Rx Status							
6/6..7	Source NodeID (8/16 bits)							
7/8	Payload Length							
8/9	Payload 1							
…	…							
8/9+N	Payload N							
9/10+N	Rx RSSI Value							

Rx Status (8 bits)

This field is used to advertise additional information about how the Z-Wave frame was received.

This field MUST be treated as a bitmask and encoded according to Table 4.1.

Source NodeID (8/16 bits)

This field is used to advertise the NodeID from which the Z-Wave Command was received.

This field MUST be encoded according to the configured NodeID base Type. Refer to *Z-Wave API Setup Set NodeID Base Type Sub Command* and Table 4.64.

Payload Length (8 bits)

This field is used to advertise the length in bytes of the *Payload* field.

Payload (N bytes)

This field is used to report the payload that was received from the Source NodeID.

Rx RSSI Value (8 bits)

This field is used to report the measured RSSI value for the received Z-Wave frame. This field MUST be encoded according to Table 4.3.

4.7.2 Z-Wave API Started Command

This command is used by the Z-Wave Module to indicate that it is ready to be operated after a reboot or reset operation.

The Z-Wave API Started Command Command Identifier is 0x0A

4.7.2.1 Frame flow

The frame flow for this command is an *Unsolicited frame*.

4.7.2.2 1. Initial data frame (host → Z-Wave Module)

None

4.7.2.3 2. Response data frame (Z-Wave Module → host)

None

4.7.2.4 3. Callback data frame (Z-Wave Module → host)

None

4.7.2.5 4. Unsolicited frame (Z-Wave Module → host)

A Z-Wave module MUST issue an unsolicited frame formatted according to Table 4.183

Table 4.183: Z-Wave API Started Command - Unsolicited data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x0A							
5	Wake Up Reason							
6	Watchdog Started							
7	Device Option Mask							
8	Generic Device Type							
9	Specific Device Type							
10	Command Class List Length							
11	Command Class List 1							
...	...							
11+N	Command Class List N							
11+N+1	Supported Protocols							

Wake Up Reason (8 bits)

This field is used to advertise the event that caused the Z-Wave API module to start. It MUST be encoded according to Table 4.184.

Table 4.184: Z-Wave API Started - Wake Up Reason encoding

Value	Description
0x00	*Reset* The Z-Wave API Module has been woken up by reset or external interrupt.
0x01	*Wake Up Timer* The Z-Wave API Module has been woken up by a timer.
0x02	*Wake Up Beam* The Z-Wave API Module has been woken up by a Wake Up Beam.
0x03	*Watchdog reset* The Z-Wave API Module has been woken up by a reset triggered by the watchdog.
0x04	*External interrupt* The Z-Wave API Module has been woken up by an external interrupt.
0x05	*Power Up* The Z-Wave API Module has been woken up by a powering up.
0x06	*USB Suspend* The Z-Wave API Module has been woken up by USB Suspend.
0x07	*Software reset* The Z-Wave API Module has been woken up by a reset triggered by software.
0x08	*Emergency Watchdog Reset* The Z-Wave API Module has been woken up by an emergency watchdog reset.
0x09	*Brownout circuit* The Z-Wave API Module has been woken up by a reset triggered by brownout circuit.
0x0A..0xFE	*Reserved values MUST NOT be used and MUST be ignored by a receiving interface*
0xFF	*Unknown* The Z-Wave API Module has been woken up by an unknown reason.

Watchdog Started (8 bits)

This field is used to advertise if the Watchdog is enabled.

The value 0x00 MUST indicate that thw watchdog is disabled. The value 0x01 MUST indicate that thw watchdog is enabled.

Device Option Mask (8 bits)

This field is used to advertise the currently configured listening capabilities configured for the Z-Wave API Module.

The host application MAY change this using the *Set Application Node Information Command*.

Generic Device Type (8 bits)

This field is used to advertise the currently configured Generic Device Type.

The host application MAY change this using the *Set Application Node Information Command*.

Specific Device Type (8 bits)

This field is used to advertise the currently configured Specific Device Type.

The host application MAY change this using the *Set Application Node Information Command*.

Command Class List Length (8 bits)

This field is used to advertise the length of the *Command Class List* in bytes.

Command Class List (N bytes)

This field is used to advertise the list of supported Command Classes advertised by the Z-Wave API Module upon request.

The host application MAY change this using the *Set Application Node Information Command*.

Supported Protocols (8 bits)

This field is used to advertise additional supported protocols by the Z-Wave API module. This field MUST be encoded as a bitmask and MUST be encoded according to Table 4.185.

Table 4.185: Z-Wave API Started Command - Supported Protocols encoding

Bit	Flag	Description
0	*Z-Wave Long Range*	This bit indicates if the Z-Wave API module supports Z-Wave Long range. • The value 0 MUST indicate that the Z-Wave API module does not support Z-Wave Long Range • The value 1 MUST indicate that the Z-Wave API module does supports Z-Wave Long Range
1..7	*Reserved*	These bits are reserved. Reserved bits MUST NOT be used and MUST be ignored by a receiving interface.

4.7.3 Application Update Command

This command is used to update node information data structures and to control SmartStart inclusion. The Application Update Command Identifier is 0x49.

This command is used by during the following conditions:

If the Z-Wave API Module runs an End Node library type (refer to Table 4.36), it MUST send this command to the host application when:

- It received a *Node Information Frame Command*.

If the Z-Wave API Module runs an Controller node library type (refer to Table 4.36), it MUST send this command to the host application when:

- It received a *Node Information Frame Command*.
- It received a *SmartStart Prime Command* and it is in SmartStart Add Mode.
- It received an *Included Node Info Frame Command* and it is in SmartStart Add Mode.
- It has the SIS Role and a node is added or excluded from the network by a controller (Primary or Inclusion controller).
- It has received a change in the Node Information Frame data for a node during network topology update process (either Automatic Controller Update or Controller Replication).

Refer to [zwave_nwk_spec] for details.

4.7.3.1 Frame flow

The frame flow for this command is an *Unsolicited frame*.

4.7.3.2 1. Initial data frame (host → Z-Wave Module)

None.

4.7.3.3 2. Response data frame (Z-Wave Module → host)

None.

4.7.3.4 3. Callback data frame (Z-Wave Module → host)

None.

4.7.3.5 4. Unsolicited frame (Z-Wave Module → host)

The Z-Wave module issues several unsolicited Application Update request frames corresponding to the

information it receives over the Z-Wave media.

4.a. Unsolicited Application Update Command generic format

A Z-Wave API module MUST issue this unsolicited frame to the host application when one of the events described in the event field happened.

A Z-Wave API module implementing a *Controller Node* library type (refer to Table 4.36) MUST also issue this unsolicited frame when it receives a *Node Information Frame* as part of a network inclusion or when a node has been excluded from the network.

A Z-Wave module MUST issue this unsolicited frame formatted according to Table 4.186.

Table 4.186: Application Update Command - Unsolicited data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x49							
5	Event							
6/6..7	Remote NodeID							
7/8	Supported Command Class List Length							
8/9	Basic Device Class							
8/10	Generic Device Type							
10/11	Specific Device Type							
(10/11)+1	Supported Command Class List 1							
…	…							
(11/12)+N	Supported Command Class List N							

Event (8 bits)

This field is used indicate which event has triggered the transmission of this command. This field MUST be encoded according to Table 4.187

Table 4.187: Application Update Command - Event encoding

Value	Description
0x10	*UPDATE_STATE_SUC_ID* The SIS NodeID has been updated.
0x20	*UPDATE_STATE_DELETE_DONE* A node has been deleted from the network.
0x40	*UPDATE_STATE_NEW_ID_ASSIGNED* A new node has been added to the network.
0x40	*UPDATE_STATE_NEW_ID_ASSIGNED* A new node has been added to the network.
0x80	*UPDATE_STATE_ROUTING_PENDING* Another node in the network has requested the Z-Wave API Module to perform a neighbor discovery.
0x81	*UPDATE_STATE_NODE_INFO_REQ_FAILED* The issued *Request Node Information Command* has not been acknowledged by the destination.

0x82	*UPDATE_STATE_NODE_INFO_REQ_DONE* The issued *Request Node Information Command* has been acknowledged by the destination.
0x83	*UPDATE_STATE_NOP_POWER_RECEIVED* Another node sent a NOP Power Command to the Z-Wave API Module. The host application SHOULD NOT power down the Z-Wave API Module.
0x84	*UPDATE_STATE_NODE_INFO_RECEIVED* A *Node Information Frame* has been received as unsolicited frame or in response to a *Request Node Information Command*.
0x85	*UPDATE_STATE_NODE_INFO_SMARTSTART_HOMEID_RECEIVED* A *SmartStart Prime Command* has been received using the Z-Wave protocol.
0x86	*UPDATE_STATE_INCLUDED_NODE_INFO_RECEIVED* A *SmartStart Included Node Information Frame* has been received (using either Z-Wave or Z-Wave Long Range protocol).
0x87	*UPDATE_STATE_NODE_INFO_SMARTSTART_HOMEID_RECEIVED_LR* A *SmartStart Prime Command* has been received using the Z-Wave Long Range protocol.

Values that are not listed in Table 4.187 are reserved and MUST NOT be used.

Remote NodeID (8/16 bits)

This field is used to advertise the NodeID of the remote node for/with which the event occurred.

Supported Command Class List Length (8 bits)

This field is used to indicates the length of the *Supported Command Class List* field in bytes.

Basic Device Class (8 bits)

Refer to *Basic Device Class (8 bits)*.

Generic Device Type (8 bits)

This field is used to advertise the *Generic Device Type* of the remote NodeID.

For a detailed description of all available Generic Device Classes, refer to [device_class_spec] for Z-Wave devices, [device_type_spec] for Z-Wave Plus devices, and [device_type_spec_v2] for Z-Wave Plus v2 devices.

Specific Device Type (8 bits)

This field is used to advertise the *Specific Device Type* of the remote NodeID.

For a detailed description of all available Specific Device Classes, refer to [device_class_spec] for Z-Wave devices, [device_type_spec] for Z-Wave Plus devices, and [device_type_spec_v2] for Z-Wave Plus v2 devices.

Supported Command Class List (N bytes)

This field is used to advertise the list of non-secure supported Command Classes by the remote node.

4.b. Unsolicited Application Update Command with SmartStart Prime events

A Z-Wave API module implementing a *Controller Node* library type (refer to Table 4.36) MUST issue a unsolicited frame formatted according to Table 4.188 to the host application when it receives a *SmartStart Prime Command* or a *SmartStart Inclusion Node Information Frame*.

Table 4.188: Application Update Command - SmartStart Prime data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x49							
5	Event							
6/6..7	Remote NodeID							
7/8	Rx Status							
8/9	NWI HomeID 1							
9/10 | NWI HomeID 2							
10/11	NWI HomeID 3							
11/12	NWI HomeID 4							
12/13	Supported Command Class List Length							
13/14	Basic Device Class							
14/15	Generic Device Type							
15/16	Specific Device Type							
16/17	Supported Command Class List 1							
…	…							
17/18+N	Supported Command Class List N							

Refer to *4.a. Unsolicited Application Update Command generic format* for fields that are not described below.

Event (8 bits)

This field is used to inform the host application that a SmartStart Prime frame has been received.

This field MUST be set to one of the following values:

- UPDATE_STATE_NODE_INFO_SMARTSTART_HOMEID_RECEIVED (0x85) (if received using the Z-Wave Protocol
- UPDATE_STATE_NODE_INFO_SMARTSTART_HOMEID_RECEIVED_LR (0x87) if received using the Z-Wave Long Range Protocol

Refer to *Application Update Command - Event encoding* for details about these values.

Rx Status (8 bits)

Refer to *Rx Status (8 bits)*

NWI HomeID (4 bytes)

This field is used to advertise the NWI HomeID on which the *SmartStart Prime Command* was received.

4.c. Unsolicited Application Update Command with Include Node Information event

A Z-Wave API module implementing a *Controller Node* library type (refer to Table 4.36) MUST issue a unsolicited frame formatted according to Table 4.189 to the host application when it receives a *SmartStart Prime Command* or a *SmartStart Inclusion Node Information Frame*.

Table 4.189: Application Update Command - SmartStart INIF data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x49							
5	Event							
6/6..7	Remote NodeID							
7/8	Reserved							
8/9	Rx Status							
9/10	NWI HomeID 1							
10/11	NWI HomeID 2							
11/12	NWI HomeID 3							
12/13	NWI HomeID 4							

Refer to *4.a. Unsolicited Application Update Command generic format* for fields that are not described below.

Event (8 bits)

This field is used to inform the host application that a SmartStart Included Node Information frame has been received. This field MUST be set to UPDATE_STATE_INCLUDED_NODE_INFO_RECEIVED (0x86).

Refer to *Application Update Command - Event encoding* for defailts about these values.

Reserved (8 bits)

This field is obsoleted.

It SHOULD be set to 0 by a Z-Wave API Module and ignored by a Host Application.

Rx Status (8 bits)

Refer to *Rx Status (8 bits)*

NWI HomeID (4 bytes)

This field is used to advertise the NWI HomeID for which the *SmartStart Inclusion Node Information Frame* was received.

4.7.4 Bridge Application Command Handler Command

This command is used by a Z-Wave module to notify a host application that a Z-Wave frame has been received. The Bridge Application Command Handler Command Identifier is 0xA8.

This command MUST only be supported by Z-Wave API Modules implementing a *Bridge Controller library* (refer to Table 4.36). Z-Wave API Modules with another library type MUST use the *Application Command Handler Command* instead.

4.7.4.1 Frame flow

The frame flow for this command is an *Unsolicited frame*.

4.7.4.2 1. Initial data frame (host → Z-Wave Module)

None.

4.7.4.3 2. Response data frame (Z-Wave Module → host)

None.

4.7.4.4 3. Callback data frame (Z-Wave Module → host)

None.

4.7.4.5 4. Unsolicited frame (Z-Wave Module → host)

A Z-Wave module MUST issue an unsolicited frame formatted according to Table 4.190.

Table 4.190: Bridge Application Command Handler Command - Unsolicited data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0xA8							
5	Rx Status							
6/6..7	Destination NodeID							
7/8..9	Source NodeID							
8/10	Payload Length							
(8/10)+1	Payload 1							
..	…							
(9/10)+N	Payload N							
10/11+N	Multicast Destination Node Mask Length							
11/12+N	Multicast Destination Node Mask 1							
..	…							

11/12+N+M	Multicast Destination Node Mask M
12/13+N+M	Received RSSI

Rx Status (8 bits)

Refer to *Rx Status (8 bits)*.

Destination NodeID (8/16 bits)

This field is used to advertise the NodeID to which the Z-Wave Command is addressed.

This field MUST be encoded according to the configured NodeID base Type. Refer to *Z-Wave API Setup Set NodeID Base Type Sub Command* and Table 4.64.

If received frame is a multicast frame, the *Destination NodeID* is not valid.

Source NodeID (8/16 bits)

This field is used to advertise the NodeID from which the Z-Wave Command was received.

This field MUST be encoded according to the configured NodeID base Type. Refer to *Z-Wave API Setup Set NodeID Base Type Sub Command* and Table 4.64.

Payload Length (8 bits)

This field is used to indicate the length of the *Payload* field in bytes.

Payload (N bytes)

This field is used to advertise payload of the received Z-Wave frame. The very first byte will be a Command Class identifier.

The length of this field, in bytes, MUST be according to the *Payload Length* field.

Multicast Destination Node Mask Length (8 bits)

This field is used to indicate the length of the *Multicast Destination Node Mask* field in bytes.

Multicast Destination Node Mask (M bytes)

This field is used to indicate the destination nodes IDs using multicast addressing.

The length of this field, in bytes, MUST be according to the *Multicast Destination Node Mask Length* field.

Received RSSI (8 bits)

This field is used to indicate the received frame RSSI value. This field value MUST be encoded according to Table 4.3.

4.8 Z-Wave API Miscellaneous Commands

This section describes *Z-Wave API Commands* that do not belong in any of the other categories.

4.8.1 Clear Tx Timers Command

This command is used to clear/reset the Z-Wave Module internal Tx timers. The Tx timers are updated by the module when a frame is sent. The Clear Tx Timers Command Identifier is 0x37

4.8.1.1 Frame flow

The frame flow for this command is an *Acknowledged frame*.

4.8.1.2 1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.191

Table 4.191: Clear Tx Timers Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x37							

4.8.1.3 2. Response data frame (Z-Wave Module → host)

None

4.8.1.4 3. Callback data frame (Z-Wave Module → host)

None

4.8.2 Get Background RSSI Command

This command is used to request the most recent background RSSI levels detected. The Get Background RSSI Command Identifier is 0x3B.

NOTE: The RSSI shall only be measured when the radio is in receive mode.

4.8.2.1 Frame flow

The frame flow for this command is an *Acknowledged frame with response*.

4.8.2.2 1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.192

Table 4.192: Get Background RSSI Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x3B							

4.8.2.3 2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.193

Table 4.193: Get Background RSSI Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x3B							
5	RSSI CH0							
6	RSSI CH1							
7	RSSI CH2							

RSSI (2 or 3 bytes)

The RSSI fields are used to indicate the measured RSSI.

If the Z-Wave Module operates on a 2 channel RF Profile, RSSI CH0, and RSSI CH1 field MUSTTt contain the RSSI values of the first and second channel.

If the Z-Wave Module operates on a 3 channel RF profile, all three RSSI fields MUST contain RSSI values.

All RSSI measurements MUST be encoded according to Table 4.3

4.8.2.4 3. Callback data frame (Z-Wave Module → host)

None.

4.8.3 Get Tx Timer Command

This command is used to request the Z-Wave Module internal Tx timer. When the module receives this command, it MUST return the Tx timer for each channels. The Get Tx Timer Command Identifier is 0x38.

4.8.3.1 Frame flow

The frame flow for this command is an *Acknowledged frame with response*.

4.8.3.2 1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.194

Table 4.194: Get Tx Timer Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x38							

4.8.3.3 2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.195

Table 4.195: Get Tx Timer Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x38							
5	Tx Timer Channel 0							
6	Tx Timer Channel 1							
7	Tx Timer Channel 2							

Tx Timer Channel 0 (8 bits)

This field is used to advertise time [in miliseconds] the tranmistter has been active on channel 0 since the last reset.

Tx Timer Channel 1 (8 bits)

This field is used to advertise time [in miliseconds] the tranmistter has been active on channel 1 since the last reset.

Tx Timer Channel 2 (8 bits)

This field is used to advertise time [in miliseconds] the tranmistter has been active on channel 2 since the last reset.

4.8.3.4 3. Callback data frame (Z-Wave Module → host)

None

4.8.4 Get Virtual Nodes Command

This command is used to request available Virtual End Nodes in a Z-Wave Network. The Get Virtual Nodes Command Identifier is 0xA5.

4.8.4.1 Frame flow

The frame flow for this command is an *Acknowledged frame with response*.

4.8.4.2 1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.196.

Table 4.196: Get Virtual Nodes Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0xA5							

4.8.4.3 2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.197 to notify the available Virtual End Nodes to the application that requests the list of virtual node using Get Virtual Nodes Command request frame.

Table 4.197: Get Virtual Nodes Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0xA5							
5	NodeMask							

NodeMask(N bytes)

This field is used to indicate the bit mask of the virtual end NodeIDs in a Z-Wave network. The field value MUST be encoded according to:

- If bit ‘n’ in the NodeMask byte ‘i’ is 1, it indicates that node (i*8)+n+1 is a virtual end node.
- If bit ‘n’ in the NodeMask byte ‘i’ is 0, it indicates that node (i*8)+n+1 is not a virtual end node.

4.8.4.4 3. Callback data frame (Z-Wave Module → host)

None.

4.8.5 Get Z-Wave Module Protocol Status Command

This command is used to request the current status of the protocol runs on the Z-Wave module. The Get Z-Wave Module Protocol Status Command Identifier is 0xBF.

4.8.5.1 Frame flow

The frame flow for this command is an *Acknowledged frame with response*.

4.8.5.2 1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.198 to request the protocol status.

Table 4.198: Get Z-Wave Module Protocol Status Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0xBF							

4.8.5.3 2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.199 for notifying the current status of the Z-Wave protocol to the host application.

Table 4.199: Get Z-Wave Module Protocol Status Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0xBF							
5	Status							

Status (8 bits)

This field is used to advertise the current status of the protocol runs on the Z-Wave module. This field MUST be encoded according to Table 4.200

Table 4.200: Z-Wave protocol status encoding

Value	Flag	Description
0x00	IDLE	Protocol is idle.
0x01	ZW_PROTOCOL_STATUS_ROUTING	Protocol is analyzing the routing table.
0x02	ZW_PROTOCOL_STATUS_SUC	SIS sends pending updates.
0x03..0xFF	Reserved	Reserved values MUST NOT be used.

4.8.5.4 3. Callback data frame (Z-Wave Module → host)

None.

4.8.6 Is Virtual Node Command

This command is used to check if a given NodeID is a virtual end node. The Is Virtual Node Command Identifier is 0xA6.

4.8.6.1 Frame flow

The frame flow for this command is an *Acknowledged frame with response*.

4.8.6.2 1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.201.

Table 4.201: Is Virtual Node Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0xA6							
5	NodeID							

NodeID (8 bits)

This field is used to indicate the NodeID on node for which virtual node status is requested.

This field MUST be encoded using 8 bits regardless of the configured NodeID base Type. Refer to *Z-Wave API Setup Set NodeID Base Type Sub Command* and Table 4.64.

4.8.6.3 2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.202.

Table 4.202: Is Virtual Node Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0xA6							
5	Virtual node characteristic							

Virtual node characteristic (8 bits)

This field is used to advertise characteristic of the request NodeID. This field MUST be encoded as follow:

- If the NodeID is a virtual node, this field MUST be set to 0x01.
- If the NodeID is not a virtual node. this field MUST be set to 0x00.

4.8.6.4 3. Callback data frame (Z-Wave Module → host)

None.

4.8.7 Set Listen Before Talk Threshold Command

This command is used to to set the “Listen Before Talk” RSSI threshold that controls at what RSSI level a Z-Wave node will refuse to transmit because of noise. The default threshold value is set to a value corresponding to the RF regulatory requirements in the specific country. The Set Listen Before Talk Threshold Command Identifier is 0x3C.

4.8.7.1 Frame flow

The frame flow for this command is an *Acknowledged frame with response*.

This command SHOULD be used once for each channel configured by the Host Application.

4.8.7.2 1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.203

Table 4.203: Set Listen Before Talk Threshold Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x3C							
5	Channel							
6	RSSI Threshold							

Channel (8 bits)

This field is used to indicate the channel number where the RSSI threshold shall be set for. Valid channel numbers are 0, 1, 2 and 3.

Channel values 0..2 MUST indicate the Z-Wave channel number. Channel value 3 MUST indicate the Z-Wave Long Range Channel.

A Z-Wave API Module without Z-Wave Long Range support will ignore the channel value 3.

RSSI Threshold (8 bits)

This field is used to indicate the RSSI threshold that MUST be used by the Z-Wave API Module to detect the channel availability.

This field MUST be encoded according to This field MUST be encoded according to *RSSI Measurements (8 bits)* and Table 4.3. The values 125, 126 and 127 MUST NOT be used by a Host Application in this command and MUST be ignored by a Z-Wave API Module if received.

4.8.7.3 2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.204

Table 4.204: Set Listen Before Talk Threshold Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x3C							
5	Status							

Status (8 bits)

This field is used to indicate status regarding the Set Listen Before Talk Threshold request command.

- If the Listen Before Talk Threshold value is accepted by the Z-Wave Module, the field value MUST be set to 0x01.
- If the Listen Before Talk Threshold value is not accepted, the field value MUST be set to 0x00.

4.8.7.4 3. Callback data frame (Z-Wave Module → host)

None.

4.8.8 Set RF Receive Mode Command

This command is used to to power down the RF when not in use e.g., expects nothing to be received. It can also be used to set the RF into receive mode. This functionality is useful in battery powered Z-Wave nodes. The RF is automatic powered up when transmitting data.

The Set RF Receive Mode Command Identifier is 0x10.

4.8.8.1 Frame flow

The frame flow for this command is an *Acknowledged frame with response*.

4.8.8.2 1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.205

Table 4.205: Set RF Receive Mode Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x10							
5	Mode							

Mode (8 bits)

This field is used to advertise information about the Set RF Receive Mode operation mode. The feild MUST encoded with the format indicated in Table 4.206.

Table 4.206: Set RF Receive Mode Command - Mode encoding

Value	Flags	Description
0x01	On	Set the RF in receive mode and starts the receive data sampling
0x00	Off	Set the RF in power down mode (for battery power save).

4.8.8.3 2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.207

Table 4.207: Set Rf Receive Mode Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x10							
5	Status							

Status (8 bits)

This field is used to advertise information about the status of the Set RF Receive Mode operation. The

feild MUST encoded according to:

- 0x01, If the operation was successfull.
- 0x00, If the operation was not successfull.

3. Callback data frame (Z-Wave Module → host)

None

4.8.9 Set RF Power Level Command

This command is used to set the power level used for RF transmission. The Set RF Power Level Command Identifier is 0x17.

NOTE: This command should only be used in an install/test link situation and the power level should always be set back to normal Power when the testing is done.

4.8.9.1 Frame flow

The frame flow for this command is an *Acknowledged frame with response*.

4.8.9.2 1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.208

Table 4.208: Set RF Power Level Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x17							
5	PowerLevel							

Powerlevel (8 bits)

This field is used to indicate the power level that shall be used for the RF transmission. This field MUST comply with the format indicated in Table 4.209.

Table 4.209: Set RF Power Level Command - Power Level encoding

Value	Description
NormalPower	Normal transmit power configured on the device.
-1dB	NormalPower - 1dB
-2dB	NormalPower - 2dB
-3dB	NormalPower - 3dB
-4dB	NormalPower - 4dB
-5dB	NormalPower - 5dB
-6dB	NormalPower - 6dB
-7dB	NormalPower - 7dB
-8dB	NormalPower - 8dB
-9dB	NormalPower - 9dB

4.8.9.3 2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.210

Table 4.210: Set RF Power Level Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x17							
5	Powerlevel							

Powerlevel (8 bits)

This field is used to indicate the actual RF power level that can used for transmitting a given test frame.

4.8.9.4 3. Callback data frame (Z-Wave Module → host)

None.

4.8.10 Set Maximum Routing Attempts Command

This command is used to set the maximum number of source routing attempts based on the routing table lookups, and this shall be used before the Z-Wave protocol layer starts the dynamic route resolution. The Set Maximum Routing Retries Command Identifier is 0xD4.

4.8.10.1 Frame flow

The frame flow for this command is an *Acknowledged frame with response*.

4.8.10.2 1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.211 for setting the maximum source routing attempts.

Table 4.211: Set Maximum Routing Attempts Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0xD4							
5	Max Routing Retries							

Max Routing Retries (8 bits)

This field is used to indicate the maximum source routing attempts that can be used before the Z-Wave module triggers dynamic route resolution.

4.8.10.3 2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.212 when it is asked to set the maximum source routing attempts.

Table 4.212: Set Maximum Routing Attempts Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0xD4							
5	Command Status							

Command Status (8 bits)

Refer *Command Status (8 bits)*.

4.8.10.4 3. Callback data frame (Z-Wave Module → host)

None.

4.8.11 Set RF Power Level Rediscovery Command

This command is used to set the power level to RF Module that can be used for finding neighboring nodes. The Set RF Power Level Rediscovery Command Identifier is 0x1E.

4.8.11.1 Frame flow

The frame flow for this command is an *Acknowledged frame*.

4.8.11.2 1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.213

Table 4.213: Set RF Power Level Rediscovery Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x1E							
5	PowerLevel							

Powerlevel (8 bits)

This field is used to indicate the power level that MUST be used by the node when performing the neighbor discovery process.

This field MUST comply with the format indicated in Table 4.210.

4.8.11.3 2. Response data frame (Z-Wave Module → host)

None.

4.8.11.4 3. Callback data frame (Z-Wave Module → host)

None.

4.8.12 Start Watchdog Command

This command is used to start Watchdog functionality on Z-Wave module. The Start Watchdog Command Identifier is 0xD2.

4.8.12.1 Frame flow

The frame flow for this command is an *Unacknowledged frame*.

4.8.12.2 1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.214 for triggering the Watchdog functionality on a Z-Wave module.

Table 4.214: Start Watchdog Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0xD2							

4.8.12.3 2. Response data frame (Z-Wave Module → host)

None.

4.8.12.4 3. Callback data frame (Z-Wave Module → host)

None.

4.8.13 Stop Watchdog Command

This command is used to stop Watchdog functionality on Z-Wave module. The Start Watchdog Command Identifier is 0xD3.

4.8.13.1 Frame flow

The frame flow for this command is an *Unacknowledged frame*.

4.8.13.2 1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.215 for stopping the Watchdog functionality on a Z-Wave module.

Table 4.215: Stop Watchdog Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0xD3							

4.8.13.3 2. Response data frame (Z-Wave Module → host)

None.

4.8.13.4 3. Callback data frame (Z-Wave Module → host)

None.

4.8.14 Set Timeouts Command

This command is used to set timeouts with 10ms ticks. The Set Timeouts Command Identifier is 0x06.

4.8.14.1 Frame flow

The frame flow for this command is an *Acknowledged frame with response*.

4.8.14.2 1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.216.

Table 4.216: Set Timeouts Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x06							
5	Rx ACK Timeout							
6	Rx BYTE Timeout							

Rx ACK Timeout (8 bits)

This field is used to indicate the maximum time to wait for ACK after frame transmission, in 10ms ticks.

Rx BYTE Timeout (8 bits)

This field is used to indicate the maximum time to wait for next byte when receiving a new frame, in 10ms ticks.

4.8.14.3 2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.217.

Table 4.217: Set Timeouts Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x06							
5	Previous Rx ACK Timeout							
6	Previous Rx BYTE Timeout							

Previous Rx ACK Timeout (8 bits)

This field is used to indicate previous Rx ACK timeout setting, in 10ms ticks.

Previous Rx BYTE Timeout (8 bits)

This field is used to indicate previous Rx BYTE timeout setting, in 10ms ticks.

4.8.14.4 3. Callback data frame (Z-Wave Module → host)

None.

4.8.15 Initiate Shutdown Command

This command is used to instruct the Z-Wave API to go to sleep in order to remove the power. The Initiate Shutdown Command Identifier is 0xD9.

4.8.15.1 Frame flow

The frame flow for this command is an *Acknowledged frame with response*.

4.8.15.2 1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.218

Table 4.218: Initiate Shutdown Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0xD9							

4.8.15.3 2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.219

Table 4.219: Initiate Shutdown Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0xD9							
5	Command Status							

Command Status (8 bits)

Refer to *Command Status (8 bits)*.

4.8.15.4 3. Callback data frame (Z-Wave Module → host)

None.

4.9 Z-Wave API Transport Commands

This section describes *Z-Wave API Commands* that are used to perform transport operations.

4.9.1 Controller Node Send Data Command

This command is used to transmit contents of a data buffer to a single node or all nodes (broadcast). The Controller Node Send Data Command Identifier is 0x13.

This command MUST only be supported by controller Z-Wave library types (refer to Table 4.36).

4.9.1.1 Frame flow

The frame flow for this command is an *Acknowledged frame with response and callback*.

4.9.1.2 1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.220

Table 4.220: Controller Node Send Data Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x13							
5/(5..6)	Destination NodeID							
6/7	Data Length							
7/8	Data 1							
...	...							
7+N/8+N	Data N							
8+N/9+N	Tx Options							
9+N/10+N	Session identifier							

Destination NodeID (8/16 bits)

This field is used to indicate the destination NodeID to send the Z-Wave Frame to.

This field MUST be encoded according to the configured NodeID base Type. Refer to *Z-Wave API Setup Set NodeID Base Type Sub Command* and Table 4.64.

Data Length (8 bits)

This field is used to indicate the length in bytes of the *Data* field. This field MUST be set to a value greater than 0.

Data (N bytes)

This field is used to advertise the data payload that MUST be transmited on the Z-Wave radio to the destination NodeID.

The length of this field, in bytes, MUST be according to the *Data Length* field.

Tx Options (8 bits)

Refer to *Tx Options (8 bits)*.

Session Identifier (8 bits)

Refer to *Session identifier (8 bits)*.

4.9.1.3 2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.221

Table 4.221: Send Data Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x13							
5	Response status							

Response status (8 bits)

Refer to *Response status (8 bits)*.

4.9.1.4 3. Callback data frame (Z-Wave Module → host)

A Z-Wave module MUST issue a callback frame formatted according to Table 4.222

Table 4.222: Send Data Command - Callback data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x13							
5	Session identifier							
6	Tx Status							
7	Tx Status Report 1							
…	…							
7+N	Tx Status Report N							

Session Identifier (8 bits)

Refer to *Session identifier (8 bits)*.

Tx Status (8 bits)

Refer to *Tx Status (8 bits)*.

Tx Status Report (N bytes)

This field is used to report detailed information about the Z-Wave frame transmission. This field MUST be omitted if the Z-Wave API module is not configured to enable Tx Status Reports in the *Z-Wave API Setup Set Tx Status Report Sub Command*.

For field description, refer to *Tx Status Report (N bytes)*.

4.9.2 Controller Node Send Data Multicast Command

This command is used to transmit a data buffer to a list of Z-Wave nodes (i.e., Multicast frame). The Controller Node Send Data Multicast Command Identifier is 0x14.

This command MUST only be supported by Controller Z-Wave library types (refer to Table 4.36). Z-Wave API Module supporting an End Node library MUST use *End Node Send Data Multicast Command* instead.

4.9.2.1 Frame flow

The frame flow for this command is an *Acknowledged frame with response and callback*.

4.9.2.2 1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.223

Table 4.223: Send Data Multicast Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x14							
5	NodeID Count							
5+1	NodeID List 1							
…	…							
5+N	NodeID List N							
6+N	Data Length							
6+N+1	Data 1							
…	…							
6+N+M	Data M							
7+N+M	Tx Options							
8+N+M	Session identifier							

NodeID Count (8 bits)

This field is used to advertise the number of NodeIDs contained in the *NodeID List* field.

For example, if there are 2 NodeIDs encoded in 4 bytes in the *NodeID List* field, this field MUST be set to 2.

NodeID List (N bytes)

This field is used to advertise the list of destination NodeID's.

Each 8 bits/16 bits groups in this field MUST represent a NodeID.

All NodeIDs in this field MUST be encoded according to the configured NodeID base Type. Refer to *Z-Wave API Setup Set NodeID Base Type Sub Command* and Table 4.64.

Data Length (8 bits)

This field is used to indicate the length in bytes of the *Data* field. This field MUST be set to a value greater than 0.

Data (M bytes)

This field is used to advertise the data payload that MUST be transmited on the Z-Wave radio to the destination NodeIDs.

The length of this field, in bytes, MUST be according to the *Data Length* field.

Tx Options (8 bits)

Refer to *Tx Options (8 bits)*.

Session Identifier (8 bits)

Refer to *Session identifier (8 bits)*.

4.9.2.3 2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.224

Table 4.224: Send Data Multicast Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x14							
5	Response status							

Response status (8 bits)

Refer to *Response status (8 bits)*.

4.9.2.4 3. Callback data frame (Z-Wave Module → host)

A Z-Wave module MUST issue a callback frame formatted according to Table 4.225

Table 4.225: Send Data Multicast Command - Callback data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x14							
5	Session identifier							
6	Tx Status							

Session Identifier (8 bits)

Refer to *Session identifier (8 bits)*.

Tx Status (8 bits)

Refer to *Tx Status (8 bits)*.

4.9.3 End Node Send Data Command

This command is used to transmit contents of a data buffer to a single node or all nodes (broadcast). The End Node Send Data Command Identifier is 0x0E. This command MUST only be supported by End node library types (refer to Table 4.36).

4.9.3.1 Frame flow

The frame flow for this command is an *Acknowledged frame with response and callback*.

4.9.3.2 1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.226

Table 4.226: End Node Send Data Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x0E							
5/(5..6)	Destination NodeID							
6/7	Data Length							
7/8	Data 1							
…	…							
7/8+N	Data N							
7/8+N	Tx Options							
7/8+N	Tx Security Options							
7/8+N	Security Keys							
7/8+N	Tx Options 2							
7/8+N	Session identifier							

Destination NodeID (8/16 bits)

This field is used to indicate the Destination NodeID.

This field MUST be encoded according to the configured NodeID base Type. Refer to *Z-Wave API Setup Set NodeID Base Type Sub Command* and Table 4.64.

Data Length (8 bits)

This field is used to indicate the length in bytes of the *Data* field. This field MUST be set to a value greater than 0.

Data (N bytes)

This field is used to advertise the data payload that MUST be transmited on the Z-Wave radio to the destination NodeID.

This field MUST represent the unencrypted data payload.

The length of this field, in bytes, MUST be according to the *Data Length* field.

Tx Options (8 bits)

Refer to *Tx Options (8 bits)*.

Tx Security Options (8 bits)

This field is used to indicate the security 2 specific options. This field MUST be encoded according to Table 4.227

Table 4.227: End Node Send Data - Tx Security Options encoding

Value	Flag	Description
0x01	S2_TXOPTION_VERIFY_DELIVERY.	This flag will activate frame delivery verification.
0x02	S2_TXOPTION_SINGLECAST_FOLLOWUP.	This flag must be present on all single cast followup messages to a multicast transmission.
0x04	S2_TXOPTION_FIRST_SINGLECAST_FOLLOWUP	. This flag must be present on the first, and only the first singlecast followup message in a multicast transmission.

Other Tx Security Options values are reserved. Reserved values MUST NOT be used and MUST be ignored by a receiving interface.

The Singlecast follow-up frames will reuse the Multicast GroupID received in the last *End Node Send Data Multicast Command*.

Security Keys (8 bits)

This field is used to advertise the security key for the transmission. This field MUST be encoded according to Table 4.228

Table 4.228: End Node Send Data - Tx Security key encoding

Value	Flag	Description
0x00	SECURITY_KEY_NONE.	Nonsecure transmission.
0x01	SECURITY_KEY_S2_UNAUTHENTICATED.	Use S2 Unauthenticated key.
0x02	SECURITY_KEY_S2_AUTHENTICATED.	Use S2 Authenticated key.
0x03	SECURITY_KEY_S2_ACCESS.	Use S2 Access key.
0x04	SECURITY_KEY_S0.	Use Security Scheme 0 key.

Other Tx Security key values are reserved. Reserved values MUST NOT be used and MUST be ignored by a receiving interface.

Tx Options 2 (8 bits)

This field is used to indicate more transmission options flags. It is reserved for future use. It MUST be set to 0x00.

Session Identifier (8 bits)

Refer to *Session identifier (8 bits)*.

4.9.3.3 2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.229

Table 4.229: End Node Send Data Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x0E							
5	Response status							

Response status (8 bits)

Refer to *Response status (8 bits)*.

4.9.3.4 3. Callback data frame (Z-Wave Module → host)

A Z-Wave module MUST issue a callback frame formatted according to Table 4.230

Table 4.230: End Node Send Data Command - Callback data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x0E							
5	Session identifier							
6	Tx Status							
7	Tx Status Report 1							
…	…							
7+N	Tx Status Report N							

Session Identifier (8 bits)

Refer to *Session identifier (8 bits)*.

Tx Status (8 bits)

Refer to *Tx Status (8 bits)*.

Tx Status Report (N bytes)

This field is used to report detailed information about the Z-Wave frame transmission. This field MUST be omitted if the Z-Wave API module is not configured to enable Tx Status Reports in the *Z-Wave API Setup Set Tx Status Report Sub Command*.

For field description, refer to *Tx Status Report (N bytes)*.

4.9.4 End Node Send Data Multicast Command

This command is used to transmit a data buffer to a list of Z-Wave nodes (i.e., S2 Multicast frame).

The End Node Send Data Multicast Command Identifier is 0x0F. This command shall only be supported by End Node library types (refer to Table 4.36).

4.9.4.1 Frame flow

The frame flow for this command is an *Acknowledged frame with response and callback*.

4.9.4.2 1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.231

Table 4.231: End Node Send Data Multicast Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x0F							
5	Data Length							
6	Data 1							
…	…							
6+N	Data N							
7+N	Tx Options							
8+N	Security Keys							
9+N	Multicast group ID							
10+N	Session identifier							

Data Length (8 bits)

This field is used to indicate the length in bytes of the *Data* field. This field MUST be set to a value greater than 0.

Data (N bytes)

This field is used to advertise the data payload that MUST be transmited on the Z-Wave radio to the destination NodeIDs.

The length of this field, in bytes, MUST be according to the *Data Length* field.

Multicast group ID (8 bits)

This field is used to indicate the destination GroupID assigned to the current Security 2 Multicast frame.

This GroupID will be used in the S2 MGRP extension. Refer to [zwave_encapsulation_cc_spec] for the S2 MGRP extension.

Session Identifier (8 bits)

Refer to *Session identifier (8 bits)*.

4.9.4.3 2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.232

Table 4.232: End Node Send Data Multicast Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x0F							
5	Response status							

Response status (8 bits)

Refer to *Response status (8 bits)*.

4.9.4.4 3. Callback data frame (Z-Wave Module → host)

A Z-Wave module MUST issue a callback frame formatted according to Table 4.233

Table 4.233: End Node Send Data Multicast Command - Callback data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x0F							
5	Session Identifier							
6	Tx Status							

Session Identifier (8 bits)

Refer to *Session identifier (8 bits)*.

Tx Status (8 bits)

Refer to *Tx Status (8 bits)*.

4.9.5 Bridge Controller Node Send Data Command

This command is used to transmit contents of a data buffer to a single node or all nodes (broadcast). The Bridge Controller Node Send Data Command Identifier is 0xA9.

This command MUST only be supported by nodes implementing a Bridge Controller library type (refer to Table 4.36).

4.9.5.1 Frame flow

The frame flow for this command is an *Acknowledged frame with response and callback*.

4.9.5.2 1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.234

Table 4.234: Bridge Controller Node Send Data Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0xA9							
5/(5..6)	Source NodeID							
6/(7..8)	Destination NodeID							
7/9	Data Length							
8/9	Data 1							
…	…							
8+N/9+N	Data N							
9+N/10+N	Tx Options							
10+N/11+N	Route							
11+N/12+N	Session identifier							

Source NodeID (8 bits / 16 bits)

This field is used to indicate the Source NodeID from which the Z-Wave Frame must be issued.

This field MUST be encoded according to the configured NodeID base Type. Refer to *Z-Wave API Setup Set NodeID Base Type Sub Command* and Table 4.64.

Setting Source NodeID to 0xFF/0xFFF will cause the protocol to automatically use the controllers native NodeID

Destination NodeID (8 bits / 16 bits)

This field is used to indicate the destination NodeID.

This field MUST be encoded according to the configured NodeID base Type. Refer to *Z-Wave API Setup Set NodeID Base Type Sub Command* and Table 4.64.

Data Length (8 bits)

This field is used to indicate the length in bytes of the *Data* field. This field MUST be set to a value greater than 0.

Data (N bytes)

This field is used to advertise the data payload that MUST be transmited on the Z-Wave radio to the destination NodeID.

The length of this field, in bytes, MUST be according to the *Data Length* field.

Tx Options (8 bits)

Refer to *Tx Options (8 bits)*.

Route (4 bytes)

This feild is used to indicate the priority route to be used for tranmiting a frame. If there are not any route, the feild MUST set to zero.

Session identifier (8 bits)

Refer to *Session identifier (8 bits)*.

4.9.5.3 2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.235

Table 4.235: Bridge Controller Node Send Data Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0xA9							
5	Response status							

Response status (8 bits)

Refer to *Response status (8 bits)*.

4.9.5.4 3. Callback data frame (Z-Wave Module → host)

A Z-Wave module MUST issue a callback frame formatted according to Table 4.236

Table 4.236: Bridge Controller Node Send Data Command - Callback data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0xA9							
5	Session identifier							
6	Tx Status							
7	Tx Status Report 1							
...	...							
7+N	Tx Status Report N							

Session Identifier (8 bits)

Refer to *Session identifier (8 bits)*.

Tx Status (8 bits)

Refer to *Tx Status (8 bits)*.

Tx Status Report (N bytes)

This field is used to report detailed information about the Z-Wave frame transmission. This field MUST be omitted if the Z-Wave API module is not configured to enable Tx Status Reports in the *Z-Wave API Setup Set Tx Status Report Sub Command*.

For field description, refer to *Tx Status Report (N bytes)*.

4.9.6 Bridge Controller Node Send Data Multicast Command

This command is used to transmit a data buffer to a list of Z-Wave nodes (i.e., Multicast frame). The Bridge Controller Node Send Data Multicast Command Identifier is 0xAB.

This command MUST only be supported by Z-Wave API Module using the Controller Bridge library types (refer to Table 4.36).

4.9.6.1 Frame flow

The frame flow for this command is an *Acknowledged frame with response and callback*.

4.9.6.2 1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.237

Table 4.237: Bridge Controller Node Send Data Multicast Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0xAB							
5/5..6	Source NodeID							
6/7	NodeID Count							
(6/7)+1	NodeID List 1							
…	…							
(6/7)+N	NodeID List N							
(7/8)+N	Data Length							
(8/9)+N+1	Data 1							
…	…							
(8/9)+N+M	Data M							
(9/10)+N+M	Tx Options							
(10/11)+N+M	Session identifier							

Source NodeID (8/16 bits)

This field is used to indicate the Source NodeID from which the Z-Wave Frame MUST be issued.

This field MUST be encoded according to the configured NodeID base Type. Refer to *Z-Wave API Setup Set NodeID Base Type Sub Command* and Table 4.64.

Setting Source NodeID to 0xFF/0xFFF will cause the protocol to automatically use the controllers native NodeID

NodeID Count (8 bits)

This field is used to advertise the number of NodeIDs contained in the *NodeID List* field.

For example, if there are 2 NodeIDs encoded in 4 bytes in the *NodeID List* field, this field MUST be set to 2.

NodeID List (N bytes)

This field is used to advertise a list of NodeID destinations.

Each 8 bits/16 bits groups in this field MUST represent a NodeID.

All NodeIDs in this field MUST be encoded according to the configured NodeID base Type. Refer to *Z-Wave API Setup Set NodeID Base Type Sub Command* and Table 4.64.

Data Length (8 bits)

This field is used to indicate the length in bytes of the *Data* field. This field MUST be set to a value greater than 0.

Data (M bytes)

This field is used to advertise the data payload that MUST be transmited on the Z-Wave radio to the destination NodeIDs.

The length of this field, in bytes, MUST be according to the *Data Length* field.

Tx Options (8 bits)

Refer to *Tx Options (8 bits)*.

Session Identifier (8 bits)

Refer to *Session identifier (8 bits)*.

4.9.6.3 2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.238

Table 4.238: Bridge Controller Node Send Data Multicast Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0xAB							
5	Response status							

Response status (8 bits)

Refer to *Response status (8 bits)*.

4.9.6.4 3. Callback data frame (Z-Wave Module → host)

A Z-Wave module MUST issue a callback frame formatted according to Table 4.239

Table 4.239: Bridge Controller Node Send Data Multicast Command - Callback data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0xAB							
5	Session identifier							

6	Tx Status

Session Identifier (8 bits)

Refer to *Session identifier (8 bits)*.

Tx Status (8 bits)

Refer to *Tx Status (8 bits)*.

4.9.7 Send Data Abort Command

This command is used to instruct the Z-Wave Module to abort an ongoing transmission started with any of the following commands:

- *Bridge Controller Node Send Data Command*
- *Bridge Controller Node Send Data Multicast Command*
- *End Node Send Data Command*
- *End Node Send Data Multicast Command*
- *Controller Node Send Data Command*
- *Controller Node Send Data Multicast Command*
- *Send NOP Command*

The Send Data Abort Command Identifier is 0x16.

4.9.7.1 Frame flow

The frame flow for this command is an *Acknowledged frame*.

If a host application aborts an ongoing transmission, the Z-Wave API Module MUST still issue a 3. callback data frame for the ongoing transmission. This is illustrated in Figure 4.21. A Z-Wave API Module MUST ignore this command if it is received when no transmission is ongoing.

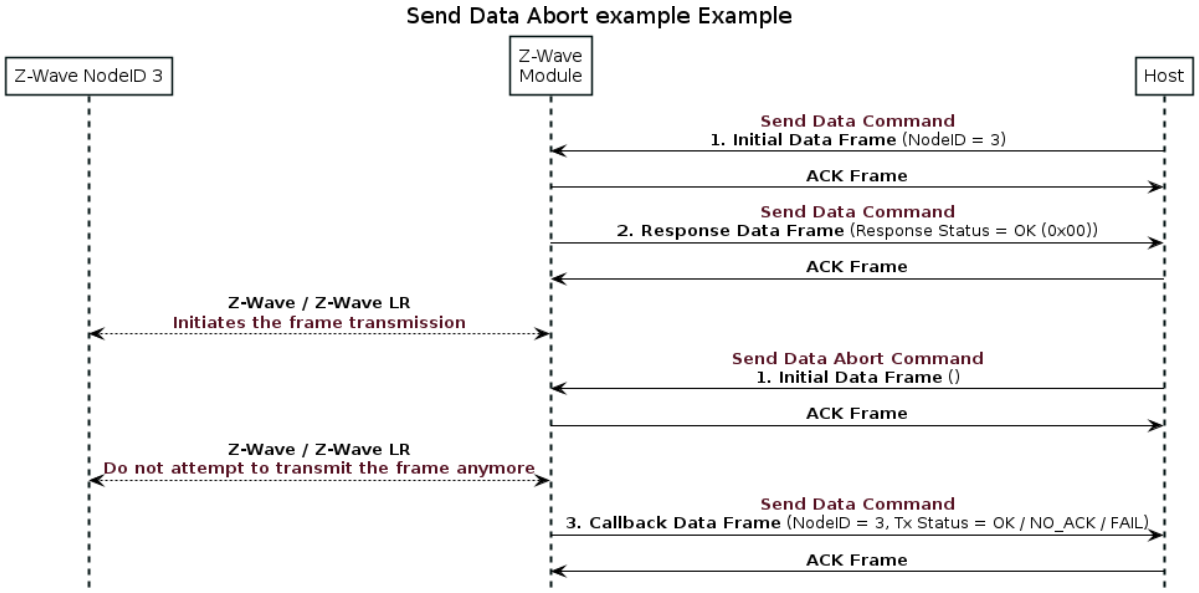

Figure 4.21: Send Data Abort Command Example

4.9.7.2 1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.240

Table 4.240: Send Data Abort Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x16							

4.9.7.3 2. Response data frame (Z-Wave Module → host)

None

4.9.7.4 3. Callback data frame (Z-Wave Module → host)

None.

4.9.8 Send Test Frame Command

This command is used to send a test frame directly to a given node without any routing. The Send Test Frame Command Identifier is 0xBE.

Note that this command shall only be used during installation and testing the wireless communication link path. And the test will be done using 9600 kbit/s transmission rate.

4.9.8.1 Frame flow

The frame flow for this command is an *Acknowledged frame with response and callback*.

4.9.8.2 1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.241.

Table 4.241: Send Test Frame Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0xBE							
5/5..6	NodeID							
6/7	Powerlevel							
7/8	Session identifier							

NodeID (8/16 bits)

This field is used to advertise the NodeID of the node where the test frame is sent to.

This field MUST be encoded according to the configured NodeID base Type. Refer to *Z-Wave API Setup Set NodeID Base Type Sub Command* and Table 4.64.

Powerlevel (8 bits)

This field is used to advertise the power level which the Z-Wave module shall use for the RF transmission of the test frame. This field MUST comply with the format indicated in *Set RF Power Level Command* initial data frame section.

Session identifer (8 bits)

Refer to *Session identifier (8 bits)*.

4.9.8.3 2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.242 when the Send Test Frame Command data is received by the Z-Wave Module.

Table 4.242: Send Test Frame Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0

4	Z-Wave API Command ID = 0xBE
5	Response status

Response status (8 bits)

Refer to *Response status (8 bits)*.

4.9.8.4 3. Callback data frame (Z-Wave Module → host)

A Z-Wave module MUST issue a callback frame formatted according to Table 4.243 when the transmission of the test frame is executed.

Table 4.243: Send Test Frame Command - Callback data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0xBE							
5	Session identifier							
6	Tx Status							

Session identifer (8 bits)

Refer to *Session identifier (8 bits)*.

Tx Status (8 bits)

Refer to *Tx Status (8 bits)*.

4.10 Z-Wave API Security Commands

This section describes *Z-Wave API Commands* that are used to perform security bootstrapping operations.

4.10.1 Security Setup Command

This command is used to set the Requested Security Keys and Requested Authentication method prior to inclusion (add). The Requested Security Keys and Authentication is requested by the protocol during S2 inclusion. The Security Setup Command Identifier is 0x9C.

This command MUST only be supported by Z-Wave API Module implementing an *End Node* library type (e.g. *End Node library*, *Enhanced 232 End Node Library** or *Routing End Node library*). Refer to Table 4.36).

4.10.1.1 Frame flow

The frame flow for this command is an *Acknowledged frame with response*.

4.10.1.2 1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.244.

Table 4.244: Security Setup Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x9C							
5	Security Mode							
6	Parameter Length							
7	Parameter 1							
..	...							
7+N	Parameter N							

Security Mode (8 bits)

This field is used to indicate the mode that represents the requested security functionalities. This field MUST be encoded according to Table 4.245

Table 4.245: The security mode value encoding

Value	Identifier and description
0x00	*GET_SECURITY_KEYS* This value used to request the security keys. If this value is used, the *Security Setup Command - Initial data frame* MUST NOT contain *Parameter Length* and *Parameter* fields.
0x01	*Obsoleted* This value is obsoleted and MUST NOT be used.

0x02	*GET_SECURITY_2_PUBLIC_DSK* This value is used to request the public DSK. If this value is used the *Security Setup Command - Initial data frame* MUST NOT contain *Parameter Length** and *Parameter* fields.
0x03..0x04	*Obsoleted* These values are obsoleted and MUST NOT be used.
0x05	*SET_SECURITY_INCLUSION_REQUESTED_KEYS* This value is used to set Requested Security Inclusion Keys.
0x06	*Obsoleted* This value is obsoleted and MUST NOT be used.
0x07..0xFD	*Reserved* These values are reserved and MUST NOT be used.
0xFE	*GET_SECURITY_CAPABILITIES* This value is used to request the supported *Security Modes* values by the Z-Wave API Module.

Parameter Length (8 bits)

This field is used to indicate the length in bytes of the *Parameter* field. This field MUST be set to a value greater than 0.

Parameter (N bytes)

This field is used to advertise additional parameters required for the value specified in the *Security Mode* field. The length of this field, in bytes, MUST be according to the *Parameter Length* field. This field MUST be encoded according to Table 4.246

Table 4.246: Security Setup Command - Initial Frame Parameter Field Encoding

Security Mode field	Parameter field
0x00 *Get Security Keys*	Omitted.
0x01 *Obsoleted*	This value is obsoleted and MUST NOT be used.
0x02 *Get Security 2 Authenticated Learn Mode DSK*	Omitted.
0x03..0x04 *Obsoleted*	These values are obsoleted and MUST NOT be used.
0x05 *Set requested network keys*	The *Parameter* field MUST contain Requested Security Classes during Security Bootstrapping.
0x06 *Obsoleted*	This value is obsoleted and MUST NOT be used.
0x07..0xFD *Reserved*	These values are reserved and MUST NOT be used.
0xFE *Get Security Capabilities*	Omitted.

0xFF *Unknown Security Modes*	N/A This value MUST only be used by a Z-Wave API Module in a 2. Response data frame in order to indicate that the received Security Mode in the 1. Initial data frame is unknown or not supported

4.10.1.3 2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.247.

Table 4.247: Security Setup Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x9C							
5	Security Mode							
6	Parameter Length							
7	Parameter 1							
..	. . .							
7+N	Parameter N							

Refer *Security Setup Command - Initial data frame* field descriptions for the fields that are not described below.

Parameter (N bytes)

This field is used to advertise parameters corresponds to the security mode flag used in *Security Mode* field. The length of this field, in bytes, MUST be according to the *Parameter Length* field. This field MUST be encoded according to Table 4.248

Table 4.248: Security Setup Command Response Frame Parameter Field Encoding

Security Mode field	Parameter field
0x00 *Get Security Keys*	The *Parameter* field MUST contain the bitmask that represents the security keys the Z-Wave module poses. The bitmask field MUST be encoded according to Table 4.249
0x01 *Obsoleted*.	This value is obsoleted and MUST NOT be used.
0x02 *Get Security 2 Authenticated Learn Mode DSK*	The *Parameter* field MUST describe if the Security 2 DSK. (Derived from the Learn Mode Authenticated ECDH key pair) Refer to [zwave_encapsulation_cc_spec] for details about the DSK.
0x03 *Obsoleted*	This value is obsoleted and MUST NOT be used.
0x04 *Obsoleted*	This value is obsoleted and MUST NOT be used.
0x05 *Set requested network keys*	The *Parameter* field MUST describe if the Requested Security Inclusion Keys is accepted or not via the Z-Wave module. The field MUST be encoded according to *Command Status (8 bits)*.
0x06 *Obsoleted*	This value is obsoleted and MUST NOT be used.

0x07..0xFD *Reserved*	These values are reserved and MUST NOT be used.
0xFE *Get Security Capabilities*	The *Parameter* field MUST be encoded as a bitmask representing the supported *Security Modes*. • Bit 0 in Byte 1 MUST represent mode 0x00 • Bit 1 in Byte 1 MUST represent mode 0x01 • Bit 2 in Byte 1 MUST represent mode 0x02 • etc. A bit set to 1 MUST indicate that the corresponding *Security Mode* is supported. A bit set to 0 MUST indicate that the corresponding *Security Mode* is not supported. The field MUST be encoded according to *Command Status (8 bits)*.
0xFF *Unknown Security Modes*	The *Parameter* field MUST be set to the unknown / unsupported *Security Mode* value that was received in the 1. Initial data frame.

Table 4.249: The security keys bitmask value encoding

Bit mask	Flag and description
0x00	*SECURITY_KEY_NONE_MASK* No network key.
0x01	*SECURITY_KEY_S2_UNAUTHENTICATED_BIT* S2 Unauthenticated network key.
0x02	*SECURITY_KEY_S2_AUTHENTICATED_BIT* S2 Authenticated network key.
0x04	*SECURITY_KEY_S2_ACCESS_BIT* S2 Access Control network key.
0x80	*SECURITY_KEY_S0_BIT* Security 0 network key.

4.9.1.1 3. Callback data frame (Z-Wave Module → host)

None.

4.9.2 Encrypt Data With AES Command

This command is used to request the Z-Wave API module to encrypt a Z-Wave frame payload using AES-128 Electronic CookBook mode. The Encrypt Data With AES Command Identifier is 0x67.

4.9.2.1 Frame flow

The frame flow for this command is an *Acknowledged frame with response*.

4.9.2.2 1. Initial data frame (host → Z-Wave Module)

The initial data frame MUST be formatted according to Table 4.250.

Table 4.250: Encrypt Data With AES Command - Initial data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x67							
5	Keys 1							
..	…							
20	keys 16							
21	Input Data 1							
..	…							
36	Input Data 16							

Keys (16 bytes)

This field is used to advertise the encryption key.

Input Data (16 bytes)

This field is used to indicate the data to be encrypted.

4.9.2.3 2. Response data frame (Z-Wave Module → host)

A Z-Wave module MUST return a response frame formatted according to Table 4.251.

Table 4.251: Encrypt Data With AES Command - Response data frame

byte\bit	7	6	5	4	3	2	1	0
4	Z-Wave API Command ID = 0x67							
5	Output Data 1							
..	…							
20	Output Data 16							

Output Data (16 bytes)

This field is used to advertise the encrypted data.

4.9.2.4 3. Callback data frame (Z-Wave Module → host)

None.

5 References

[device_type_spec_v2]	Z-Wave Alliance, Z-Wave Plus v2 Device Type Specification
[device_type_spec]	Z-Wave Alliance, Z-Wave Plus Device Type Specification
[device_class_spec]	Z-Wave Alliance, Z-Wave Device Class Specification
[zwave_nwk_spec]	Z-Wave Alliance, ZWA_Z-Wave and Z-Wave Long Range Network Layer Specifi- cation_SPE
[zwave_manufacturer_ids]	Z-Wave Alliance, List of defined Manufacturer IDs
[zwave_management_cc_spec]	Z-Wave Alliance, Z-Wave Management Command Class Specification
[zwave_encapsulation_cc_spec]	Z-Wave Alliance, Z-Wave Transport-Encapsulation Command Class Specification

