BeeStack Consumer Blackbox Interface

User's Guide

Document Number: BSCONBBIUG
Rev. 1.8
2/2012

freescale™
semiconductor

How to Reach Us:

Home Page:
www.freescale.com

E-mail:
support@freescale.com

USA/Europe or Locations Not Listed:
Freescale Semiconductor
Technical Information Center, CH370
1300 N. Alma School Road
Chandler, Arizona 85224
+1-800-521-6274 or +1-480-768-2130
support@freescale.com

Europe, Middle East, and Africa:
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
support@freescale.com

Japan:
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku,
Tokyo 153-0064, Japan
0120 191014 or +81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:
Freescale Semiconductor Hong Kong Ltd.
Technical Information Center
2 Dai King Street
Tai Po Industrial Estate
Tai Po, N.T., Hong Kong
+800 2666 8080
support.asia@freescale.com

***For Literature Requests Only*:**
Freescale Semiconductor Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
1-800-521-6274 or 303-675-2140
Fax: 303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

Information in this document is provided solely to enable system and software implementers to use Freescale Semiconductor products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits or integrated circuits based on the information in this document.

Freescale Semiconductor reserves the right to make changes without further notice to any products herein. Freescale Semiconductor makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale Semiconductor assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. “Typical” parameters that may be provided in Freescale Semiconductor data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including “Typicals”, must be validated for each customer application by customer’s technical experts. Freescale Semiconductor does not convey any license under its patent rights nor the rights of others. Freescale Semiconductor products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Freescale Semiconductor product could create a situation where personal injury or death may occur. Should Buyer purchase or use Freescale Semiconductor products for any such unintended or unauthorized application, Buyer shall indemnify and hold Freescale Semiconductor and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Freescale Semiconductor was negligent regarding the design or manufacture of the part.

Freescale™ and the Freescale logo are trademarks of Freescale Semiconductor, Inc. All other product or service names are the property of their respective owners.

© Freescale Semiconductor, Inc. 2008, 2009, 2010, 2011, 2012. All rights reserved.

Contents

About This Book

This user's guide provides a detailed description of the BeeStack Consumer Blackbox Interface, communication packet structure, available services and usage.

Audience

This reference manual is intended for application designers and users of the BeeStack Consumer Blackbox interface.

Organization

This document contains the following chapters:

Chapter 1 BeeStack Consumer BlackBox Overview - Briefly describes BeeStack Consumer BlackBox functionality and usage.

Chapter 2 Interface Description - Describes the BlackBox interfaces which can be either a three wire UART connection or a two wire I^2C connection to interface with the system.

Chapter 3 BeeStack Consumer BlackBox Messages - Details the messages that the BlackBox exchanges with the host.

Revision History

The following table summarizes revisions to this manual since the previous release (Rev. 1.7).

Revision History

Date / Author	Description / Location of Changes
Feb 2012, Dev Team	Changes in Chapter 3.

Definitions, Acronyms, and Abbreviations

The following list defines the abbreviations used in this document.

API	Application Programming Interface
CE	Consumer Electronics
I2C	Inter - Integrated Circuit
LQI	Link Quality Indicator
NW Layer	Network Layer
PAN	Personal Area Network
NV	Non volatile
NVM	Non volatile Memory
ZRC	Zigbee Remote Control

References

The following sources were referenced to produce this book:

1. RF4CE Specification version 1.0.0, Document 080002r04
2. IEEE 802.15.4 Standard -2003, Part 14.5: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs), The Institute of Electrical and Electronics Engineers, Inc. October 2003
3. *BeeStack Consumer Application Reference Manual* (BSCONRM)
4. *BeeStack Consumer Application User's Guide* (BSCONAUG)
5. *Freescale BeeKit Wireless Connectivity Toolkit User's Guide* (BKWCTKUG)

Chapter 1
BeeStack Consumer BlackBox Overview

The BeeStack Consumer BlackBox is an embedded application built on the BeeStack Consumer Network platform. The BeeStack Consumer BlackBox offers access to all the BeeStack Consumer Control Network features over a UART or a I^2C interface. This allows BeeStack Consumer Network connectivity to be added to any system with limited modifications, because only a serial port is needed.

The BeeStack Consumer Network is a software networking layer that sits on top of the IEEE 802.15.4 MAC and PHY layers. It is designed for Wireless Personal Area Networks (WPANs) and conveys information over short distances among the participants in the network. It enables small, power efficient, inexpensive solutions to be implemented for a wide range of applications. Some key characteristics of a BeeStack Consumer network are:

- An over the air data rate of 250 kbit/s in the 2.4 GHz band
- Three independent communication channels in the 2.4 GHz band
- Two network node types, controller node and respectively target node
- Channel agility mechanism
- Provides robustness and ease of use
- Includes essential functionality to build and support a CE network

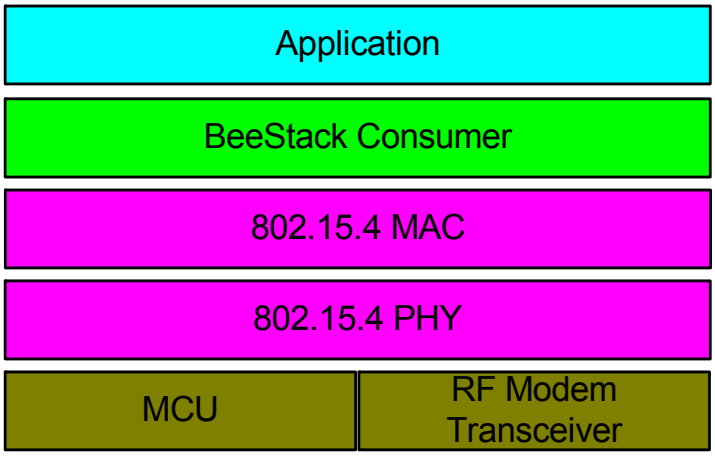

Figure 1-1. Network Layer Overview

Chapter 2
Interface Description

The Blackbox uses one of the following interfaces for communications:

- A 3-wire UART connection
- A 2-wire I^2C connection

2.1 UART Overview and Packet Structure

The BlackBox UART packet structure sends and receives messages as shown in Figure 2-1. This structure is specific to the UART interface and is designed to offer the best communication reliability. The BlackBox device is expecting messages in little-endian format and responds with messages in little-endian format.

Figure 2-1. UART Packet Structure

2.1.1 UART Packet Field Description

Stx — (1 bytes) Used for synchronization over the serial interface. The value is always 0x02.

Header — (2 bytes) Used to distinguish between different network sublayers (e.g. NLME or NLDE) and to distinguish between different services on each sublayer in part.

First byte is named **OpcodeGroup** and the second is named **Opcode**.

Example 0xD0 0x00

0xD0 represent the NLME sublayer

0x00 represent the RESET service provided by NLME

Length — (2 bytes) The length of the packet payload, excluding the header and FCS. The length field content shall be provided in little endian format.

Payload — (Variable length) (Optional) Payload of the BlackBox structure.

FCS — (1 byte) Field used to check the data integrity of the packet.

The FCS is computed by xor-ing all the fields except Stx filed and FCS field. This value is then compared to the received FCS field. If they are not equal, then the packet is considered corrupted and is dropped. If the FCS is good, then the message is processed.

2.2 I²C Overview and Packet Structure

The BlackBox I²C packet structure sends and receives messages as shown in Figure 2-2. This structure is designed to offer the best communication reliability. The BlackBox device is expecting messages in little-endian format and responds with messages in little-endian format.

Figure 2-2. I²C Packet Structure

2.2.1 I²C Packet Field Description

Stx — (1 bytes) Used for synchronization over the serial interface. The value is always 0x02.

Header — (2 bytes) Used to distinguish between different network sublayers (e.g. NLME or NLDE) and to distinguish between different services on each sublayer in part.

First byte is named **OpcodeGroup** and the second is named **Opcode**.

Example 0xD0 0x00

0xD0 represent the NLME sublayer

0x00 represent the RESET service provided by NLME

Length — (2 bytes) The length of the packet payload, excluding the header and FCS. The length field content shall be provided in little endian format.

Payload — (Variable length) (Optional) Payload of the BlackBox structure.

FCS — (1 byte) Field used to check the data integrity of the packet.

The FCS is computed by xor-ing all the fields except Stx filed and FCS field. This value is then compared to the received FCS field. If they are not equal, then the packet is considered corrupted and is dropped. If the FCS is good, then the message is processed.

Chapter 3
BeeStack Consumer Blackbox Messages

Table 3-1 lists the types of messages the BlackBox exchanges with the host. The messages allow access to all BeeStack Consumer Control Network services exposed by the BlackBox and utility services used to access a range of device or application options.

Table 3-1. BlackBox Messages

BeeStack Consumer Requests		
OpcodeGroup	**Opcode**	**Service**
D0	00	RF4CE_NLME_Reset.Request
D0	01	RF4CE_NLME_Start.Request
D0	02	RF4CE_NLME_Discovery.Request
D0	03	RF4CE_NLME_Discovery.Response
D0	04	RF4CE_NLME_Pair.Request
D0	05	RF4CE_NLME_Pair.Response
D0	06	RF4CE_NLME_UnPair.Request
D0	07	RF4CE_NLME_Get.Request
D0	08	RF4CE_NLME_Set.Request
D0	09	RF4CE_NLME_RxEnable.Request
D0	0A	RF4CE_NLME_UnPair.Response
D0	0B	RF4CE_NLME_AutoDiscovery.Request
D0	0C	RF4CE_NLME_UpdateKey.Request
D2	00	RF4CE_NLDE_Data.Request
D4	00	RF4CE_NWK_SetMacAddress.Request
D4	01	RF4CE_NWK_GetMacAddress.Request
D4	02	RF4CE_NWK_SetMaxPairingTableEntries.Request
D4	03	RF4CE_NWK_GetMaxPairingTableEntries.Request
D4	04	RF4CE_NWK_SetNodeCapabilities.Request
D4	05	RF4CE_NWK_GetNodeCapabilies.Request
D4	06	RF4CE_NWK_SetVendorIdentifier.Request
D4	07	RF4CE_NWK_GetVendorIdentifier.Request
D4	08	RF4CE_NWK_SetVendorString.Request

Table 3-1. BlackBox Messages (continued)

D4	09	RF4CE_NWK_GetVendorString.Request
D4	0A	RF4CE_NWK_SetFrameCounterWindow.Request
D4	0B	RF4CE_NWK_GetFrameCounterWindow.Request
D4	0C	RF4CE_NWK_AddNewPairTableEntry.Request
D4	0D	RF4CE_NWK_SavePersistentData.Request
D4	0E	RF4CE_NWK_GenerateShortAddress.Request
D4	0F	RF4CE_NWK_GenerateSecurityKey.Request
D4	10	RF4CE_NWK_SaveFrameCounter.Request
D4	11	RF4CE_NWK_GetLastPacketLQI.Request
D4	12	RF4CE_NWK_GetNodePanId.Request
D4	13	RF4CE_NWK_GetNodeShortAddress.Request
D4	14	RF4CE_NWK_IsIdle.Request
D4	15	RF4CE_NWK_GetAllowedLowPowerInterval.Request
D6	00	PBP_PushButtonPairOrig.Request
D6	01	PBP_PushButtonPairRecip.Request
E0	00	ZRCProfile_AbortProcess.Request
D6	03	PBP_PushButtonPairOrigContinue.Response
D6	04	PBP_PushButtonPairRecipContinue.Response
D6	05	PBP_AbortProcess.Reques
DD	00	ZRCProfile_Command.Request
E0	01	ZRCProfile_GetAttr.Request
E0	02	ZRCProfile_SetAttr.Request
E0	03	ZRCProfile_SetZRCSupportedCmds.Request
E0	04	ZRCProfile_GetZRCSupportedCmds.Request
DA	00	FSLProfile_FragTx.Request
DA	01	FSLProfile_SetFragTxRxBufferState.Request
DA	02	FSLProfile_GetFragTxRxBufferState.Request
DA	03	FSLProfile_PollConfig.Request
DA	04	FSLProfile_Poll.Request
DA	05	FSLProfile_PollDataAvailable.Request
DA	06	FSLProfile_RmtPair.Request
DA	07	FSLProfile_RmtPairResponse
DA	08	FSLProfile_BrowseMenuReq.Request
DA	09	FSLProfile_DisplayMenuHeaderReq.Request

Table 3-1. BlackBox Messages (continued)

DA	0A	FSLProfile_DisplayMenuEntry.Request
DA	0B	FSLProfile_DisplayMenuMessage.Request
DA	0C	FSLProfile_DisplayCompleteIndToBrowser.Request
DA	0D	FSLProfile_DisplayMenuExit.Request
DA	0E	FSLProfile_GetSupportedFeatures.Request
BeeStack Consumer Confirms and Indications		
OpcodeGroup	**Opcode**	**Service**
D1	00	RF4CE_NLME_Start.Confirm
D1	01	RF4CE_NLME_AutoDiscovery.Confirm
D1	02	RF4CE_NLME_Discovery.Confirm
D1	03	RF4CE_NLME_Discovery.Indication
D1	04	RF4CE_NLME_Pair.Confirm
D1	05	RF4CE_NLME_Pair.Indication
D1	06	RF4CE_NLME_UnPair.Confirm
D1	07	RF4CE_NLME_UnPair.Indication
D1	08	RF4CE_NLME_CommStatus.Indication
D1	A0	RF4CE_NLME_Get.Confirm
D1	A1	RF4CE_NLME_Set.Confirm
D1	A2	RF4CE_NLME_Reset.Confirm
D1	A3	RF4CE_NLME_RxEnable.Confirm
D1	A4	RF4CE_NLME_UnPairResponse.Confirm
D1	A5	RF4CE_NLME_UpdateKey.Confirm
D3	00	RF4CE_NLDE_Data.Confirm
D3	01	RF4CE_NLDE_Data.Indication
D5	00	RF4CE_NWK_SetMacAddress.Confirm
D5	01	RF4CE_NWK_GetMacAddress.Confirm
D5	02	RF4CE_NWK_SetMaxPairingTableEntries.Confirm
D5	03	RF4CE_NWK_GetMaxPairingTableEntries.Confirm
D5	04	RF4CE_NWK_SetNodeCapabilities.Confirm
D5	05	RF4CE_NWK_GetNodeCapabilies.Confirm
D5	06	RF4CE_NWK_SetVendorIdentifier.Confirm
D5	07	RF4CE_NWK_GetVendorIdentifier.Confirm
D5	08	RF4CE_NWK_SetVendorString.Confirm

Table 3-1. BlackBox Messages (continued)

D5	09	RF4CE_NWK_GetVendorString.Confirm
D5	0A	RF4CE_NWK_SetFrameCounterWindow.Confirm
D5	0B	RF4CE_NWK_GetFrameCounterWindow.Confirm
D5	0C	RF4CE_NWK_AddNewPairTableEntry.Confirm
D5	0D	RF4CE_NWK_SavePersistentData.Confirm
D5	0E	RF4CE_NWK_GenerateShortAddress.Confirm
D5	0F	RF4CE_NWK_GenerateSecurityKey.Confirm
D5	10	RF4CE_NWK_SaveFrameCounter.Confirm
D5	11	RF4CE_NWK_GetLastPacketLQI.Confirm
D5	12	RF4CE_NWK_GetNodePanId.Confirm
D5	13	RF4CE_NWK_GetNodeShortAddress.Confirm
D5	14	RF4CE_NWK_IsIdle.Confirm
D5	15	RF4CE_NWK_GetAllowedLowPowerInterval.Confirm
D7	00	PBP_PushButtonPairOrig.Confirm
D7	01	PBP_PushButtonPairRecip.Confirm
D7	04	PBP_PushButtonPairOrigContinue.Indication
D7	05	PBP_PushButtonPairRecipContinue.Indication
E1	00	ZRCProfile_AbortProcess.Confirm
D7	E1	PBP_PushButtonPairOrigContinue.Confirm
D7	E2	PBP_PushButtonPairRecipContinue.Confirm
D7	E3	PBP_AbortProcess.Confirm
DB	00	FSLProfile_Frag.Confirm
DB	01	FSLProfile_StartFrag.Indication
DB	02	FSLProfile_Frag.Indication
DB	03	FSLProfile_Poll.Confirm
DB	04	FSLProfile_PollEvent
DB	05	FSLProfile_Poll.Indication
DB	06	FSLProfile_RmtPair.Confirm
DB	07	FSLProfile_RmtPair.Indication
DB	08	FSLProfile_RmtPairRsp.Confirm
DB	09	FSLProfile_MenuBrowse.Confirm
DB	0A	FSLProfile_MenuBrowseComplete.Indication
DB	0B	FSLProfile_MenuBrowse.Indication
DB	0C	FSLProfile_DisplayMenu.Confirm

Table 3-1. BlackBox Messages (continued)

DB	0D	FSLProfile_DisplayMenuHeader.Indication
DB	0E	FSLProfile_DisplayMenuEntry.Indication
DB	0F	FSLProfile_DisplayMenuComplete.Indication
DB	10	FSLProfile_DisplayMenuMessage.Indication
DB	11	FSLProfile_DisplayMenuExit.Indication
DB	12	FSLProfile_GetSupportedFeatures.Confirm
DB	E0	FSLProfile_SetFragTxRxBufferState.Confirm
DB	E1	FSLProfile_GetFragTxRxBufferState.Confirm
DB	E2	FSLProfile_PollConfig.Confirm
DB	E3	FSLProfile_PollDataAvailable.Confirm
DE	02	ZRCProfile_Command.Indication
DE	03	ZRCProfile_Command.Confirm
E1	01	ZRCProfile_GetAttr.Confirm
E1	02	ZRCProfile_SetAttr.Confirm
E1	03	ZRCProfile_SetZRCSupportedCmds.Confirm
E1	04	ZRCProfile_GetZRCSupportedCmds.Confirm
DE	06	ZRCProfile_DiscoveryCmd.Confirm
Low power control messages		
OpcodeGroup	**Opcode**	**Service**
A3	40	ZTC-WakeUpConfig.Request
A4	40	ZTC-WakeUpConfig.Confirm
A4	41	ZTC-WakeUp.Indication
ZTC control messages		
A3	30	ZTC-WriteMemoryBlock.Request
A4	30	ZTC-WriteMemoryBlock.Confirm
A3	31	ZTC-ReadMemoryBlock.Request
A4	31	ZTC-ReadMemoryBlock.Confirm
A3	44	ZTC-GetLastPacketLQI.Request
A4	45	ZTC-GetLastPacketLQI.Confirm
A3	42	ZTC-StackStatus.Request
A4	43	ZTC-StackStatus.Confirm
A3	08	ZTC-CPU_Reset.Request
A3	00	ZTC-ModeSelect.Request

Table 3-1. BlackBox Messages (continued)

A4	00	ZTC-ModeSelect.Confirm
A3	02	ZTC-GetMode.Request
A4	02	ZTC-GetMode.Confirm
A3	DB	ZTC-WriteExtAddr.RequestA3
A4	DB	ZTC-WriteExtAddr.Confirm
A3	D2	ZTC-ReadExtAddr.Request
A4	D2	ZTC-ReadExtAddr.Confirm

3.1 BlackBox Message Structure

The following section describes BlackBox message structure that allows access to BeeStack Consumer Control Network services or utility functions. The messages are grouped according to the type of service they provide.

For messages allowing access to BeeStack Consumer Control Network services, no usage description of the structure is provided. See the Freescale BeeStack Consumer Application Reference Manual for a description of the service with the same name.

3.2 BlackBox Access to BeeStack Consumer Control Network Services

NOTE

All services described below expect the length field in little endian order, that is, the least significant byte is sent first.

The Freescale Test Tool software allows users to exercise the BeeStack Consumer BlackBox features by sending and receiving messages over the UART or I^2C interface connection.

The examples shown in this chapter start with the name of the command, followed by the actual bytes sent over the interface being used. The individual fields of the command are also shown in big-endian format. The exception to this is the Header field, which is little-endian (its actually two fields concatenated, OpcodeGroup and Opcode).

3.2.1 BeeStack Consumer Control Network RESET Service

3.2.1.1 NLME Reset.Request

Description

Allows the application entity to request a reset of the NWK layer.

Payload

```
bSetDefaultNib - 1 byte
```

Example

```
RF4CE_NLME_Reset.Request 02 D0 00 01 00 01 D0
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = D0 00
PayloadLength [2 bytes] = 00 01
SetDefaultNIB [1 byte ] = 01 (true)
Checksum      [1 byte ] = D0
```

3.2.1.2 NLME Reset.Confirm

Description

Allows the NLME sublayer to notify the application of the status of its request to reset the NWK layer.

Payload

```
Status – 1 byte
```

Example

```
RF4CE_NLME_Reset.Confirm 02 D1 A2 01 00 00 72
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = D1 A2
PayloadLength [2 bytes] = 00 01
Status        [1 byte ] = 00 (gNWSuccess_c)
Checksum      [1 byte ] = 72
```

NOTE

If the NLME_Reset.Confirm message informs the host about the success of a cold reset, then after receiving the NLME Reset.Confirm message the host application shall not initiate any service request to the BlackBox for the next 300 ms. The BlackBox needs to store sensitive information in its non volatile memory and while doing this it is not able to ‘hear’ any incoming packet on the serial interface because the platform interrupts are disabled.

3.2.2 BeeStack Consumer Control Network START Services

3.2.2.1 NLME Start.Request

Description

Allows the application to request the NLME to start a network.

Payload

This command has no payload; according to this the length field needs to be 0x0000.

Example

```
RF4CE_NLME_Start.Request 02 D0 01 00 00 D1
```

```
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = D0 01
PayloadLength [2 bytes] = 00 00
Checksum      [1 byte ] = D1
```

3.2.2.2 NLME Start.Confirm

Description

Allows the NLME to notify the application of the status of its request to start a network.

Payload

```
Status – 1 byte
```

Example

```
RF4CE_NLME_Start.Confirm 02 D1 00 01 00 00 D0
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = D1 00
PayloadLength [2 bytes] = 00 01
Status        [1 byte ] = 00 (gNWSuccess_c)
Checksum      [1 byte ] = D0
```

NOTE

If the Start.Confirm message informs the host about the success of a start operation performed on a target node for the first time after programming it or for the first time after a cold reset, then after receiving the NLME Start.Confirm message, the host application shall not initiate any service request to the BlackBox for the next 300 ms. The BlackBox needs to store sensitive information in its non volatile memory and while doing this it is not able to ‘hear’ any incoming packet on the serial interface, because the platform interrupts are disabled.

3.2.3 BeeStack Consumer Comm Status Service

Description

Allows the NLME to notify the application of the status of its response request (discovery response or pair response).

Payload

```
PairingRef – 1 byte
DstPANId – 2 bytes
DstAddrMode – 2 bytes
DstAddr – 8 bytes
Status - 1 byte
```

Example

```
RF4CE_NLME_CommStatus.Indication 02 D1 08 0D 00 00 FF FF 01 FF FF FF FF FF FF FF FF 00 D5
```

```
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = D1 08
PayloadLength [2 bytes] = 00 0D
PairingRef    [1 byte ] = 00
DstPANId      [2 bytes] = FF FF
DstAddrMode   [1 byte ] = 01
DstAddr       [8 bytes] = FF FF FF FF FF FF FF FF
Status        [1 byte ] = 00 (gNWSuccess_c)
Checksum      [1 byte ] = D5
```

NOTE

If the CommStatus.Ind message informs the host about the success of a pair recipient operation, then after receiving the NLME Comm Status.Indication message, the host application shall not initiate any service request to the BlackBox for the next 300 ms. The BlackBox needs to store sensitive information in its non volatile memory and while doing this it is not able to ‘hear’ any incoming packet on the serial interface, because the platform interrupts are disabled.

3.2.4 BeeStack Consumer Control Network DISCOVERY Service

3.2.4.1 NLME Discovery.Request

Description

Allows the application to request the NLME to discover other devices of interest that operate in the POS of the given device.

Payload

```
DstPANId – 2 bytes
DstNwkAddr – 2 bytes
RecDevType – 1 byte
OrigAppCapabilities_UserStringSpecified – 1 byte
OrigAppCapabilities_NoOfSupportedDeviceTypes – 1 byte
OrigAppCapabilities_NoOfSupportedProfiles – 1 byte
DiscDuration – 4 bytes
DiscProfileIdListSize – 1 byte
DevTypeList – array of bytes with length given by
OrigAppCapabilities_NoOfSupportedDeviceTypes field
ProfileIdList – array of bytes with length given by
OrigAppCapabilities_NoOfSupportedProfiles field
```

Example

```
RF4CE_NLME_Discovery.Request 02 D0 02 10 00 FF FF FF FF 02 00 01 01 36 6E 01 00 01 01 01
01 99
StartOfFrame                                    [1 byte ] = 02
Header                                          [2 bytes] = D0 02
PayloadLength                                   [2 bytes] = 00 10
DstPANId                                        [2 bytes] = FF FF
DstNwkAddr                                      [2 bytes] = FF FF
RecDevType                                      [1 byte ] = 02 (TV)
OrigAppCapabilities_UserStringSpecified     [1 byte ] = 00 (UserStringNotIncludedInFrame)
OrigAppCapabilities_NoOfSupportedDeviceTypes [1 byte ] = 01
(OneDeviceTypeInDeviceTypeList)
OrigAppCapabilities_NoOfSupportedProfiles    [1 byte ] = 01
(OneSupportedProfilesInProfileIdList)
DiscDuration                                    [4 bytes] = 00 01 6E 36
DiscProfileIdListSize                           [1 byte ] = 01
DevTypeList                                     [1 byte ] = 01
        DevTypeList[0] = 01
ProfileIdList                                   [1 byte ] = 01
        ProfileIdList[0] = 01
DiscProfileIdList                               [1 byte ] = 01
        DiscProfileIdList[0] = 01
Checksum                                        [1 byte ] = 99
```

3.2.4.2 NLME Discovery.Confirm

Description

Allows the NLME to notify the application of the status of its request to perform a network discovery.

Payload

```
Status – 1 byte
NumNodes – 1 byte
NodeDescList – array of bytes.
```

Example

```
RF4CE_NLME_Discovery.Confirm 02 D1 02 33 00 00 01 00 0F 4F AB BB BB BB BB BB BB BB 1B 03
05 00 56 45 4E 44 4F 52 00 13 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 02 01 B7 01
B7 00 05 24 08 2C B7 0B
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = D1 02
PayloadLength [2 bytes] = 00 33
Status        [1 byte ] = 00 (gNWSuccess_c)
NumNodes      [1 byte ] = 01
NodeDescList  [49 bytes] = 00 0F 4F AB BB BB BB BB BB BB BB 1B 03 05 00 56 45 4E 44 4F
52 00 13 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 02 01 B7 01 B7 00 05 24 08 2C B7
NodeDescList[0] = B7 2C 08 24 05 00 B7 01 B7 01 02 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 13 00 52 4F 44 4E 45 56 00 05 03 1B BB BB BB BB BB BB BB AB 4F 0F 00
Checksum      [1 byte ] = 0B
```

3.2.4.3 NLME Discovery.Indication

Description

Signal the application that a discovery request frame was received over the air.

Payload

```
Status - 1 byte
SrcIEEEAddr - 8 bytes
OrgNodeCapabilities - 1 byte
OrgVendorId - 2 bytes
OrgVendorString - 7 byte
OrigAppCapabilities_UserStringSpecified - 1 byte
OrigAppCapabilities_NoOfSupportedDeviceTypes - 1 byte
OrigAppCapabilities_NoOfSupportedProfiles - 1 byte
OrgUserString – 0 or 15 bytes
OrgDevTypeList – array of bytes
OrgProfileIdList – array of bytes
RecDevType - 1 byte
RxLinkQuality - 1 byte
```

Example

```
RF4CE_NLME_Discovery.Indication 02 D1 03 1A 00 00 AA AA AA AA AA AA AA 1A 00 05 00 56 45
4E 44 4F 52 00 00 01 01 01 01 02 AA D1
StartOfFrame                                  [1 byte ] = 02
```

```
Header                                          [2 bytes] = D1 03
PayloadLength                                   [2 bytes] = 00 1A
Status                                          [1 byte ] = 00 (gNWSuccess_c)
SrcIEEEAddr                                     [8 bytes] = 1A AA AA AA AA AA AA AA
OrgNodeCapabilities                             [1 byte ] = 00
OrgVendorId                                     [2 bytes] = 00 05
OrgVendorString                                 [7 bytes] = 00 52 4F 44 4E 45 56
OrigAppCapabilities_UserStringSpecified         [1 byte ] = 00
OrigAppCapabilities_NoOfSupportedDeviceTypes    [1 byte ] = 01
OrigAppCapabilities_NoOfSupportedProfiles       [1 byte ] = 01
OrgUserString                                   [0 bytes] =
OrgDevTypeList                                  [1 byte ] = 01
OrgDevTypeList[0] = 01
OrgProfileIdList                                [1 byte ] = 01
OrgProfileIdList[0] = 01
RecDevType                                      [1 byte ] = 02 (TV)
RxLinkQuality                                   [1 byte ] = AA
Checksum                                        [1 byte ] = D1
```

3.2.4.4 NLME Discovery.Response

Description

Allows the application to request that the NLME respond to the discovery request frame.

Payload

```
Status – 1 byte
DstIEEEAddr – array of 8 bytes
OrigAppCapabilities_UserStringSpecified – 1 byte
OrigAppCapabilities_NoOfSupportedProfiles – 1 byte
DiscReqLQI – 1 byte
DevTypeList – array of bytes.
ProfileIdList – array of bytes.
```

Example

```
RF4CE_NLME_Discovery.Response 02 D0 03 0F 00 00 D1 DD DD DD DD DD DD DD DD 00 01 01 F3 01
01 23
StartOfFrame                                    [1 byte ] = 02
Header                                          [2 bytes] = D0 03
Status                                          [1 byte ] = 00 (gNWSuccess_c)
DstIEEEAddr                                     [8 bytes] = DD DD DD DD DD DD DD D1
OrigAppCapabilities_UserStringSpecified     [1 byte ] = 00 (UserStringNotIncludedInFrame)
OrigAppCapabilities_NoOfSupportedDeviceTypes [1 byte ] = 01
(OneDeviceTypeInDeviceTypeList)
OrigAppCapabilities_NoOfSupportedProfiles    [1 byte ] = 01
(OneSupportedProfilesInProfileIdList)
DiscReqLQI                                      [1 byte ] = F3
DevTypeList                                     [1 byte ] = 01
        DevTypeList[0] = 01
ProfileIdList                                   [1 byte ] = 01
        ProfileIdList[0] = 01
Checksum                                        [1 byte ] = 23
```

3.2.5 BeeStack Consumer Control Network PAIR Service

3.2.5.1 NLME Pair.Request

Description

Allows the application to request the NLME to pair with another device. This request is normally issued after a discovery process.

Payload

```
LogicalChannel – 1 byte
DstPANId – 2 bytes
DstIEEEAddr – array of 8 bytes
OrigAppCapabilities_UserStringSpecified – 1 byte
OrigAppCapabilities_NoOfSupportedDeviceTypes – 1 byte
OrigAppCapabilities_NoOfSupportedProfiles – 1 byte
KeyExTransferCount – 1 byte
DevTypeList – array of bytes with length given by
OrigAppCapabilities_NoOfSupportedDeviceTypes field
ProfileIdList – array of bytes with length given by
OrigAppCapabilities_NoOfSupportedProfiles field
```

Example

```
RF4CE_NLME_Pair.Request 02 D0 04 11 00 0F 95 CA C1 CC CC CC CC CC CC CC 00 01 01 0A 01 01 92
StartOfFrame                                    [1 byte ] = 02
Header                                          [2 bytes] = D0 04
LogicalChannel                                  [1 byte ] = 0F
DstPANId                                        [2 bytes] = CA 95
DstIEEEAddr                                     [8 bytes] = CC CC CC CC CC CC CC C1
OrigAppCapabilities_UserStringSpecified     [1 byte ] = 00 (UserStringNotIncludedInFrame)
OrigAppCapabilities_NoOfSupportedDeviceTypes [1 byte ] = 01
(OneDeviceTypeInDeviceTypeList)
OrigAppCapabilities_NoOfSupportedProfiles    [1 byte ] = 01
(OneSupportedProfilesInProfileIdList)
KeyExTransferCount                              [1 byte ] = 0A
DevTypeList                                     [1 byte ] = 01
        DevTypeList[0] = 01
ProfileIdList                                   [1 byte ] = 01
        ProfileIdList[0] = 01
Checksum                                        [1 byte ] = 92
```

3.2.5.2 NLME Pair.Confirm

Description

Allows the NLME to notify the application of the status of its request to pair with another device.

Payload

```
Status – 1 byte
PairingRef – 1 byte
RecVendorId – 2 bytes
RecVendorString – array of 7 bytes
RecAppCapabilities_UserStringSpecified – 1 byte
RecAppCapabilities_NoOfSupportedDeviceTypes – 1 byte
RecAppCapabilities_NoOfSupportedProfiles – 1 byte
RecUserString – array of bytes
RecDevTypeList – array of bytes
RecProfileIdList – array of bytes
```

Example

```
RF4CE_NLME_Pair.Confirm 02 D1 04 1F 00 00 00 05 00 56 45 4E 44 4F 52 00 0F 01 01 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 02 01 C7
StartOfFrame                                  [1 byte ] = 02
Header                                        [2 bytes] = D1 04
PayloadLength                                 [2 bytes] = 00 1F
Status                                        [1 byte ] = 00 (gNWSuccess_c)
PairingRef                                    [1 byte ] = 00
RecVendorId                                   [2 bytes] = 00 05
RecVendorString                               [7 bytes] = 00 52 4F 44 4E 45 56
RecAppCapabilities_UserStringSpecified        [1 byte ] = 0F
RecAppCapabilities_NoOfSupportedDeviceTypes [1 byte ] = 01
RecAppCapabilities_NoOfSupportedProfiles      [1 byte ] = 01
RecUserString                                 [15 bytes] = 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00
         RecUserString[0] = 00
         RecUserString[1] = 00
         RecUserString[2] = 00
         RecUserString[3] = 00
         RecUserString[4] = 00
         RecUserString[5] = 00
         RecUserString[6] = 00
         RecUserString[7] = 00
         RecUserString[8] = 00
         RecUserString[9] = 00
         RecUserString[10] = 00
         RecUserString[11] = 00
         RecUserString[12] = 00
         RecUserString[13] = 00
         RecUserString[14] = 00
RecDevTypeList                                [1 byte ] = 02
         RecDevTypeList[0] = 02
RecProfileIdList                              [1 byte ] = 01
         RecProfileIdList[0] = 01
Checksum                                      [1 byte ] = C7
```

NOTE

If the NLME_Pair.Confirm message informs the host that the pair process completed successfully (status field set to gNWSuccess_c or gNWDuplicatePairing_c) then after receiving the NLME Pair.Confirm message the host application shall not initiate any service request to the BlackBox for the next 300 ms. The BlackBox needs to store sensitive information in its non volatile memory and while doing this it is not able to 'hear' any incoming packet on the serial interface, because the platform interrupts are disabled.

3.2.5.3 NLME Pair.Indication

Description

Signal the application that a pair request frame was received over the air.

Payload

```
Status – 1 byte
SrcPANId – 2 bytes
SrcIEEEAddr – 8 bytes
OrgNodeCapabilities – 1 byte
OrgVendorId – 2 bytes
OrgVendorString – 7 bytes
OrigAppCapabilities_UserStringSpecified – 1 byte
OrigAppCapabilities_NoOfSupportedDeviceTypes – 1 byte
OrigAppCapabilities_NoOfSupportedProfiles – 1 byte
OrgUserString – 15 bytes
OrgDevTypeList – array of bytes
OrgProfileIdList – array of bytes
KeyExTransferCount - 1 byte
ProvPairingRef – 1 byte
```

Example

```
RF4CE_NLME_Pair.Indication 02 D1 05 1C 00 00 FF FF FF FF FF FF FF FF FF 0C 05 00 56
45 4E 44 4F 52 00 00 01 01 01 01 0F 00 CA
StartOfFrame                                    [1 byte ] = 02
Header                                          [2 bytes] = D1 05
PayloadLength                                   [2 bytes] = 00 1C
Status                                          [1 byte ] = 00 (gNWSuccess_c)
SrcPANId                                        [2 bytes] = FF FF
SrcIEEEAddr                                     [8 bytes] = FF FF FF FF FF FF FF FF
OrgNodeCapabilities                             [1 byte ] = 0C
OrgVendorId                                     [2 bytes] = 00 05
OrgVendorString                                 [7 bytes] = "VENDOR"
OrigAppCapabilities_UserStringSpecified         [1 byte ] = 00
OrigAppCapabilities_NoOfSupportedDeviceTypes    [1 byte ] = 01
OrigAppCapabilities_NoOfSupportedProfiles       [1 byte ] = 01
OrgUserString                                   [0 bytes] =
OrgDevTypeList                                  [1 byte ] = 01
OrgDevTypeList[0] = 01
OrgProfileIdList                                [1 byte ] = 01
```

```
OrgProfileIdList[0] = 01
KeyExTransferCount                          [1 byte ] = 0F
ProvPairingRef                              [1 byte ] = 00
Checksum                                    [1 byte ] = CA
```

3.2.5.4 NLME Pair.Response

Description

Allows the application to request that the NLME respond to a pair request frame.

Payload

```
Status – 1 byte
ProvPairingRef – 1 byte
DestPanId – 2 bytes
DestAddr – array of 8 bytes
OrigAppCapabilities_UserStringSpecified – 1 byte
OrigAppCapabilities_NoOfSupportedDeviceTypes – 1 byte
OrigAppCapabilities_NoOfSupportedProfiles – 1 byte
DevTypeList – array of bytes with length given by
OrigAppCapabilities_NoOfSupportedDeviceTypes field
ProfileIdList – array of bytes with length given by
OrigAppCapabilities_NoOfSupportedProfiles field
```

Example

```
RF4CE_NLME_Pair.Response 02 D0 05 11 00 00 00 FF FF D1 DD DD DD DD DD DD DD 00 01 01 01
01 C8
StartOfFrame                                [1 byte ] = 02
Header                                      [2 bytes] = D0 05
Status                                      [1 byte ] = 00 (gNWSuccess_c)
ProvPairingRef                              [1 byte ] = 00
DestPanId                                   [2 bytes] = FF FF
DestAddr                                    [8 bytes] = DD DD DD DD DD DD DD D1
OrigAppCapabilities_UserStringSpecified     [1 byte ] = 00 (UserStringNotIncludedInFrame)
OrigAppCapabilities_NoOfSupportedDeviceTypes [1 byte ] = 01
(OneDeviceTypeInDeviceTypeList)
OrigAppCapabilities_NoOfSupportedProfiles    [1 byte ] = 01
(OneSupportedProfilesInProfileIdList)
DevTypeList                                 [1 byte ] = 01
        DevTypeList[0] = 01
ProfileIdList                               [1 byte ] = 01
        ProfileIdList[0] = 01
Checksum                                    [1 byte ] = C8
```

3.2.6 BeeStack Consumer Control Network UNPAIR Service

3.2.6.1 NLME Unpair.Request

Description

Allows the application to request the NLME to remove a pairing link both in the local and remote pairing tables.

Payload

```
PairingRef – 1 byte
```

Example

```
RF4CE_NLME_UnPair.Request 02 D0 06 01 00 00 D7
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = D0 06
PayloadLength [2 bytes] = 00 01
PairingRef    [1 byte ] = 00
Checksum      [1 byte ] = D7
```

3.2.6.2 NLME Unpair.Confirm

Description

Allows the NLME to notify the application of the status of its request to remove a pairing link.

Payload

```
Status – 1 byte
PairingRef – 1 byte
```

Example

```
RF4CE_NLME_UnPair.Confirm 02 D1 06 02 00 00 00 D5
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = D1 06
PayloadLength [2 bytes] = 00 02
Status        [1 byte ] = 00 (gNWSuccess_c)
PairingRef    [1 byte ] = 00
Checksum      [1 byte ] = D5
```

NOTE

If the NLME_Unpair.Confirm message informs the host that the unpair process completed successfully then after receiving the NLME Unpair.Confirm message the host application shall not initiate any service request to the BlackBox for the next 300 ms. The BlackBox needs to store sensitive information in its non volatile memory and while doing this it is not able to ‘hear’ any incoming packet on the serial interface, because the platform interrupts are disabled.

3.2.6.3 NLME Unpair.Indication

Description

Signal the application that an unpair request frame was received over the air.

Payload

```
PairingRef – 1 byte
```

Example

```
RF4CE_NLME_UnPair.Indication 02 D1 07 01 00 00 D7
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = D1 07
PayloadLength [2 bytes] = 00 01
PairingRef    [1 byte ] = 00
Checksum      [1 byte ] = D7
```

3.2.6.4 NLME Unpair.Response

Description

Allows the application to notify the NLME that the pairing link. Can be removed from the pairing table.

Payload

```
PairingRef – 1 byte
```

Example

```
RF4CE_NLME_UnPair.Response 02 D0 0A 01 00 00 DB
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = D0 0A
PayloadLength [2 bytes] = 01 00
PairingRef    [1 byte ] = 00
Checksum      [1 byte ] = DB
```

3.2.6.5 NLME UnpairResponse.Confirm

Description

Allows the NLME to notify the application of the status of its request to respond to an unpair request command.

Payload

```
Status – 1 byte
```

Example

```
RF4CE_NLME_UnPairResponse.Confirm 02 D1 A4 01 00 B2 C6
StartOfFrame  [1 byte ] = 02
```

```
Header        [2 bytes] = D1 A4
PayloadLength [2 bytes] = 00 01
Status        [1 byte ] = B2 (gNWDeviceIdNotPaired_c)
Checksum      [1 byte ] = C6
```

NOTE

If the NLME_UnpairResponse.Confirm message informs the host that the unpair response process completed successfully then after receiving the NLME UnpairResponse.Confirm message the host application shall not initiate any service request to the BlackBox for the next 300 ms. The BlackBox needs to store sensitive information in its non volatile memory and while doing this it is not able to 'hear' any incoming packet on the serial interface, because the platform interrupts are disabled.

3.2.7 BeeStack Consumer Control Network GET Service

3.2.7.1 NWK Get.Request

Description

Allows the application to request the values of a NIB attribute from NLME.

Payload

```
NIBAttribute – 1 byte
NIBAttributeIndex – 1 byte
```

Example

```
RF4CE_NLME_Get.Request 02 D0 07 02 00 64 00 B1
StartOfFrame      [1 byte ] = 02
Header            [2 bytes] = D0 07
PayloadLength   [2 bytes] = 02 00
NIBAttribute      [1 byte ] = 64 (nwkDutyCycle)
NIBAttributeIndex [1 byte ] = 00
Checksum          [1 byte ] = B1
```

3.2.7.2 NWK Get.Confirm

Description

Allows the NLME to notify the application of the status of its request for the value of a NIB attribute.

Payload

```
NIBAttributeSize – 1 byte
Status – 1 byte
NIBAttribute – 1 byte
NIBAttributeIndex – 1 byte
NIBAttributeValue – array of bytes
```

Example

```
RF4CE_NLME_Get.Confirm 02 D1 A0 06 00 02 00 64 00 00 00 11
StartOfFrame      [1 byte ] = 02
Header            [2 bytes] = D1 A0
PayloadLength     [2 bytes] = 00 06
NIBAttributeSize  [1 byte ] = 02
Status            [1 byte ] = 00 (gNWSuccess_c)
NIBAttribute      [1 byte ] = 64 (nwkDutyCycle)
NIBAttributeIndex [1 byte ] = 00
NIBAttributeValue [2 bytes] = 00 00
         NIBAttributeValue[0] = 00
         NIBAttributeValue[1] = 00
Checksum          [1 byte ] = 11
```

3.2.8 BeeStack Consumer Control Network SET Service

3.2.8.1 NWK Set.Request

Description

Allows the application to request the NLME to change the value of a NIB attribute.

Payload

```
NIBAttribute – 1 byte
NIBAttributeIndex – 1 byte
NIBAttributeValue – array of bytes
```

Example

```
RF4CE_NLME_Set.Request 02 D0 08 06 00 6E 00 05 00 00 00 B5
StartOfFrame      [1 byte ] = 02
Header            [2 bytes] = D0 08
PayloadLength    [2 bytes] = 00 03
NIBAttribute      [1 byte ] = 6E (nwkScanDuration)
NIBAttributeIndex [1 byte ] = 00
NIBAttributeValue [1 bytes] = 05
Checksum          [1 byte ] = B5
```

3.2.8.2 NWK Set.Confirm

Description

Allows the NLME to notify the application of the status of its request to change the value of a NIB attribute.

Payload

```
Status – 1 byte
NIBAttribute – 1 byte
NIBAttributeIndex – 1 byte
```

Example

```
RF4CE_NLME_Set.Confirm 02 D1 A1 03 00 E8 6E 00 F5
StartOfFrame      [1 byte ] = 02
Header            [2 bytes] = D1 A1
PayloadLength     [2 bytes] = 00 03
Status            [1 byte ] = 00 (gNWSuccess_c)
NIBAttribute      [1 byte ] = 6E (nwkScanDuration)
NIBAttributeIndex [1 byte ] = 00
Checksum          [1 byte ] = 1D
```

NOTE

If the NLME_Set.Confirm message informs the host that the set process completed successfully then after receiving the NLME Set.Confirm message the host application shall not initiate any service request to the BlackBox for the next 300 ms. The BlackBox needs to store sensitive information in its non volatile memory and while doing this it is not able to 'hear' any incoming packet on the serial interface, because the platform interrupts are disabled.

3.2.9 BeeStack Consumer Control Network RX_ENABLE Service

3.2.9.1 NWK RX_Enable.Request

Description

Allows the application to request that the receiver is either enabled (for a finite period or until further notice) or disabled.

Payload

```
RxOnDuration – 4 bytes
```

Example

```
RF4CE_NLME_RxEnable.Request 02 D0 09 04 00 FF FF FF 00 22
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = D0 09
PayloadLength [2 bytes] = 04 00
RxOnDuration  [4 bytes] = 00 FF FF FF
Checksum      [1 byte ] = 22
```

3.2.9.2 NWK RX_Enable.Confirm

Description

Allows the NLME to inform the application of the status of its request to enable or disable the receiver.

Payload

```
Status – 1 byte
```

Example

```
RF4CE_NLME_RxEnable.Confirm 02 D1 A3 01 00 00 73
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = D1 A3
PayloadLength [2 bytes] = 00 01
Status        [1 byte ] = 00 (gNWSuccess_c)
Checksum      [1 byte ] = 73
```

NOTE

If the NLME_RxEnable.Confirm message informs the host that the NLME Rx Enable process completed successfully then after receiving the NLME RxEnable.Confirm message the host application shall not initiate any service request to the BlackBox for the next 300 ms. The BlackBox needs to store sensitive information in its non volatile memory and while doing this it is not able to 'hear' any incoming packet on the serial interface, because the platform interrupts are disabled.

3.2.10 BeeStack Consumer Control Network AUTO_DISCOVERY Service

3.2.10.1 NLME AutoDiscovery.Request

Description

Allows the application to request the NLME to automatically respond to incoming discovery request command frames.

Payload

```
RecAppCapabilities_UserStringSpecified – 1 byte
RecAppCapabilities_NoOfSupportedDeviceTypes – 1 byte
RecAppCapabilities_NoOfSupportedProfiles – 1 byte
AutoDiscDuration – 4 bytes
RecDevTypeList – array of bytes with length given by
ecAppCapabilities_NoOfSupportedDeviceTypes field
RecProfileIdList – array of bytes with length given by
RecAppCapabilities_NoOfSupportedProfiles field
```

Example

```
RF4CE_NLME_AutoDiscovery.Request 02 D0 0B 09 00 00 01 01 36 6E 01 00 01 01 8B
StartOfFrame                                [1 byte ] = 02
Header                                      [2 bytes] = D0 0B
PayloadLength                               [2 bytes] = 00 09
RecAppCapabilities_UserStringSpecified     [1 byte ] = 00 (UserStringNotIncludedInFrame)
RecAppCapabilities_NoOfSupportedDeviceTypes [1 byte ] = 01
(OneDeviceTypeInDeviceTypeList)
RecAppCapabilities_NoOfSupportedProfiles    [1 byte ] = 01
(OneSupportedProfilesInProfileIdList)
AutoDiscDuration                            [4 bytes] = 00 01 6E 36
RecDevTypeList                              [1 byte ] = 01
        RecDevTypeList[0] = 01
RecProfileIdList                            [1 byte ] = 01
```

```
        RecProfileIdList[0] = 01
Checksum                                    [1 byte ] = 8B
```

3.2.10.2 NLME AutoDiscovery.Confirm

Description

Allows the NLME to notify the application of the status of its request to enter auto discovery response mode.

Payload

```
Status – 1 byte
OrigIEEEAddr – array of 8 bytes
OrigPANId – 2 bytes
```

Example

```
RF4CE_NLME_AutoDiscovery.Confirm 02 D1 01 0B 00 B8 00 00 00 00 00 00 00 00 00 00 63
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = D1 01
PayloadLength [2 bytes] = 00 0B
Status        [1 byte ] = B8 (gNWDiscoveryTimeout_c)
OrigIEEEAddr  [8 bytes] = 00 00 00 00 00 00 00 00
OrigPANId     [2 bytes] = 00 00
Checksum      [1 byte ] = 63
```

3.2.11 BeeStack Consumer Control Network UPDATE_KEY Service

3.2.11.1 NLME UpdateKey.Request

Description

Allows the application to request the NLME to change the security link key of an entry in the pairing table.

Payload

```
PairingRef – 1 byte
NewLinkKey – array of 16 bytes
```

Example

```
RF4CE_NLME_UpdateKey.Request 02 D0 0C 11 00 00 AB 89 00 00 00 00 00 00 00 00 00 00 00 00
34 12 C9
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = D0 0C
PayloadLength [2 bytes] = 11 00
PairingRef    [1 byte ] = 00
NewLinkKey    [16 bytes] = 12 34 00 00 00 00 00 00 00 00 00 00 00 00 89 AB
Checksum      [1 byte ] = C9
```

3.2.11.2 NLME UpdateKey.Confirm

Description

Allows the NLME to notify the application of the status of its request to change the security link key of a pairing table entry.

Payload

```
Status – 1 byte
PairingRef – 1 byte
```

Example

```
RF4CE_NLME_UpdateKey.Confirm 02 D1 A7 02 00 B2 00 C6
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = D1 A7
PayloadLength [2 bytes] = 00 02
Status        [1 byte ] = B2 (gNWDeviceIdNotPaired_c)
PairingRef    [1 byte ] = 00
Checksum      [1 byte ] = C6
```

NOTE

If the NLME_UpdateKey.Confirm message informs the host that the update key process completed successfully then after receiving the NLME UpdateKey.Confirm message the host application shall not initiate any service request to the BlackBox for the next 300 ms. The BlackBox needs to store sensitive information in its non volatile memory and while doing this it is not able to ‘hear’ any incoming packet on the serial interface, because the platform interrupts are disabled.

3.2.12 BeeStack Consumer Control Network DATA Service

3.2.12.1 NLDE Data.Request

Description

Requests the transfer of data unit (NSDU).

Payload

```
PairingRef – 1 byte
ProfileId – 1 byte
VendorId – 2 bytes
TxOptions – 1 byte
nsduLength – 1 byte
nsdu – array of bytes with length given by nsduLength field
```

Example

```
RF4CE_NLDE_Data.Request 02 D2 00 08 00 00 01 05 00 46 02 01 02 99
StartOfFrame [1 byte ] = 02
```

```
Header       [2 bytes] = D2 00
PayloadLength[2 bytes] = 08 00
PairingRef   [1 byte ] = 00
ProfileId    [1 byte ] = 01
VendorId     [2 bytes] = 00 05
TxOptions    [1 byte ] = 46
nsduLength   [1 byte ] = 02
nsdu         [2 bytes] = 01 02
        nsdu[0] = 01
        nsdu[1] = 02
Checksum     [1 byte ] = 99
```

3.2.12.2 NLDE Data.Confirm

Description

Inform the application of the status of its request to transfer a data unit (NSDU).

Payload

```
Status – 1 byte
PairingRef – 1 byte
ProfileId - 1 byte
```

Example

```
RF4CE_NLDE_Data.Confirm 02 D3 00 03 00 00 00 01 D1
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = D3 00
PayloadLength [2 bytes] = 00 03
Status        [1 byte ] = 00 (gNWSuccess_c)
PairingRef    [1 byte ] = 00
ProfileId     [1 byte ] = 01
Checksum      [1 byte ] = D1
```

3.2.12.3 NLDE Data.Indication

Description

Signal the application that a data was received over the air.

Payload

```
PairingRef – 1 byte
ProfileId – 1 byte
VendorId -2 bytes
RxLinkQuality – 1 byte
RxFlags – 1 byte
nsduLength – 1 byte
nsdu – array of byte with the length given by nsduLength field.
```

Example

```
RF4CE_NLDE_Data.Indication 02 D3 01 09 00 00 01 05 00 B9 04 02 01 02 63
StartOfFrame  [1 byte ] = 02
```

```
Header         [2 bytes] = D3 01
PayloadLength [2 bytes] = 00 09
PairingRef     [1 byte ] = 00
ProfileId      [1 byte ] = 01
VendorId       [2 bytes] = 00 05
RxLinkQuality [1 byte ] = B9
RxFlags        [1 byte ] = 04
nsduLength     [1 byte ] = 02
nsdu           [2 bytes] = 01 02
         nsdu[0] = 01
         nsdu[1] = 02
Checksum       [1 byte ] = 63
```

3.2.13 BeeStack Consumer Push Button Pairing Service

3.2.13.1 PBP_PushButtonPairOrig.Request

Description

Establish a link between two devices. This request starts the discovery pairing process described in ZRC Profile. It is issued to an originator (controller or target) node.

Payload

```
RecipPanId                                    - 2 bytes
RecipShortAddress                             - 2 bytes
RecipDeviceType                               - 1 byte
OrigAppCapabilities_UserStringSpecified       - 1 byte
OrigAppCapabilities_NoOfSupportedDeviceTypes - 1 byte
OrigAppCapabilities_NoOfSupportedProfiles     - 1 byte
OrigDevTypeList - OrigAppCapabilities_NoOfSupportedDeviceTypes bytes
OrigProfileIdList - OrigAppCapabilities_NoOfSupportedProfiles bytes
DiscProfileIdListSize                         - 1 byte
DiscProfileIdList                             - DiscProfileIdListSize bytes
KeyExTransferCount                            - 1 byte
RequestAppAcceptToPair                        - 1 byte
TimeToWaitAppAcceptToPair                     - 2 bytes
```

Example

```
PBP_PushButtonPairOrig.Request 02 D6 00 10 00 FF FF FF FF 01 00 01 01 01 01 01 01 0F 00
E8 03 23
StartOfFrame                                    [1 byte ] = 02
Header                                          [2 bytes] = D6 00
PayloadLength                                   [2 bytes] = 00 10
RecipPanId                                      [2 bytes] = FF FF
RecipShortAddress                               [2 bytes] = FF FF
RecipDeviceType                                 [1 byte ] = 01
OrigAppCapabilities_UserStringSpecified     [1 byte ] = 00 (UserStringNotIncludedInFrame)
OrigAppCapabilities_NoOfSupportedDeviceTypes [1 byte ] = 01
(OneDeviceTypeInDeviceTypeList)
OrigAppCapabilities_NoOfSupportedProfiles    [1 byte ] = 01
(OneSupportedProfilesInProfileIdList)
OrigDevTypeList                                 [1 byte ] = 01
OrigDevTypeList[0] = 01
```

```
OrigProfileIdList                           [1 byte ] = 01
OrigProfileIdList[0] = 01
DiscProfileIdListSize                       [1 byte ] = 01
DiscProfileIdList                           [1 byte ] = 01
DiscProfileIdList[0] = 01
KeyExTransferCount                          [1 byte ] = 0F
RequestAppAcceptToPair                      [1 byte ] = 00 (FALSE)
TimeToWaitAppAcceptToPair                   [2 bytes] = 03 E8
Checksum                                    [1 byte ] = 23
```

3.2.13.2 PBP_PushButtonPairRecip.Request

Description

Establish a link between two devices. This request starts the discovery pairing process described in ZRC Profile. It is issued to a recipient (target) node.

Payload

```
OrigAppCapabilities_UserStringSpecified      - 1 byte
OrigAppCapabilities_NoOfSupportedDeviceTypes - 1 byte
OrigAppCapabilities_NoOfSupportedProfiles    - 1 byte
OrigDevTypeList - OrigAppCapabilities_NoOfSupportedDeviceTypes bytes
OrigProfileIdList - OrigAppCapabilities_NoOfSupportedProfiles bytes
DiscLQIThreshold                             - 1 byte
RequestAppAcceptToPair                       - 1 byte
TimeToWaitPairInd                            - 2 bytes
TimeToWaitAppAcceptToPair                    - 2 bytes
```

Example

```
PBP_PushButtonPairRecip.Request 02 D6 01 0B 00 00 01 01 01 01 01 00 E8 03 E8 03 DD
StartOfFrame                                [1 byte ] = 02
Header                                      [2 bytes] = D6 01
PayloadLength                               [2 bytes] = 00 0B
OrigAppCapabilities_UserStringSpecified    [1 byte ] = 00 (UserStringNotIncludedInFrame)
OrigAppCapabilities_NoOfSupportedDeviceTypes [1 byte ] = 01
(OneDeviceTypeInDeviceTypeList)
OrigAppCapabilities_NoOfSupportedProfiles    [1 byte ] = 01
(OneSupportedProfilesInProfileIdList)
OrigDevTypeList                             [1 byte ] = 01
OrigDevTypeList[0] = 01
OrigProfileIdList                           [1 byte ] = 01
OrigProfileIdList[0] = 01
DiscLQIThreshold                            [1 byte ] = 01
RequestAppAcceptToPair                      [1 byte ] = 00 (FALSE)
TimeToWaitPairInd                           [2 bytes] = 03 E8
TimeToWaitAppAcceptToPair                   [2 bytes] = 03 E8
Checksum                                    [1 byte ] = DD
```

3.2.13.3 PBP_PushButtonPairOrigContinue.Response

Description

PBP_PushButtonPairOrigContinueResponse instructs the ZRC profile layer to continue or not the controller-side push-button pairing procedure with the pair process. If the application chooses to not continue with the pair process, the push-button pairing procedure is aborted immediately.

Payload

```
Continue - 1 byte
```

Example

```
PBP_PushButtonPairOrigContinue.Response 02 D6 03 01 00 01 D5
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = D6 03
PayloadLength [2 bytes] = 00 01
Continue      [1 byte ] = 01 (TRUE)
Checksum      [1 byte ] = D5
```

3.2.13.4 PBP_PushButtonPairRecipContinue.Response

Description

PBP_PushButtonPairRecipContinueResponse instructs the ZRC profile layer to continue or not the target-side push-button pairing procedure with the pair process. If the application chooses to not continue with the pair process, the push-button pairing procedure is aborted immediately.

Payload

```
Continue - 1 byte
```

Example

```
PBP_PushButtonPairRecipContinue.Response 02 D6 04 01 00 01 D2
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = D6 04
PayloadLength [2 bytes] = 00 01
Continue      [1 byte ] = 01 (TRUE)
Checksum      [1 byte ] = D2
```

3.2.13.5 PBP_PushButtonPairOrig.Confirm

Description

Confirmation for PBP_PushButtonPairOrig.Request.Only the status field is received when it is not success. If the status is success, all the message fields are received.

Payload

```
Status                                    -1 byte
```

```
PairingRef                                    -1 byte
RecVendorId                                   -2 bytes
RecVendorString                               -7 bytes
RecAppCapabilities_UserStringSpecified      -1 byte
RecAppCapabilities_NoOfSupportedDeviceTypes -1 byte
RecAppCapabilities_NoOfSupportedProfiles    -1 byte
RecUserString                          - 15 bytes if RecAppCapabilities_UserStringSpecified
is TRUE
RecDevTypeList                          - RecAppCapabilities_NoOfSupportedDeviceTypes bytes
RecProfileIdList                           -RecAppCapabilities_NoOfSupportedProfiles bytes
```

Example

```
PBP_PushButtonPairOrig.Confirm 02 D7 00 10 00 00 00 05 00 56 45 4E 44 4F 52 00 00 01 01
01 01 C6
StartOfFrame                                  [1 byte ] = 02
Header                                        [2 bytes] = D7 00
PayloadLength                                 [2 bytes] = 00 10
Status                                        [1 byte ] = 00 (gNWSuccess_c)
PairingRef                                    [1 byte ] = 00
RecVendorId                                   [2 bytes] = 00 05
RecVendorString                               [7 bytes] = "VENDOR"
RecAppCapabilities_UserStringSpecified      [1 byte ] = 00
RecAppCapabilities_NoOfSupportedDeviceTypes [1 byte ] = 01
RecAppCapabilities_NoOfSupportedProfiles    [1 byte ] = 01
RecUserString                                 [0 bytes] =
RecDevTypeList                                [1 byte ] = 01
RecDevTypeList[0] = 01
RecProfileIdList                              [1 byte ] = 01
RecProfileIdList[0] = 01
Checksum                                      [1 byte ] = C6
```

NOTE

If the PBP_PushButtonPairOrig.Confirm message informs the host that the ZRC pair process completed successfully then after receiving the PBP_PushButtonPairOrig.Confirm message the host application shall not initiate any service request to the BlackBox for the next 300 ms. The BlackBox needs to store sensitive information in its non volatile memory and while doing this it is not able to ‘hear’ any incoming packet on the serial interface, because the platform interrupts are disabled.

3.2.13.6 PBP_PushButtonPairRecip.Confirm

Description

Confirmation for PBP_PushButtonPairRecip.Request.Only the status field is received when it is not success. If the status is success, all the message fields are received.

Payload

```
Status                                        - 1 byte
SrcPANId                                      - 2 bytes
SrcIEEEAddr                                   - 8 bytes
OrgNodeCapabilities                           - 1 byte
```

```
OrgVendorId                                    - 2 bytes
OrgVendorString                                - 7 bytes
OrigAppCapabilities_UserStringSpecified      - 1 byte
OrigAppCapabilities_NoOfSupportedDeviceTypes - 1 byte
OrigAppCapabilities_NoOfSupportedProfiles    - 1 byte
OrgUserString                                  - 15 bytes if
OrigAppCapabilities_UserStringSpecified is TRUE, else 0 byte
OrgDevTypeList                                 - 1 byte
OrgProfileIdList                               - 1 byte
KeyExTransferCount                             - 1 byte
DeviceId                                       - 1 byte
```

Example

```
PBP_PushButtonPairRecip.Confirm 02 D7 01 1C 00 00 FF FF FF FF FF FF FF FF FF FF 0C 05 00
56 45 4E 44 4F 52 00 00 01 01 01 01 0F 00 C8
StartOfFrame                                   [1 byte ] = 02
Header                                         [2 bytes] = D7 01
PayloadLength                                  [2 bytes] = 00 1C
Status                                         [1 byte ] = 00 (gNWSuccess_c)
SrcPANId                                       [2 bytes] = FF FF
SrcIEEEAddr                                    [8 bytes] = FF FF FF FF FF FF FF FF
OrgNodeCapabilities                            [1 byte ] = 0C
OrgVendorId                                    [2 bytes] = 00 05
OrgVendorString                                [7 bytes] = "VENDOR"
OrigAppCapabilities_UserStringSpecified      [1 byte ] = 00
OrigAppCapabilities_NoOfSupportedDeviceTypes [1 byte ] = 01
OrigAppCapabilities_NoOfSupportedProfiles    [1 byte ] = 01
OrgUserString                                  [0 bytes] =
OrgDevTypeList                                 [1 byte ] = 01
OrgDevTypeList[0] = 01
OrgProfileIdList                               [1 byte ] = 01
OrgProfileIdList[0] = 01
KeyExTransferCount                             [1 byte ] = 0F
DeviceId                                       [1 byte ] = 00
Checksum                                       [1 byte ] = C8
```

NOTE

If the PBP_PushButtonPairRecip.Confirm message informs the host that the ZRC pair process completed successfully then after receiving the PBP_PushButtonPairRecip.Confirm message the host application shall not initiate any service request to the BlackBox for the next 300 ms. The BlackBox needs to store sensitive information in its non volatile memory and while doing this it is not able to ‘hear’ any incoming packet on the serial interface, because the platform interrupts are disabled.

3.2.13.7 PBP_PushButtonPairOrigContinue.Indication

Description

The PBP_PushButtonPairOrigContinue.Indication message informs the application about the successful completion of the Discovery process of the push-button pairing originator procedure and asks for its permission to continue with the pair process.

Payload

```
Status                                          - 1 byte
RecipChannel                                    - 1 byte
RecipPanId                                      - 2 bytes
RecipMacAddress                                 - 8 bytes
RecipCapabilities                               - 1 byte
RecipVendorId                                   - 2 bytes
RecipVendorString                               - 7 bytes
RecipAppCapabilities_UserStringSpecified      - 1 byte
RecipAppCapabilities_NoOfSupportedDeviceTypes - 1 byte
RecipAppCapabilities_NoOfSupportedProfiles    - 1 byte
RecipUserString                                 - 15 bytes if
RecipAppCapabilities_UserStringSpecified is TRUE
RecipDevTypeList                          - RecipAppCapabilities_NoOfSupportedDeviceTypes
bytes
RecipProfileIdList                           - RecipAppCapabilities_NoOfSupportedProfiles
bytes
requestLQI                                      - 1 byte
```

Example

```
PBP_PushButtonPairOrigContinue.Indication 02 D7 04 1C 00 00 0F BD 51 FF FF FF FF FF FF
FF FF 0F 05 00 56 45 4E 44 4F 52 00 00 01 01 01 01 D7 F5
StartOfFrame                                    [1 byte ] = 02
Header                                          [2 bytes] = D7 04
PayloadLength                                   [2 bytes] = 00 1C
Status                                          [1 byte ] = 00 (gNWSuccess_c)
RecipChannel                                    [1 byte ] = 0F
RecipPanId                                      [2 bytes] = 51 BD
RecipMacAddress                                 [8 bytes] = FF FF FF FF FF FF FF FF
RecipCapabilities                               [1 byte ] = 0F
RecipVendorId                                   [2 bytes] = 00 05
RecipVendorString                               [7 bytes] = "VENDOR"
RecipAppCapabilities_UserStringSpecified      [1 byte ] = 00
RecipAppCapabilities_NoOfSupportedDeviceTypes [1 byte ] = 01
RecipAppCapabilities_NoOfSupportedProfiles    [1 byte ] = 01
RecipUserString                                 [0 bytes] =
RecipDevTypeList                                [1 byte ] = 01
RecipDevTypeList[0] = 01
RecipProfileIdList                              [1 byte ] = 01
RecipProfileIdList[0] = 01
requestLQI                                      [1 byte ] = D7
Checksum                                        [1 byte ] = F5
```

3.2.13.8 PBP_PushButtonPairRecipContinue.Indication

Description

The PBP_PushButtonPairRecipContinue.Indication message informs the application about the reception of a successfully pair indication message in the push-button pairing target-side procedure and asks for application's permission to continue the procedure by responding to pair.

Payload

```
Status                                      - 1 byte
SrcPANId                                    - 2 bytes
SrcIEEEAddr                                 - 8 bytes
OrgNodeCapabilities                         - 1 byte
OrgVendorId                                 - 2 bytes
OrgVendorString                             - 7 bytes
OrigAppCapabilities_UserStringSpecified     - 1 byte
OrigAppCapabilities_NoOfSupportedDeviceTypes - 1 byte
OrigAppCapabilities_NoOfSupportedProfiles   - 1 byte
OrgUserString                               - 15 bytes if
OrigAppCapabilities_UserStringSpecified is et TRUE
OrgDevTypeList                            - OrigAppCapabilities_NoOfSupportedDeviceTypes
bytes
OrgProfileIdList                        - OrigAppCapabilities_NoOfSupportedProfiles bytes
KeyExTransferCount                          - 1 byte
DeviceId                                    - 1 byte
```

Example

```
PBP_PushButtonPairRecipContinue.Indication 02 D7 05 1C 00 00 FF FF FF FF FF FF FF FF FF
FF 0C 05 00 56 45 4E 44 4F 52 00 00 01 01 01 01 0F 00 CC
StartOfFrame                                [1 byte ] = 02
Header                                      [2 bytes] = D7 05
PayloadLength                               [2 bytes] = 00 1C
Status                                      [1 byte ] = 00 (gNWSuccess_c)
SrcPANId                                    [2 bytes] = FF FF
SrcIEEEAddr                                 [8 bytes] = FF FF FF FF FF FF FF FF
OrgNodeCapabilities                         [1 byte ] = 0C
OrgVendorId                                 [2 bytes] = 00 05
OrgVendorString                             [7 bytes] = "VENDOR"
OrigAppCapabilities_UserStringSpecified     [1 byte ] = 00
OrigAppCapabilities_NoOfSupportedDeviceTypes [1 byte ] = 01
OrigAppCapabilities_NoOfSupportedProfiles   [1 byte ] = 01
OrgUserString                               [0 bytes] =
OrgDevTypeList                              [1 byte ] = 01
OrgDevTypeList[0] = 01
OrgProfileIdList                            [1 byte ] = 01
OrgProfileIdList[0] = 01
KeyExTransferCount                          [1 byte ] = 0F
DeviceId                                    [1 byte ] = 00
Checksum                                    [1 byte ] = CC
```

3.2.13.9 PBP_PushButtonPairOrigContinue.Confirm

Description

Confirmation for PBP_PushButtonPairOrigContinue.Response.

Payload

```
Status - 1 byte
```

Example

```
PBP_PushButtonPairOrigContinue.Confirm 02 D7 E1 01 00 00 37
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = D7 E1
PayloadLength [2 bytes] = 00 01
Status        [1 byte ] = 00
Checksum      [1 byte ] = 37
```

3.2.13.10 PBP_PushButtonPairRecipContinue.Confirm

Description

Confirmation for PBP_PushButtonPairRecipContinue.Response.

Payload

```
Status - 1 byte
```

Example

```
PBP_PushButtonPairRecipContinue.Confirm 02 D7 E2 01 00 00 34
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = D7 E2
PayloadLength [2 bytes] = 00 01
Status        [1 byte ] = 00
Checksum      [1 byte ] = 34
```

3.2.13.11 PBP_AbortProcess.Request

Description

Abort a Push Button Pair process on an originator or recipient node (it is a sync request).

Payload

None

Example

```
PBP_AbortProcess.Request 02 D6 05 00 00 D3
        Sync    [1 byte ]  = 02
        OpGroup [1 byte ]  = D6
        OpCode  [1 byte ]  = 05
```

```
        Length  [2 bytes]  = 00 00
        CRC     [1 byte ]  = D3
```

3.2.13.12 PBP_AbortProcess.Confirm

Description

Confirmation for PBP_AbortProcess.Request.

Payload

```
Status – 1 byte
```

Example

```
PBP_AbortProcess.Confirm 02 D7 E3 01 00 00 35
        Sync    [1 byte ]  = 02
        OpGroup [1 byte ]  = D7
        OpCode  [1 byte ]  = E3
        Length  [2 bytes]  = 00 01
        Status  [1 byte ]  = 00
        CRC     [1 byte ]  = 35
```

3.2.14 BeeStack Consumer ZRC Profile Abort Service

3.2.14.1 ZRCProfile_AbortProcess.Request

Description

Abort a Push Button Pair or a Send Command process on an originator or recipient node (it is a sync request).

Payload

None

Example

```
ZRCProfile_AbortProcess.Request 02 E0 00 00 00 E0
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = E0 00
PayloadLength [2 bytes] = 00 00
Checksum      [1 byte ] = E0
```

3.2.14.2 ZRCProfile_AbortProcess.Confirm

Description

Confirmation for ZRCProfile_AbortProcess.Request.

Payload

```
Status – 1 byte
```

Example

```
ZRCProfile_AbortProcess.Confirm 02 E1 00 01 00 80 60
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = E1 00
PayloadLength [2 bytes] = 00 01
Status        [1 byte ] = 80
Checksum      [1 byte ] = 60
```

3.2.15 BeeStack Consumer ZRC Profile Command Services

3.2.15.1 ZRCProfile_Command.Request

Description

Sends a ZRC command to a device from its pairing table.

Payload

```
PairingRef    - 1 byte
CommandCode   - 1 byte
Command       - 1 byte
VendorId      - 2 bytes
TxOptions     - 1 byte
DataLength    - 1 byte
Data          - DataLength bytes
```

Example

```
ZRCProfile_Command.Request 02 DD 00 0C 00 00 01 01 05 00 04 05 00 00 00 00 00 D5
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = DD 00
PayloadLength [2 bytes] = 00 0C
PairingRef    [1 byte ] = 00
CommandCode   [1 byte ] = 01
Command       [1 byte ] = 01
VendorId      [2 bytes] = 00 05
TxOptions     [1 byte ] = 04
DataLength    [1 byte ] = 05
Data          [5 bytes] = 00 00 00 00 00
         Data[0] = 00
         Data[1] = 00
         Data[2] = 00
         Data[3] = 00
         Data[4] = 00
Checksum      [1 byte ] = D5
```

NOTE

If the data is vendor specific (i.e. the vendor specific data bit in the TxOptions field is set), the commandCode and Command fields should be ignored and the data payload should contains the vendor specific data payload.

If the data is not vendor specific (i.e. the vendor specific data bit in the TxOptions field is zero), the CommandCode byte should specify only the command code assigned for the user control pressed, release and discovery request commands.

If the CommandCode field specifies that the request is an user control pressed command (gZRC_CmdCode_UserCtrlPressed_c), the rest of the fields should describe the HDMI-CEC command.

If the CommandCode field specifies that the request is a user control released command (gZRC_CmdCode_UserCtrlReleased_c), the Command field should specifies what command should be released and the DataLength field should be set zero (no data payload).

If the CommandCode field specifies that the request is a discovery request command (gZRC_CmdCode_DiscoveryRequest_c), the Command fields is ignored and and the DataLength field should be set zero (no data payload).

If the CommandCode field specifies that the request is an user control pressed command with repetitions (gZRC_CmdCode_UserCtrlPressedAndRepeat_c), the node starts repeating the command (using the user control repeat command frame) until the user control released command is performed.

3.2.15.2 ZRCProfile_Command.Confirm

Description

Confirmation for ZRCProfile_Command.Request.Only the status field is received when it is not success. If the status is success, all the message fields are received.

Payload

```
Status – 1 byte
PairingRef – 1 byte
```

Example

```
RF4CE_NLDE_Data.Confirm 02 D3 00 03 00 00 00 01 D1
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = D3 00
PayloadLength [2 bytes] = 00 03
Status        [1 byte ] = 00 (gNWSuccess_c)
PairingRef    [1 byte ] = 00
Checksum      [1 byte ] = D1
```

3.2.15.3 ZRCProfile_DiscoveryCmd.Confirm

Description

Confirmation for ZRCProfile_Command.Request when a discovery command was sent.Only the status field is received when it is not success. If the status is success, all the message fields are received.

Payload

```
Status – 1 byte
DeviceId – 1 byte
CommandsSupportedBitMap - 32 bytes
```

Example

```
ZRCProfile_Command.Request DD 00 08 00 00 04 01 05 00 04 01 00
StartOfFrame  [1 byte ] = 02
Header      [2 bytes] = DD 00
PairingRef  [1 byte ] = 00
CommandCode [1 byte ] = 04
Command     [1 byte ] = 01
VendorId    [2 bytes] = 00 05
TxOptions   [1 byte ] = 04
DataLength  [1 byte ] = 01
Data        [1 byte ] = 00
         Data[0] = 00
Checksum      [1 byte ] = D0


ZRCProfile_DiscoveryCmd.Confirm DE 06 22 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

```
StartOfFrame  [1 byte ] = 02
Header                  [2 bytes] = DE 06
PayloadLength           [2 bytes] = 00 22
Status                  [1 byte ] = 00 (gNWSuccess_c)
DeviceId                [1 byte ] = 00
CommandsSupportedBitMap [32 bytes] = 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
Checksum      [1 byte ] = FA
```

3.2.15.4 ZRCProfile_Command.Indication

Description

Signal the application that a ZRC command frame was received over-the-air.

Payload

```
DeviceId      - 1 byte
DataLength    - 1 byte
VendorId      - 2 bytes
RxLinkQuality - 1 byte
RxFlags       - 1 byte
CommandCode   - 1 byte
Command       - 1 byte
Data          - DataLength bytes
```

Example

```
ZRCProfile_Command.Indication 02 DE 02 0D 00 00 05 C1 D2 BA 00 01 01 00 00 00 00 00 7D
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = DE 02
PayloadLength [2 bytes] = 00 0D
DeviceId      [1 byte ] = 00
DataLength    [1 byte ] = 05
VendorId      [2 bytes] = D2 C1
RxLinkQuality [1 byte ] = BA
RxFlags       [1 byte ] = 00
CommandCode   [1 byte ] = 01
Command       [1 byte ] = 01
Data          [5 bytes] = 00 00 00 00 00
        Data[0] = 00
        Data[1] = 00
        Data[2] = 00
        Data[3] = 00
        Data[4] = 00
Checksum      [1 byte ] = 7D
```

3.2.16 BeeStack Consumer ZRC Profile Set/Get Attribute Services

3.2.16.1 ZRCProfile_GetAttr.Request

Description

This request will get the ZRC local attribute specified by the attribute identifier.

Payload

```
attrId - 1 byte
```

Example

```
ZRCProfile_GetAttr.Request E0 01 01 00 82
StartOfFrame  [1 byte ] = 02
Header [2 bytes] = E0 01
attrId [1 byte ] = 82 (gKeyExTransferCountAttrId_c)
Checksum      [1 byte ] = 62
```

3.2.16.2 ZRCProfile_GetAttr.Confirm

Description

.Confirmation for ZRCProfile_GetAttr.Request.

Payload

```
Status - 1 byte
AttributeSize - 1 byte
AttributeValue - AttributeSize bytes
```

Example

```
ZRCProfile_GetAttr.Confirm E1 01 03 00 00 01 03
StartOfFrame  [1 byte ] = 02
Header         [2 bytes] = E1 01
PayloadLength  [2 bytes] = 00 03
Status         [1 byte ] = 00 (gNWSuccess_c)
AttributeSize  [1 byte ] = 01
AttributeValue [1 byte ] = 03
         AttributeValue[0] = 03
Checksum      [1 byte ] = E1
```

3.2.16.3 ZRCProfile_SetAttr.Request

Description

This request will set the ZRC local attribute specified by the attribute identifier.

Payload

```
attrId - 1 byte
```

```
AttrValue - attribute value
```

Example

```
ZRCProfile_SetAttr.Request E0 02 02 00 82 03
StartOfFrame  [1 byte ] = 02
Header    [2 bytes] = E0 02
attrID    [1 byte ] = 82 (gKeyExTransferCountAttrId_c)
AttrValue [1 byte ] = 03
Checksum      [1 byte ] = 61
```

3.2.16.4 ZRCProfile_SetAttr.Confirm

Description

.Confirmation for ZRCProfile_SetAttr.Request.

Payload

```
Status - 1 byte
```

Example

```
ZRCProfile_SetAttr.Confirm E1 02 01 00 00
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = E1 02
PayloadLength [2 bytes] = 00 01
Status        [1 byte ] = 00 (gNWSuccess_c)
Checksum      [1 byte ] = E2
```

3.2.16.5 ZRCProfile_SetZRCSupportedCmds.Request

Description

This request will set the bitmap of the ZRC supported commands .

Payload

```
ZRCCmdSupportedBitMap - the bitmap with the supported ZRC commands (32 bytes).
```

Example

```
ZRCProfile_SetZRCSupportedCmds.Request 02 E0 03 20 00 FF FF FF 0F 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 33
         Sync                  [1 byte ]  = 02
         OpGroup               [1 byte ]  = E0
         OpCode                [1 byte ]  = 03
         Length                [2 bytes]  = 20 00
         ZRCCmdSupportedBitMap [32 bytes]  = FF FF FF 0F 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
         CRC                   [1 byte ]  = 33
```

3.2.16.6 ZRCProfile_SetZRCSupportedCmds.Confirm

Description

.Confirmation for ZRCProfile_SetZRCSupportedCmds.Request.

Payload

```
Status - 1 byte
```

Example

```
ZRCProfile_SetZRCSupportedCmds.Confirm 02 E1 03 01 00 00 E3
         Sync    [1 byte ]  = 02
         OpGroup [1 byte ]  = E1
         OpCode  [1 byte ]  = 03
         Length  [2 bytes]  = 01 00
         Status  [1 byte ]  = 00 (gNWSuccess_c)
         CRC     [1 byte ]  = E3
```

3.2.16.7 ZRCProfile_GetZRCSupportedCmds.Request

Description

This request will get the ZRC supported commands bitmap.

Example

```
ZRCProfile_GetZRCSupportedCmds.Request 02 E0 04 00 00 E4
         Sync    [1 byte ]  = 02
         OpGroup [1 byte ]  = E0
         OpCode  [1 byte ]  = 04
         Length  [2 bytes]  = 00 00
         CRC     [1 byte ]  = E4
```

3.2.16.8 ZRCProfile_GetZRCSupportedCmds.Confirm

Description

.Confirmation for ZRCProfile_GetZRCSupportedCmds.Request.

Payload

```
ZRCCmdSupportedBitMap - the bitmap with the supported ZRC commands (32 bytes).
```

Example

```
ZRCProfile_GetZRCSupportedCmds.Confirm 02 E1 04 20 00 FF FF FF 0F 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 35
         Sync                  [1 byte ]  = 02
         OpGroup               [1 byte ]  = E1
         OpCode                [1 byte ]  = 04
         Length                [2 bytes]  = 20 00
```

```
         ZRCCmdSupportedBitMap [32 bytes]  = FF FF FF 0F 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
         CRC                   [1 byte ]  = 35
```

3.2.17 BeeStack Consumer Set MAC Address Service

3.2.17.1 NWK SetMacAddress.Request

Description

Set the MAC address of the BeeStack Consumer node. The MAC address is not written to flash.

Payload

```
IsMacAddressNull – 1 byte
MacAddress – 8 bytes(IEEE Address)
```

Example

```
RF4CE_NWK_SetMacAddress.Request 02 D4 00 09 00 00 DD DD DD DD DD DD DD DD DD
StartOfFrame     [1 byte ] = 02
Header           [2 bytes] = D4 00
PayloadLength    [2 bytes] = 00 09
IsMacAddressNull [1 byte ] = 00 (false)
MacAddress       [8 bytes] = DD DD DD DD DD DD DD DD
Checksum         [1 byte ] = DD
```

3.2.17.2 NWK SetMacAddress.Confirm

Description

Confirmation for SetMacAddress.Request.

Payload

```
Status – 1 byte
```

Example

```
RF4CE_NWK_SetMacAddress.Confirm 02 D5 00 01 00 00 D4
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = D5 00
PayloadLength [2 bytes] = 00 01
Status        [1 byte ] = 00 (gNWSuccess_c)
Checksum      [1 byte ] = D4
```

3.2.18 BeeStack Consumer Get MAC Address Service

3.2.18.1 NWK GetMacAddress.Request

Description

Get the MAC address of the BeeStack Consumer node from RAM (may be different from the MAC address in flash if a NWK_SetMacAddress.Request has been previously issued).

Payload

```
No payload.
```

Example

```
RF4CE_NWK_GetMacAddress.Request 02 D4 01 00 00 D5
StartOfFrame [1 byte ] = 02
Header       [2 bytes] = D4 01
PayloadLength[2 bytes] = 00 00
Checksum     [1 byte ] = D5
```

3.2.18.2 NWK GetMacAddress.Confirm

Description

Confirmation for GetMacAddress.Request.

Payload

```
Status – 1 byte
MacAddress – array of 8 bytes(IEEE Address)
```

Example

```
RF4CE_NWK_GetMacAddress.Confirm 02 D5 01 09 00 00 DD DD DD DD DD DD DD DD DD
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = D5 01
PayloadLength [2 bytes] = 00 09
Status        [1 byte ] = 00 (gNWSuccess_c)
MacAddress    [8 bytes] = DD DD DD DD DD DD DD DD
Checksum      [1 byte ] = DD
```

3.2.19 BeeStack Consumer Set Maximum Pairing Table Entries Service

3.2.19.1 NWK SetMaxPairingTableEntries.Request

Description

Set the Maximum Pairing Table Entries.

Payload

nwkcMaxPairingTableEntries – 1 byte.

Example

```
RF4CE_NWK_SetMaxPairingTableEntries.Request 02 D4 02 01 00 05 D2
StartOfFrame               [1 byte ] = 02
Header                     [2 bytes] = D4 02
PayloadLength              [2 bytes] = 00 01
nwkcMaxPairingTableEntries [1 byte ] = 05
Checksum                   [1 byte ] = D2
```

3.2.19.2 NWK SetMaxPairingTableEntries.Confirm

Description

Confirmation for SetMaxPairingTableEntries.Request.

Payload

Status – 1 byte

Example

```
RF4CE_NWK_SetMaxPairingTableEntries.Confirm 02 D5 02 01 00 00 D6
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = D5 02
PayloadLength [2 bytes] = 00 01
Status        [1 byte ] = 00 (gNWSuccess_c)
Checksum      [1 byte ] = D6
```

3.2.20 BeeStack Consumer Get Maximum Pairing Table Entries Service

3.2.20.1 NWK GetMaxPairingTableEntries.Request

Description

Get the Maximum Pairing Table Entries.

Payload

No payload.

Example

```
RF4CE_NWK_GetMaxPairingTableEntries.Request 02 D4 03 00 00 D7
StartOfFrame [1 byte ] = 02
Header       [2 bytes] = D4 03
PayloadLength[2 bytes] = 00 00
Checksum     [1 byte ] = D7
```

3.2.20.2 NWK GetMaxPairingTableEntries.Confirm

Description

Confirmation for GetMaxPairingTableEntries.Request.

Payload

```
nwkcMaxPairingTableEntries – 1 byte
```

Example

```
RF4CE_NWK_GetMaxPairingTableEntries.Confirm 02 D5 03 01 00 05 D2
StartOfFrame               [1 byte ] = 02
Header                     [2 bytes] = D5 03
PayloadLength              [2 bytes] = 00 01
nwkcMaxPairingTableEntries [1 byte ] = 05
Checksum                   [1 byte ] = D2
```

3.2.21 BeeStack Consumer Set Node Capabilities Service

3.2.21.1 NWK SetNodeCapabilities.Request

Description

Set the node capabilities.

Payload

```
nwkcNodeCapabilities – 1 byte
```

Example

```
RF4CE_NWK_SetNodeCapabilities.Request 02 D4 04 01 00 00 D1
StartOfFrame         [1 byte ] = 02
Header               [2 bytes] = D4 04
PayloadLength        [2 bytes] = 00 01
nwkcNodeCapabilities [1 byte ] = 00
Checksum             [1 byte ] = D1
```

3.2.21.2 NWK SetNodeCapabilities.Confirm

Description

Confirmation for SetNodeCapabilities.Request.

Payload

```
Status – 1 byte
```

Example

```
RF4CE_NWK_SetNodeCapabilities.Confirm 02 D5 04 01 00 00 D0
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = D5 04
PayloadLength [2 bytes] = 00 01
Status        [1 byte ] = 00 (gNWSuccess_c)
Checksum      [1 byte ] = D0
```

3.2.22 BeeStack Consumer Get Node Capabilities Service

3.2.22.1 NWK GetNodeCapabilities.Request

Description

Get the node capabilities.

Payload

```
No payload.
```

Example

```
RF4CE_NWK_GetNodeCapabilities.Request 02 D4 05 00 00 D1
StartOfFrame [1 byte ] = 02
Header       [2 bytes] = D4 05
PayloadLength[2 bytes] = 00 00
Checksum     [1 byte ] = D1
```

3.2.22.2 NWK GetNodeCapabilies.Confirm

Description

Confirmation for GetNodeCapabilities.Request.

Payload

```
nwkcNodeCapabilities - 1 byte
```

Example

```
RF4CE_NWK_GetNodeCapabilies.Confirm 02 D5 05 01 00 00 D1
StartOfFrame         [1 byte ] = 02
Header               [2 bytes] = D5 05
PayloadLength        [2 bytes] = 00 01
nwkcNodeCapabilities [1 byte ] = 00
Checksum             [1 byte ] = D1
```

3.2.23 BeeStack Consumer Set Vendor Identifier Service

3.2.23.1 NWK SetVendorIdentifier.Request

Description

Set the Vendor Identifier.

Payload

```
nwkcVendorIdentifier – 2 byte
```

Example

```
RF4CE_NWK_SetVendorIdentifier.Request 02 D4 06 02 00 05 00 D5
StartOfFrame         [1 byte ] = 02
Header               [2 bytes] = D4 06
PayloadLength        [2 bytes] = 00 02
nwkcVendorIdentifier [2 bytes] = 00 05
Checksum             [1 byte ] = D5
```

3.2.23.2 NWK SetVendorIdentifier.Confirm

Description

Confirmation for SetVendorIdentifier.Request.

Payload

```
Status – 1 byte
```

Example

```
RF4CE_NWK_SetVendorIdentifier.Confirm 02 D5 06 01 00 00 D2
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = D5 06
PayloadLength [2 bytes] = 00 01
Status        [1 byte ] = 00 (gNWSuccess_c)
Checksum      [1 byte ] = D2
```

3.2.24 BeeStack Consumer Get Vendor Identifier Service

3.2.24.1 NWK GetVendorIdentifier.Request

Description

Get the Vendor Identifier.

Payload

No payload.

Example

```
RF4CE_NWK_GetVendorIdentifier.Request 02 D4 07 00 00 D3
StartOfFrame [1 byte ] = 02
Header       [2 bytes] = D4 07
PayloadLength[2 bytes] = 00 00
Checksum     [1 byte ] = D3
```

3.2.24.2 NWK GetVendorIdentifier.Confirm

Description

Confirmation for GetVendorIdentifier.Request.

Payload

```
nwkcVendorId – 2 byte
```

Example

```
RF4CE_NWK_GetVendorIdentifier.Confirm 02 D5 07 02 00 05 00 D5
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = D5 07
PayloadLength [2 bytes] = 00 02
nwkcVendorId  [2 bytes] = 00 05
Checksum      [1 byte ] = D5
```

3.2.25 BeeStack Consumer Set Vendor String Service

3.2.25.1 NWK SetVendorString.Request

Description

Set the node Vendor String.

Payload

```
nwkcVendorString – array of 7 bytes.
```

Example

```
RF4CE_NWK_SetVendorString.Request 02 D4 08 07 00 61 62 63 64 65 66 67 BB
StartOfFrame     [1 byte ] = 02
Header           [2 bytes] = D4 08
PayloadLength    [2 bytes] = 00 07
nwkcVendorString [7 bytes] = "abcdefg"
Checksum         [1 byte ] = 24
```

3.2.25.2 NWK SetVendorString.Confirm

Description

Confirmation for SetVendorString.Request.

Payload

```
Status – 1 byte
```

Example

```
RF4CE_NWK_SetVendorString.Confirm 02 D5 08 01 00 00 DC
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = D5 08
PayloadLength [2 bytes] = 00 01
Status        [1 byte ] = 00 (gNWSuccess_c)
Checksum      [1 byte ] = DC
```

3.2.26 BeeStack Consumer Get Vendor String Service

3.2.26.1 NWK GetVendorString.Request

Description

Get the node Vendor String.

Payload

```
No payload.
```

Example

```
RF4CE_NWK_GetVendorString.Request 02 D4 09 00 00 DD
StartOfFrame [1 byte ] = 02
Header       [2 bytes] = D4 09
PayloadLength[2 bytes] = 00 00
Checksum     [1 byte ] = DD
```

3.2.26.2 NWK GetVendorString.Confirm

Description

Confirmation for GetVendorString.Request.

Payload

```
nwkcVendorString – array of 7 bytes.
```

Example

```
RF4CE_NWK_GetVendorString.Confirm 02 D5 09 07 00 61 62 63 64 65 66 67 BB
StartOfFrame     [1 byte ] = 02
Header           [2 bytes] = D5 09
PayloadLength    [2 bytes] = 00 07
nwkcVendorString [7 bytes] = "abcdefg"
Checksum         [1 byte ] = BB
```

3.2.27 BeeStack Consumer Get Frame Counter Window Service

3.2.27.1 NWK GetFrameCounterWindow.Request

Description

Get the Node Frame Counter Window.

Payload

```
No payload.
```

Example

```
RF4CE_NWK_GetFrameCounterWindow.Request 02 D4 0B 00 00 DF
StartOfFrame [1 byte ] = 02
Header       [2 bytes] = D4 0B
PayloadLength[2 bytes] = 00 00
Checksum     [1 byte ] = DF
```

3.2.27.2 NWK GetFrameCounterWindow.Confirm

Description

Confirmation for GetFrameCounterWindow.Request.

Payload

```
nwkcFrameCounterWindow – 4 bytes.
```

Example

```
RF4CE_NWK_GetFrameCounterWindow.Confirm 02 D5 0B 04 00 00 04 00 00 DE
```

```
StartOfFrame           [1 byte ] = 02
Header                 [2 bytes] = D5 0B
PayloadLength          [2 bytes] = 00 04
nwkcFrameCounterWindow [4 bytes] = 00 00 04 00
Checksum               [1 byte ] = DE
```

3.2.28 BeeStack Consumer Set Frame Counter Window Service

3.2.28.1 NWK SetFrameCounterWindow.Request

```
Description
```

Set the Node Frame Counter Window.

Payload

```
nwkcFrameCounterWindow – 4 bytes.
```

Example

```
RF4CE_NWK_SetFrameCounterWindow.Request 02 D4 0A 04 00 00 08 00 00 D2
StartOfFrame           [1 byte ] = 02
Header                 [2 bytes] = D4 0A
PayloadLength          [2 bytes] = 00 04
nwkcFrameCounterWindow [4 bytes] = 00 00 08 00
Checksum               [1 byte ] = D2
```

3.2.28.2 NWK SetFrameCounterWindow.Confirm

Description

Confirmation for SetFrameCounterWindow.Request.

Payload

```
Status – 1 bytes.
```

Example

RF4CE_NWK_SetFrameCounterWindow.Confirm 02 D5 0A 01 00 00 DE

```
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = D5 0A
PayloadLength [2 bytes] = 00 01
Status        [1 byte ] = 00 (gNWSuccess_c)
Checksum      [1 byte ] = DE
```

3.2.29 BeeStack Consumer Add New Pair Table Entry Service

3.2.29.1 NWK AddNewPairTableEntry.Request

Description

Add a new entry in Pair Table.

Payload

```
LocalShortAddress – 2 bytes.
RecipChannel      - 1 byte.
RecipMacAddress   - 8 bytes.
RecipPanId        - 2 bytes.
RecipShortAddress – 2 bytes.
RecipCapabilities - 1 byte.
SecurityKey       - 16 bytes.
RecipUserString   - 15 bytes.
```

Example

```
RF4CE_NWK_AddNewPairTableEntry.Request 02 D4 0C 2F 00 E7 7F 0F 08 07 06 05 04 03 02 01
FE CA 35 12 04 08 07 06 05 04 03 02 01 08 07 06 05 04 03 02 01 61 61 61 61 61 61 61 61
61 61 61 61 61 61 61 1E
StartOfFrame      [1 byte ] = 02
Header            [2 bytes] = D4 0C
PayloadLength     [2 bytes] = 00 2F
LocalShortAddress [2 bytes] = 7F E7
RecipChannel      [1 byte ] = 0F
RecipMacAddress   [8 bytes] = 01 02 03 04 05 06 07 08
RecipPanId        [2 bytes] = CA FE
RecipShortAddress [2 bytes] = 12 35
RecipCapabilities [1 byte ] = 04
SecurityKey       [16 bytes] = 01 02 03 04 05 06 07 08 01 02 03 04 05 06 07 08
RecipUserString   [15 bytes] = "aaaaaaaaaaaaaaa"
Checksum          [1 byte ] = 1E
```

3.2.29.2 NWK AddNewPairTableEntry.Confirm

Description

Confirmation for AddNewPairTableEntry.Request.

Payload

```
Index – 1 bytes.
```

Example

```
RF4CE_NWK_AddNewPairTableEntry.Confirm 02 D5 0C 01 00 01 D9
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = D5 0C
PayloadLength [2 bytes] = 00 01
Index         [1 byte ] = 01
```

```
Checksum      [1 byte ] = D9
```

NOTE

If the NWK AddNewPairTableEntry.Confirm message informs the host that the adding of a new pair table entry process completed successfully then after receiving the AddNewPairTableEntry.Confirm message the host application shall not initiate any service request to the BlackBox for the next 300 ms. The BlackBox needs to store sensitive information in its non volatile memory and while doing this it is not able to 'hear' any incoming packet on the serial interface, because the platform interrupts are disabled.

3.2.30 BeeStack Consumer Save Persistent Data Service

3.2.30.1 NWK SavePersistentData.Request

Description

Save network or application persistent data in flash.

Payload

```
No payload.
```

Example

```
NWK_SavePersistentData.Request 02 D4 0D 00 00 D9
StartOfFrame [1 byte ] = 02
Header       [2 bytes] = D4 0D
PayloadLength[2 bytes] = 00 00
Checksum     [1 byte ] = D9
```

3.2.30.2 NWK SavePersistentData.Confirm

Description

Confirmation for SavePersistentData.Request.

Payload

```
Status – 1 bytes.
```

Example

```
RF4CE_NWK_SavePersistentData.Confirm 02 D5 0D 01 00 00 D9
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = D5 0D
PayloadLength [2 bytes] = 00 01
Status        [1 byte ] = 00 (gNWSuccess_c)
Checksum      [1 byte ] = D9
```

NOTE

If the NWK SavePersistentData.Confirm message informs the host that the saving of persistent data process completed successfully then after receiving the NWK SavePersistentData.Confirm message the host application shall not initiate any service request to the BlackBox for the next 300 ms. While the BlackBox stores sensitive information in its non volatile memory it is not able to 'hear' any incoming packet on the serial interface, because the platform interrupts are disabled.

3.2.31 BeeStack Consumer Generate Short Address Service

3.2.31.1 NWK GenerateShortAddress.Request

Description

Generate a random short address.

Payload

```
No payload.
```

Example

```
NWK_SavePersistentData.Request 02 D4 0D 00 00 D9
StartOfFrame [1 byte ] = 02
Header       [2 bytes] = D4 0D
PayloadLength[2 bytes] = 00 00
Checksum     [1 byte ] = D9
```

3.2.31.2 NWK GenerateShortAddress.Confirm

Description

Confirmation for GenerateShortAddress.Request.

Payload

```
ShortAddress  – 2 bytes.
```

Example

```
RF4CE_NWK_GenerateShortAddress.Confirm 02 D5 0E 02 00 C4 76 6B
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = D5 0E
PayloadLength [2 bytes] = 00 02
ShortAddress  [2 bytes] = 76 C4
Checksum      [1 byte ] = 6B
```

3.2.32 BeeStack Consumer Generate Security Key Service

3.2.32.1 NWK GenerateSecurityKey.Request

Description

Generate a security key.

Payload

```
No payload.
```

Example

```
RF4CE_NWK_GenerateSecurityKey.Request 02 D4 0F 00 00 DB
StartOfFrame [1 byte ] = 02
Header       [2 bytes] = D4 0F
PayloadLength[2 bytes] = 00 00
Checksum     [1 byte ] = DB
```

3.2.32.2 NWK GenerateSecurityKey.Confirm

Description

Confirmation for GenerateSecurityKey.Request.

Payload

```
SecurityKey   - array of 16 bytes.
```

Example

```
RF4CE_NWK_GenerateSecurityKey.Confirm 02 D5 0F 10 00 A5 9E E1 73 C1 C2 2E 21 E3 44 31 35
8E 48 19 80 93
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = D5 0F
PayloadLength [2 bytes] = 00 10
SecurityKey   [16 bytes] = 80 19 48 8E 35 31 44 E3 21 2E C2 C1 73 E1 9E A5
Checksum      [1 byte ] = 93
```

3.2.33 BeeStack Consumer Save Frame Counter Service

3.2.33.1 NWK SaveFrameCounter.Request

Description

Store Frame Counter in flash.

Payload

```
No payload.
```

Example

```
RF4CE_NWK_SaveFrameCounter.Request 02 D4 10 00 00 C4
StartOfFrame [1 byte ] = 02
Header       [2 bytes] = D4 10
PayloadLength[2 bytes] = 00 00
Checksum     [1 byte ] = C4
```

3.2.33.2 NWK SaveFrameCounter.Confirm

Description

Confirmation for SaveFrameCounter.Request.

Payload

```
Status – 1 bytes.
```

Example

```
RF4CE_NWK_SaveFrameCounter.Confirm 02 D5 10 01 00 00 C4
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = D5 10
PayloadLength [2 bytes] = 00 01
Status        [1 byte ] = 00 (gNWSuccess_c)
Checksum      [1 byte ] = C4
```

NOTE

If the NWK SaveFrameCounter.Confirm message informs the host that the saving of the frame counter in the non-volatile memory process completed successfully then after receiving the NWK SaveFrameCounter.Confirm message the host application shall not initiate any service request to the BlackBox for the next 300 ms. The BlackBox is not able to ‘hear’ any incoming packet on the serial interface while writing in the non-volatile memory.

3.2.34 BeeStack Consumer Get Last Packet LQI Service

3.2.34.1 NWK GetLastPacketLQI.Request

Description

Get the LQI of the last packet.

Payload

```
No payload.
```

Example

```
RF4CE_NWK_GetLastPacketLQI.Request 02 D4 11 00 00 C5
StartOfFrame [1 byte ] = 02
```

```
Header       [2 bytes] = D4 11
PayloadLength[2 bytes] = 00 00
Checksum     [1 byte ] = C5
```

3.2.34.2 NWK GetLastPacketLQI.Confirm

Description

Confirmation for GetLastPacketLQI.Request.

Payload

```
LQI – 1 bytes.
```

Example

```
RF4CE_NWK_GetLastPacketLQI.Confirm 02 D5 11 01 00 B2 77
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = D5 11
PayloadLength [2 bytes] = 00 01
LQI           [1 byte ] = B2
Checksum      [1 byte ] = 77
```

3.2.35 BeeStack Consumer Get Node Short Address Service

3.2.35.1 NWK GetNodeShortAddress.Request

Description

Get the node short address.

Payload

```
No payload.
```

Example

```
RF4CE_NWK_GetNodeShortAddress.Request 02 D4 13 00 00 C7
StartOfFrame [1 byte ] = 02
Header       [2 bytes] = D4 13
PayloadLength[2 bytes] = 00 00
Checksum     [1 byte ] = C7
```

3.2.35.2 NWK GetNodeShortAddress.Confirm

Description

Confirmation for GetNodeShortAddress.Request.

Payload

```
Status – 1 bytes.
```

```
ShortAddress – 2 bytes.
```

Example

```
RF4CE_NWK_GetNodeShortAddress.Confirm 02 D5 13 03 00 00 00 00 C5
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = D5 13
PayloadLength [2 bytes] = 00 03
Status        [1 byte ] = 00 (gNWSuccess_c)
ShortAddress  [2 bytes] = 00 00
Checksum      [1 byte ] = C5
```

3.2.36 BeeStack Consumer GetAllowedLowPowerInterval Service

3.2.36.1 NWK GetAllowedLowPowerInterval.Request

Description

Returns the amount of time the network is available for entering low power mode, starting with the moment this service call is made.

Payload

```
No payload.
```

Example

```
RF4CE_NWK_GetAllowedLowPowerInterval.Request 02 D4 15 00 00 BF
StartOfFrame [1 byte ] = 02
Header       [2 bytes] = D4 15
PayloadLength[2 bytes] = 00 00
Checksum     [1 byte ] = BF
```

3.2.36.2 NWK RF4CE_GetAllowedLowPowerInterval.Confirm

Description

Confirmation for RF4CE_GetAllowedLowPowerInterval.Request.

Payload

```
interval – 4 bytes.
```

Example

```
RF4CE_NWK_GetAllowedLowPowerInterval.Confirm 02 D5 15 04 00 00 00 00 00 BE
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = D5 15
PayloadLength [2 bytes] = 00 04
Status        [1 byte ] = 00 00 00 00
Checksum      [1 byte ] = BE
```

3.2.37 BeeStack Consumer Is Network In Idle State Service

3.2.37.1 NWK IsIdle.Request

Description

Check if the network is in idle state.

Payload

```
No payload.
```

Example

```
RF4CE_NWK_IsIdle.Request 02 D4 14 00 00 C0
StartOfFrame [1 byte ] = 02
Header       [2 bytes] = D4 14
PayloadLength[2 bytes] = 00 00
Checksum     [1 byte ] = C0
```

3.2.37.2 NWK IsIdle.Confirm

Description

Confirmation for IsIdle.Request.

Payload

```
Status – 1 bytes.
```

Example

```
RF4CE_NWK_IsIdle.Confirm 02 D5 14 01 00 01 C1
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = D5 14
PayloadLength [2 bytes] = 00 01
Status        [1 byte ] = 01 (Idle)
Checksum      [1 byte ] = C1
```

3.2.38 Freescale Profile Fragmentation Service

3.2.38.1 FSLProfile_FragTx.Request

Description

Sends over-the-air a big buffer using fragmentation process. The big buffer is limited and depends of the RAM usage.

Payload

```
UseSecurityFlag - 1 byte
DeviceId        - 1 byte
DataLenght      - 2 bytes
Data            - DataLenght bytes
```

Example

```
FSLProfile_FragTx.Request 02 DA 00 68 00 01 00 64 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 D7
StartOfFrame    [1 byte ] = 02
Header          [2 bytes] = DA 00
PayloadLength   [2 bytes] = 00 68
UseSecurityFlag [1 byte ] = 01 (true)
DeviceId        [1 byte ] = 00
DataLenght      [2 bytes] = 00 64
Data            [100 bytes] = 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
         Data[0] = 00
         Data[1] = 00
         Data[2] = 00
         Data[3] = 00
         Data[4] = 00
         Data[5] = 00
         Data[6] = 00
         Data[7] = 00
         Data[8] = 00
         Data[9] = 00
.....................................................................................
Checksum        [1 byte ] = D7
```

3.2.38.2 FSLProfile_Frag.Confirm

Description

Confirmation for FSLProfile_FragTx.Request.

Payload

```
Status               - 1 byte
FragRxMaxAcceptedLen - 2 bytes
```

Example

```
FSLProfile_Frag.Confirm 02 DB 00 03 00 00 00 00 D8
StartOfFrame         [1 byte ] = 02
Header               [2 bytes] = DB 00
PayloadLength        [2 bytes] = 00 03
Status               [1 byte ] = 00 (gNWSuccess_c)
FragRxMaxAcceptedLen [2 bytes] = 00 00
Checksum             [1 byte ] = D8
```

3.2.38.3 FSLProfile_StartFrag.Indication

Description

Informs the application layer about the start of a fragmented reception initiated by one of the nodes in its Pair Table.

Payload

```
DeviceId      - 1 byte
fragDataLen   - 2 bytes
```

Example

```
FSLProfile_StartFrag.Indication 02 DB 01 03 00 00 64 00 BD
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = DB 01
PayloadLength [2 bytes] = 00 03
DeviceId      [1 byte ] = 00
fragDataLen   [2 bytes] = 00 64
Checksum      [1 byte ] = BD
```

3.2.38.4 FSLProfile_Frag.Indication

Description

Signal the Application when all fragments are received (the big buffer is entirely received).

Payload

```
Status          - 1 byte
DeviceId        - 1 byte
```

```
FragSecuredFlag - 1 byte
fragDataLen     - 2 bytes
FragTxRxBuffer  - fragDataLen bytes
```

Example

```
FSLProfile_Frag.Indication 02 DB 02 69 00 00 00 01 64 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 D5
StartOfFrame    [1 byte ] = 02
Header          [2 bytes] = DB 02
PayloadLength   [2 bytes] = 00 69
Status          [1 byte ] = 00
DeviceId        [1 byte ] = 00
FragSecuredFlag [1 byte ] = 01 (true)
fragDataLen     [2 bytes] = 00 64
FragTxRxBuffer  [100 bytes] = 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
         FragTxRxBuffer[0] = 00
         FragTxRxBuffer[1] = 00
         FragTxRxBuffer[2] = 00
         FragTxRxBuffer[3] = 00
         FragTxRxBuffer[4] = 00
.....................................................................................
Checksum        [1 byte ] = D5
```

3.2.38.5 FSLProfile_SetFragTxRxBufferState.Request

Description

Set the Fragmentation RxTx buffer state. The RxTx buffer can be used by the FSL Profile (when receiving or transmitting bytes), used by the Application, or “free to use”. The Application Layer must set the RxTx buffer state “free” to receive the fragments in the buffer.

Payload

```
newBufferState - 1 byte)
```

Example

```
FSLProfile_SetFragTxRxBufferState.Request 02 DA 01 01 00 00 DA
StartOfFrame   [1 byte ] = 02
Header         [2 bytes] = DA 01
PayloadLength  [2 bytes] = 00 01
newBufferState [1 byte ] = 00 (gFragTxRxBufferFree_c)
Checksum       [1 byte ] = DA
```

3.2.38.6 FSLProfile_SetFragTxRxBufferState.Confirm

Description

Confirmation for FSLProfile_SetFragTxRxBufferState.Request.

Payload

```
Status – 1 byte
```

Example

```
FSLProfile_SetFragTxRxBufferState.Confirm 02 DB E0 01 00 00 3A
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = DB E0
PayloadLength [2 bytes] = 00 01
Status        [1 byte ] = 00 (gNWSuccess_c)
Checksum      [1 byte ] = 3A
```

3.2.38.7 FSLProfile_GetFragTxRxBufferState.Request

Description

Get the Fragmentation RxTx buffer state. The RxTx buffer can be used by FSL Profile (when receives or transmits bytes), used by the Application, or “free to use”.

Payload

```
None
```

Example

```
FSLProfile_GetFragTxRxBufferState.Request 02 DA 02 00 00 D8
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = DA 02
PayloadLength [2 bytes] = 00 00
Checksum      [1 byte ] = D8
```

3.2.38.8 FSLProfile_GetFragTxRxBufferState.Confirm

Description

Confirmation for FSLProfile_GetFragTxRxBufferState.Request.

Payload

```
RxTxBufferState - 1 byte
```

Example

```
FSLProfile_GetFragTxRxBufferState.Confirm 02 DB E1 01 00 00 3B
StartOfFrame    [1 byte ] = 02
Header          [2 bytes] = DB E1
```

```
PayloadLength   [2 bytes] = 00 01
RxTxBufferState [1 byte ] = 00 (gFragTxRxBufferFree_c)
Checksum        [1 byte ] = 3B
```

3.2.39 Freescale Profile Poll Data Service

3.2.39.1 FSLProfile_PollConfig.Request

Description

Configure the poll process parameters (PollInterval and RxOnInterval).

Payload

```
PollInterval - 4 bytes
RxOnInterval - 2 bytes
```

Example

```
FSLProfile_PollConfig.Request 02 DA 03 06 00 B8 0B 00 00 00 00 6C
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = DA 03
PayloadLength [2 bytes] = 00 06
PollInterval  [4 bytes] = 00 00 0B B8
RxOnInterval  [2 bytes] = 00 00
Checksum      [1 byte ] = 6C
```

3.2.39.2 FSLProfile_PollConfig.Confirm

Description

Confirmation for FSLProfile_PollConfig.Request.

Payload

```
Status – 1 byte
```

Example

```
FSLProfile_PollConfig.Confirm 02 DB E2 01 00 00 38
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = DB E2
PayloadLength [2 bytes] = 00 01
Status        [1 byte ] = 00 (gNWSuccess_c)
Checksum      [1 byte ] = 38
```

3.2.39.3 FSLProfile_Poll.Request

Description

FSLProfile_PollRequest starts or stops the polling process for a specified device Id. This request cannot be called without previously having configured the poll parameters.

Payload

```
DeviceId - 1 byte
PollEnable - 1 byte
```

Example

```
FSLProfile_Poll.Request 02 DA 04 02 00 00 01 DD
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = DA 04
PayloadLength [2 bytes] = 00 02
DeviceId      [1 byte ] = 00
PollEnable    [1 byte ] = 01 (true)
Checksum      [1 byte ] = DD
```

3.2.39.4 FSLProfile_Poll.Confirm

Description

Confirmation for FSLProfile_Poll.Request.

Payload

```
Status – 1 byte
```

Example

```
FSLProfile_Poll.Confirm 02 DB 03 01 00 00 D9
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = DB 03
PayloadLength [2 bytes] = 00 01
Status        [1 byte ] = 00
Checksum      [1 byte ] = D9
```

3.2.39.5 FSLProfile_Poll.Indication

Description

Signal the Application when an over-the-air poll frame was received and the Application has data available (the Application has to tell that it has data to send or not using FSLProfile_PollDataAvailable.Request). If Data is not available, no Poll.Indication message is received on the recipient.

Payload

```
DeviceId -1 byte
rxOnInterval -2 bytes
```

Example

```
FSLProfile_Poll.Indication 02 DB 05 03 00 00 00 00 DD
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = DB 05
PayloadLength [2 bytes] = 00 03
DeviceId      [1 byte ] = 00
```

```
rxOnInterval  [2 bytes] = 00 00
Checksum      [1 byte ] = DD
```

3.2.39.6 FSLProfile_PollEvent

Description

This message is received on the originator (the node that sends a FSLProfile_Poll.Request) when the RxOnInterval is 0x0000.

When a poll frame is sent over-the-air, the originator waits a Poll response frame from the recipient. Having RxOnInterval parameter set to zeros, it will consider that the received response frame is the expected DATA, and will forward it to the Application using a FSLProfile_PollEvent message.

Payload

```
DeviceId - 1 byte
DataAvailable - 1 byte
```

Example

```
FSLProfile_PollEvent 02 DB 04 02 00 00 00 DD
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = DB 04
PayloadLength [2 bytes] = 00 02
DeviceId      [1 byte ] = 00
DataAvailable [1 byte ] = 00 (false)
Checksum      [1 byte ] = DD
```

3.2.39.7 FSLProfile_PollDataAvailable.Request

Description

This request signals the FSL Profile Layer that the Application has data to send or not. If the Application has data to send than FSLProfile_Poll.Indication messages will arrive when over-the-air poll frames occur.

Payload

```
DeviceId - 1 byte
DataAvailable -1 byte
```

Example

```
FSLProfile_PollDataAvailable.Request 02 DA 05 02 00 00 01 DC
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = DA 05
PayloadLength [2 bytes] = 00 02
DeviceId      [1 byte ] = 00
DataAvailable [1 byte ] = 01 (true)
Checksum      [1 byte ] = DC
Checksum         [1 byte ] = DD
```

3.2.39.8 FSLProfile_PollDataAvailable.Confirm

Description

Confirmation for FSLProfile_PollDataAvailable.Request.

Payload

```
Status – 1 byte
```

Example

```
FSLProfile_PollDataAvailable.Confirm 02 DB E3 01 00 00 39
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = DB E3
PayloadLength [2 bytes] = 00 01
Status        [1 byte ] = 00 (gNWSuccess_c)
Checksum      [1 byte ] = 39
```

3.2.40 Freescale Profile Remote Pair Service

3.2.40.1 FSLProfile_RmtPair.Request

Description

Establish a communication link between two devices from Remote Pair Originator's Pair Table.

Payload

```
AppRecipRspTimeOut                          - 2 bytes
DeviceId1                                   - 1 byte
Dev1AppCapabilities_UserStringSpecified     - 1 byte
Dev1AppCapabilities_NoOfSupportedDeviceTypes - 1 byte
Dev1AppCapabilities_NoOfSupportedProfiles   - 1 byte
Dev1TypeList - Dev1AppCapabilities_NoOfSupportedDeviceTypes bytes
Dev1ProfileIdList - Dev1AppCapabilities_NoOfSupportedProfiles bytes
DeviceId2                                   - 1 byte
Dev2AppCapabilities_UserStringSpecified     - 1 byte
Dev2AppCapabilities_NoOfSupportedDeviceTypes - 1 byte
Dev2AppCapabilities_NoOfSupportedProfiles   - 1 byte
Dev2TypeList - Dev2AppCapabilities_NoOfSupportedDeviceTypes bytes
Dev2ProfileIdList - Dev2AppCapabilities_NoOfSupportedProfiles bytes
```

Example

```
FSLProfile_RmtPair.Request 02 DA 06 0E 00 FF F0 00 00 01 01 01 01 01 00 01 01 01 01 DC
StartOfFrame                                [1 byte ] = 02
Header                                      [2 bytes] = DA 06
PayloadLength                               [2 bytes] = 00 0E
AppRecipRspTimeOut                          [2 bytes] = F0 FF
DeviceId1                                   [1 byte ] = 00
Dev1AppCapabilities_UserStringSpecified    [1 byte ] = 00 (UserStringNotIncludedInFrame)
Dev1AppCapabilities_NoOfSupportedDeviceTypes [1 byte ] = 01
(OneDeviceTypeInDeviceTypeList)
```

```
Dev1AppCapabilities_NoOfSupportedProfiles    [1 byte ] = 01
(OneSupportedProfilesInProfileIdList)
Dev1TypeList                                 [1 byte ] = 01
        Dev1TypeList[0] = 01
Dev1ProfileIdList                            [1 byte ] = 01
        Dev1ProfileIdList[0] = 01
DeviceId2                                    [1 byte ] = 01
Dev2AppCapabilities_UserStringSpecified     [1 byte ] = 00 (UserStringNotIncludedInFrame)
Dev2AppCapabilities_NoOfSupportedDeviceTypes [1 byte ] = 01
(OneDeviceTypeInDeviceTypeList)
Dev2AppCapabilities_NoOfSupportedProfiles    [1 byte ] = 01
(OneSupportedProfilesInProfileIdList)
Dev2TypeList                                 [1 byte ] = 01
        Dev2TypeList[0] = 01
Dev2ProfileIdList                            [1 byte ] = 01
        Dev2ProfileIdList[0] = 01
Checksum                                     [1 byte ] = DC
```

3.2.40.2 FSLProfile_RmtPair.Confirm

Description

Confirmation for FSLProfile_RmtPair.Request.

Payload

```
Status – 1 byte
```

Example

```
FSLProfile_RmtPair.Confirm 02 DB 06 01 00 00 DC
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = DB 06
PayloadLength [2 bytes] = 00 01
Status        [1 byte ] = 00 (gNWSuccess_c)
Checksum      [1 byte ] = DC
```

NOTE

If the FSLProfile_RmtPair.Confirm message informs the host about the success of a remote pair then after receiving the FSLProfile_RmtPair.Confirm message the host application shall not initiate any service request to the BlackBox for the next 300 ms. The BlackBox needs to store sensitive information in its non volatile memory and while doing this it is not able to ‘hear’ any incoming packet on the serial interface, because the platform interrupts are disabled.

3.2.40.3 FSLProfile_RmtPair.Indication

Description

When a Remote Pair request frame is received, the FSL Profile will signal the Application Layer that the frame was received by generating and sending a FSLProfile_RmtPair.Indication message.

Payload

```
Status                                      - 1 byte
DeviceId                                    - 1 byte
AppRspTimeOut                               - 2 bytes
DevAppCapabilities_UserStringSpecified      - 1 byte
DevAppCapabilities_NoOfSupportedDeviceTypes - 1 byte
DevAppCapabilities_NoOfSupportedProfiles    - 1 byte
DevTypeList - DevAppCapabilities_NoOfSupportedDeviceTypes bytes
ProfileIdList - DevAppCapabilities_NoOfSupportedProfiles bytes
```

Example

```
FSLProfile_RmtPair.Indication 02 DB 07 09 00 00 01 FF F0 00 01 01 01 01 DB
StartOfFrame                                [1 byte ] = 02
Header                                      [2 bytes] = DB 07
PayloadLength                               [2 bytes] = 00 09
Status                                      [1 byte ] = 00 (gNWSuccess_c)
DeviceId                                    [1 byte ] = 01
AppRspTimeOut                               [2 bytes] = F0 FF
DevAppCapabilities_UserStringSpecified      [1 byte ] = 00
DevAppCapabilities_NoOfSupportedDeviceTypes [1 byte ] = 01
DevAppCapabilities_NoOfSupportedProfiles    [1 byte ] = 01
DevTypeList                                 [1 byte ] = 01
         DevTypeList[0] = 01
ProfileIdList                               [1 byte ] = 01
         ProfileIdList[0] = 01
Checksum                                    [1 byte ] = DB
```

3.2.40.4 FSLProfile_RmtPairResponse

Description

On the recipient, the Application will respond issuing a FSLProfile_RmtPairResponse(when a Remote Pair frame is received).

Payload

```
Status – 1 byte
```

Example

```
FSLProfile_RmtPairResponse 02 DA 07 01 00 00 DC
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = DA 07
PayloadLength [2 bytes] = 00 01
Status        [1 byte ] = 00 (gNWSuccess_c)
Checksum      [1 byte ] = DC
```

3.2.40.5 FSLProfile_RmtPairRsp.Confirm

Description

Confirmation for FSLProfile_RmtPairResponse.

Payload

```
Status – 1 byte
DeviceId - 1 byte
```

Example

```
FSLProfile_RmtPairRsp.Confirm 02 DB 08 02 00 00 01 D0
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = DB 08
PayloadLength [2 bytes] = 00 02
Status        [1 byte ] = 00
DeviceId      [1 byte ] = 01
Checksum      [1 byte ] = D0
```

NOTE

If the FSLProfile_RmtPairResp.Confirm message informs the host about the success of a remote pair then after receiving the FSLProfile_RmtPairResp.Confirm message the host application shall not initiate any service request to the BlackBox for the next 300 ms. The BlackBox needs to store sensitive information in its non volatile memory and while doing this it is not able to 'hear' any incoming packet on the serial interface, because the platform interrupts are disabled.

3.2.41 Freescale Profile OTA Menu Browser Service

3.2.41.1 FSLProfile_BrowseMenuReq.Request

Description

FSLProfile_BrowseMenuRequest tells the profile layer to transmit a menu browse command frame to a menu owner.

Payload

```
PairingRef      - 1 byte
Direction       - 1 byte
UseSecurityFlag - 1 byte
```

Example

```
FSLProfile_BrowseMenuReq.Request 02 DA 08 03 00 00 01 01 D1
StartOfFrame    [1 byte ] = 02
Header          [2 bytes] = DA 08
PayloadLength   [2 bytes] = 00 03
PairingRef      [1 byte ] = 00
Direction       [1 byte ] = 01
UseSecurityFlag [1 byte ] = 01 (true)
Checksum        [1 byte ] = D1
```

3.2.41.2 FSLProfile_MenuBrowse.Confirm

Description

Confirmation for FSLProfile_BrowseMenuReq.Request.

Payload

```
Status – 1 byte
```

Example

```
RF4CE_NLDE_Data.Confirm 02 D3 00 03 00 00 00 F0 20
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = D3 00
PayloadLength [2 bytes] = 00 03
Status        [1 byte ] = 00 (gNWSuccess_c)
PairingRef    [1 byte ] = 00
Checksum      [1 byte ] = 20
```

3.2.41.3 FSLProfile_MenuBrowseComplete.Indication

Description

Informs the application about the result of its menu browsing request.

Payload

```
Status        1 byte
DeviceId      1 byte
UserString    15 bytes if it is specifies(0 bytes else)
```

Example

```
FSLProfile_MenuBrowseComplete.Indication 02 DB 0A 11 00 02 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 C2
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = DB 0A
PayloadLength [2 bytes] = 00 11
Status        [1 byte ] = 02
DeviceId      [1 byte ] = 00
UserString    [15 bytes] = ""
Checksum      [1 byte ] = C2
```

3.2.42 Freescale Profile OTA Menu Owner Service

3.2.42.1 FSLProfile_DisplayMenuHeaderReq.Request

Description

Instructs the profile layer to transmit a display menu header request frame to a menu displayer.

Payload

```
PairingRef          - 1 byte
UseSecurityFlag     - 1 byte
IndexSelectedEntry  - 1 byte
NrMenuItemsInWindow - 1 byte
NrMenuItemsInMenu   - 2 bytes
FirstEntryNumber    - 2 bytes
ContentType         - 1 byte
MenuTextLength      - 1 byte
MenuText            - MenuTextLength bytes
```

Example

```
FSLProfile_DisplayMenuHeaderReq.Request 02 DA 09 0E 00 00 00 01 03 03 00 03 00 01 04 4D
65 6E 75 E9
StartOfFrame        [1 byte ] = 02
Header              [2 bytes] = DA 09
PayloadLength       [2 bytes] = 00 0E
PairingRef          [1 byte ] = 00
UseSecurityFlag     [1 byte ] = 00 (false)
IndexSelectedEntry  [1 byte ] = 01
NrMenuItemsInWindow [1 byte ] = 03
NrMenuItemsInMenu   [2 bytes] = 00 03
FirstEntryNumber    [2 bytes] = 00 03
ContentType         [1 byte ] = 01
MenuTextLength      [1 byte ] = 04
MenuText            [1 byte ] = "Menu"
Checksum            [1 byte ] = E9
```

3.2.42.2 FSLProfile_DisplayMenuEntry.Request

Description

It instructs the profile layer to transmit a display menu entry request frame to a menu displayer.

Payload

```
PairingRef       - 1 byte
UseSecurityFlag  - 1 byte
EntryIndex       - 1 byte
EntryType        - 1 byte
ContentType      - 1 byte
EntryValueLength - 1 byte
EntryTextLength  - 1 byte
EntryValue       - EntryValueLength bytes
```

```
EntryText        - EntryTextLength bytes
```

Example

```
FSLProfile_DisplayMenuEntry.Request 02 DA 0A 0C 00 00 01 00 00 01 01 04 00 4D 65 6E 75 EA
StartOfFrame     [1 byte ] = 02
Header           [2 bytes] = DA 0A
PayloadLength    [2 bytes] = 00 0C
PairingRef       [1 byte ] = 00
UseSecurityFlag  [1 byte ] = 01 (true)
EntryIndex       [1 byte ] = 00
EntryType        [1 byte ] = 00
ContentType      [1 byte ] = 01
EntryValueLength [1 byte ] = 01
EntryTextLength  [1 byte ] = 04
EntryValue       [1 byte ] = 00
         EntryValue[0] = 00
EntryText        [1 byte ] = "Menu"
Checksum         [1 byte ] = EA
```

3.2.42.3 FSLProfile_DisplayMenuMessage.Request

Description

Informs the profile layer of a menu owner to transmit a display menu message request frame to a menu displayer.

Payload

```
PairingRef      - 1 byte
UseSecurityFlag - 1 byte
MessageType     - 1 byte
MenuTextLength  - 1 byte
MenuText        - MenuTextLength bytes
```

Example

```
FSLProfile_DisplayMenuMessage.Request 02 DA 0B 08 00 00 01 02 04 4D 65 6E 75 ED
StartOfFrame    [1 byte ] = 02
Header          [2 bytes] = DA 0B
PayloadLength   [2 bytes] = 00 08
PairingRef      [1 byte ] = 00
UseSecurityFlag [1 byte ] = 01 (true)
MessageType     [1 byte ] = 02
MenuTextLength  [1 byte ] = 04
MenuText        [1 byte ] = "Menu"
Checksum        [1 byte ] = ED
```

3.2.42.4 FSLProfile_DisplayCompleteIndToBrowser.Request

Description

Tells the profile layer of a menu owner to inform the menu browser whether the menu displayer has acknowledged the display menu frame requests.

Payload

```
BrowserDeviceId   - 1 byte
DisplayerDeviceId - 1 byte
UseSecurityFlag   - 1 byte
Status            - 1 byte
```

Example

```
FSLProfile_DisplayCompleteIndToBrowser.Request 02 DA 0C 04 00 00 00 01 02 D1
StartOfFrame      [1 byte ] = 02
Header            [2 bytes] = DA 0C
PayloadLength     [2 bytes] = 00 04
BrowserDeviceId   [1 byte ] = 00
DisplayerDeviceId [1 byte ] = 00
UseSecurityFlag   [1 byte ] = 01 (true)
Status            [1 byte ] = 02
Checksum          [1 byte ] = D1
```

3.2.42.5 FSLProfile_DisplayMenuExit.Request

Description

Tells the profile layer of a menu owner to inform the menu displayer that the menu has become inactive.

Payload

```
DeviceId        - 1 byte
UseSecurityFlag - 1 byte
```

Example

```
FSLProfile_DisplayMenuExit.Request 02 DA 0D 02 00 00 01 D4
StartOfFrame    [1 byte ] = 02
Header          [2 bytes] = DA 0D
PayloadLength   [2 bytes] = 00 02
DeviceId        [1 byte ] = 00
UseSecurityFlag [1 byte ] = 01 (true)
Checksum        [1 byte ] = D4
```

3.2.42.6 FSLProfile_DisplayMenu.Confirm

Description

The Display Menu Confirm message informs the application about the transmission status of a display menu frame (menu header, menu entry or menu message).

Payload

```
Status – 1 byte
DeviceId -1 byte
```

Example

```
FSLProfile_DisplayMenu.Confirm 02 DB 0C 02 00 00 00 D5
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = DB 0C
PayloadLength [2 bytes] = 00 02
Status        [1 byte ] = 00
DeviceId      [1 byte ] = 00
Checksum      [1 byte ] = D5
```

3.2.43 Freescale Profile OTA Menu Displayer Service

3.2.43.1 FSLProfile_DisplayMenuHeader.Indication

Description

Informs the application of a menu displayer about the arrival of display menu header request from a menu owner.

Payload

```
PairingRef          - 1 byte
IndexSelectedEntry  - 1 byte
NrMenuItemsInWindow - 1 byte
NrMenuItemsInMenu   - 2 bytes
FirstEntryNumber    - 2 bytes
ContentType         - 1 byte
MenuTextLength      - 1 byte
MenuText            - MenuTextLength bytes
```

Example

```
FSLProfile_DisplayMenuHeader.Indication 02 DB 0D 0D 00 00 01 03 03 00 03 00 01 04 4D 65
6E 75 EF
StartOfFrame        [1 byte ] = 02
Header              [2 bytes] = DB 0D
PayloadLength       [2 bytes] = 00 0D
PairingRef          [1 byte ] = 00
IndexSelectedEntry  [1 byte ] = 01
NrMenuItemsInWindow [1 byte ] = 03
NrMenuItemsInMenu   [2 bytes] = 00 03
FirstEntryNumber    [2 bytes] = 00 03
ContentType         [1 byte ] = 01
MenuTextLength      [1 byte ] = 04
MenuText            [1 byte ] = "Menu"
Checksum            [1 byte ] = EF
```

3.2.43.2 FSLProfile_DisplayMenuEntry.Indication

Description

Informs the application of a menu displayer about the arrival of display menu entry request from a menu owner.

Payload

```
PairingRef       - 1 byte
EntryIndex       - 1 byte
EntryType        - 1 byte
ContentType      - 1 byte
EntryValueLength - 1 byte
EntryTextLength  - 1 byte
EntryValue       - EntryValueLength bytes
EntryText        - EntryTextLength bytes
```

Example

```
FSLProfile_DisplayMenuEntry.Indication 02 DB 0E 0B 00 00 00 00 01 01 04 00 4D 65 6E 75 E9
StartOfFrame     [1 byte ] = 02
Header           [2 bytes] = DB 0E
PayloadLength    [2 bytes] = 00 0B
PairingRef       [1 byte ] = 00
EntryIndex       [1 byte ] = 00
EntryType        [1 byte ] = 00 (false)
ContentType      [1 byte ] = 01
EntryValueLength [1 byte ] = 01
EntryTextLength  [1 byte ] = 04
EntryValue       [1 byte ] = 00
         EntryValue[0] = 00
EntryText        [1 byte ] = "Menu"
Checksum         [1 byte ] = E9
```

3.2.43.3 FSLProfile_DisplayMenuComplete.Indication

Description

Informs the application of the reception of complete menu window (one menu header and multiple menu entries).

Payload

```
Status        - 1 byte
DeviceId      - 1 byte
```

Example

```
FSLProfile_DisplayMenuComplete.Indication 02 DB 0F 02 00 B3 00 65
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = DB 0F
PayloadLength [2 bytes] = 00 02
Status        [1 byte ] = B3
DeviceId      [1 byte ] = 00
Checksum      [1 byte ] = 65
```

3.2.43.4 FSLProfile_DisplayMenuMessage.Indication

Description

Informs the application of the reception of a display menu message request frame.

Payload

```
PairingRef     - 1 byte
MessageType    - 1 byte
MenuTextLength - 1 byte
MenuText       - MenuTextLength bytes
```

Example

```
FSLProfile_DisplayMenuMessage.Indication 02 DB 10 07 00 00 02 04 4D 65 6E 75 F9
StartOfFrame   [1 byte ] = 02
Header         [2 bytes] = DB 10
PayloadLength  [2 bytes] = 00 07
PairingRef     [1 byte ] = 00
MessageType    [1 byte ] = 02
MenuTextLength [1 byte ] = 04
MenuText       [1 byte ] = "Menu"
Checksum       [1 byte ] = F9
```

3.2.43.5 FSLProfile_DisplayMenuExit.Indication

Description

Informs the application of the reception of a display menu exit request frame.

Payload

```
DeviceId – 1 byte
```

Example

```
FSLProfile_DisplayMenuExit.Indication 02 DB 11 01 00 00 CB
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = DB 11
PayloadLength [2 bytes] = 00 01
DeviceId      [1 byte ] = 00
Checksum      [1 byte ] = CB
```

3.2.44 Freescale Profile Utilities Service

3.2.44.1 FSLProfile_GetSupportedFeatures.Request

Description

Get the bitmap of the supported features.

Payload

```
DeviceId        - 1 byte
UseSecurityFlag - 1 byte
```

Example

```
FSLProfile_GetSupportedFeatures.Request 02 DA 0E 02 00 00 01 D7
```

```
StartOfFrame    [1 byte ] = 02
Header          [2 bytes] = DA 0E
PayloadLength   [2 bytes] = 00 02
DeviceId        [1 byte ] = 00
UseSecurityFlag [1 byte ] = 01 (true)
Checksum        [1 byte ] = D7
```

3.2.44.2 FSLProfile_GetSupportedFeatures.Confirm

Description

Confirmation for FSLProfile_GetSupportedFeatures.Request.

Payload

```
Status               - 1 byte
DeviceId             - 1 byte
SupportedFeaturesMap - 4 bytes
```

Example

```
FSLProfile_GetSupportedFeatures.Confirm 02 DB 12 06 00 00 00 C0 01 00 00 0E
StartOfFrame         [1 byte ] = 02
Header               [2 bytes] = DB 12
PayloadLength        [2 bytes] = 00 06
Status               [1 byte ] = 00
DeviceId             [1 byte ] = 00
SupportedFeaturesMap [4 bytes] = 00 00 01 C0
Checksum             [1 byte ] = 0E
```

3.2.45 Freescale Low Power Control messages

The BlackBox application contain built-in support for minimizing the power consumption by entering into a low power state. At a given moment in time, from the host application point of view, the BlackBox can be in one of 2 states:

1. Active - MCU and transceiver are working normally.
2. Inactive - MCU and transceiver are put into a low power state. The low power states the MCU and transceiver will enter when the BlackBox is inactive cannot be configured. The MCU will always use STOP3 mode while the transceiver will always use the Doze mode.

The host application can ask the BlackBox to change it’s state through the use of a connection line (called wakeUp line) between one pin on the host processor and one pin on the BlackBox processor. By setting this line to a particular level (high or low), the host can tell the BlackBox which exact state it wants the BlackBox to be in (active or inactive).

The host application can configure the following aspects of the low power mode:

1. The pin on the BlackBox processor used for the wakeUp line. The application can choose one of the 8 pins on PORT A (keyboard port) of the MC1321x. The MC1323x offers 12 pins that can wake up the processor: 8 pins on PORT B and the first 4 pins of PORT C (all keyboard pins). The MC1322x offers 4 pins that can wake up the processor: KBD 7-4.

2. The way the BlackBox will interpret the level of the wakeUp line. This allows the application to choose whether it wants the BlackBox to be active when the wakeUp line is kept high or when it is kept low.
3. Whether it wants the BlackBox to signal the host application with a message when it switches from inactive to active mode as a result of the wakeUp line state change. This may be useful for the host application because the BlackBox needs a certain amount of time for waking up from the inactive state to the active state. The message is sent to the host as fast as the BlackBox is awake and ready to process any incoming requests, either from the host through the serial interface or from the radio.

Configuring all three of these options is accomplished through the ZTC-WakeUpConfig.Request message. As long as the low power mode is not configured using the already mentioned service, the BlackBox will always be active.

The BlackBox application is able to change state from active to inactive only when the following two conditions are met:

1. The BlackBox is idle (meaning there is no MAC, network, profile or application process ongoing). If the second condition is met and the wakeUp line asks the BlackBox to switch to the inactive state, it does that as soon as all the processes are complete and it transitions to idle state.
2. The radio receiver is closed. The application must explicitly turn off the receiver before asking the BlackBox to switch to the inactive state to actually enter low power. The state of the receiver can be previously configured using the NLME_RxEnableRequest network service. For the particular case where the receiver is configured to intermittently turn on and off, the BlackBox application will be in low power mode only during that time interval when the receiver is off and will be in active mode for the time intervals when the receiver is on, regardless if the wakeUp line requests the BlackBox to be inactive. In this case, there will be no wakeUp indication message sent to the application every time the BlackBox switches from inactive to active as a result of the receiver being switched from off to on. The wakeUp message is sent to the host only when the host changes the state of the wakeUp line, requesting the BlackBox to become active.

3.2.45.1 ZTC-WakeUpConfig.Request

Description

The request selects and configures:

1. The BlackBox pin where the wakeUp line will be connected. The pin will be provided as an 8 bit mask having all bits set to 0 except for the one to be used for wakeUp line which should be set to 1.
2. Whether BlackBox should wake up (switch from inactive to active) on the rising edge or on the falling edge of the wakeUp line. In the first case the BlackBox will be active when the wakeUp line is high and inactive when the wakeUp line is low. In the second case, the BlackBox will be active when the wakeUp line is low and inactive when the wakeUp line is high.
3. Whether the BlackBox should send a wakeUp indication message after the wakeUp line has been toggled requesting that the BlackBox become active.

Payload

```
KBI Pin mask              - 1 byte
```

```
WakeUp MCU on edges       - 1 byte
Signal Host  When Wake Up - 1 byte
```

Example

```
ZTC-WakeUpConfig.Request 02 A3 40 03 00 04 00 01 E5
StartOfFrame              [1 byte ] = 02
Header                    [2 bytes] = A3 40
PayloadLength             [2 bytes] = 00 03
KBI Pin mask              [1 byte ] = 04 (KBIP2)
WakeUp MCU on edges       [1 byte ] = 00 (FallingEdges)
Signal Host  When Wake Up [1 byte ] = 01 (TRUE)
Checksum                  [1 byte ] = E5
```

NOTE

On the MC1321x, all keyboard pins (KBIP0-KBIP7) can be configured to wake the MCU (and thus the BlackBox) on falling edges, but only the pins from KBIP4 to KBIP7 can be configured to wake the MCU on rising edges. Configuring the wakeUp line to be an active high and be connected to a KBIP0-KBIP3 triggers the ‘gZtcError_c’ error.

3.2.45.2 ZTC-WakeUpConfig.Confirm

Description

Confirmation for ZTC-WakeUpConfig.Request

Payload

```
Status - 1 byte
```

Example

```
ZTC-WakeUpConfig.Confirm 02 A4 40 01 00 00 E5
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = A4 40
PayloadLength [2 bytes] = 00 01
Status        [1 byte ] = 00 (gSuccess)
Checksum      [1 byte ] = E5
```

3.2.45.3 ZTC-WakeUp.Indication

Description

This message is used by the BlackBox to signal the host that it is active. It is used only as a response for the host’s action of toggling the wakeUp line requesting the BlackBox to switch to active mode.

Payload

```
None
```

Example

```
ZTC-WakeUp.Indication 02 A4 41 00 00 E5
StartOfFrame  [1 byte ] = 02
Header        [2 bytes] = A4 41
PayloadLength [2 bytes] = 00 00
Checksum      [1 byte ] = E5
```

3.2.46 ZTC Control messages

3.2.46.1 ZTC-WriteMemoryBlock.Request

Description

This request writes a memory block of data.

Payload

```
StartAddress - the address where to write the data
NumberOfBytes - the number of bytes
Data - the bytes
```

Example

```
ZTC-WriteMemoryBlock.Request 02 A3 30 08 00 00 10 00 00 03 11 11 11 99
        Sync          [1 byte ]  = 02
        OpGroup       [1 byte ]  = A3
        OpCode        [1 byte ]  = 30
        Length        [2 bytes]  = 08 00
        StartAddress  [4 bytes]  = 00 00 10 00
        NumberOfBytes [1 byte ]  = 03
        Data          [3 bytes]  = 11 11 11
        CRC           [1 byte ]  = 9
```

3.2.46.2 ZTC-WriteMemoryBlock.Confirm

Description

.Confirmation for ZTC-WriteMemoryBlock.Request.

Payload

```
NumberOfBytesWritten - number of bytes that was written (1 byte)
```

Example

```
ZTC-WriteMemoryBlock.Confirm 02 A4 30 01 00 03 96
        Sync                [1 byte ]  = 02
        OpGroup             [1 byte ]  = A4
        OpCode              [1 byte ]  = 30
        Length              [2 bytes]  = 01 00
        NumberOfBytesWritten [1 byte ]  = 03
        CRC                 [1 byte ]  = 96
```

3.2.46.3 ZTC-ReadMemoryBlock.Request

Description

This request reads a memory block of data.

Payload

```
StartAddress - the address from where to read the data
NumberOfBytes - the number of bytes
```

Example

```
ZTC-ReadMemoryBlock.Request 02 A3 31 05 00 00 10 00 00 03 84
        Sync          [1 byte ]  = 02
        OpGroup       [1 byte ]  = A3
        OpCode        [1 byte ]  = 31
        Length        [2 bytes]  = 05 00
        StartAddress  [4 bytes]  = 00 00 10 00
        NumberOfBytes [1 byte ]  = 03
        CRC           [1 byte ]  = 84
```

3.2.46.4 ZTC-ReadMemoryBlock.Confirm

Description

.Confirmation for ZTC-ReadMemoryBlock.Request.

Payload

```
ReadData - the data read from memory
```

Example

```
ZTC-ReadMemoryBlock.Confirm 02 A4 31 03 00 11 11 11 87
        Sync     [1 byte ]  = 02
        OpGroup  [1 byte ]  = A4
        OpCode   [1 byte ]  = 31
        Length   [2 bytes]  = 03 00
        ReadData [3 byte ]  = 11 11 11
        CRC      [1 byte ]  = 87
```

3.2.46.5 ZTC-GetLastPacketLQI.Request

Description

This request gets the LQI of the last received packet.

Example

```
ZTC-GetLastPacketLQI.Request 02 A3 44 00 00 E7
        Sync    [1 byte ]  = 02
        OpGroup [1 byte ]  = A3
        OpCode  [1 byte ]  = 44
        Length  [2 bytes]  = 00 00
```

```
        CRC     [1 byte ]  = E7
```

3.2.46.6 ZTC-GetLastPacketLQI.Confirm

Description

.Confirmation for ZTC-GetLastPacketLQI.Request.

Payload

```
LQI - the LQI of the last received packet.
```

Example

```
ZTC-GetLastPacketLQI.Confirm 02 A4 45 01 00 00 E0
        Sync    [1 byte ]  = 02
        OpGroup [1 byte ]  = A4
        OpCode  [1 byte ]  = 45
        Length  [2 bytes]  = 01 00
        LQI     [1 byte ]  = 00
        CRC     [1 byte ]  = E0
```

3.2.46.7 ZTC-StackStatus.Request

Description

This request gets information about the space of RAM allocated for Stack (Stack size, Stack maximum usage).

NOTE

This request should not be used when the NVM module is enable. If the NVM module is enable, the ZTC can not keep track of the RAM Stack changes to build the confirm because the NVM is using the RAM Stack to read/write data from/to flash memory.

Example

```
ZTC-StackStatus.Request 02 A3 42 00 00 E1
        Sync    [1 byte ]  = 02
        OpGroup [1 byte ]  = A3
        OpCode  [1 byte ]  = 42
        Length  [2 bytes]  = 00 00
        CRC     [1 byte ]  = E1
```

3.2.46.8 ZTC-StackStatus.Confirm

Description

.Confirmation for ZTC-StackStatus.Request.

Payload

```
Status - the status: 0x00 - Read Successful, 0x01 - RAM Stack is NOT initialized, 0x02 -
NVM module is enabled.
StackDimension - the RAM Stack size
StackMaxUsed - the maximum used size
```

Example

```
ZTC-StackStatus.Confirm 02 A4 43 05 00 02 00 02 00 00 E2
        Sync           [1 byte ]  = 02
        OpGroup        [1 byte ]  = A4
        OpCode         [1 byte ]  = 43
        Length         [2 bytes]  = 05 00
        Status         [1 byte ]  = 02 (Err)
        StackDimension [2 bytes]  = 02 00
        StackMaxUsed   [2 bytes]  = 00 00
        CRC            [1 byte ]  = E2
```

3.2.46.9 ZTC-CPU_Reset.Request

Description

This request resets the CPU. No confirm is generated for this request.

Example

```
ZTC-CPU_Reset.Request 02 A3 08 00 00 AB
        Sync    [1 byte ]  = 02
        OpGroup [1 byte ]  = A3
        OpCode  [1 byte ]  = 08
        Length  [2 bytes]  = 00 00
        CRC     [1 byte ]  = AB
```

3.2.46.10 ZTC-ModeSelect.Request

Description

The ZTC injects or monitors the messages to/from different SAPs. There are three modes to control the SAP messages:

- disable mode - the SAP messages are ignored.
- hook mode - the SAP messages are not sent to the upper layer but are sent to Test Client (Test Tool). In this mode, the Test Client can replace the layer whose SAP is hooked.
- monitor mode - capture all messages received by a SAP.

This request selects the available modes to control each SAP (Service Access Points); it also sets the UART transmission to be blocking or not.

Payload

```
UARTTxBlocking - the UART transmission option
MCPS - select the mode for MCPS SAP
MLME - select the mode for MLME SAP
```

```
ASP - select the mode for ASP SAP
RF4CENLDE - select the mode for RF4CENLDE SAP
RF4CENLME - select the mode for RF4CENLME SAP
RF4CENWKUtilities - select the mode for RF4CENWKUtilities SAP
```

Example

```
ZTC-ModeSelect.Request 02 A3 00 07 00 01 00 00 00 02 02 02 A7
        Sync              [1 byte ]  = 02
        OpGroup           [1 byte ]  = A3
        OpCode            [1 byte ]  = 00
        Length            [2 bytes]  = 07 00
        UARTTxBlocking    [1 byte ]  = 01
        MCPS              [1 byte ]  = 00 (DisableMode)
        MLME              [1 byte ]  = 00 (DisableMode)
        ASP               [1 byte ]  = 00 (DisableMode)
        RF4CENLDE         [1 byte ]  = 02 (MonitorMode)
        RF4CENLME         [1 byte ]  = 02 (MonitorMode)
        RF4CENWKUtilities [1 byte ]  = 02 (MonitorMode)
        CRC               [1 byte ]  = A7
```

3.2.46.11 ZTC-ModeSelect.Confirm

Description

.Confirmation for ZTC-ModeSelect.Request.

Payload

```
Status - the status of the request
```

Example

```
ZTC-ModeSelect.Confirm 02 A4 00 01 00 00 A5
        Sync    [1 byte ]  = 02
        OpGroup [1 byte ]  = A4
        OpCode  [1 byte ]  = 00
        Length  [2 bytes]  = 01 00
        Status  [1 byte ]  = 00 (gSuccess)
        CRC     [1 byte ]  = A5 [1 byte ]  = 96
```

3.2.46.12 ZTC-GetMode.Request

Description

This request gets the selected modes which control each SAP (Service Access Points); it also gets the UART transmission option (is blocking or not).

Example

```
ZTC-GetMode.Request 02 A3 02 00 00 A1
        Sync    [1 byte ]  = 02
        OpGroup [1 byte ]  = A3
        OpCode  [1 byte ]  = 02
```

```
        Length  [2 bytes]  = 00 00
        CRC     [1 byte ]  = A1
```

3.2.46.13 ZTC-GetMode.Confirm

Description

.Confirmation for ZTC-GetMode.Request.

Payload

```
Status - the status of the request
UARTTxBlocking - the UART transmission option
MCPS - the selected mode for MCPS SAP
MLME - the selected mode for MLME SAP
ASP - the selected mode for ASP SAP
RF4CENLDE - the selected mode for RF4CENLDE SAP
RF4CENLME - the selected mode for RF4CENLME SAP
RF4CENWKUtilities - the selected mode for RF4CENWKUtilities SAP
```

Example

```
ZTC-GetMode.Confirm 02 A4 02 0C 00 00 01 FF FF FF 02 02 02 02 02 02 02 56
        Sync            [1 byte ]  = 02
        OpGroup         [1 byte ]  = A4
        OpCode          [1 byte ]  = 02
        Length          [2 bytes]  = 0C 00
        Status          [1 byte ]  = 00 (gSuccess)
        UARTTxBlocking  [1 byte ]  = 01
        MCPS            [1 byte ]  = FF (NotSupported)
        MLME            [1 byte ]  = FF (NotSupported)
        ASP             [1 byte ]  = FF (NotSupported)
        RF4CENLDE       [1 byte ]  = 02 (MonitorMode)
        RF4CENLME       [1 byte ]  = 02 (MonitorMode)
        RF4CENWK        [1 byte ]  = 02 (MonitorMode)
        CRC             [1 byte ]  = 56
```

3.2.46.14 ZTC-WriteExtAddr.Request

Description

This request writes the IEEE address.

Payload

```
Address - the IEEE address
```

Example

```
ZTC-WriteExtAddr.Request 02 A3 DB 08 00 AA AA AD FF CF CC EE AA 65
        Sync    [1 byte ]  = 02
        OpGroup [1 byte ]  = A3
        OpCode  [1 byte ]  = DB
        Length  [2 bytes]  = 08 00
```

```
        Address [8 bytes]  = AA EE CC CF FF AD AA AA
        CRC     [1 byte ]  = 65
```

3.2.46.15 ZTC-WriteExtAddr.Confirm

Description

.Confirmation for ZTC-WriteExtAddr.Request.

Payload

```
Status - the status of the request
```

Example

```
ZTC-WriteExtAddr.Confirm 02 A4 DB 01 00 00 7E
        Sync    [1 byte ]  = 02
        OpGroup [1 byte ]  = A4
        OpCode  [1 byte ]  = DB
        Length  [2 bytes]  = 01 00
        Status  [1 byte ]  = 00
        CRC     [1 byte ]  = 7E
```

3.2.46.16 ZTC-ReadExtAddr.Request

Description

This request reads the IEEE address.

Example

```
ZTC-ReadExtAddr.Request 02 A3 D2 00 00 71
        Sync    [1 byte ]  = 02
        OpGroup [1 byte ]  = A3
        OpCode  [1 byte ]  = D2
        Length  [2 bytes]  = 00 00
        CRC     [1 byte ]  = 71
```

3.2.46.17 ZTC-ReadExtAddr.Confirm

Description

.Confirmation for ZTC-ReadExtAddr.Request.

Payload

```
Status - the status of the request
DeviceAddr - the device IEEE address
```

Example

```
ZTC-ReadExtAddr.Confirm 02 A4 D2 09 00 00 AA AA AD FF CF CC EE AA 6A
        Sync       [1 byte ]  = 02
        OpGroup    [1 byte ]  = A4
```

```
OpCode      [1 byte ]  = D2
Length      [2 bytes]  = 09 00
Status      [1 byte ]  = 00
DeviceAddr  [8 bytes]  = AA EE CC CF FF AD AA AA
CRC         [1 byte ]  = 6A
```

3.3 ZigBee Input Device (ZID) Profile Messages and Services

Before using the ZID ZTC commands from the following sections, users must ensure that the Zigbee Input Device (ZID) Profile specification have been read and understood.

The ZTC Node application from BeeKit software can be configured to generate a ZID Class device node or a ZID Adaptor node. To generate a ZID Class device node, the user should include the “ZID Class Device library” from the “Freescale BeeStack Consumer ZID Profile” component, and the “Push Button Pair Originator feature” from the “Freescale BeeStack Consumer Push Button Pair” component. To generate a ZID Adaptor node, the user should include the “ZID Adaptor library” from the “Freescale BeeStack Consumer ZID Profile” component, and the “Push Button Pair Recipient feature” from the “Freescale BeeStack Consumer Push Button Pair” component.

3.3.1 ZID Class Device Node Default Configuration

The ZID Class device has the capability to configure and store maximum five reports, four non-standard descriptors and four NULL reports. The ZID Class device manages three tables that can be configured:

- The non-standard descriptors table. It has 4 entries and uses a memory pool (gaZtcNonStdDescCompsPool) of 250 (gZtcZIDDeviceNonStdDescCompsPoolSize_c) bytes to store all non-standard descriptors. The user can configure and store a non-standard descriptor using ZIDClassDevice-SetLocalAttribute.Req (0xF1) command.
- The reports table (gaClassDevReportsList). It has 5 entries and uses a memory pool (gaZtcReportsPool) of 50 (gZtcZIDDeviceMaxNumOfReports_c) bytes to store all devices’ reports. The user can configure and store reports using ZIDClassDevice-ConfigureReportData.Req command.
- The non-standard NULL reports table (gaNonStdNullReportsList). It has 4 entries and uses a memory pool (gaZtcZIDDeviceNullReportsPool) of 50 (gZtcZIDDeviceNullReportsPoolSize_c) bytes to store all non-standard NULL reports. The user can configure and store a NULL report using ZIDClassDevice-SetNonStdNULLReportData.Req.

NOTE

There tables are declared in `ZIDClassDeviceGlobals.c` file and their sizes can be configured from ZtcInterface.h file. These tables, besides others, are saved in flash using the application NVM data set. Care must be taken when the sizes of these tables are changed, so that not to exceed the allocated application NVM data sector.

Note that each table configuration depends on the following ZID attributes configuration:

- Number of standard descriptor components (gZidAttrId_ZidNumStdDescComps_c - 0xE9)
- Standard descriptor list (gZidAttrId_ZidStdDescCompsList_c -0xEA)

- Number of non-standard NULL reports (gZidAttrId_ZidNumNullReports_c - 0xEB)
- Number of non-standard descriptors components (gZidAttrId_ZidNumNonStdDescComps_c – 0xF0)

By default the ZID Class device is configured to have one standard report (mouse) and one non-standard report (non-standard keys). The non-standard report is described by a descriptor and a NULL report.

The default reports are the followings:

- A mouse report: standard report, report id =0x01, has 3 byte of data allocated on gaZtcReportsPool pool from offset 0 to 2. Use the ZIDClassDevice-GetConfiguredReportData command with EntryIndex=0 to receive the report information.
- A non-standard keys report: non standard report, report id =0x10, has 1 byte of data allocated on gaZtcReportsPool pool from offset 3. Use ZIDClassDevice-GetConfiguredReportData command with EntryIndex=1 to receive the report information. This report is described by the non-standard descriptor 0xF1 (gZidAttrId_ZidNumNonStdDescCompSpec1_c) and a NULL report with the value: 0x00.

3.3.1.1 Using the ZID Class device default configuration

1. Generate two ZTC Node applications from BeeKit: one configured as a ZID Class Device and another as a ZID Adaptor device.
2. Load the boards, one with the ZID Class device image and another with the ZID adaptor image.
3. Configure Test Tool so that it can communicate with the boards:
   — baud rate-38400
   — ZTC Length-Auto
   — Use Flow Control-enabled
4. From Test Tool, start the Command Console and ensure that the `ZigBeeRF4CE.xml` file is selected.
5. Start the Adaptor as a target device and the ZID Class Device as a controller device, running the following commands: ZTC-WriteExtAddr.Request, RF4CE_NLME_Reset, RF4CE_NWK_SetNodeCapabilities, RF4CE_NLME_Start .
6. Pair and configure the devices using the ZIDClassDevice_PBPConfig.Req and respectively ZIDAdaptor_PBPConfig.Req commands.
7. After the configuration completes successfully, exercise the <ZID commands> from `ZigBeeRF4CE.xml`. For example, use the ZIDClassDevice-ReportData.Req or ZIDClassDevice-SendReportIdsList.Req commands to send the report frame which can include one or more reports.

3.3.1.2 Reconfigure the ZID Class Device with New Reports

The ZID Class device can be reconfigured with different reports that overwrite the default reports. This must be done before starting the node.

To reconfigure the ZID Class device perform the following tasks:

1. Configure the ZID attributes using the ZIDClassDevice-SetLocalAttribute.Req command. The user can configure attributes such as gZidAttrId_ZidNumStdDescComps_c, gZidAttrId_ZidStdDescCompsList_c , gZidAttrId_ZidNumNullReports_c, gZidAttrId_ZidNumNonStdDescComps_c, gZidAttrId_ZidNumNonStdDescCompSpec1_c etc.
2. Configure the supported reports data using the ZIDClassDevice-ConfigureReportData.Req command.
3. Set the NULL report (if any) using the ZIDClassDevice-SetNonStdNULLReportData.Req commands .
4. At this step the ZID Class device is considered configured with new reports. Start the node and the paring and configuration process using ZIDClassDevice_PBPConfig.Req command.

3.3.2 Observations

- During the Configuration phase, the ZID Class device receives the ZIDClassDevice_CompatibilityCheck.Indication message which is the Get Attribute Response message sent by the Adaptor. The ZTC application should response to this message within 100 milliseconds, accepting or denying to continue configuring (sends the ZIDClassDevice-CompatibilityCheckResp.Req with the parameter ContinueConnect TRUE or FALSE).
- The ReportData.Req command (on ZID Adaptor or ZID Class device) receives as parameter the list of report records. Please refer to ZID profile specification to see how a report record is defined.
- The ZIDClassDevice-SendReportIdsList.Req command uses a list of report IDs to form and send the report records. Setting the “action” parameter, the built report frame can be sent only once or repeated at the specified ReportRepeatInterval attribute. To stop repeating the formed report frame, the ZIDClassDevice-SendReportIdsList.Req command have to be sent again with the same device ID parameter and the “action” parameter set to gStopRepeatReportFrame_c; in this case the rest of the parameters are ignored. In order to start repeating another report frame, which can include one or more reports, the user must stop the current repetition.
- The ZID Adaptor Device has by default the Data pending flag for each connection set to FALSE (the device has no data pending for the remote node), this means that when a Heartbeat frame is received, the ZID profile layer automatically responds with Generic Response frame. To receive the HeartBeat.Indication message, set the Data pending to TRUE, using ZIDAdaptor-SetDataPending.Req command.

3.3.3 ZID ZTC Command and Message List

The commands and messages are ordered based on Opcode Group and Opcode values:

Table 3-2. ZID Commands and Messages

Opcode Group	Opcode	Service
Opcode Group	Opcode	Service
0xE3	0x02	ZID_GetAttributes.Confirm

Table 3-2. ZID Commands and Messages (continued)

Opcode Group	Opcode	Service
0xE3	0x03	ZID_PBPConfig.Confirm
0xE3	0x01	ZID_ReportData.Confirm
0xE3	0x00	ZID_ReportData.Indication
0xE2	0xCA	ZIDAdaptor_AbortProcess.Req
0xE3	0xC6	ZIDAdaptor_AbortProcess.Confirm
0xE2	0xC9	ZIDAdaptor_DeviceIsIdle.Req
0xE3	0xC5	ZIDAdaptor_DeviceIsIdle.Confirm
0xE2	0xC4	ZIDAdaptor_GetAttributes.Req
0xE2	0xCC	ZIDAdaptor_GetConnectionInfo.Req
0xE3	0xC7	ZIDAdaptor_GetConnectionInfo.Confirm
0xE2	0xC2	ZIDAdaptor_GetLocalAttribute.Req
0xE3	0xC1	ZIDAdaptor_GetLocalAttribute.Confirm
0xE2	0xC8	ZIDAdaptor_GetNonStandardDescComp.Req
0xE3	0xC4	ZIDAdaptor_GetNonStandardDescComp.Confirm
0xE2	0xC5	ZIDAdaptor_GetReport.Req
0xE3	0x0B	ZIDAdaptor_GetReport.Confirm
0xE3	0x09	ZIDAdaptor_Heartbeat.Indication
0xE2	0xC1	ZIDAdaptor_PBPConfig.Req
0xE3	0x0A	ZIDAdaptor_PushAttr.Indication
0xE2	0xC7	ZIDAdaptor_RemoveConfiguredDevice.Req
0xE3	0xC3	ZIDAdaptor_RemoveConfiguredDevice.Confirm
0xE2	0xCB	ZIDAdaptor_ReportData.Req
0xE2	0xCD	ZIDAdaptor_SetDataPending.Req
0xE3	0xC8	ZIDAdaptor_SetDataPending.Req.Confirm
0xE2	0xC3	ZIDAdaptor_SetLocalAttribute.Req
0xE3	0xC2	ZIDAdaptor_SetLocalAttribute.Confirm
0xE2	0xC6	ZIDAdaptor_SetReport.Req
0xE3	0x0C	ZIDAdaptor_SetReport.Confirm
0xE2	0xC0	ZIDAdaptor_StartWithNVM.Req
0xE3	0xC0	ZIDAdaptor_StartWithNVM.Confirm
0xE2	0x87	ZIDClassDevice_AbortProcess.Req
0xE3	0x85	ZIDClassDevice_AbortProcess.Confirm
0xE3	0x04	ZIDClassDevice_CompatibilityCheck.Indication

Table 3-2. ZID Commands and Messages (continued)

Opcode Group	Opcode	Service
0xE2	0x8D	ZIDClassDevice_CompatibilityCheckResp.Req
0xE3	0x88	ZIDClassDevice_CompatibilityCheckResp.Confirm
0xE2	0x88	ZIDClassDevice_ConfigureReportData.Req
0xE3	0x86	ZIDClassDevice_ConfigureReportData.Confirm
0xE2	0x86	ZIDClassDevice_DeviceIsIdle.Req
0xE3	0x84	ZIDClassDevice_DeviceIsIdle.Confirm
0xE2	0x8C	ZIDClassDevice_GetAttributes.Req
0xE2	0x89	ZIDClassDevice_GetConfiguredReportData.Req
0xE3	0x87	ZIDClassDevice_GetConfiguredReportData.Confirm
0xE2	0x82	ZIDClassDevice_GetLocalAttribute.Req
0xE3	0x81	ZIDClassDevice_GetLocalAttribute.Confirm
0xE2	0x90	ZIDClassDevice_GetNonStdNULLReportData.Req
0xE3	0x8A	ZIDClassDevice_GetNonStdNULLReportData.Confirm
0xE2	0x8B	ZIDClassDevice_Heartbeat.Req
0xE3	0x07	ZIDClassDevice_Heartbeat.Confirm
0xE2	0x81	ZIDClassDevice_PBPConfig.Req
0xE2	0x8A	ZIDClassDevice_PushAttr.Req
0xE3	0x06	ZIDClassDevice_PushAttr.Confirm
0xE2	0x85	ZIDClassDevice_RemoveConfiguredDevice.Req
0xE3	0x83	ZIDClassDevice_RemoveConfiguredDevice.Confirm
0xE2	0x84	ZIDClassDevice_ReportData.Req
0xE2	0x8E	ZIDClassDevice_SendReportIdsList.Req
0xE3	0x08	ZIDClassDevice_SendReportIdsList.Confirm
0xE2	0x83	ZIDClassDevice_SetLocalAttribute.Req
0xE3	0x82	ZIDClassDevice_SetLocalAttribute.Confirm
0xE2	0x8F	ZIDClassDevice_SetNonStdNULLReportData.Req
0xE3	0x89	ZIDClassDevice_SetNonStdNULLReportData.Confirm
0xE3	0x05	ZIDClassDevice_SetReport.Indication
0xE2	0x80	ZIDClassDevice_StartWithNVM.Req

3.3.4 ZID Messages

3.3.4.1 ZID_GetAttributes.Confirm

Description

The confirm for ZID_GetAttributes.Req command.

Parameters

Table 3-3. ZID_GetAttributes.Confirm Parameters

Parameter	Size (bytes)	Comments
OpGroup	1	0xE3
OpCode	1	0x02
Length	1	Length in bytes of the following parameters
Status	1	The returned status specifies the ZID Generic response code field or any NWK status. Possible values: 0x00: gZIDResp_Success_c (Successy) 0x01: gZIDResp_UnsupportedRequest_c (UnsupportedRequest) 0x02: gZIDResp_InvalidParameter_c (Invalid Parameter) 0x40: gZIDResp_InvalidReportId_c (Invalid ReportId)
DeviceID	1	The deviceID confirmed.
AttrRspRecordsListSize	1	The attribute records List Size.
AttrRspRecordsList	AttrRspRecordsListSize	Get Attribute Response payload.

3.3.4.2 ZID_PBPConfig.Confirm

Description

The confirm for ZIDAdaptor_PBPConfig.Req or ZIDClassDevice_PBPConfig.Req commands.

Parameters

Table 3-4. ZID_PBPConfig.Confirm Parameters

Parameter	Size (bytes)	Comments
OpGroup	1	0xE3
OpCode	1	0x03
Length	1	Length in bytes of the following parameters

Table 3-4. ZID_PBPConfig.Confirm Parameters

Status	1	The status of the request. Possible values: 0x00: gNWSuccess_c (Requested action was completed successfully 0x90: gDeviceConfigFailure_d (The cofiguration failed) 0x91: gDeviceConfigNoCapacity_d (The device has no capacity) 0xB3: gNWNoResponse_c (No response) 0x80: gNWDenied_c (Denied)
DeviceID	1	The deviceID confirmed.
ConnectionIndex	1	The index in connection table.

3.3.4.3 ZID_ReportData.Confirm

Description

The ZID report data confirm.

Parameters

Table 3-5. ZID_ReportData.Confirm Parameters

Parameter	Size (bytes)	Comments
OpGroup	1	0xE3
OpCode	1	0x01
Length	1	Length in bytes of the following parameters
Status	1	The status of the Request. Possible values: any RF4CE network status.
DeviceID	1	The deviceID confirmed.

3.3.4.4 ZID_ReportData.Indication

Description

The ZID Report Data Indication message.

Parameters

Table 3-6. ZID_ReportData.Indication Parameters

Parameter	Size (bytes)	Comments
OpGroup	1	0xE3
OpCode	1	0x00
Length	1	Length in bytes of the following parameters
DeviceID	1	From where the the list of report records is received.
ProfileId	1	The identifier of the profile indicating the format of the received data.

Table 3-6. ZID_ReportData.Indication Parameters

VendorId	2	If the RxFlags parameter specifies that the data is vendor specific, this parameter specifies the vendor identifier. If the RxFlags parameter specifies that the data is not vendor specific this parameter is ignored.
RxLinkQuality	1	LQI value measured during the reception of the NPDU.
RxFlags	1	Reception indication flags for this NSDU.
ReportRecordsListSize	1	The number of octets contained by payload (report records list).
ReportRecordsList	ReportRecords ListSize	The list of report records.

3.3.4.5 ZIDAdaptor_AbortProcess.Req

Description

Abort the ZID profile current running process.

Parameters

Table 3-7. ZIDAdaptor_AbortProcess.Req Parameters

Parameter	Size (Bytes)	Comments
OpGroup	1	0xE2
OpCode	1	0xCA
Length	1	0x00 - This message does not have any parameters

3.3.4.6 ZIDAdaptor_AbortProcess.Confirm

Description

The confirm for ZIDAdaptor_AbortProcess.Req command.

Parameters

Table 3-8. ZIDAdaptor_AbortProcess.Confirm Parameters

Parameter	Size (Bytes)	Comments
OpGroup	1	0xE3
OpCode	1	0xC6
Length	1	Length in bytes of the following parameters
Status	1	The returned status. Possible values: 0x00: gNWSuccess_c (Success) 0x80: gNWDenied_c (Denied)

3.3.4.7 ZIDAdaptor_DeviceIsIdle.Req

Description

Check if the ZID Adaptor device is in idle state.

Parameters

Table 3-9. ZIDAdaptor_DeviceIsIdle.Req Parameters

Parameter	Size (Bytes)	Comments
OpGroup	1	0xE2
OpCode	1	0xC9
Length	1	0x00 - This message does not have any parameters

3.3.4.8 ZIDAdaptor_DeviceIsIdle.Confirm

Description

The confirm for ZIDAdaptor_DeviceIsIdle.Req command.

Parameters

Table 3-10. ZIDAdaptor_DeviceIsIdle.Confirm Parameters

Parameter	Size (Bytes)	Comments
OpGroup	1	0xE3
OpCode	1	0xC5
Length	1	Length in bytes of the following parameters
IdleState	1	The returned status. Possible values: 0x00: FALSE (Device is not in idle state) 0x01: TRUE (Device is in idle state)

3.3.4.9 ZIDAdaptor_GetAttributes.Req

Description

Get HID attributes from a remote node.

Parameters

Table 3-11. ZIDAdaptor_GetAttributes.Req Parameters

Parameter	Size (Bytes)	Comments
OpGroup	1	0xE2

Table 3-11. ZIDAdaptor_GetAttributes.Req Parameters

OpCode	1	0xC4
Length	1	Length in bytes of the following parameters
DstDeviceID	1	The destination device ID.
DataPendingFlag	1	Signal the ZID Class device that the node has data pending. Possible values: 0x00: FALSE 0x01: TRUE
AttrIDsListLength	1	The length of attribute IDs list.
AttrIDsList	AttrIDsListLength	The attribute IDs List.

3.3.4.10 ZIDAdaptor_GetConnectionInfo.Req

Description

Get information about a connection.

Parameters

Table 3-12. ZIDAdaptor_GetConnectionInfo.Req Parameters

Parameter	Size (Bytes)	Comments
OpGroup	1	0xE2
OpCode	1	0xCC
Length	1	Length in bytes of the following parameters
DeviceID	1	The device ID.

3.3.4.11 ZIDAdaptor_GetConnectionInfo.Confirm

Description

The confirm for ZIDAdaptor_GetConnectionInfo.Req command.

Parameters

Table 3-13. ZIDAdaptor_GetConnectionInfo.Confirm Parameters

Parameter	Size (Bytes)	Comments
OpGroup	1	0xE3
OpCode	1	0xC7
Length	1	Length in bytes of the following parameters

Table 3-13. ZIDAdaptor_GetConnectionInfo.Confirm Parameters

Status	1	The returned status. Possible values: 0x00: gNWSuccess_c (Success) 0xe8: gNWInvalidParam_c (Invalid Parameter)
DeviceID	1	Device Id.
zidParserVersion	2	ZID Parser Version.
zidDeviceSubclass	1	ZID Device Subclass.
zidProtocolCode	1	ZID Protocol Code.
zidCountryCode	1	ZID Country Code.
zidDeviceReleaseNumber	2	ZID Device Release Number.
zidVendorId	2	ZID Vendor Id.
zidProductId	2	ZID Product Id.
zidNumEndpoints	1	Number of Endpoints.
zidPollInterval	1	ZID Poll Interval.
zidNumStdDescComps	1	Number of standard descriptor components.
zidStdDescCompsList	zidNumStdDescComps	Standard descriptor components list.
zidNumOfNonStdNullReports	1	Number of non-standard NULL reports.
zidNumOfNonStdDescComps	1	Number of non-standard descriptor components.
DeviceIDleRate	1	device idle rate.
CurrentProtocol	1	The current protocol.

3.3.4.12 ZIDAdaptor_GetLocalAttribute.Req

Description

Get a local attribute from the Adaptor device.

Parameters

Table 3-14. ZIDAdaptor_GetLocalAttribute.Req Parameters

Parameter	Size (Bytes)	Comments
OpGroup	1	0xE2
OpCode	1	0xC2

Table 3-14. ZIDAdaptor_GetLocalAttribute.Req Parameters

Length	1	Length in bytes of the following parameters
attrId	1	Attribute ID. Possible values: 0xA0: gZIDAttrId_ZIDProfileVersion_c (ZID Profile Version) 0xA1: gZIDAttrId_IntPipeUnsafeTxWindowTime_c (The Interrupt pipe unsafe transmissions window duration) 0xE0: gZIDAttrId_ZIDParserVersion_c (The ZID Parser Version) 0xE3: gZIDAttrId_ZIDCountryCode_c (The ZID Country Code) 0xE4: gZIDAttrId_ZIDDeviceReleaseNumber_c (The ZID Device Release Number) 0xE5: gZIDAttrId_ZIDVendorId_c (The ZID Vendor ID) 0xE6: gZIDAttrId_ZIDProductId_c (The ZID Product ID)

3.3.4.13 ZIDAdaptor_GetLocalAttribute.Confirm

Description

The confirm for ZIDAdaptor_GetLocalAttribute.Req command.

Parameters

Table 3-15. ZIDAdaptor_GetLocalAttribute.Confirm Parameters

Parameter	Size (Bytes)	Comments
OpGroup	1	0xE3
OpCode	1	0xC1
Length	1	Length in bytes of the following parameters
Status	1	The returned status. Possible values: 0x00: gNWSuccess_c (Success) 0xf4: gNWUnsupportedAttribute_c (Unsupported Attribute)
AttrSize	1	Attribute Size.
AttrValue	AttrSize	Attribute Value.

3.3.4.14 ZIDAdaptor_GetNonStandardDescComp.Req

Description

Get non-standard descriptor component of a connected device.

Parameters

Table 3-16. ZIDAdaptor_GetNonStandardDescComp.Req Parameters

Parameter	Size (Bytes)	Comments
OpGroup	1	0xE2

Table 3-16. ZIDAdaptor_GetNonStandardDescComp.Req Parameters

OpCode	1	0xC8
Length	1	Length in bytes of the following parameters
DeviceID	1	The device ID.
Report non-standard Desc Attr ID:	1	What attribute ID describe the non-standard report. Possible values: 0xF1 - 0xF4

3.3.4.15 ZIDAdaptor_GetNonStandardDescComp.Confirm

Description

The confirm for ZIDAdaptor_GetNonStandardDescComp.Req command.

Parameters

Table 3-17. ZIDAdaptor_GetNonStandardDescComp.Confirm Parameters

Parameter	Size (Bytes)	Comments
OpGroup	1	0xE3
OpCode	1	0xC4
Length	1	Length in bytes of the following parameters
Status	1	The returned status. Possible values: 0x00: gNWSuccess_c (Success) 0xe8: gNWInvalidParam_c (Invalid Parameter)
DeviceID	1	DeviceID.
NonStdDescCompAttrID:	1	What attribute ID describe this non-standard descriptor component. Possible values: 0xF1 -0xF4
DescriptorType	1	Descriptor type. Possible values: 0x22: gZIDDescType_Report_c (Report type) 0x23: gZIDDescType_Physical_c (Physical type)
DescriptorSize	2	Descriptor size.
DescriptorID	1	Descriptor ID.
Non-standard Descriptor	DescriptorSize	Non-standard descriptor.

3.3.4.16 ZIDAdaptor_GetReport.Req

Description

Get a report from a remote node.

Parameters

Table 3-18. ZIDAdaptor_GetReport.Req Parameters

Parameter	Size (Bytes)	Comments
OpGroup	1	0xE2
OpCode	1	0xC5
Length	1	Length in bytes of the following parameters
DstDeviceID	1	The destination device ID.
DataPendingFlag	1	Signal the ZID Class device that the node has data pending. Possible values: 0x00: FALSE 0x01: TRUE
ReportType	1	Report Type. Possible values: 0x01: gZIDReportType_Input_c (Input Report) 0x02: gZIDReportType_Output_c (Output Report) 0x03: gZIDReportType_Feature_c (Feature report)
ReportId	1	The Report ID.

3.3.4.17 ZIDAdaptor_GetReport.Confirm

Description

The confirm for ZIDAdaptor_GetReport.Req command.

Parameters

Table 3-19. ZIDAdaptor_GetReport.Confirm Parameters

Parameter	Size (Bytes)	Comments
OpGroup	1	0xE3
OpCode	1	0x0B
Length	1	Length in bytes of the following parameters
Status	1	The returned status specifies the ZID Handshake Response code field OR any NWK status. Possible values: 0x00: gZIDResp_Success_c (Successy) 0x01: gZIDResp_UnsupportedRequest_c (UnsupportedRequest) 0x02: gZIDResp_InvalidParameter_c (Invalid Parameter) 0x40: gZIDResp_InvalidReportId_c (Invalid ReportId)
DeviceID	1	The deviceID confirmed.
ReportDataSize	1	The report data size.
ReportData	ReportDataSize	The report data.

3.3.4.18 ZIDAdaptor_Heartbeat.Indication

Description

This message is generate by ZID profile when a HeartBeat command frame was received over the air.

Parameters

Table 3-20. ZIDAdaptor_Heartbeat.Indication Parameters

Parameter	Size (Bytes)	Comments
OpGroup	1	0xE3
OpCode	1	0x09
Length	1	Length in bytes of the following parameters
DeviceID	1	From where the the list of report records is received.
ProfileId	1	The identifier of the profile indicating the format of the received data.
VendorId	2	If the RxFlags parameter specifies that the data is vendor specific, this parameter specifies the vendor identifier. If the RxFlags parameter specifies that the data is not vendor specific this parameter is ignored.
RxLinkQuality	1	LQI value measured during the reception of the NPDU.
RxFlags	1	Reception indication flags: 0x01 - broadcast mask; 0x02 - use security mask; 0x80 - HRC bit is set.

3.3.4.19 ZIDAdaptor_PBPConfig.Req

Description

Start the ZID Adaptor pairing and configuration phase.

Parameters

Table 3-21. ZIDAdaptor_PBPConfig.Req Parameters

Parameter	Size (Bytes)	Comments
OpGroup	1	0xE2
OpCode	1	0xC1
Length	1	Length in bytes of the following parameters
OrigAppCapabilities_UserStringSpecified	1	The application capabilities of this node: user string specified. Possible values: 0x01: UserStringIncludedInFrame 0x00: UserStringNotIncludedInFrame
OrigAppCapabilities_NoOfSupportedDeviceTypes	1	The application capabilities of this node: number of supported device types.
OrigAppCapabilities_NoOfSupportedProfiles	1	The application capabilities of this node: number of supported profiles.

Table 3-21. ZIDAdaptor_PBPConfig.Req Parameters

OrigDevTypeList	OrigAppCapabilities_NoOfSupportedDeviceTypes	The list of device types supported by this node.
OrigProfileIdList	OrigAppCapabilities_NoOfSupportedProfiles	The list of profile identifiers supported by this node.
DiscLQIThreshold	1	The Discovery LQI Threshold.
RequestAppAcceptToPair	1	Request the application to accept pairing. Possible values: 0x00: FALSE 0x01: TRUE
TimeToWaitPairInd	2	The amount of time in milliseconds the PBP layer will wait for a pair indication to arrive after the autoDiscovery part of the Push Button Pair process has completed successfully. If no pairIndication message arrives during this interval of time, the PBP layer will abort the Push Button Pair Recipient process with a 'No Response' error status.
TimeToWaitAppAcceptToPair	2	The amount of time in milliseconds the PBP layer will wait for the application to call PushButtonPairRecipContinueResponse() service after previously receiving the PBP_PushButtonPairRecipContinueInd message. If no call to the PushButtonPairRecipContinueResponse() is done by the application during this interval of time, the PBP layer will abort the Push Button Pair Recipient process with a 'No Response' error status.

3.3.4.20 ZIDAdaptor_PushAttr.Indication

Description

This message is generated by the ZID profile layer when a Push Attributes frame is received over the air.

Parameters

Table 3-22. ZIDAdaptor_PushAttr.Indication Parameters

Parameter	Size (Bytes)	Comments
OpGroup	1	0xE3
OpCode	1	0x0A
Length	1	Length in bytes of the following parameters
DeviceID	1	from where the The list of report records is received.
ProfileId	1	The identifier of the profile indicating the format of the received data.
VendorId	2	If the RxFlags parameter specifies that the data is vendor specific, this parameter specifies the vendor identifier. If the RxFlags parameter specifies that the data is not vendor specific this parameter is ignored.
RxLinkQuality	1	LQI value measured during the reception of the NPDU.

Table 3-22. ZIDAdaptor_PushAttr.Indication Parameters

RxFlags	1	Reception indication flags: 0x01 - broadcast mask; 0x02 - use security mask; 0x80 - HRC bit is set.
AttrRecordsLength	1	The number of octets contained by payload (the attributes records List).
AttrRecordsList	AttrRecordsLength	The attributes records List.

3.3.4.21 ZIDAdaptor_RemoveConfiguredDevice.Req

Description

Remove a configured remote node.

Parameters

Table 3-23. ZIDAdaptor_RemoveConfiguredDevice.Req Parameters

Parameter	Size (Bytes)	Comments
OpGroup	1	0xE2
OpCode	1	0xC7
Length	1	Length in bytes of the following parameters
DeviceID	1	The device ID.

3.3.4.22 ZIDAdaptor_RemoveConfiguredDevice.Confirm

Description

The confirm for ZIDAdaptor_RemoveConfiguredDevice.Req command.

Parameters

Table 3-24. ZIDAdaptor_RemoveConfiguredDevice.Confirm Parameters

Parameter	Size (Bytes)	Comments
OpGroup	1	0xE3
OpCode	1	0xC3
Length	1	Length in bytes of the following parameters
Status	1	The returned status. Possible values: 0x00: gNWSuccess_c (Success) 0xE8: gNWInvalidParam_c (Invalid Param)

3.3.4.23 ZIDAdaptor_ReportData.Req

Description

Send a report data over the air.

Parameters

Table 3-25. ZIDAdaptor_ReportData.Req Parameters

Parameter	Size (Bytes)	Comments
OpGroup	1	0xE2
OpCode	1	0xCB
Length	1	Length in bytes of the following parameters
DstDeviceID	1	The destination device Id.
TxOptions	1	Masks: 0x04 - Control Pipe (unicast with ACK, multiple channel) mask; 0x01 - Control Pipe Broadcast mask; 0x10 - Interrupt pipe mask; 0x08 - Use Security mask; 0x80 - Set data pending bit mask.
ReportRecordsListSize	1	The size of the report records .
ReportRecordsList	ReportRecords ListSize	The list of report records.

3.3.4.24 ZIDAdaptor_SetDataPending.Req

Description

Set data pending for a connection. This will signal the ZID profile that the application has data to be sent over the air.

Parameters

Table 3-26. ZIDAdaptor_SetDataPending.Req Parameters

Parameter	Size (Bytes)	Comments
OpGroup	1	0xE2
OpCode	1	0xCD
Length	1	Length in bytes of the following parameters
Connection Index	1	The connection index .
Data pending flag	1	Set the data pending flag. Possible values: 0x00: FALSE (The adaptor has no data pending) 0x01: TRUE (The adaptor has data pending)

3.3.4.25 ZIDAdaptor_SetDataPending.Req.Confirm

Description

The confirm for ZIDAdaptor_SetDataPending.Req command.

Parameters

Table 3-27. ZIDAdaptor_SetDataPending.Req.Confirm Parameters

Parameter	Size (Bytes)	Comments
OpGroup	1	0xE3
OpCode	1	0xC8
Length	1	Length in bytes of the following parameters
Status	1	The returned status. Possible values: 0x00: gNWSuccess_c (Success) 0xe8: gNWInvalidParam_c (Invalid parameter)

3.3.4.26 ZIDAdaptor_SetLocalAttribute.Req

Description

Set a local attribute for the Adaptor device.

Parameters

Table 3-28. ZIDAdaptor_SetLocalAttribute.Req Parameters

Parameter	Size (Bytes)	Comments
OpGroup	1	0xE2
OpCode	1	0xC3
Length	1	Length in bytes of the following parameters
attrID	1	attribute ID. Possible values: 0xA0: gZIDAttrId_ZIDProfileVersion_c (ZID Profile Version) 0xA1: gZIDAttrId_IntPipeUnsafeTxWindowTime_c (The Interrupt pipe unsafe transmissions window duration) 0xE0: gZIDAttrId_ZIDParserVersion_c (The ZID Parser Version) 0xE3: gZIDAttrId_ZIDCountryCode_c (The ZID Country Code) 0xE4: gZIDAttrId_ZIDDeviceReleaseNumber_c (The ZID Device Release Number) 0xE5: gZIDAttrId_ZIDVendorId_c (The ZID Vendor ID) 0xE6: gZIDAttrId_ZIDProductId_c (The ZID Product ID)
AttrValue	Variable	The value to set the attribute to. "Union" type parameter. Its structure is based on the value of parameter attrID. See detailed table below for parameter structure.

Table 3-29. AttrValue Parameter Structure

attrID	Structure Parameter	Size (bytes)	Comments
0xA0	gZIDAttrId_ZIDProfileVersion_c	2	The ZID Profile version.
0xA1	gZIDAttrId_IntPipeUnsafeTxWindowTime_c	2	The Interrupt pipe unsafe transmissions window duration.
0xE0	gZIDAttrId_ZIDParserVersion_c	2	ZID Parser Version.
0xE3	gZIDAttrId_ZIDCountryCode_c	1	ZID Country Code.
0xE4	gZIDAttrId_ZIDDeviceReleaseNumber_c	2	ZID Device Release Number.
0xE5	gZIDAttrId_ZIDVendorId_c	2	ZID Vendor ID.
0xE6	gZIDAttrId_ZIDProductId_c	2	ZID Product ID.

3.3.4.27 ZIDAdaptor_SetLocalAttribute.Confirm

Description

The confirm for ZIDAdaptor_SetLocalAttribute.Req command.

Parameters

Table 3-30. ZIDAdaptor_SetLocalAttribute.Confirm Parameters

Parameter	Size (Bytes)	Comments
OpGroup	1	0xE3
OpCode	1	0xC2
Length	1	Length in bytes of the following parameters
Status	1	The returned status. Possible values: 0x00: gNWSuccess_c (Success) 0xf4: gNWUnsupportedAttribute_c (Unsupported Attribute)

3.3.4.28 ZIDAdaptor_SetReport.Req

Description

Set a report on a remote node.

Parameters

Table 3-31. ZIDAdaptor_SetReport.Req Parameters

Parameter	Size (Bytes)	Comments
OpGroup	1	0xE2
OpCode	1	0xC6

Table 3-31. ZIDAdaptor_SetReport.Req Parameters

Length	1	Length in bytes of the following parameters
DstDeviceID	1	The destination device Id.
DataPendingFlag	1	Signal the ZID Class device that the node has data pending. Possible values: 0x00: FALSE 0x01: TRUE
ReportType	1	Report Type. Possible values: 0x01: gZIDReportType_Input_c (Input Report) 0x02: gZIDReportType_Output_c (Output Report) 0x03: gZIDReportType_Feature_c (Feature report)
ReportId	1	The Report Id.
ReportDataSize	1	The size of the report data.
ReportData	ReportDataSize	The report data.

3.3.4.29 ZIDAdaptor_SetReport.Confirm

Description

The confirm for ZIDAdaptor_SetReport.Req command.

Parameters

Table 3-32. ZIDAdaptor_SetReport.Confirm Parameters

Parameter	Size (Bytes)	Comments
OpGroup	1	0xE3
OpCode	1	0x0C
Length	1	Length in bytes of the following parameters
Status	1	The returned status specifies the ZID Handshake Response code field OR any NWK status. Possible values: 0x00: gZIDResp_Success_c (Successy) 0x01: gZIDResp_UnsupportedRequest_c (Unsupported Request) 0x02: gZIDResp_InvalidParameter_c (Invalid Parameter) 0x40: gZIDResp_InvalidReportId_c (Invalid Report ID)
DeviceID	1	The deviceID confirmed.

3.3.4.30 ZIDAdaptor_StartWithNVM.Req

Description

Start the ZID Adaptor with NVM data.

Parameters

Table 3-33. ZIDAdaptor_StartWithNVM.Req Parameters

Parameter	Size (Bytes)	Comments
OpGroup	1	0xE2
OpCode	1	0xC0
Length	1	0x00 - This message does not have any parameters

3.3.4.31 ZIDAdaptor_StartWithNVM.Confirm

Description

The confirm for ZIDAdaptor_StartWithNVM.Req.

Parameters

Table 3-34. ZIDAdaptor_StartWithNVM.Confirm Parameters

Parameter	Size (Bytes)	Comments
OpGroup	1	0xE3
OpCode	1	0xC0
Length	1	Length in bytes of the following parameters
Status	1	status.

3.3.4.32 ZIDClassDevice_AbortProcess.Req

Description

Abort the current running process.

Parameters

Table 3-35. ZIDClassDevice_AbortProcess.Req Parameters

Parameter	Size (Bytes)	Comments
OpGroup	1	0xE2
OpCode	1	0x87
Length	1	0x00 - This message does not have any parameters

3.3.4.33 ZIDClassDevice_AbortProcess.Confirm

Description

The confirm for ZIDClassDevice_AbortProcess.Req command.

Parameters

Table 3-36. ZIDClassDevice_AbortProcess.Confirm Parameters

Parameter	Size (Bytes)	Comments
OpGroup	1	0xE3
OpCode	1	0x85
Length	1	Length in bytes of the following parameters
Status	1	The returned status. Possible values: 0x00: gNWSuccess_c (Success) 0x80: gNWDenied_c (Denied)

3.3.4.34 ZIDClassDevice_CompatibilityCheck.Indication

Description

This message is received during the configuration phase to check the compatibility. It contains the Get Attributes response received from the ZID Adaptor device.

Parameters

Table 3-37. ZIDClassDevice_CompatibilityCheck.Indication Parameters

Parameter	Size (Bytes)	Comments
OpGroup	1	0xE3
OpCode	1	0x04
Length	1	Length in bytes of the following parameters
Status	1	The returned status specifies the ZID Generic Response code field or any NWK status. Possible values: 0x00: gZIDResp_Success_c (Success) 0x01: gZIDResp_UnsupportedRequest_c (Unsupported Request) 0x02: gZIDResp_InvalidParameter_c (Invalid Parameter) 0x40: gZIDResp_InvalidReportId_c (Invalid Report ID)
DeviceID	1	The deviceID confirmed.
AttrRspRecordsSize	1	The Attribute Response payload size.
AttrRspRecordsList	AttrRspRecords Size	Attribute records list.

3.3.4.35 ZIDClassDevice_CompatibilityCheckResp.Req

Description

Send the compatibility check response during the configuration phase.

Parameters

Table 3-38. ZIDClassDevice_CompatibilityCheckResp.Req Parameters

Parameter	Size (Bytes)	Comments
OpGroup	1	0xE2
OpCode	1	0x8D
Length	1	Length in bytes of the following parameters
ContinueConnect	1	Continue to Connect. Possible values: 0x00: FALSE 0x01: TRUE

3.3.4.36 ZIDClassDevice_CompatibilityCheckResp.Confirm

Description

The confirm for ZIDClassDevice_CompatibilityCheckResp.Req command.

Parameters

Table 3-39. ZIDClassDevice_CompatibilityCheckResp.Confirm Parameters

Parameter	Size (Bytes)	Comments
OpGroup	1	0xE3
OpCode	1	0x88
Length	1	Length in bytes of the following parameters
Status	1	The returned status. Possible values: 0x00: gNWSuccess_c (Success) 0x80: gNWDenied_c (Denied)

3.3.4.37 ZIDClassDevice_ConfigureReportData.Req

Description

Add and configure a local report data.

Parameters

Table 3-40. ZIDClassDevice_ConfigureReportData.Req Parameters

Parameter	Size (Bytes)	Comments
OpGroup	1	0xE2
OpCode	1	0x88
Length	1	Length in bytes of the following parameters

Table 3-40. ZIDClassDevice_ConfigureReportData.Req Parameters

EntryIndex	1	What report entry to set.
ReportType	1	Report Type. Possible values: 0x01: gZIDReportType_Input_c (Input Report) 0x02: gZIDReportType_Output_c (Output Report) 0x03: gZIDReportType_Feature_c (Feature report)
ReportId	1	Set the Report Id.
Offset in Reports data pool	1	Offset in Reports data pool.
ReportDataSize	1	The size of the report data.
ReportData	ReportDataSize	The report data.

3.3.4.38 ZIDClassDevice_ConfigureReportData.Confirm

Description

The confirm for ZIDClassDevice_ConfigureReportData.Req command.

Parameters

Table 3-41. ZIDClassDevice_ConfigureReportData.Confirm Parameters

Parameter	Size (Bytes)	Comments
OpGroup	1	0xE3
OpCode	1	0x86
Length	1	Length in bytes of the following parameters
Status	1	The returned status. Possible values: 0x00: gNWSuccess_c (Success) 0xf4: gNWUnsupportedAttribute_c (Unsupported Attribute) 0xE8: gNWInvalidParam_c (Invalid Parameter)

3.3.4.39 ZIDClassDevice_DeviceIsIdle.Req

Description

Checks if the device Is Idle.

Parameters

Table 3-42. ZIDClassDevice_DeviceIsIdle.Req Parameters

Parameter	Size (Bytes)	Comments
OpGroup	1	0xE2

Table 3-42. ZIDClassDevice_DeviceIsIdle.Req Parameters

OpCode	1	0x86
Length	1	0x00 - This message does not have any parameters

3.3.4.40 ZIDClassDevice_DeviceIsIdle.Confirm

Description

The confirm for ZIDClassDevice_DeviceIsIdle.Req command.

Parameters

Table 3-43. ZIDClassDevice_DeviceIsIdle.Confirm Parameters

Parameter	Size (Bytes)	Comments
OpGroup	1	0xE3
OpCode	1	0x84
Length	1	Length in bytes of the following parameters
IdleState	1	The returned status. Possible values: 0x00: FALSE (Device is not in idle state) 0x01: TRUE (Device is in idle state)

3.3.4.41 ZIDClassDevice_GetAttributes.Req

Description

Get HID attributes from a remote node.

Parameters

Table 3-44. ZIDClassDevice_GetAttributes.Req Parameters

Parameter	Size (Bytes)	Comments
OpGroup	1	0xE2
OpCode	1	0x8C
Length	1	Length in bytes of the following parameters
DstDeviceID	1	The destination device ID.
AttrIDsListLength	1	The length of attributes IDs list.
AttrIDsList	AttrIDsListLength	The attribute IDs List.

3.3.4.42 ZIDClassDevice_GetConfiguredReportData.Req

Description

Get the local report data information.

Parameters

Table 3-45. ZIDClassDevice_GetConfiguredReportData.Req Parameters

Parameter	Size (Bytes)	Comments
OpGroup	1	0xE2
OpCode	1	0x89
Length	1	Length in bytes of the following parameters
EntryIndex	1	Entry Index.

3.3.4.43 ZIDClassDevice_GetConfiguredReportData.Confirm

Description

The confirm for ZIDClassDevice_GetConfiguredReportData.Req command.

Parameters

Table 3-46. ZIDClassDevice_GetConfiguredReportData.Confirm Parameters

Parameter	Size (Bytes)	Comments
OpGroup	1	0xE3
OpCode	1	0x87
Length	1	Length in bytes of the following parameters
Status	1	The returned status. Possible values: 0x00: gNWSuccess_c (Success) 0xf4: gNWUnsupportedAttribute_c (Unsupported Attribute) 0xE8: gNWInvalidParam_c (Invalid Parameter)
EntryIndex	1	Entry Index.
ReportType	1	ReportType. Possible values: 0x01: gZIDReportType_Input_c (Input Report) 0x02: gZIDReportType_Output_c (Output Report) 0x03: gZIDReportType_Feature_c (Feature report)
ReportId	1	The Report ID.
ReportDataSize	1	The report data size.
ReportDataPayload	ReportDataSize	The report data payload.

3.3.4.44 ZIDClassDevice_GetLocalAttribute.Req

Description

Get a local attribute from the ZID Class device.

Parameters

Table 3-47. ZIDClassDevice_GetLocalAttribute.Req Parameters

Parameter	Size (Bytes)	Comments
OpGroup	1	0xE2
OpCode	1	0x82
Length	1	Length in bytes of the following parameters
attrId	1	Attribute ID.

3.3.4.45 ZIDClassDevice_GetLocalAttribute.Confirm

Description

The confim for ZIDClassDevice_GetLocalAttribute.Req command.

Parameters

Table 3-48. ZIDClassDevice_GetLocalAttribute.Confirm Parameters

Parameter	Size (Bytes)	Comments
OpGroup	1	0xE3
OpCode	1	0x81
Length	1	Length in bytes of the following parameters
Status	1	The returned status. Possible values: 0x00: gNWSuccess_c (Success) 0xF4: gNWUnsupportedAttribute_c (Unsupported Attribute)
AttributeSize	1	The attribute size.
AttributeValue	AttributeSize	Attribute Value.

3.3.4.46 ZIDClassDevice_GetNonStdNULLReportData.Req

Description

Get the non-standard NULL report Data.

Parameters

Table 3-49. ZIDClassDevice_GetNonStdNULLReportData.Req Parameters

Parameter	Size (Bytes)	Comments
OpGroup	1	0xE2
OpCode	1	0x90
Length	1	Length in bytes of the following parameters
EntryIndex	1	What NULL report entry to get.

3.3.4.47 ZIDClassDevice_GetNonStdNULLReportData.Confirm

Description

The confirm for ZIDClassDevice_GetNonStdNULLReportData.Req command.

Parameters

Table 3-50. ZIDClassDevice_GetNonStdNULLReportData.Confirm Parameters

Parameter	Size (Bytes)	Comments
OpGroup	1	0xE3
OpCode	1	0x8A
Length	1	Length in bytes of the following parameters
Status	1	The returned status. Possible values: 0x00: gNWSuccess_c (Success) 0xf4: gNWUnsupportedAttribute_c (Unsupported Attribute) 0xE8: gNWInvalidParam_c (Invalid Parameter)
EntryIndex	1	Entry Index.
NULLReportId	1	The NULL Report Id.
NULLReportDataSize	1	The report data size.
NULLReportDatPayload	NULLReportDataSize	The NULL report data payload.

3.3.4.48 ZIDClassDevice_Heartbeat.Req

Description

Send a Heartbeat command frame over the air.

Parameters

Table 3-51. ZIDClassDevice_Heartbeat.Req Parameters

Parameter	Size (Bytes)	Comments
OpGroup	1	0xE2
OpCode	1	0x8B
Length	1	Length in bytes of the following parameters
DstDeviceID	1	The destination device Id.

3.3.4.49 ZIDClassDevice_Heartbeat.Confirm

Description

The confirm for ZIDClassDevice_Heartbeat.Req command.

Parameters

Table 3-52. ZIDClassDevice_Heartbeat.Confirm Parameters

Parameter	Size (Bytes)	Comments
OpGroup	1	0xE3
OpCode	1	0x07
Length	1	Length in bytes of the following parameters
Status	1	The status of the Request. Possible values: any RF4CE network status.
DeviceID	1	The deviceID confirmed.

3.3.4.50 ZIDClassDevice_PBPConfig.Req

Description

Start the ZID class device pairing process and cofiguration phase.

Parameters

Table 3-53. ZIDClassDevice_PBPConfig.Req Parameters

Parameter	Size (Bytes)	Comments
OpGroup	1	0xE2
OpCode	1	0x81
Length	1	Length in bytes of the following parameters
RecipPanId	2	The PAN identifier of the device with which to pair. Set to 0xFFFF to indicate a wildcard.

Table 3-53. ZIDClassDevice_PBPConfig.Req Parameters

RecipShortAddress	2	The recipient short address of the device with which to pair. Set to 0xFFFF to indicate a wildcard.
RecipDeviceType	1	The recipent device type of the device with which to pair.
OrigAppCapabilities_UserStringSpecified	1	The application capabilities of this node: user string specified. Possible values: 0x01: UserStringIncludedInFrame 0x00: UserStringNotIncludedInFrame
OrigAppCapabilities_NoOfSupportedDeviceTypes	1	The application capabilities of this node: number of supported device types.
OrigAppCapabilities_NoOfSupportedProfiles	1	The application capabilities of this node: number of supported.
OrigDevTypeList	OrigAppCapabilities_NoOfSupportedDeviceTypes	The list of device types supported by this node.
OrigProfileIdList	OrigAppCapabilities_NoOfSupportedProfiles	The list of profile identifiers supported by this node.
DiscProfileIdListSize	1	The discovery ProfileIds List size.
DiscProfileIdList	DiscProfileIdListSize	The discovery ProfileIds List.
KeyExTransferCount	1	The number of transfers the target should use to exchange the link key with the pairing originator.
RequestAppAcceptToPair	1	Request the application to accept pairing. Possible values: 0x00: FALSE 0x01: TRUE
TimeToWaitAppAcceptToPair	2	The amount of time in milliseconds the Push Button Pair layer will wait for the application to call PushButtonPairOrigContinueResponse() service after previously receiving the PushButtonPairOrigContinueInd message. If no call to the PushButtonPairOrigContinueResponse() is done by the application during this interval of time, the Push Button Pair layer will abort the Push Button Pair Originator process with a 'No Response' error status.

3.3.4.51 ZIDClassDevice_PushAttr.Req

Description

Send a Push attribute request.

Parameters

Table 3-54. ZIDClassDevice_PushAttr.Req Parameters

Parameter	Size (Bytes)	Comments
OpGroup	1	0xE2

Table 3-54. ZIDClassDevice_PushAttr.Req Parameters

OpCode	1	0x8A
Length	1	Length in bytes of the following parameters
DstDeviceID	1	The destination device Id.
AttrIDsListLength	1	The length of attributes IDs list.
AttrIDsList	AttrIDsListLength	The attribute IDs List.

3.3.4.52 ZIDClassDevice_PushAttr.Confirm

Description

The confirm for ZIDClassDevice_PushAttr.Req command.

Parameters

Table 3-55. ZIDClassDevice_PushAttr.Confirm Parameters

Parameter	Size (Bytes)	Comments
OpGroup	1	0xE3
OpCode	1	0x06
Length	1	Length in bytes of the following parameters
Status	1	The status of the Request. Possible values: any RF4CE network status.
DeviceID	1	The deviceID confirmed.

3.3.4.53 ZIDClassDevice_RemoveConfiguredDevice.Req

Description

Remove a configured remote node.

Parameters

Table 3-56. ZIDClassDevice_RemoveConfiguredDevice.Req Parameters

Parameter	Size (Bytes)	Comments
OpGroup	1	0xE2
OpCode	1	0x85
Length	1	Length in bytes of the following parameters
DeviceID	1	The device Id.

3.3.4.54 ZIDClassDevice_RemoveConfiguredDevice.Confirm

Description

The confirm for ZIDClassDevice_RemoveConfiguredDevice.Req command.

Parameters

Table 3-57. ZIDClassDevice_RemoveConfiguredDevice.Confirm Parameters

Parameter	Size (Bytes)	Comments
OpGroup	1	0xE3
OpCode	1	0x83
Length	1	Length in bytes of the following parameters
Status	1	The returned status. Possible values: 0x00: gNWSuccess_c (Success) 0xE8: gNWInvalidParam_c (Invalid Param)

3.3.4.55 ZIDClassDevice_ReportData.Req

Description

Send a report data over-the-air.

Parameters

Table 3-58. ZIDClassDevice_ReportData.Req Parameters

Parameter	Size (Bytes)	Comments
OpGroup	1	0xE2
OpCode	1	0x84
Length	1	Length in bytes of the following parameters
DstDeviceID	1	The destination device Id.
TxOptions	1	Masks: 0x04 - Control Pipe (unicast with ACK, multiple channel) mask; 0x01 - Control Pipe Broadcast mask; 0x10 - Interrupt pipe mask; 0x08 - Use Security mask; 0x80 - Set data pending bit mask.
ReportRecordsListSize	1	The size of the list of the report records.
ReportRecordsList	ReportRecords ListSize	The list of report records.

3.3.4.56 ZIDClassDevice_SendReportIdsList.Req

Description

Send a list of report IDs to form and send the report records.

Parameters

Table 3-59. ZIDClassDevice_SendReportIdsList.Req Parameters

Parameter	Size (Bytes)	Comments
OpGroup	1	0xE2
OpCode	1	0x8E
Length	1	Length in bytes of the following parameters
DstDeviceID	1	The destination device Id.
Action	1	Select to send the built report frame only once, repeat it or stop repeating it . Possible values: 0x00: gSendReportFrameOnlyOnce_c (Send the report frame only once) 0x01: gRepeatReportFrame_c (Repeat the report frame) 0x02: gStopRepeatReportFrame_c (Stop repeating the report frame)
TxOptions	1	Masks: 0x04 - Control Pipe (unicast with ACK, multiple channel) mask; 0x01 - Control Pipe Broadcast mask; 0x10 - Interrupt pipe mask; 0x08 - Use Security mask; 0x80 - Set data pending bit mask.
ReportIDsListSize	1	The size of the report IDs list.
ReportIDsList	ReportIDsListSize	The list of report IDs.

3.3.4.57 ZIDClassDevice_SendReportIdsList.Confirm

Description

The confirm for ZIDClassDevice_SendReportIdsList.Req command.

Parameters

Table 3-60. ZIDClassDevice_SendReportIdsList.Confirm Parameters

Parameter	Size (Bytes)	Comments
OpGroup	1	0xE3
OpCode	1	0x08
Length	1	Length in bytes of the following parameters
Status	1	The status of the Request. Possible values: any RF4CE network status.
DeviceID	1	The deviceID confirmed.

3.3.4.58 ZIDClassDevice_SetLocalAttribute.Req

Description

Set a local attribute for the ZID Class device.

Parameters

Table 3-61. ZIDClassDevice_SetLocalAttribute.Req Parameters

Parameter	Size (Bytes)	Comments
OpGroup	1	0xE2
OpCode	1	0x83
Length	1	Length in bytes of the following parameters
attrID	1	attribute ID. Possible values: ZID Class device attributes IDs (refer to next tables).
AttrValue	Variable	The value to set the attribute to. "Union" type parameter. Its structure is based on the value of parameter attrID. See detailed table below for parameter structure.

Table 3-62. AttrValue Parameter Structure

attrID	Parameter	Size (Bytes)	Comments
0xA0	gZIDAttrId_ZIDProfileVersion_c	2	The ZID Profile version.
0xA1	gZIDAttrId_IntPipeUnsafeTxWindowTime_c	2	The Interrupt pipe safe transmissions window duration.
0xA2	gZIDAttrId_ZIDReportRepeatInterval_c	1	Report Repeat Interval.
0xE0	gZIDAttrId_ZIDParserVersion_c	2	ZID Parser Version.
0xE1	gZIDAttrId_ZIDDeviceSubClass_c	1	ZID Device SubClass.
0xE2	gZIDAttrId_ZIDProtocolCode_c	1	ZID Protocol Code.
0xE3	gZIDAttrId_ZIDCountryCode_c	1	ZID Country Code.
0xE4	gZIDAttrId_ZIDDeviceReleaseNumber_c	2	ZID Device Release Number.
0xE5	gZIDAttrId_ZIDVendorId_c	2	ZID Vendor ID.
0xE6	gZIDAttrId_ZIDProductId_c	2	ZID Product ID.
0xE7	gZIDAttrId_ZIDNumEndpoints_c	1	Number of endpoints.
0xE8	gZIDAttrId_ZIDPollInterval_c	1	Poll Interval.
0xE9	gZIDAttrId_ZIDNumStdDescComps_c	1	Number of standard descriptor components.
0xEA	gZIDAttrId_ZIDStdDescCompsList_c	Variable	The standard descriptors components list. Structure type parameter. See detailed table below for parameter structure.
0xEB	gZIDAttrId_ZIDNumNullReports_c	1	The number of non-standard NULL reports.
0xF0	gZIDAttrId_ZIDNonStdDescComps_c	1	Number of non-standard descriptor components.
0xF1	gZIDAttrId_ZIDNonStdDescCompSpec1_c	Variable	ZID non-standard descriptor component 1. Structure type parameter. See detailed table below for parameter structure.
0xF2	gZIDAttrId_ZIDNonStdDescCompSpec2_c	Variable	ZID non-standard descriptor component 2. Structure type parameter. See detailed table below for parameter structure.

Table 3-62. AttrValue Parameter Structure

attrID	Parameter	Size (Bytes)	Comments
0xF3	gZIDAttrId_ZIDNonStdDescCompSpec3_c	Variable	ZID non-standard descriptor component 3. Structure type parameter. See detailed table below for parameter structure.
0xF4	gZIDAttrId_ZIDNonStdDescCompSpec4_c	Variable	ZID non-standard descriptor component 4. Structure type parameter. See detailed table below for parameter structure.

Table 3-63. gZIDAttrId_ZIDStdDescCompsList_c Parameter Structure

Parameter	Size (Bytes)	Comments
NumberOfStandardDescriptor components	1	Number Of Standard Descriptor components.
StandardDescriptorList	NumberOfStandardDescriptor components	Standard descriptor components List.

Table 3-64. gZIDAttrId_ZIDNonStdDescCompSpecX_c Parameter Structure

Parameter	Size (Bytes)	Comments
OffsetInNonStandarddescriptor componentsPool	2	Offset in non-standard descriptors pool.
AttrLength	1	Attribute Length.
Attribute Value	Variable	Attribute Value. Structure type parameter. See detailed table below for parameter structure.

Table 3-65. Attribute Value Parameter Structure

Parameter	Size (Bytes)	Comments
DescriptorType	1	Descriptor Type. Possible values: 0x22: gZIDDescType_Report_c (Report type) 0x23: gZIDDescType_Physical_c (Physical type)
NonStandardDescriptorSize	2	Descriptor Size.
Parameter	Size (Bytes)	Comments
Non-standard Descriptor	NonStandardDescriptorSize	Non-standard Descriptor payload.

3.3.4.59 ZIDClassDevice_SetLocalAttribute.Confirm

Description

The confirm for ZIDClassDevice_SetLocalAttribute.req service.

Parameters

Table 3-66. ZIDClassDevice_SetLocalAttribute.Confirm Parameters

Parameter	Size (Bytes)	Comments
OpGroup	1	0xE3
OpCode	1	0x82
Length	1	Length in bytes of the following parameters
Status	1	The returned status. Possible values: 0x00: gNWSuccess_c (Success) 0xF4: gNWUnsupportedAttribute_c (Unsupported Attribute) 0xE8: gNWInvalidParam_c (Invalid Parameter)

3.3.4.60 ZIDClassDevice_SetNonStdNULLReportData.Req

Description

Set non-standard null report data.

Parameters

Table 3-67. ZIDClassDevice_SetNonStdNULLReportData.Req Parameters

Parameter	Size (Bytes)	Comments
OpGroup	1	0xE2
OpCode	1	0x8F
Length	1	Length in bytes of the following parameters
EntryIndex	1	What NULL report entry to set.
NULL Report Id	1	Set the non-standard NULL Report Id.
Offset in NULL Reports data pool	1	Offset in NULL Reports data pool.
NULLReportDataSize	1	The size of the NULL report data.
NULLReportData	NULLReportDataSize	The NULL report data.

3.3.4.61 ZIDClassDevice_SetNonStdNULLReportData.Confirm

Description

The confirm for ZIDClassDevice_SetNonStdNULLReportData.Req command.

Parameters

Table 3-68. ZIDClassDevice_SetNonStdNULLReportData.Confirm Parameters

Parameter	Size (Bytes)	Comments
OpGroup	1	0xE3
OpCode	1	0x89
Length	1	Length in bytes of the following parameters
Status	1	The returned status. Possible values: 0x00: gNWSuccess_c (Success) 0xf4: gNWUnsupportedAttribute_c (Unsupported Attribute) 0xE8: gNWInvalidParam_c (Invalid Parameter)

3.3.4.62 ZIDClassDevice_SetReport.Indication

Description

This message is generated by the ZID profile layer when a Set Report frame is received over the air.

Parameters

Table 3-69. ZIDClassDevice_SetReport.Indication Parameters

Parameter	Size (Bytes)	Comments
OpGroup	1	0xE3
OpCode	1	0x05
Length	1	Length in bytes of the following parameters
DeviceID	1	From where the the list of report records is received.
ProfileId	1	The identifier of the profile indicating the format of the received data.
VendorId	2	If the RxFlags parameter specifies that the data is vendor specific, this parameter specifies the vendor identifier. If the RxFlags parameter specifies that the data is not vendor specific this parameter is ignored.
RxLinkQuality	1	LQI value measured during the reception of the NPDU.
RxFlags	1	Reception indication flags for this NSDU.
SetReportDataSize	1	The number of octets contained by payload (Set Report Data).
SetReportData	SetReportData Size	Set Report payload.

3.3.4.63 ZIDClassDevice_StartWithNVM.Req

Description

Start the ZID class device with NVM data.

Parameters

Table 3-70. ZIDClassDevice_StartWithNVM.Req Parameters

Parameter	Size (Bytes)	Comments
OpGroup	1	0xE2
OpCode	1	0x80
Length	1	0x00 - This message does not have any parameters