SynkroRF Network

Reference Manual

Document Number: SYNKRORM
Rev. 1.4
06/2011

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

Contents

Chapter 3

SynkroRF Data Definitions

Chapter 4
SynkroRF Frame Format

About This Book

This reference manual describes in detail the Freescale SynkroRF Network which will allow users to develop upper layer or application code for this product.

The Freescale SynkroRF Network software is designed for use with the following families of short range, low power, 2.4 GHz Industrial, Scientific, and Medical (ISM) band transceivers:

- Freescale MC1319x and MC1320x families, designed for use with the HC(S)08 GT/GB MCU.
- Freescale MC1321x and MC1323x families, that incorporate a low power 2.4 GHz radio frequency transceiver and an 8-bit microcontroller into a single LGA package.
- Freescale MC1322x Platform-In-Package, that combines a low power 2.4 GHz frequency transceiver and a 32-bit ARM7 microcontroller into a single LGA package.
- Freescale MC1323x low cost System-on-Chip (SoC) platform for the IEEE® 802.15.4 Standard that incorporates a complete, low power, 2.4 GHz radio frequency transceiver with Tx/Rx switch, an 8-bit HCS08 CPU, and a functional set of MCU peripherals into a 48-pin QFN package.

Audience

This reference manual is intended for application designers and users of the SynkroRF Network library.

Organization

This document contains the following chapters:

Chapter 1	SynkroRF Software Overview – Provides an introduction to the SynkroRF Network.
Chapter 2	SynkroRF Software Usage – Provides a description of the network interfaces
Chapter 3	SynkroRF Data Definitions – Provides a brief description of the network used data structures
Chapter 4	SynkroRF Frames Format – Provides a description of the over the air command frames

Revision History

The following table summarizes revisions to this manual since the previous release (Rev. 1.3).

Revision History

Doc. Version	Date / Author	Description / Location of Changes
1.4	November 2009, Dev Team	Updates for CW 10 support

Definitions, Acronyms, and Abbreviations

The following list defines the abbreviations used in this document.

API	Application Programming Interface
CE	Consumer Electronics
LQI	Link Quality Indicator
NW Layer	Network Layer
PAN	Personal Area Network
Poll map	Map (40 bit array) in which each bit corresponds to a location in the Pair Table. Bit i is set when a poll request will be sent to device i from the Pair Table.
NHR	Network Header
CHR	Command Header
SOPT	Size of Pair Table

Chapter 1
SynkroRF Software Overview

This chapter presents a brief overview of the SynkroRF Network software.

1.1 SynkroRF Network Introduction

The SynkroRF 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, cost-effective solutions to be implemented for a wide range of applications. Some key characteristics of a SynkroRF network are:

- An over the air data rate of 250 kbit/s in the 2.4 GHz band.
- 3 independent communication channels in the 2.4 GHz band (15, 20, and 25).
- 2 network node types, controller and controlled nodes.
- Channel Agility mechanism.
- Low Latency Tx mode automatically enabled in conditions of radio interference.
- Fragmented mode transmissions and reception, automatically enabled in conditions of radio interference.
- Robustness and ease of use.
- Essential functionality to build and support a CE network.
- Figure 1-1 shows the software architecture of an application using the SynkroRF Network.

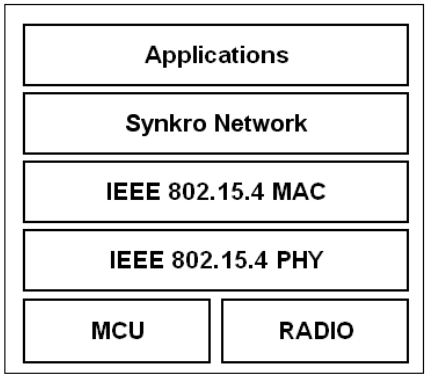

Figure 1-1. SynkroRF Network Application Structure

The SynkroRF Network defines a peer to peer network topology where communication can occur between any two controller & controlled nodes or between any two controlled nodes, if these nodes have been previously paired. Communication between controller nodes is also possible under the use of SynkroRF's

cloning service.The communication between nodes is performed exclusively using the IEEE 802.15.4 MAC Data Request primitives in non beacon mode.

As shown in Figure 1-2, if a node wants to join a SynkroRF Network, it must pair with a node that is already part of the network. Controller nodes can only be paired with controlled nodes, but controlled nodes can be paired with both controller and controlled nodes.

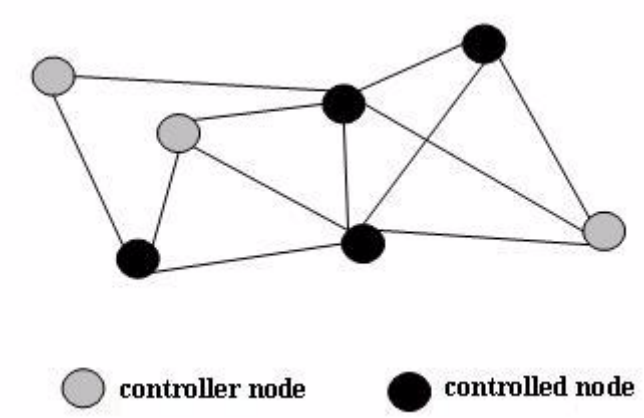

Figure 1-2. SynkroRF Network Topology

1.2 Node Types

The basic node types for the Freescale SynkroRF Network are as follows:

- **Controller Node -** This device contains a subset of the SynkroRF features. The controller node type is used in applications that intend to exercise control over other nodes in the network. It is the single node type that can initiate a pair, remote pair or clone process. This node does not start a PAN. During the pairing process, it receives a Pan Id and Short Address from the controlled node it has paired with. These values will be used in future communications with that controlled node.
- **Controlled Node -** This device contains a subset of the SynkroRF features. The controlled node type is intended for use with consumer electronic device applications (TVs, DVD players, etc.). It is intended to be used in devices that are powered by a continuous power source. Starting a controlled node will always start a PAN. A controlled node cannot initiate pair, remote pair or clone processes.

To allow users to better manage memory resources, Freescale provides both controller and controlled subsets as precompiled libraries.

Chapter 2
SynkroRF Software Usage

SynkroRF handles all access to the physical radio channel and is responsible for the following tasks:

- Supporting network formation by pairing and un-pairing between nodes.
- Supporting remote device pairing.
- Supporting device cloning.
- Supporting fragmented transmission mode.
- Supporting Low Latency transmission mode.
- Supporting Channel Agility mechanism
- Providing a reliable link between two paired SynkroRF entities.

NOTE

Constants that are specified and maintained by the SynkroRF layer are written in the text of this document in italics. Constants have a general prefix of "g" *gNWSuccess_c*

2.1 SynkroRF Service Specifications

SynkroRF implements an interface between the application and the IEEE.802.15.4 MAC layer includes a management entity that provides the service interfaces through which layer management functions may be invoked. This management entity is also responsible for maintaining a database of managed objects pertaining to the SynkroRF layer. This database is referred to as the SynkroRF NodeData Database. The SynkroRF system is shown in Figure 2-1.

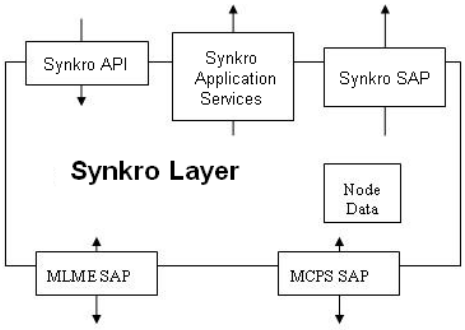

Figure 2-1. Components and interfaces of the SynkroRF layer

SynkroRF provides 36 services, accessed through the SynkroRF APIare listed in Table 2-1 and have the following characteristics:

- SynkroRF API function calls start the services

- The execution status is communicated to the application using confirm messages
- Network arrival information informs the application layer using indication messages
- Ask the application to validate certain actions using service requests implemented with callback functions

The SynkroRF SAP provides the interface for confirm and indication messages. The SynkroRF Application Services provides the interface for the callback functions.

Table 2-1. SynkroRF Services List

SynkroRF Service	Request	Confirm	Indication	Application Service Request
Synkro_Start	2.1.1	2.1.2		
Synkro_SearchRequest	2.1.3	2.1.4, 2.1.6		2.1.5
Synkro_PairRequest	2.1.7	2.1.8, 2.1.10		2.1.9
Synkro_PairRemoteDevices	2.1.11	2.1.12, 2.1.14		2.1.13
Synkro_CloneDevice	2.1.15	2.1.16, 2.1.18		2.1.17, 2.1.19
Synkro_SendCommand	2.1.20	2.1.21	2.1.22	
Synkro_SetBulkBufferState	2.1.23			
Synkro_GetBulkBufferState	2.1.24			
Synkro_SendBulkData	2.1.25	2.1.26	2.1.27, 2.1.28	
Synkro_PollConfig	2.1.29			
Synkro_PollDevice	2.1.30	2.1.31	2.1.32	
Synkro_PollDataAvailable	2.1.33			
Synkro_UpdateCapabilities	2.1.34	2.1.35	2.1.36	
Synkro_RefreshCapabilities	2.1.37	2.1.38	2.1.39	
Synkro_ClearPairingInformation	2.1.40			
Synkro_SetNewMACAddress	2.1.41	2.1.42		
Synkro_GetMACAddress	2.1.43			
Synkro_Sleep	2.1.44	2.1.45		
Synkro_Wake	2.1.46			
Synkro_SetReceiveMode	2.1.47			
Synkro_SetPowerLevel	2.1.48			
Synkro_IsFeatureSetAvailable	2.1.49			
Synkro_GetPairedDeviceCapabilities	2.1.50			
Synkro_GetPairedDeviceInfo	2.1.51			
Synkro_GetLocalNodeInfo	2.1.52			
Synkro_GenerateNewShortAddress	2.1.53			

Table 2-1. SynkroRF Services List

Synkro_GenerateNewSecurityKey	2.1.54			
Synkro_AddEntryInControllerPairTable	2.1.55			
Synkro_AddEntryInControlledPairTable	2.1.56			
Synkro_SavePersistentDataInFlash	2.1.57			
Synkro_GetLastLQI	2.1.58			
Synkro_SetSearchThreshold	2.1.59			
Synkro_SetPairingThreshold	2.1.60			
Synkro_SetCloningThreshold	2.1.61			
Synkro_GetNwkStatus	2.1.62			
Synkro_IsIdle	2.1.63			

2.1.1 Synkro_Start Request

The Synkro_Start service makes a request for a SynkroRF node to start. This service is available on both controller and controlled nodes.

2.1.1.1 Service Request Semantics

The semantics of the Synkro_Start Request service are as follows:

```
Synkro Start(
        nodeType
        pMACAddr
        bUseDataFromNV
        bNwkAutoRePairResponse
        )
```

Table 2-2 specifies the parameters for the Synkro_Start Request service.

Table 2-2. Synkro_Start Request Service Parameters

Name	Type	Dir	Valid range	Description
nodeType	uint8_t	IN	{gNodeType_Controller, gNodeType_Controlled }	Selects one of the two possible SynkroRF node types for the current node.
pMACAddr	uint8_t*	IN	-	This parameter allows the application to replace the default address loaded at manufacturing . If parameter is NULL, the node's MAC address remains untouched and the one set at manufacturing is used. A value of all 1's or all 0's is not accepted.

Table 2-2. Synkro_Start Request Service Parameters

bUseDataFromNv	bool_t	IN	{TRUE,FALSE}	If TRUE, the Node Data database will be filled with information existing in Non-VolatileMemory (Flash). If FALSE, Node Data database is cleared.
bNwkAutoRePairResponse	bool_t	IN	{TRUE,FALSE}	Ignored on controller nodes. If TRUE and nodeType is gNodeType_Controlled , when a pair request is received from a device that already exists in the Node Data pair table, a positive pair response will automatically be sent by the SynkroRF layer, without requiring the application's permission. If FALSE, the application will always be asked to accept a pair request, regardless of the pair status of the requesting node.

The possible return values for the Synkro_Start Request service API call are shown in the following table.

Table 2-3. Synkro_Start Request API Call Return Values

Type	Possible values	Description
uint8_t	gNWInvalidParam_c gNWNodeAlreadyStarted_c gNWDenied_c gNWNoTimers_c gNWSuccess_c	All possible return values are fully described in Section 2.1.1.3, "Effect on Receipt".

2.1.1.2 When Generated

The Synkro_Start Request service is generated by the application layer when a SynkroRF node is started. The Synkro_Start request is transmitted to the network layer by calling the Synkro_Start() function in the SynkroRF API.

2.1.1.3 Effect on Receipt

On receipt of Synkro_Start service request, the SynkroRF layer verifies if all the conditions to begin a Start process are met.

First, the SynkroRF layer performs the validation of the parameters. If the application uses a SynkroRF controller node library and tries to start a controlled node, or uses a SynkroRF controlled node library and tries to start a controller node, the SynkroRF layer returns *gNWInvalidParam_c*. If nodeType parameter has another value other than *gNodeType_Controller* or *gNodeType_Controlled*, then*gNWInvalidParam_c* is returned. If the pMACAddr is not null, but points to a 64 bit array which has all the bits set to 1 or all the bits set to 0, *gNWInvalidParam_c* is returned. If no timers are available for the network to allocate for its internal operations, *gNWNoTimers_c* is returned.

If none of the above conditions are met, SynkroRF checks its internal state machine and status variables, to determine if it can accept or deny the Synkro_Start request. A SynkroRF node can be started only once, so if the node is already started, the function exits with return value of *gNWNodeAlreadyStarted_c.* The

SynkroRF layer can only process one request at a time, so if another service request is currently being processed, the function exits with return value *gNWDenied_c*.

If the SynkroRF layer accepts the Synkro_Start request service for processing, the *gNWSuccess_c* value is returned.

When the processing of the service request completes, the network informs the application about the status of the operation using a Synkro_Start confirm message that is sent to the application layer through the SynkroRF SAP. For more details about how the Synkro_Start request is processed inside the SynkroRF layer see Section 2.1.2, “Synkro_Start Confirm”.

2.1.2 Synkro_Start Confirm

The Synkro_Start Confirm message indicates to the application layer that a previous Synkro_Start Request service has been processed and also provides the status of the operation. The Synkro_Start confirm message is sent to the application layer trough the SynkroRF SAP. It can be received by the application layer of both controller and controlled nodes.

2.1.2.1 Service Confirmation Semantics

The semantics of the Synkro_Start.confirm message are as follows:

```
synkroStartCnf(
        result
        )
```

Table 2-4-specifies the fields available in the SynkroRF Start Confirm message structure.

Table 2-4. Synkro_Start Confirm message structure

Field	Type	Possible values	Description
result	uint8_t	gNWNoMemory_c gNWSuccess_c	Specifies the result of a Synkro_Startprocessed request

2.1.2.2 When Generated

The Synkro_Start Confirm message is generated by the SynkroRF layer entity in response to a Synkro_Start service request.

2.1.2.2.1 Synkro_Start Processing

When starting to process a Synkro_Start request, the SynkroRF layer does the following:

- If the node type to be started is a controller node, then the current channel is set to the one in the NodeData database and a confirm message is sent to the application, having the result field set to *gNWSuccess_c*. The start process stops here for a controller node type.
- If the node type to be started is a controlled node, then an IEEE.802.15.4 Active Scan is initiated on all three SynkroRF channels, to find other IEEE.802.15.4 PAN devices already started and active on these channels. This operation is done for the controlled node to start a PAN using a not already assigned PAN Id and Short Address. If no MAC message could be allocated for the Active

Scan request from the MAC message pools, than the process of starting the node is aborted and a confirm message is sent to the application with the field result sets to *gNWNoMemory_c*. After completing the Active Scan process, a randomly generated PanId and ShortAddress values are chosen for the node being started. These values are verified not to be identical with any of the Pan Ids or Short Addresses returned by the Active Scan operation. The last operation in the start process of a controlled node is to actually start the PAN. If no MAC message could be allocated for the 802.15.4 Start.Request primitive from the MAC message pools, than the process of starting the node is aborted and a confirm message is sent to the application with the field result set to *gNWNoMemory_c*. If the start of the PAN was successful, then the start of the controlled node is considered successfully done and a confirm message is sent to the application with the result field set to *gNWSuccess_c*.

2.1.2.3 Effect on Receipt

On receipt of the Synkro_Start Confirm message, the application layer of the initiating device is notified of the result of its request to start the node.

2.1.3 Synkro_Search Request

The SynkroRF Search service makes a request for a controller node to find controlled devices are active in its proximity. This service is available only on controller nodes.

2.1.3.1 Service Request Semantics

The semantics of the Synkro_Search Request service are as follows:

```
Synkro_Search(
         deviceType
         pSearchData
         searchDataLength
         timeout
         )
```

Table 2-5 specifies the parameters for the SynkroRF Search Request service.

Table 2-5. Synkro_Search Request Service Parameters

Name	Type	Dir	Valid Range	Description
deviceType	uint8_t	IN	gDeviceType_Max	Type of device searched. All public device types are listed in Table 4-4. Wildcard 0xFF value can be used if a search for controlled devices of any device type is required
pSearchData	uint8_t*	IN	-	Address in memory where the data string the application wishes to send to the searched devices during the search process is located.
searchDataLength	uint8_t	IN	0 – *gMaxSearchPayload_c*	Length of the Search Request string sent by the application to the searched device.
Timeout	uint16_t	IN	1 65535	Interval in ms for the network to wait for search responses on one channel from the list of channels for responses to arrive.

The possible return values for the Synkro_Search Request service API call are shown in the following table.

Table 2-6. Synkro_Search Request API Call Return Values

Type	Possible values	Description
uint8_t	gNWInvalidParam_c gNWNodeNotStarted_c gNWDenied_c gNWSuccess_c	All possible return values are fully described in Section 2.1.3.3, "Effect on Receipt"

2.1.3.2 When Generated

The Synkro_Search Request service is generated by the application layer when it needs to look for controlled nodes that are active in the of the controller. The SynkroRF Search request is transmitted to the network layer, calling the Synkro_SearchRequest() function in the SynkroRF API.

2.1.3.3 Effect on Receipt

On receipt of a Synkro_Search service request, the SynkroRF layer verifies if all the conditions to begin a search process are met.

SynkroRF first checks the validity of the received parameters. If timeout is set to zero, the function exits with the *gNWInvalidParam_c* error code. If payload length is not in the accepted boundaries (0 - *gMaxSearchPayload_c*), the function exits with the *gNWInvalidParam_c* error code. If payload length is not zero but the pointer to the payload string is NULL, the function exits with the *gNWInvalidParam_c* error code.Although the valid range of the deviceType field specifies the 0-*gDeviceType_Max* interval or the 0xFF wildcard value, there is no test inside SynkroRF for the validity of this parameter, as proprietary deviceTypes are also supported.

SynkroRF then checks its internal state machine and status variables, to determine if it can accept or deny the SynkroRF Search Request. The SynkroRF node should already be started before a call to this function. If the node is not yet started, the function exits with *gNWNodeNotStarted_c* error code. If the node is in Sleep mode or if another SynkroRF process is already in place, the function exits with the *gNWDenied_c* error code.

If all parameters are within accepted ranges, the SynkroRF layer starts a Search process and returns the *gNWSuccess_c* value.

When the processing of the service request is complete, the network will inform the application about the status of the operation using a SynkroRF Search Confirm message that will be sent to the application layer through the SynkroRF SAP. For more details about how the Search process takes place inside the SynkroRF layer see Section 2.1.4, "Synkro_Search Confirm".

2.1.4 Synkro_Search Confirm

The SynkroRF Search Confirm message indicates to the application layer that the search process initiated by a previous Search Request has been completed and informs it about the status of its completion. This

message can be received only by the application layer of a controller node. The SynkroRF Search Confirm message is received by the application layer trough the SynkroRF SAP.

2.1.4.1 Service Confirmation Semantics

The semantics of the SynkroRF Search Confirm message are as follows:

```
synkroSearchCnf(
        status
        numNodes
        pSearchDescriptorBlocks
        )
```

Table 2-7-specifies the fields available in the Synkro_Search Confirm message structure.

Table 2-7. Synkro_Search Confirm message structure

Field	Type	Possible values	Description
Status	uint8_t	gNWNoTimers_c gNWNoMemory_c gNWFailed_c gNWSuccess_c	Specifies the final result of a Search Request that has been processed.
numNodes	uint8_t	0- gMaxReportedSearchDescriptors_c	Number of nodes that have been found during the search process
pSearchDescriptorBlocks	searchDescriptorBlock_t	-	Pointer to a chained list containing network and application information of each responding node

2.1.4.2 When Generated

The SynkroRF Search Confirm message is generated by the SynkroRF layer in response to an accepted Search Request.

When SynkroRF processes a Search Request, it performs the following steps.

- Start a wait response timeout operation timer. If a timer could not be started, the search process is aborted and a SynkroRF Search Confirm message with the result set to *gNWNoTimers_c* value is sent to the application.
- Send over the air a SynkroRF Search Request command. If no memory could be allocated to build the command, the Search process is aborted and a SynkroRF Search Confirm message with the result set to *gNWNoMemory_c* value is sent to the application. If the command is sent successfully, the network tries to enable the radio receiver of the node, to wait for Search Responses. If for some reasons this operation cannot be performed, the Search process is aborted and a SynkroRF Search Confirm message with the result set to *gNWFailed_c* value is sent to the application.

If no Search Responses are received in the interval specified by the timeout parameter, the SearchRequest command is repeated on the next channel. This operation repeats for a total of three (3) times, one per each channel. At the end of the process, a SynkroRF Search Confirm message with the result set to *gSuccess_c* value is sent to the application, having the numNodes field set to the number of responding devices and the pSearchDescriptorBlocks pointer set to the location where the chained list containing information about the found nodes starts.

The communication between the controller (initiating the search request) node and the controlled (initiating the search response) node(s) occurs as follows:

- From controller node to controlled node using:
  - 64-bit MAC Address of the controller node
  - For destination identifying, Pan Id 0xFFFF and Short Address 0xFFFF, which means the Search Request message is a broadcast message
- From the controlled node to the controller node using:
  - Pan Id and the Mac Address of the controlled node
  - For destination identifying, the 64 bit MAC Address of the controller node

2.1.4.3 Effect on Receipt

On receipt of the SynkroRF Search Confirm message, the application layer is notified about the completion and also about the status of its request to search the nearby radio space for active controlled nodes.

2.1.5 SynkroRF CallbackSearch

The CallbackSearch Application Service makes a request for the application layer to accept or deny a Search Request received from another controller device. This Application Service is available only on controlled nodes.

2.1.5.1 Service Request Semantics

```
CallbackSearch(
         macAddr
         nwkVersion
         nodeDescriptor
         LQI
         dataLength
         pData
         )
```

Table 2-8 specifies the parameters for the CallbackSearch application service.

Table 2-8. CallbackSearch Application Service Parameters

Name	Type	Possible values	Description
macAddr	uint8_t*		Pointer to a 8 byte array containing the MAC address of the controller node who initiated the search request
nwkVersion	uint8_t*		Pointer to a 2 byte array containing the version of the SynkroRF network running on the controller node who initiated the search request
nodeDescriptor	nodeDescriptor_t*	-	Address in memory to a structure which contains all application relevant information about the controller node who initiated the search request

Table 2-8. CallbackSearch Application Service Parameters

LQI	uint8_t	0 – 255	Link Quality Indicator for the IEEE 802.15.4 packet, containing the SynkroRF Search Request command
dataLength		0– gMaxPairCommandPayload_c	Length of the string sent by the application on the search requesting device, during the search process.
pData	uint8_t	-	Address in memory where the string sent by the application on the search requesting device starts

2.1.5.2 When Generated

The CallbackSearch Application Service is generated by SynkroRF when a Search Request SynkroRF internal command is received from a controller node. The network layer needs to find out if the application on the receiving node wants to respond to the search request or not.

To obtain this information, the network asks the application by calling the CallbackSearch Application Service.

Using the parameters of the function, the network passes to the application the following information:

- search request payload string transmitted by the requesting device
- radio quality of the received request
- requesting node's application defined information

2.1.5.3 Effect on Receipt

The return value of the CallbackSearch Application Service is used by the application to respond to the network's request.

The response is encapsulated in the appSearchCallbackResponse_t structure. A value of the status member set to *gNWSuccess_c* will indicate that application is instructing the network to respond to the just arrived Search Request while a value of this member set to any other value will inform the network not to respond to the Search Request. In the case the Search Request is accepted, a Search Response will be immediately sent by the network to the requesting device, The pData and dataLength members are used for setting the Search Response payload string. This string will be received by the requesting device, through the Search Confirm message.

2.1.6 Synkro_SearchResponse Confirm

The Synkro_Search Response Confirm message indicates to the application layer that the SearchResponse process has been completed and informs it about the status of its completion. This message can be received only by the application layer of a controlled node. The Synkro_Search Response Confirm message is sent to the application layer trough the SynkroRF SAP.

2.1.6.1 Service Confirmation Semantics

The semantics of the Synkro_Search Response.confirm message is as follows:

```
synkroSearchRespCnf(
        macAddr
        status
        )
```

Table 2-9-specifies the fields available in the Synkro_Search Response Confirm message structure.

Table 2-9. Synkro_Search Response Confirm Message Structure

Field	Type	Possible values	Description
macAddress	int8_t*		Pointer to an 8 byte array containing the MAC address of the node to which the search response was sent to
Status	int8_t	gNWFailed_c gNWSuccess_c	Specifies the status of the operation of sending a search response to a controller node

2.1.6.2 When Generated

The Synkro_Search Response Confirm message is generated by the SynkroRF layer entity in response to an accepted Search Request from a controlled node.

When SynkroRF is starting to process an accepted Search Request, it performs the following the sequence:

- Sends a SynkroRF internal Search Response command over the air to the requesting node
- If a successful confirm for the transmission of this command is not received from the MAC, a Search Response Confirm message is sent to the application with the status field set to *gNWFailed_c*.
- If a successful confirm is received from the MAC, a Search Response Confirm with the status field set to *gNWSuccess_c* is sent to the application, through the SynkroRF SAP

Sending the Search Response command from the controlled node to the controller node is accomplished using the following information:

- For source identifying, the Pan Id and the Mac Address of the controlled node
- For destination identifying, the 64 bit MAC Address of the controller node

2.1.6.3 Effect on Receipt

On receipt of the Synkro_Search Response Confirm message, the application layer is notified about the completion of an accepted search request.

2.1.7 Synkro_Pair Request

The Synkro_pair service makes a request for a SynkroRF controller node to attempt establishing a communication link with a controlled node of a specified deviceType. This service is available only on controller nodes.

2.1.7.1 Service Request Semantics

The semantics of the Synkro_Pair Request service are as follows:

```
Synkro_Pair(
        deviceType
        pMACAddr
        deviceId
        pPairingData
        pairingDataLength
        timeout
        )
```

Table 2-10 specifies the parameters for the Synkro_Pair Request service.

Table 2-10. Synkro_Pair Request Service Parameters

Name	Type	Dir	Valid range	Description
deviceType	uint8_t	IN	gDeviceType_Max	Type of device searched to pair with. All public device types are listed in Table 4-4
pMACAddr	uint8_t*	IN		Either a NULL value or a pointer to an 8 byte array containing the MAC address of the controlled node to pair with. If a NULL value is provided, the pair request would be attempted to any node that meets the deviceType criteria.In this case, only one positive pair response (first one to arrive) will be accepted.
deviceId	uint8_t	IN	0–(SOPT - 1)	Location in the Pair Table where information about the paired node will be copied, if the pair process completes successfully.
pPairingData	uint8_t*	IN	-	Address in memory where the string the application wants to send to the device during the pair process begins
pairingDataLength	uint8_t	IN	0– gMaxPairCommandPayload_c	Length of the Pair Request string sent by the application to the searched device.
timeout	uint16_t	IN	1 65535	Interval in ms for the pair process to take place. If the pair process does not complete by the end of this time interval, it will be aborted.

The possible return values for the Synkro_Pair Request service API call are shown in the following table.

Table 2-11. Synkro_Pair Request API Call Return Values

Type	Possible values	Description
uint8_t	gNWInvalidParam_c gNWNodeNotStarted_c gNWUnavailable_c gNWDenied_c gNWSuccess_c	All possible return values are fully described in Section 2.1.7.3, “Effect on Receipt”

2.1.7.2 When Generated

The Synkro_Pair Request service is generated by the application layer when it needs to establish a communication link with the application running on another node. The Synkro_pair request is transmitted to the network layer by calling the Synkro_PairRequest() function in the SynkroRF API.

2.1.7.3 Effect on Receipt

On receipt of Synkro_Pair service request, the SynkroRF layer verifies if all the conditions to begin a pair process are met.

SynkroRF first checks the validity of the received parameters. If timeout is set to zero, the function exits with the *gNWInvalidParam_c* error code. If payload length is not within the accepted boundaries, the function exits with the *gNWInvalidParam_c* error code. If payload length is not zero but the pointer to the payload string is NULL, the function exits with the *gNWInvalidParam_c* error code.If deviceId does not point to a location inside the boundaries of the Pair Table, the function exits with the *gNWInvalidParam_c* error code.Although the valid range of the deviceType field specifies the 0-gDeviceType_Max interval, there is no test inside SynkroRF for the validity of this parameter, because proprietary deviceTypes are also supported.

SynkroRF then checks its internal state machine and status variables to determine if it can accept or deny the Synkro_Pair Request. The SynkroRF node should already be started before calling this function, so if the node is not yet the function exits with *gNWNodeNotStarted_c* error code. If the node is in Sleep mode or if another SynkroRF process is already in place, the function exits with the *gNWDenied_c* error code.

If metal parameters pass validation checks, the SynkroRF layer starts a pair process and returns the *gNWSuccess_c* value.

When the processing of the service request completes, the network will inform the application about the status of the operation using an Synkro_Pair Confirm message that will be sent to the application layer through the SynkroRF SAP. For more details about how the Pair process takes place inside the SynkroRF layer see Section 2.1.8, “Synkro_Pair Confirm”.

2.1.8 Synkro_Pair Confirm

The Synkro_Pair Confirm message indicates to the application layer that the pair process initiated by a previously Pair Request has been completed and informs it about the status of its completion. This message can be received only by the application layer of a controller node. The Synkro_Pair Confirm message is received by the application layer trough the SynkroRF SAP.

2.1.8.1 Service Confirmation Semantics

The semantics of the Synkro_Pair Confirm message is as follows:

```
synkroPairCnf(
        result
        pReceivedData
        receivedDataLength
        )
```

Table 2-12-specifies the fields available in the Synkro_Pair Confirm message structure.

Table 2-12. Synkro_Pair Confirm message structure

Field	Type	Possible values	Description
Result	uint8_t	gNWNoTimers_c gNWNoMemory_c gNWTimeOut_c gNWFailed_c gNWSuccess_c	Specifies the final result of a Pair Request that has been processed
pReceiveData	uint8_t*	-	Address in memory where the string sent by the application of the device who accepted the Pair Request begins
receivedDataLength	uint8_t	gMaxPairCommandPayload_c	Length of the string sent by the application on the device that accepted the Pair Request

2.1.8.2 When Generated

The Synkro_Pair Confirm message is generated by the SynkroRF layer entity in response to a Pair Request that has been processed.

When processing a Pair Request, the following steps are observed:

- Start a timeout abort operation timer, having the interval specified in the received Pair Request. If a timer could not be started, the Pair process is aborted and a SynkroRF Pair Confirm message with the result set to *gNWNoTimers_c* value is sent to the application.
- Send over the air a SynkroRF internal Pair Request command. If no memory could be allocated to build the command, the Pair process is aborted and a SynkroRF Pair Confirm message with the result set to *gNWNoMemory_c* value is sent to the application. If the command is sent successfully, the network tries to enable the radio receiver of the node, to listen for a Pair Response. If for some reasons this operation cannot be performed, the Pair process is aborted and a SynkroRF Pair Confirm message with the result set to *gNWFailed_c* value is sent to the application.

If no Pair Response is received in *gPairReqRetryIntervalTx_c* interval of time, the Pair Request command is repeated. If no Pair Response is received when the abort timer interval elapses, the pair process is aborted and a SynkroRF Pair Confirm message with the result set to *gNWTimeout_c* value is sent to the application.If a Pair Response is received before the expiring of the abort timer interval, the Pair process is considered successful and a SynkroRF Pair Confirm message with the result set to *gNWSuccess_c* value is sent to the application.

The communication between the controller (requesting a pair) node and the controlled (accepting a pair request) node follows the steps below:

- From controller node to controlled node using:
- For source identifying, the 64 bit MAC Address of the controller node
- For destination identifying:
- When pMACAddr parameter of the Pair Request is NULL: the Pan Id 0xFFFF and Short Address 0xFFFF, starting an 802.15.4 broadcast message.
- When pMACAddr parameter of the Pair Request is not NULL: the Pan Id 0xFFFF and Mac address indicated by pMACAddr parameter, starting an 802.15.4 unicast acknowledged message.

- From the controlled node to the controller node using:
- For source identifying, the Pan Id and the Short Address of the controlled node
- For destination identifying, the 64 bit MAC Address of the controller node

2.1.8.3 Effect on Receipt

On receipt of the Synkro_Pair Confirm message, the application layer is notified about the completion and also about the status of its Pair Request.

2.1.9 SynkroRF CallbackPairing

The CallbackPairing Application Service makes a SynkroRF request for the application layer to accept or deny a Pair Request received from a controller device. This Application Service is available only on controlled nodes.

2.1.9.1 Service Request Semantics

The semantics of the CallbackPairing service are as follows:

```
CallbackPairing(
        pData
        length
        LQI
        deviceId
        nodeDescriptor
        )
```

Table 2-13-specifies the parameters for the CallbackPairing application service.

Table 2-13. CallbackPairing Application Service Parameters

Name	Type	Possible values	Description
pData	uint8_t	-	Address in memory where the string sent by the pair requesting device begins.
Length		0-gMaxPairCommandPayload_c	Length of the string sent by the application on the pair requesting device,
LQI	uint8_t	0255	Link Quality Indicator for the IEEE 802.15.4 received packet, containing the SynkroRF Pair Request command
deviceId		(SOPT-1), 0xFF	Specifies if the Pair Request is received from a node already paired or from a new node. In the first case, the deviceId indicates the location in the Pair Table where information about the already paired node can be found. In the second case, the deviceID is set to 0xFF, informing the application that the Pair Request is received from a node which is not found is the Pair Table.
nodeDescriptor	nodeDescriptor_t*	-	Address in memory to a structure which contains all application relevant information about the node requesting the pairing.

The return value of the CallbackPairing function is of type appPairCallbackResponse_t, where appPairCallbackResponse_t is a structure having the following definition:

```
typedef struct appPairCallbackResponse_tag{
        uint8_t  deviceId
        uint8_t* pData
        uint8_t  length
        }
```

2.1.9.2 When Generated

The CallbackPairing Application Service is generated by the SynkroRF layer when a Pair Request SynkroRF internal command having the LQI greater than the pair threshold is received from a controller node. The network layer needs to find out if the application accepts this request or not.To obtain this information, the network asks the application by calling the CallbackPairing Application Service.

Using the parameters of the function, the network passes the application the following information:

- Pair Request payload string transmitted by the requesting device
- Radio quality of the received request
- Whether or not the requesting device already is paired to the requesting node
- Requesting node's application defined information

2.1.9.3 Effect on Receipt

The return value of the CallbackPairing Application Service is used by the application to respond to the network's request.

The response is encapsulated in the appPairCallbackResponse_t structure. A value of the deviceId member inside the boundaries of the Pair Table will indicate that application is accepting the just arrived Pair Request while a value of this member set outside the boundaries of the Pair Table will inform the network that the Pair Request was refused by the application. In the case the Pair Request is accepted, a Pair Response will be immediately sent by the network to the requesting device. If the delivery of the Pair Response is confirmed by the MAC, the pair requesting node's information is copied in the Pair Table, at the location specified by the deviceId. If pair information was already present in that location, it will be overwritten.The pData and length members are used for setting the Pair Response payload string. This string will be received by the pair requesting device, trough the Pair Confirm message.If the Pair Request was refused by the application, the network layer will perform no further processing and will not submit a response to the requesting device.

2.1.10 Synkro_PairResponse Confirm

The Synkro_Pair Response Confirm message indicates to the application layer that the pair process initiated by a previously accepted Pair Request has been completed and informs it about the status of its completion. This message can be received only by the application layer of a controlled node. The Synkro_Pair Response Confirm message is sent to the application layer trough the SynkroRF SAP.

2.1.10.1 Service Confirmation Semantics

The semantics of the Synkro_Pair Response.confirm message is as follows:

```
synkroPairRespCnf(
        deviceId
        status
        )
```

Table 2-14 specifies the fields available in the Synkro_Pair Response Confirm message structure.

Table 2-14. Synkro_Pair Response Confirm Message Structure

Field	Type	Possible values	Description
deviceId	uint8_t	0 – (SOPT-1)	Position in the table where information about the Pair requesting node can be found, in case the Pair process completes successfully
status	uint8_t	gNWFailed_c gNWSuccess_c	Specifies the final result of an accepted Pair Request that has been processed

2.1.10.2 When Generated

The Synkro_Pair Response Confirm message is generated by the SynkroRF layer entity in response to an accepted Pair Request from a controller device.

When SynkroRF is processing an accepted Pair Request, it will perform the following steps:

- Send an SynkroRF internal Pair Response command, over the air, to the requesting node.
- If a successful confirm for the transmission of this command is not received from the MAC, the Pair process is aborted and a Pair Response Confirm message is send the application with the status field set to *gNWFailed_c*.
- If a successful confirm is received from the MAC, the Pair process successfully completed. A Pair Response Confirm with the status field set to *gNWSuccess_c* is sent to the application, trough the SynkroRF SAP. The message will also contain in the deviceId field the location in the Pair Table where the information about the successfully paired node can be found.

The sending of the Pair Response command from the controlled (pair accepting) node to the controller (pair requesting) node is done using:

- For source identifying, the Pan Id and the Short Address of the controlled node
- For destination identifying, the 64 bit MAC Address of the controller node

2.1.10.3 Effect on Receipt

On receipt of the Synkro_Pair Response Confirm message, the application layer is notified about the completion of an accepted Pair Request process.

2.1.11 Synkro_RemotePair Request

The Synkro_RemotePair service makes a request for a SynkroRF controller node to try to establish a communication link between two nodes in its Pair Table. This service is available only on controller nodes.

2.1.11.1 Service Request Semantics

The semantics of the Synkro_RemotePair Request service are as follows:

```
Synkro_RemotePair(
         deviceId1
         deviceId2
         timeout
         )
```

Table 2-15 specifies the parameters for the Synkro_RemotePair Request service.

Table 2-15. Synkro_RemotePair Request Service Parameters

Name	Type	Dir	Valid range	Description
deviceId1	uint8_t	IN	0 – (SOPT - 1)	Location in the Pair Table where information about the first node involved in the RemotePair process can be found.
deviceId2	uint8_t	IN	0 – (SOPT - 1)	Location in the Pair Table where information about the second node involved in the RemotePair process can be found.
timeout	uint16_t	IN	1 65535	Interval in ms for the RemotePair process to take place. If the RemotePair process does not complete during this time interval, it will be aborted.

The possible return values for the Synkro_RemotePair Request service API call are shown in the following table.

Table 2-16. Synkro_RemotePair Request API Call Return Values

Type	Possible values	Description
uint8_t	gNWInvalidParam_c gNWNodeNotStarted_c gNWUnavailable_c gNWDeviceIdNotPaired_c gNWDenied_c gNWSuccess_c	All possible return values are fully described in Section 2.1.11.3, “Effect on Receipt”

2.1.11.2 When Generated

The Synkro_RemotePair Request service is generated by the application layer when it needs to establish a communication link between the applications of two nodes from its Pair Table. The Synkro_RemotePair request is transmitted to the network layer by calling the Synkro_RemotePair() function.

2.1.11.3 Effect on Receipt

On receipt of Synkro_RemotePair service request, the SynkroRF layer verifies if all the conditions to begin a remote pair process are met.

SynkroRF first checks the validity of the received parameters. If timeout is set to zero, the function exits with the *gNWInvalidParam_c* error code. If deviceId1 has the same value as deviceId2, the function exits with the *gNWInvalidParam_c* error code.If deviceId1 or deviceId2 do not point to a location inside the boundaries of the Pair Table, the function exits with the *gNWInvalidParam_c* error code.

SynkroRF then checks its internal state machine and status variables, to determine if it can accept or not the Synkro_RemotePair Request. The SynkroRF node should already be started before calling this function, so if the node is not yet started the function exits with *gNWNodeNotStarted_c* error code. If the node is in Sleep mode or if another SynkroRF process is already in place, the function exits with the *gNWDenied_c* error code. If the service request is received on a controlled node, then the function exits with the *gNWUnavailable_c* error code. If there is no information in the Pair Table at any of the locations specified by deviceId1 or deviceId2, the function exits with the *gNWDeviceIdNotPaired_c* error code.

Once the validity of the parameters is verified, the SynkroRF layer starts a RemotePair process and returns the *gNWSuccess_c* value.

After the process completes, the network will inform the application about the status of the operation using a Synkro_RemotePair Confirm message that will be sent to the application layer through the SynkroRF SAP. For more details about how the RemotePair process takes place inside the SynkroRF layer see Section 2.1.12, “Synkro_RemotePair Confirm”.

2.1.12 Synkro_RemotePair Confirm

The Synkro_RemotePair Confirm message indicates to the application layer that the RemotePair process initiated by a previously RemotePair Request has been completed and informs it about the status of its completion. This message can be received only by the application layer of a controller node. The Synkro_RemotePair Confirm message is received by the application layer trough the SynkroRF SAP.

2.1.12.1 Service Confirmation Semantics

The semantics of the Synkro_RemotePair Response.confirm message is as follows:

```
synkroPairRemoteDevicesCnf(
        result
        )
```

Table 2-17-specifies the fields available in the Synkro_RemotePair Confirm message structure.

Table 2-17. Synkro_RemotePair Confirm message structure

Field	Type	Possible values	Description
Result	uint8_t	gNWNoTimers_c gNWNoMemory_c gNWTimeOut_c gNWFailed_c gNWDenied_c gNWSuccess_c	Specifies the final result of aRemotePair Request that has been processed.

2.1.12.2 When Generated

The Synkro_RemotePair Confirm message is generated by the SynkroRF layer entity in response to a RemotePair Request that has been processed.

When SynkroRF is processing a RemotePair Request, it performs the following steps:

- Start a timeout abort operation timer, having the interval specified in the received RemotePair Request. If a timer could not be started, the RemotePair process is aborted and a SynkroRF RemotePair Confirm message with the result set to *gNWNoTimers_c* value is sent to the application.
- Send a SynkroRF internal RemotePair Request command, over the air, to the first of the two nodes to remote pair. If no memory could be allocated to build the command, the RemotePair process is aborted and a SynkroRF RemotePair Confirm message with the result set to *gNWNoMemory_c* value is sent to the application. If the command is sent successfully, the network tries to enable the radio receiver of the node. If for some reasons this operation cannot be performed, the RemotePair process is aborted and a SynkroRF RemotePair Confirm message with the result set to *gNWFailed_c* value is sent to the application.If no RemotePair Response is received in *gPairReqRetryIntervalTx_c* interval of time, the RemotePair Request command is repeated. If no RemotePair Response is received when the abort timer interval expires, the RemotePair process is aborted and a SynkroRF RemotePair Confirm message with the result set to *gNWTimeout_c* value is sent to the application.
- If a RemotePair Response SynkroRF internal command is received, the Status field in the packet is verified (for mode details about the structure of the command, see Section 4.2.1.4, “Remote Pair Response Command Frame”). If it is set to FALSE, this means that the node that has sent the RemotePair Response has rejected the RemotePair Request. The RemotePair process is in this case aborted and a SynkroRF RemotePair Confirm message with the result set to *gNWDenied_c* value is sent to the application.
- If the RemotePair Request is accepted by the first node, then SynkroRF sends it the information in its Pair Table relating to the second node. If memory for building the packet containing this information cannot be allocated, the RemotePair process is aborted and a SynkroRF RemotePair Confirm message with the result set to *gNWNoMemory_c* value is sent to the application. If the MAC confirmation for the send of this packet is not successful, the RemotePair process is aborted and a SynkroRF RemotePair Confirm message with the result set to *gNWFailed_c* value is sent to the application. If the MAC confirmation is successful, then the same procedure as in case of first node is followed in with the second node.
- If a timeout of the abort timer occurs while RemotePair process is taking place, the process is aborted and a SynkroRF RemotePair Confirm message with the result set to *gNWTimeout_c* value is sent to the application.
- If all information has been exchanged before the expiring of the abort timer interval, the RemotePair process is considered successful and a SynkroRF RemotePair Confirm message with the result set to *gNWSuccess_c* value is sent to the application.

The communication between the controller node and the two controlled nodes is done as follows:

- From controller node to any of the controlled nodes using:
  - For source identifying, the Pan Id of the controlled node and the Short Address received by the controller during the pairing with that specific controlled node
  - For destination identifying, the Pan Id and Short Address of the controlled node
- From any of the controlled nodes to the controller node using:
  - For source identifying, the Pan Id and the Short Address of the controlled node

— For destination identifying, the 64 bit MAC Address of the controller node

2.1.12.3 Effect on Receipt

On receipt of the Synkro_RemotePair Confirm message, the application layer is notified about the completion and also about the status of its request to RemotePair two nodes.

2.1.13 SynkroRF CallbackRmtPairing

The CallbackRmtPairing Application Service makes a SynkroRF request for the application layer to accept or deny a RemotePair Request received from a controller device. This Application Service is available only on controlled nodes.

2.1.13.1 Service Request Semantics

The semantics of the CallbackRmtPairing service are as follows:

```
CallbackRmtPairing(
        deviceId
        nodeDescriptor
        )
```

Table 2-18 specifies the parameters for the CallbackRmtPairing application service.

Table 2-18. CallbackRmtPairing Application Service parameters

Name	Type	Possible values	Description
deviceId		(SOPT1)0xFF	Specifies if the RemotePair Request involves a controlled node already paired or if it involves a new controlled node. In the first case, the deviceId indicates the location in the Pair Table where information about the controlled node already paired can be found. In the second case, the deviceID is set to 0xFF, informing the application that the Pair Request is involves a controlled node which is not found is the Pair Table.
nodeDescriptor	nodeDescriptor_t*	-	Address in memory to a structure which contains all application relevant information about the controlled node requesting the pairing.

The return value of the CallbackRmtPairing function is of type uint8_t.

2.1.13.2 When Generated

The CallbackRmtPairing Application Service is generated by SynkroRF when a RemotePair Request SynkroRF internal command is received from a controller node. The network layer needs to find out if the application is accepting this request or not.

To obtain this information, the network asks the application by calling the CallbackRmtPairing Application Service.Using the parameters of the function, the network passes the following information to the application:

- Whether or not the controlled node that remotely (using the controller node) requests to pair is already paired to the receiving node
- Requesting node's application defined information

2.1.13.3 Effect on Receipt

The return value of the CallbackRmtPairing Application Service is used by the application to respond to the network's request.

A return value set inside the boundaries of the Pair Table indicates that application is accepting the just arrived RemotePair Request while a return value set outside the boundaries of the Pair Table informs the network that the RemotePair Request is refused by the application. In the case the RemotePair Request is accepted, a RemotePair Response will be immediately sent by the network to the controller device. If the delivery of the RemotePair Response is confirmed by the MAC, the controlled node's information is copied in the Pair Table, at the location specified by the return value of the CallbackRmtPairing.

A deviceId parameter of the callback different from 0xFF, informs the application that the node requesting the RemotePair is already in the Pair Table. In this case, a RemotePair Response will be sent to that requesting node no matter the return value of the CallbackRmtPairing function.

2.1.14 Synkro_RemotePair Response Confirm

The Synkro_RemotePair Response Confirm message indicates the application layer that the RemotePair process initiated by a previously accepted RemotePair Request has been completed and informs it about the status of its completion. This message can be received only by the application layer of a controlled node. The Synkro_RemotePair Response Confirm message is sent to the application layer trough the SynkroRF SAP.

2.1.14.1 Service Confirmation Semantics

The semantics of the Synkro_RemotePair Response.confirm message is as follows:

```
synkroRmtPairRespCnf(
        deviceId
        status
        )
```

Table 2-19-specifies the fields available in the Synkro_RemotePair Response Confirm message structure.

Table 2-19. Synkro_RemotePair Response Confirm message structure

Field	Type	Possible values	Description
deviceId	uint8_t	0 – (SOPT1)	Position in the table where information about the RemotePair controlled node can be found, in case the RemotePair process completes successfully
Status	uint8_t	gNWFailed_c gNWSuccess_c	Specifies the final result of an accepted RemotePair Request that has been processed

2.1.14.2 When Generated

The Synkro_RemotePair Response Confirm message is generated by the SynkroRF layer entity in response to an accepted RemotePair Request from a controller device.

When SynkroRF is processing an accepted RemotePair Request, it will perform the following steps:

- Send a SynkroRF internal RemotePair Response command, over the air, to the remote pair requesting controller node.
- If a successful confirm is not received from the MAC, the RemotePair process is aborted and a RemotePair Response Confirm message is send the application with the status field set to *gNWFailed*.
- If a successful confirm is received from the MAC, the RemotePair process is successfully completed. A RemotePair Response Confirm with the status field set to *gNWSuccess_c* is sent to the application, trough the SynkroRF SAP. The message will also contain in the deviceId field the location in the Pair Table where the information about the successfully remotely paired controlled node can be found.

The sending of the RemotePair Response command from the controlled (remote pair accepting) node to the controller (remote pair requesting) node is done using:

- For source identifying, the Pan Id and the Short Address of the controlled node
- For destination identifying, the 64 bit MAC Address of the controller node

2.1.14.3 Effect on Receipt

On receipt of the Synkro_RemotePair Response Confirm message, the application layer is notified about the completion of an accepted RemotePair request process.

2.1.15 Synkro_CloneDevice Request

The Synkro_CloneDevice service makes a request for a SynkroRF node to try to copy itself on another controller node. This service is available only on controller nodes.

2.1.15.1 Service Request Semantics

The semantics of the Synkro_CloneDevice Request service are as follows:

```
Synkro_CloneDevice(
        timeout
        )
```

Table 2-20 specifies the parameters for the Synkro_CloneDevice Request service.

Table 2-20. Synkro_CloneDevice Request Service Parameters

Name	Type	Dir	Valid Range	Description
timeout	uint16_t	IN	1 – 65535	Interval in ms for the Clone process to take place. If the Clone process does not complete by the expiration of this time interval, it will be aborted.

The possible return values for the Synkro_CloneDevice Request service API call are shown in the following table.

Table 2-21. Synkro_CloneDevice Request API Call Return Values

Type	Possible values	Description
uint8_t	gNWNodeNotStarted_c gNWUnavailable_c gNWNothingToClone_c gNWDenied_c gNWSuccess_c	All possible return values are fully described in Section 2.1.15.3, "Effect on Receipt"

2.1.15.2 When Generated

The Synkro_CloneDevice Request service is generated by the application layer when a SynkroRF node wants to be cloned on another SynkroRF controller node. The Synkro_CloneDevice request is transmitted to the network layer by calling the Synkro_CloneDevice() function in the SynkroRF API.

2.1.15.3 Effect on Receipt

On receipt of Synkro_CloneDevice service request, the SynkroRF layer verifies if all the conditions to begin a Clone process are met.

SynkroRF first checks its internal state machine and status variables, to determine if it can accept or not the Synkro_CloneDevice Request. The SynkroRF node should already be started before calling this function, so if the node is not yet started, the function exits with *gNWNodeNotStarted_c* error code. If the node is in Sleep mode or if another SynkroRF process is already in place, the function exits with the *gNWDenied_c* error code. If the service request is received on a controlled node, then the function exits with the *gNWUnavailable_c* error code. If no pair information is found in any of the Pair Table locations, the function exits with the *gNWNothingToClone_c* error code.

Once the validity of all parameters has been verified, the SynkroRF layer accepts the Synkro_CloneDevice Request for processing and returns the *gNWSuccess_c* value.

When the processing of the service request completes, the network will inform the application about the status of the operation using a Synkro_CloneDevice Confirm message that will be sent to the application layer through the SynkroRF SAP. For more details about how the Clone process takes place inside the SynkroRF layer see Section 2.1.16, "Synkro_CloneDevice Confirm".

2.1.16 Synkro_CloneDevice Confirm

The Synkro_CloneDevice Confirm message indicates the application layer that the Clone process initiated by a previously Clone Request has been completed and informs it about the status of its completion. This message can be received only by the application layer of a controller node. The Synkro_CloneDevice Confirm message is sent to the application layer trough the SynkroRF SAP.

2.1.16.1 Service Confirmation Semantics

The semantics of the Synkro_CloneDevice Confirm message is as follows:

```
synkroCloneCnf(
        result
        )
```

Table 2-22 specifies the fields available in the Synkro_CloneDevice Confirm message structure.

Table 2-22. Synkro_CloneDevice Confirm message structure

Field	Type	Possible values	Description
Result	uint8_t	gNWNoTimers_c gNWNoMemory_c gNWTimeOut_c gNWFailed_c gNWCloningRejected_c gNWSuccess_c	Specifies the final result of a CloneRequest that has been processed

2.1.16.2 When Generated

The Synkro_CloneDevice Confirm message is generated by the SynkroRF layer entity in response to a Clone Request that has been processed.

When SynkroRF is starting to process a Clone Request, the following steps are performed:

- Start a timeout abort operation timer, having the interval specified in the received Clone Request. If a timer could not be started, the Clone process is aborted and a SynkroRF CloneDevice Confirm message with the result set to *gNWNoTimers_c* value is sent to the application.
- Send a SynkroRF internal Clone Request command over the air. If no memory could be allocated to build the command, the Clone process is aborted and a SynkroRF CloneDevice Confirm message with the result set to *gNWNoMemory_c* value is sent to the application. If the command is sent successfully, the network tries to enable the radio receiver of the node. If for some reasons this operation cannot be performed, the Clone process is aborted and a SynkroRF CloneDevice Confirm message with the result set to *gNWFailed_c* value is sent to the application.
- If no Clone Response is received in *gCloneReqRetryInterval_c* interval of time, the Clone Request command is repeated. If no Clone Response is received yet until the elapsing of the abort timer interval, the Clone process is aborted and a SynkroRF CloneDevice Confirm message with the result set to *gNWTimeout_c* value is sent to the application.
- If a Clone Response SynkroRF internal command is received, the status field in the packet is verified. If it is set to FALSE, this means that the node that has sent the Clone Response has rejected the Clone Request. The Clone process is in this case aborted and a SynkroRF CloneDevice Confirm message with the result set to *gNWCloningRejected_c* value is sent to the application.
- If the clone Request is accepted, then SynkroRF starts to send the information of the nodes it is paired with, one by one. If memory for building the packets containing information of one nodes cannot be allocated, the Clone process is aborted and a SynkroRF CloneDevice Confirm message with the result set to *gNWNoMemory_c* value is sent to the application. If the MAC confirmation for the transmission of one of these packets is not successful, the Clone process is aborted and a

SynkroRF CloneDevice Confirm message with the result set to *gNWFailed_c* value is sent to the application.

- When the response to a packet containing one node's information is received, its status field is checked. If this is set to FALSE, this means that the device becoming a clone has refused the previously sent node information (also known as entry). In this case the Clone process is aborted on the device initiating the clone and a SynkroRF CloneDevice Confirm message with the result set to *gNWCloningRejected_c* value is sent to the application. The clone process is also aborted on the device that initially accepted the clone request and its pair table is restored to its previous state. If the CloneEntry Response packet has the status field set to TRUE, then SynkroRF will send the next clone entry. This process is repeated until the last clone entry is sent to the device becoming a clone.
- If a timeout of the abort timer occurs while sending the entries, then the Clone process is aborted and a SynkroRF CloneDevice Confirm message with the result set to *gNWTimeout_c* value is sent to the application.

After all the entries have been sent to the device becoming a clone, before the expiring of the abort timer interval, the clone process is considered successfully completed and a SynkroRF CloneDevice Confirm message with the result set to *gNWSuccess_c* value is sent to the application.

The communication between the first controller (clone requesting) node and the second controller (clone accepting) node is done using:

- For source identifying, the 64 bit MAC Address of the first controller node
- For destination identifying, the 64 bit MAC Address of the second controller node, obtained from the CloneResponse message.

2.1.16.3 Effect on Receipt

On receipt of the Synkro_CloneDevice Confirm message, the application layer is notified about the completion and also about the status of its request to Clone the node.

2.1.17 CallbackCloneDevice

The CallbackCloneDevice Application Service makes a SynkroRF request for the application layer to accept or deny a Clone Request received from another controller device. This Application Service is available only on controller nodes.

2.1.17.1 Service Request Semantics

The semantics of the CallbackCloneDevice service are as follows:

```
CallbackCloneDevice(
        entriesCount
        LQI
        )
```

Table 2-23-specifies the parameters for the CallbackCloneDevice application service.

Table 2-23. CallbackCloneDevice Application Service parameters

Name	Type	Possible values	Description
entriesCount	uint8_t	0 SOPT	Number of paired devices in the Clone requesting node's Pair Table
LQI	uint8_t	0255	Link Quality Indicator for the IEEE 802.15.4 received packet, containing the SynkroRF Clone Request command

The value returned by the CallbackCloneDevice is of type uint8_t.

2.1.17.2 When Generated

The CallbackCloneDevice Application Service is generated by SynkroRF when a CloneRequest SynkroRF internal command having the LQI greater than the Clone Threshold is received from a controller node. The network layer needs to find out if the application is accepting this request or not. To obtain this information, the network asks the application by calling the CallbackCloneDevice Application Service.

Using the parameters of the function, the network passes the application information related to the number of paired devices with the clone requesting node, and also related to the radio quality of the received command.

2.1.17.3 Effect on Receipt

The return value of the CallbackCloneDevice Application Service is used by the application to respond to the network's request.

A return value of TRUE will indicate that application is accepting the Clone Request while a return value of FALSE will inform the network that the Clone Request was refused by the application.

2.1.18 Synkro_CloneDevice Response Confirm

The Synkro_CloneDevice Response Confirm message indicates to the application layer that the Clone process initiated by a previously accepted Clone Request from a controller device has been completed and informs it about the status of its completion. This message can be received only by the application layer of a controller node. The Synkro_CloneDevice Response Confirm message is sent to the application layer trough the SynkroRF SAP.

2.1.18.1 Service Confirmation Semantics

The semantics of the Synkro_CloneDevice Response.confirm message is as follows:

```
synkroCloneRespCnf(
        result
        macAddr
        )
```

Table 2-24 specifies the fields available in the Synkro_CloneDevice Response Confirm message structure.

Table 2-24. Synkro_CloneDevice Response Confirm message structure

Field	Type	Possible values	Description
Result	uint8_t	gNWTimeout_c gNWFailed_c gNWCloningRejected_c gNWSuccess_c	Specifies the final result of an accepted CloneRequest that has been processed
macAddr	uint8_t*	-	Relevant only when the result is *gNWSuccess_c*. Pointer to a 64 bit memory location containing the IEEE 802.15.4 64 bit MAC address of the node that has been cloned

2.1.18.2 When Generated

The Synkro_CloneDevice Response Confirm message is generated by the SynkroRF layer entity in response to an accepted Clone Request from a controller device.

When SynkroRF processes a Clone Request, the procedure below is observed:

- Starts an timeout abort operation timer.
- Waits for a clone entry containing the information of one of the nodes paired with the clone requesting device to arrive.
- When a clone entry information arrives, calls the CallbackCloneEntry Application Service and find out if the information is accepted by the application. If not accepted, abort the whole Clone process and send the application a CloneDevice Response Confirm with the result set to *gNWCloningRejected_c*. If accepted, send a CloneEntryResponse to the clone requesting device, to confirm the receipt of the entry information. If Clone Entry Response cannot be sent successfully, abort the Clone process and send the application a CloneDevice Response Confirm with the result set to *gNWFailed_c*.
- If the abort timer elapses before receiving all clone entries, abort the Clone process and send the application a CloneDevice Response Confirm with the result set to *gNWTimeout_c*. The pair table will be restored to its previous content.
- If last entry is received before the abort interval expires, and the CloneEntryResponse for this last entry is sent successfully to the clone requesting device, then the Clone process is successfully completed. A CloneDevice Response Confirm with the result set to *gNWSuccess_c* will be sent to the application, trough the SynkroRF SAP. The message will also contain in the macAddr field the IEEE 802.15.4 64 bit MAC address of the device that has requested the clone.

The communication between the second controller (clone accepting) node and the first controller (clone requesting) node is done using:

- For source identifying, the 64 bit MAC Address of the second controller node.
- For destination identifying, the 64 bit MAC Address of the first controller node obtained from the CloneResponse message.

2.1.18.3 Effect on Receipt

On receipt of the Synkro_CloneDevice Response Confirm message, the application layer is notified about the completion of an accepted Clone request process.

2.1.19 CallbackCloneEntry

The CallbackCloneEntry Application Service makes a SynkroRF request for the application layer to provide a position in the Pair Table where information about one of the clone's paired nodes should be copied. This Application Service is available only on controller nodes.

2.1.19.1 Service Request Semantics

The semantics of the CallbackCloneEntry service are as follows:

```
CallbackCloneEntry(
        deviceId
        nodeDescriptor
        )
```

Table 2-25-specifies the parameters for the CallbackCloneEntry application service.

Table 2-25. CallbackCloneEntry Application Service parameters

Name	Type	Possible values	Description
deviceId	uint8_t	0 – (SOPT1)	Position in the clone request device's Pair Table where the arrived node information is placed.
nodeDescriptor	nodeDescriptor_t*	-	Memory address of a structure containing the arrived node information.

The value returned by the CallbackCloneEntry function is of type uint8_t.

2.1.19.2 When Generated

The CallbackCloneEntry Application Service is generated by the SynkroRF layer during an already started Clone process, on the device that is trying to become the clone. During the Clone process, the device to be cloned sends the information of the nodes it is paired with, one by one, to the clone device. The network layer on the receiving device receives one node's information but needs to find out if the application is accepting it, and if it does, at what precise location in the Pair Table this information should be stored.

To obtain this information, the network asks the application by calling the CallbackCloneEntry Application Service.

Using the parameters of the function, the network passes to the application the entire node structure information, and also its position in the clone requesting device's Pair Table.

2.1.19.3 Effect on Receipt

The return value of the CallbackCloneEntry Application Service is used by the application to respond to the network's request.

A return value inside the boundaries of the clone device's Pair Table will indicate that application is accepting the just arrived information. This information will be stored at the return value position in the device's Pair Table. Any previously stored paired information in this location will be overwritten. A CloneEntryResponse will be sent to the clone requesting device, having the status field set to TRUE.

A value outside the boundaries of the clone device's Pair Table will indicate that application is not accepting the just arrived information. This information is dropped. A CloneEntryResponse will be sent to the clone requesting device, having the Status field set to FALSE. This will cause the whole Clone process to be aborted. For more details about the structure of the CloneEntry Response command packet, see Section 4.2.1.8, "Clone Entry Response Command Frame".

2.1.20 Synkro_SendCommand Request

The Synkro_SendCommand service makes a request for the SynkroRF layer to send an application defined command to one or all the nodes in the Pair Table. This service is available both on controller and controlled nodes.

2.1.20.1 Service Request Semantics

The semantics of the Synkro_SendCommand Request service are as follows:

```
Synkro_SendCommand(
         deviceId
         cmdId
         paramLength
         paramData
         deviceTypeBroadcast
         txOptions
         )
```

Table 2-26 specifies the parameters for the Synkro_SendCommand Request service.

Table 2-26. Synkro_SendCommand Request Service Parameters

Name	Type	Dir	Valid range	Description
deviceId	uint8_t	IN	0 – (SOPT -1)0xFF	Position in the Pair Table where the information about the destination node should be found. If 0xFF, the command should be sent to all nodes in the Pair Table
cmdId	uint16_t	IN	1 – 32767	Identifier of the application command
paramLength	uint8_t	IN	090	Length of the command payload
paramData	uint8_t*	IN	-	Address in memory where the command payload string starts
deviceTypeBroadcast	uint8_t	IN	-	Considered only when deviceId is set to 0xFF, meaning a broadcast command was requested. It filters the type of devices to send the broadcast command to. If is equal to 0xFF the command will be sent to all devices in the pair table. Ignored if deviceId is not 0xFF.
txOptions	uint8_t	IN	-	Bit map containing the options for transmitting the command: Bit 0-1 for transmission with ack 0 for transmission without ack

The possible return values for the Synkro_SendCommand Request service API call are shown in the following table.

Table 2-27. Synkro_SendCommand Request API Call Return Values

Type	Possible values	Description
uint8_t	gNWNodeNotStarted_c gNWInvalidParam_c gNWDeviceIdNotPaired_c gNWDenied_c gNWSuccess_c	All possible return values are fully described in Section 2.1.20.3, "Effect on Receipt"

2.1.20.2 When Generated

The Synkro_SendCommand service is generated by the application layer when it needs to send an application command to one or all the nodes in the Pair Table.

2.1.20.3 Effect on Receipt

On receipt of Synkro_SendCommand Request service, the SynkroRF layer verifies if all the conditions to begin a SendCommand process are accomplished.

SynkroRF first checks the validity of the parameters. If the deviceId parameter exceeds the boundaries of the Pair Table, but is different from 0xFF, the function returns the *gNWInvalidParameter_c* value. If the commandId parameter is not in the specified range, the function exits with the *gNWInvalidParam_c* value. If the commandId is not found in the supported by the application commands (defined in NwkCommands.c), the function exits with the *gNWInvalidParam_c* value. If the direction constraints or the payload constraints of the defined command do not match the direction or payload specified in the request, the function exit with the *gNWInvalidParam_c* value. If there is no information in the Pair Table at the position indicated by deviceId parameter, then the function exits with the *gNWDeviceIdNotPaired_c* value. If the application command set to which the command belongs is found to be unsupported by the destination node, the function exits with the *gNWInvalidParam_c* value.

SynkroRF then checks its internal state machine and status variables, to determine if it can accept or not the Synkro_SendCommand Request. The SynkroRF node should already be started before calling this function, so if the node is not yet the function exits with *gNWNodeNotStarted_c* error code. If another SynkroRF process is already in place, the function exits with the *gNWDenied_c* error code.

If none of the conditions enumerated above are met, the SynkroRF layer starts processing the Synkro_SendCommand Request and the *gNWSuccess_c* value is returned.

When the SendCommand process will complete, the network will inform the application about the status of the operation using an Synkro_SendCommand Confirm message that will be sent to the application layer through the SynkroRF SAP.

2.1.21 Synkro_SendCommand Confirm

The Synkro_SendCommand Confirm message indicates the application layer that a previous Synkro_SendCommand Request service has been processed and informs it about the status of the operation. This message can be received by the application layer of both controller and controlled nodes. The Synkro_SendCommand Confirm message is sent to the application layer trough the SynkroRF SAP.

2.1.21.1 Service Confirmation Semantics

The semantics of the Synkro_SendCommand.confirm message is as follows:

```
synkroCommandCnf(
        result
        map
        )
```

Table 2-28 specifies the fields in the Synkro_SendCommand Confirm message structure.

Table 2-28. Synkro_SendCommand Confirm message structure

Field	Type	Possible values	Description
result	uint8_t	gNWFailed gNWPartialSuccess_c gNWSuccess_c	Specifies the result of a Synkro_SendCommand processed request
map	uint8_t*	-	32 bit array. Relevant only when the Synkro_SendCommand request has asked for the transmission of a broadcast command. Each bit indicates if the transmission to the node on the corresponding position in the Pair Table was successful (bit set to 1) or has failed (bit set to 0)

2.1.21.2 When Generated

The Synkro_SendCommand Confirm message is generated by the SynkroRF layer entity in response to a Synkro_SendCommand service request.

When SynkroRF processes a Synkro_SendCommand request, the procedure below is observed:

- Reserves a 32 bit array in the RAM memory and sets all the bits inside to 0. This is a map in which each bit corresponds to a location in the Pair Table.
- If deviceId parameter is not 0xFF, it tries to send the command to the node having the corresponding position in the Pair Table. A confirm message is sent to the application, having the result set to *gNWSuccess_c* or *gNWFailed_c,* depending on the transmission status.
- If deviceId parameter is set to 0xFF, SynkroRF sends the command to all nodes in the pair table, one at a time. After every successful transmission, the corresponding bit in the map array is set to 1. After the command is sent to all paired nodes, the map array is computed, considering only the bits corresponding to positions in the Pair Table where pair information exists. If all the values are set to 1, the result in the Confirm message is set to *gNWSuccess_c*. If all the values in the array are set to 0, the result in the Confirm message is set to *gNWFailed_c.* In the other cases, the result in the Confirm message is set to *gNwPartialSuccess_c*. The map array is also sent in the Confirm message, for the application to identify the nodes where the command has been successfully transmitted to from the nodes where the command failed to be transmitted.

The communication between a transmitter (command requesting) node and a receiver (command accepting) node is done as follows:

- From a controller node to a controlled node using:
  - For source identifying, the Pan Id of the controlled node and the Short Address of the controller node received during the pairing with the controlled node
  - For destination identifying, the Pan Id and the Short Address of the controlled node

- From a controlled node to a controller node using:
  - For source identifying, the Pan Id and the Short Address of the controlled node
  - For destination identifying, the 64 bit MAC Address of the controller node
- From a controlled node to a controlled node using:
  - For source identifying, the Pan Id and the Short Address of the transmitter controlled node
  - For destination identifying, the Pan Id and the Short Address of the receiver controlled node

2.1.21.3 Effect on Receipt

On receipt of the Synkro_SendCommand Confirm message, the application layer of the initiating device is notified of the result of its request to send an application command to one or all of its paired nodes.

2.1.22 Synkro_Command Indication

The Synkro_Command Indication message informs the application layer about the arrival of a command from one of the nodes in the Pair Table. This message can be received by the application layer of both a controller and a controlled node. The Synkro_CommandIndication message is sent to the application layer trough the SynkroRF SAP.

2.1.22.1 Service Indication Semantics

The semantics of the Synkro_Command Indication message is as follows:

```
synkroCommandInd(
         deviceId
         cmdId
         pDataPayload
         dataPayloadLength
         LQI
         hasUpdatedCapabilities
         )
```

Table 2-29 specifies the fields available in the Synkro_Command Indication message structure.

Table 2-29. Synkro_Command Indication message structure

Field	Type	Possible values	Description
deviceId	uint8_t	0 – (SOPT -1)	Specifies the position in the Pair Table of the node the command is received from
cmdId	uint16_t	1 – 32767	Identifier of the command
pDataPayload	uint8_t*	-	Address in memory where the command payload information starts
dataPayloadLength	uint8_t	090	Length of the command payload

Table 2-29. Synkro_Command Indication message structure

LQI	uint8_t	0255	Link Quality Indicator for the IEEE 802.15.4 received packet, containing the SynkroRF command
hasUpdatedCapabilities	uint8_t	{TRUE , FALSE}	Always set to TRUE on a controller node. On a controlled node specifies if the device from which the command is received has successfully updated its information related to supported command sets by the receiving device (parameter set to TRUE), or it has not successfully updated this information (parameter set to FALSE). In the second case the application on the node receiving the Command Indication message can take the decision to send an UpdateCapabilities command to the sender of the first command, in order for it to update the supported connections information of the node it previously sent the command to.

2.1.22.2 When Generated

The Synkro_Command Indication message is generated by the SynkroRF layer asynchronously, without any connection to a previously service request, when a command from one of the nodes in the Pair Table is received.

2.1.22.3 Effect on Receipt

On receipt of the Synkro_Command Indication message, the application layer is notified about the arrival of a command from one of the nodes in the Pair Table.

2.1.23 Synkro_SetBulkBufferState Request

The Synkro_SetBulkBufferState service makes a request for the SynkroRF layer to change the bulk buffer state. This service is available on both controller and controlled nodes.

The bulk buffer is an application buffer used by SynkroRF in bulk data transfers. Its maximum length is specified by the application and must be less than 64 Kbytes. This buffer can be in one of the following states:

Returns

gBufferFree_c	The bulk data buffer is free
gBufferBusyAppR_c	The buffer is reserved by application for reading the data inside (when the SynkroRF layer receives the last packet of a bulk data transfer, it sets the bulk buffer state to gBufferBusyAppR_c and notify the application that the bulk data buffer can be read).
gBufferBusyAppW_c	The buffer is reserved by application for writing the data inside (before sending bulk data, the application must set the bulk buffer state to gBufferBusyAppW_c, then write data in the buffer, and then pass the control to the SynkroRF layer).
gBufferBusyRx_c	The buffer is reserved by SynkroRF for receiving data inside (when a bulk data start packet is received and the transfer can be accepted, the SynkroRF sets the bulk data buffer state to gBufferBusyRx_c).

gBufferBusyTx_c — The buffer is reserved by SynkroRF for transmitting the data inside (when the application asks SynkroRF to begin a bulk data transfer, SynkroRF sets the bulk buffer state to gBufferBusyTx_c)

2.1.23.1 Service Request Semantics

The semantics of the Synkro_SetBulkBufferState Request service are as follows:

```
Synkro_SetBulkBufferState(
        newState
        )
```

Table 2-30 specifies the parameters for the Synkro_SetBulkBufferState Request service.

Table 2-30. Synkro_SetBulkBufferState Service Parameters

Name	Type	Dir	Valid range	Description
newState	uint8_t	IN	gBufferFree_c gBufferBusyAppR_c gBufferBusyAppW_c	Specifies the new bulk buffer state.

The possible return values for the Synkro_SetBulkBufferState Request service API call are shown in the following table.

Table 2-31. Synkro_SetBulkBufferState Request API Call Return Values

Type	Possible values	Description
uint8_t	gNWInvalidParam_c gNWDenied_c gNWSuccess_c	All possible return values are fully described in Section 2.1.23.3, “Effect on Receipt”.

2.1.23.2 When Generated

The Synkro_SetBulkBufferState Request is generated by the application layer when it wants to change the bulk buffer state (the application wants to mark the bulk buffer as free or as busy).

2.1.23.3 Effect on Receipt

On receipt of Synkro_SetBulkBufferState Request service, the SynkroRF layer entity will check if the new state requested can be set by the application and if the current state can be changed. After that the old bulk buffer state will be replaced with the new one.

Returns

gNWInvalidParam_c — Shows that the desired state can’t be set by the application (the application can’t set the buffer state to gBufferBusyRx_c or to gBufferBusyTx_c)

gNWDenied_c — Shows that the bulk buffer is reserved by the SynkroRF, and the application can’t change its state (the SynkroRF is involved in another bulk data transfer)

gNWSuccces_c — Shows that the bulk buffer state was changed to the desired one

2.1.24 Synkro_GetBulkBufferState Request

The Synkro_GetBulkBufferState service makes a request for the SynkroRF layer to return the bulk buffer state. This service is available on both controller and controlled nodes.

2.1.24.1 Service Request Semantics

The semantics of the Synkro_GetBulkBufferState Request service are as follows:

```
Synkro_GetBulkBufferState(
        )
```

Synkro_GetBulkBufferState Request does not pass any parameters to SynkroRF layer.

The possible return values for the Synkro_GetBulkBufferState Request service API call are shown in the following table.

Table 2-32. Synkro_GetBulkBufferState Request API Call Return Values

Type	Possible values	Description
uint8_t	gBufferFree_c gBufferBusyAppR_c gBufferBusyAppW_c gBufferBusyRx_c gBufferBusyTx_c	The bulk data buffer is free The application reads data from the bulk data buffer The application writes data in the bulk data buffer The transport layer transmits data from the bulk data buffer The transport layer receives data in the bulk data buffer

2.1.24.2 When Generated

The Synkro_GetBulkBufferState Request is generated by the application layer when it wants to obtain the current bulk buffer state.

2.1.24.3 Effect on Receipt

On receipt of Synkro_GetBulkBufferState Request service, the SynkroRF layer entity will return the bulk buffer state.

2.1.25 Synkro_SendBulkData Request

The Synkro_SendBulkData service makes a request for the SynkroRF layer to send bulk data to a node in the Pair Table. This service is available both on controller and controlled nodes.

2.1.25.1 Service Request Semantics

The semantics of the Synkro_SendBulkData Request service are as follows:

```
Synkro_SendBulkData(
        deviceId
        data
        length
        )
```

Table 2-33 specifies the parameters for the Synkro_SendBulkData Request service.

Table 2-33. Synkro_SendBulkData Request Service Parameters

Name	Type	Dir	Valid range	Description
deviceId	uint8_t	IN	0 – (SOPT 1)	Position in the Pair Table where the information about the destination node is found.
data	uint8_t*	IN	-	Address in memory where the bulk data string starts
length	uint16_t	IN	165535	Length of the bulk data

The possible return values for the Synkro_SendBulkData Request service API call are shown in the following table.

Table 2-34. Synkro_SendBulkData Request API Call Return Values

Type	Possible values	Description
uint8_t	gNWNodeNotStarted_c gNWInvalidParam_c gNWDeviceIdNotPaired_c gNWBufferBusy_c; gNWDenied_c gNWSuccess_c	All possible return values are fully described in Section 2.1.25.3, “Effect on Receipt”.

2.1.25.2 When Generated

The Synkro_SendBulkData service is generated by the application layer when it needs to send bulk data to a node in the Pair Table.

2.1.25.3 Effect on Receipt

On receipt of Synkro_SendBulkData Request service, the SynkroRF layer verifies if all the conditions to begin a SendBulkData process are met.

SynkroRF first checks the validity of the parameters. If the deviceId parameter exceeds the boundaries of the Pair Table, the function returns the *gNWInvalidParameter_c* value. If there is no information in the Pair Table at the position indicated by deviceId parameter, then the function exits with the *gNWDeviceIdNotPaired_c* value. If the bulk data length is 0 or the bulk data start address is null, the function exits with *gNWInvalidParameter_c* error code

SynkroRF then checks its internal state machine and status variables, to determine if it can accept or not the Synkro_ SendBulkData Request. The SynkroRF node should already be started before calling this function, so if the node is not yet started, the function exits with *gNWNodeNotStarted_c* error code. If the bulk buffer is in use by SynkroRF layer for another bulk transfer, the function exits with the *gNWBufferBusy_c* error code. If another SynkroRF process is already in place, the function exits with the *gNWDenied_c* error code.

Once the validity of all parameters has been verified, the SynkroRF layer accepts the Synkro_SendBulkData Request for processing, and the *gNWSuccess_c* value is returned.

When the SendBulkData process completes, the network will inform the application about the status of the operation using a Synkro_SendBulkData Confirm message that will be sent to the application layer through the SynkroRF SAP.

2.1.26 Synkro_SendBulkData Confirm

The Synkro_SendBulkData Confirm message indicates to the application layer that a previous Synkro_SendBulkData Request service has been processed and informs it about the status of the operation. This message can be received by the application layer of both controller and controlled nodes. The Synkro_SendBulkData Confirm message is sent to the application layer trough the SynkroRF SAP.

2.1.26.1 Service Confirmation Semantics

The semantics of the Synkro_SendBulkData.confirm message is as follows:

```
synkroBulkDataCnf(
        result
        deviceId
        )
```

Table 2-35 specifies the fields available in the Synkro_SendBulkData Confirm message structure.

Table 2-35. Synkro_SendBulkData Confirm message structure

Field	Type	Possible values	Description
Result	uint8_t	gNWNoTimers_c gNWTimeOut_c gNWNoMemory_c gNWFailed_c gNWBufferBusy_c gNWLengthTooBig_c gNWSuccess_c	Specifies the result of an Synkro_SendBulkData processed request
deviceId	uint8_t	0 – (SOPT 1)	Position in the Pair Table where the information about the destination node is found.

2.1.26.2 When Generated

The Synkro_SendBulkData Confirm message is generated by the SynkroRF layer entity in response to a Synkro_SendBulkData service request.

When SynkroRF is processing a Synkro_SendBulkData request, it performs the following:

- Sends a bulk data start packet to the destination node (the packet contains the bulk data length) and waits the bulk data start response.
- If timeout occurs, SynkroRF sends a Synkro_SendBulkData Confirm to the application with *gNWTimeOut*.
- If the response status is *gNWSuccess_c*, SynkroRF sends the first bulk data packet and waits the bulk data response.
- When the response is received, the SynkroRF layer will send the next bulk data packet and the process repeats until all the bulk data is transferred to its intended destination.

- If timeout occurs while waiting responses, SynkroRF sends a Synkro_SendBulkData Confirm to the application with *gNWTimeOut_c* status.
- When the response for the last bulk data packet arrives, SynkroRF sends a Synkro_SendBulkData Confirm to the application with gNWSuccess_c status.

The communication between a transmitter (bulk data transfer requesting) node and a receiver (bulk data transfer accepting) node is done as follows:

- From a controller node to a controlled node using:
  - For source identifying, the Pan Id of the controlled node and the Short Address of the controller node received during the pairing with the controlled node
  - For destination identifying, the Pan Id and the Short Address of the controlled node
- From a controlled node to a controller node using:
  - For source identifying, the Pan Id and the Short Address of the controlled node
  - For destination identifying, the 64 bit MAC Address of the controller node
- From a controlled node to a controlled node using:
  - For source identifying, the Pan Id and the Short Address of the transmitter controlled node
  - For destination identifying, the Pan Id and the Short Address of the receiver controlled node

2.1.26.3 Effect on Receipt

On receipt of the Synkro_SendBulkData Confirm message, the application layer of the initiating device is notified of the result of its request to send bulk data to one paired node.

2.1.27 Synkro_BulkDataStart Indication

The Synkro_BulkDataStart Indication message informs the application layer about the start of a bulk data receive process as a result of a bulk transfer initiated by one of the nodes in the Pair Table. The bulkData will be received inside the *bulkDataBuffer* array reserved in the application's RAM space. This message can be received by the application layer of both a controller and a controlled node. The Synkro_BulkDataStart Indication message is sent to the application layer trough the SynkroRF SAP.

2.1.27.1 Service Indication Semantics

The semantics of the Synkro_BulkDataStart Indication message is as follows:

```
synkroBulkDataStartInd(
        deviceId
        dataPayloadLength
        LQI
        )
```

Table 2-36 specifies the fields available in the Synkro_BulkDataStart Indication message structure.

Table 2-36. Synkro_BulkDataStart Indication message structure

Field	Type	Possible Values	Description
deviceId	uint8_t	0 – (SOPT 1)	Specifies the position in the Pair Table of the node the bulk data is received from
dataPayloadLength	uin16_t	165535	Length of the bulk data
LQI	uint8_t	0255	Link Quality Indicator for the IEEE 802.15.4 received packet, containing the first bulk data packet

2.1.27.2 When Generated

The Synkro_BulkDataStart Indication message is generated by the SynkroRF layer asynchronously, without any connection to a previously service request, when the first bulk data packet from one of the nodes in the Pair Table is received; before that, a bulk data transfer must be initiated by the source (by sending a bulk data start packet to this node), and this transfer must be accepted by the current node.

2.1.27.3 Effect on Receipt

On receipt of the Synkro_BulkDataStart Indication message, the application layer is notified about the start of a bulk data transfer, initiated by one of the nodes in the Pair Table.

2.1.28 Synkro_BulkData Indication

The Synkro_BulkData Indication message informs the application layer about the end of a bulk data transfer initiated by one of the nodes in the Pair Table. This message can be received by the application layer of both a controller and a controlled node. The Synkro_BulkData Indication message is sent to the application layer trough the SynkroRF SAP.

2.1.28.1 Service Indication Semantics

The semantics of the Synkro_BulkData Indication message is as follows:

```
synkroBulkDataInd(
         result
         deviceId
         dataPayloadLength
         LQI
         )
```

Table 2-37 specifies the fields available in the Synkro_BulkData Indication message structure.

Table 2-37. Synkro_BulkData Indication message structure

Field	Type	Possible values	Description
Result	uint8_t	gNWTimeOut_c gNWNoMemory_c gNWFailed_c gNWSuccess_c	Specifies the result of the bulk data transfer

Table 2-37. Synkro_BulkData Indication message structure

deviceId	uint8_t	0 – (SOPT -1)	Specifies the position in the Pair Table of the node the bulk data is received from
dataPayloadLength	Uin16_t	165535	Length of the bulk data received if the result field is set to gNWSuccess_c. Ignored otherwise.
LQI	uint8_t	0255	Link Quality Indicator for the IEEE 802.15.4 received packet, containing the last bulk data packet

2.1.28.2 When Generated

The Synkro_BulkData Indication message is generated by the SynkroRF layer asynchronously, without any connection to a previously service request, when the last bulk data packet from one of the nodes in the Pair Table is received.

2.1.28.3 Effect on Receipt

On receipt of the Synkro_BulkData Indication message, the application layer is notified about the end of a bulk data transfer, initiated by one of the nodes in the Pair Table. The Synkro_BulkData Indication message will always be received after a previous Synkro_BulkDataStart Indication message. The received bulkData resides inside the application's bulkData buffer, starting with position 0. The length of the received bulkData is indicated in the dataPayloadField of the indication message. At the moment when this message is received by application, the state of the BulkBuffer is already set, automatically, by , to gBufferBusyAppR_c.

2.1.29 Synkro_PollConfig Request

The Synkro_PollConfig service makes a request for the SynkroRF layer to set the poll interval and the receiver enabled time interval for periodic poll request. It must be called prior to calling Synkro_PollDevice function, whenever the application needs to perform a poll request to one or many controlled devices. This service is available only on controller node.

2.1.29.1 Service Request Semantics

The semantics of the Synkro_PollConfig Request service are as follows:

```
Synkro_PollConfig(
        pollInterval
        rxOnInterval
        )
```

Table 2-38 specifies the parameters for the Synkro_PollConfig Request service.

Table 2-38. Synkro_PollConfig Request Service Parameters

Name	Type	Dir	Valid range	Description
pollInterval	uint32_t	IN	(rxOnInterval + 1) – 262000(ms)	The time interval between two consecutive poll requests to all the devices in the poll map.
rxOnInterval	uint16_t	IN	1 – 65535(ms)	The time period that the receiver will be enabled for data reception from the controlled node.

The possible return values for the Synkro_PollConfig Request service API call are shown in the following table.

Table 2-39. Synkro_PollConfig Request API Call Return Values

Type	Possible values	Description
uint8_t	gNWNodeNotStarted_c gNWUnavailable_c gNWInvalidParam_c gNWSuccess_c	All possible return values are fully described in Section 2.1.29.3, "Effect on Receipt"

2.1.29.2 When Generated

The Synkro_PollConfig service is generated by the application layer when it needs to configure the poll request mechanism.

2.1.29.3 Effect on Receipt

On receipt of Synkro_PollConfig Request service, the SynkroRF layer verifies if all the conditions to set the poll interval and receiver enabled interval are met.

The SynkroRF node should already be started before calling this function, so if the node is not yet started, the function exits with *gNWNodeNotStarted_c* error code. If the request is received on a controlled node, the function exits with the *gNWUnavailable_c* error code.If the pollInterval is smaller then the rxOnInterval or if it is equal to zero, the function returns the *gNWInvalidParameter_c* value.

Once the validity of the parameters has been verified, the SynkroRF layer saves the poll request parameters into NVM and the *gNWSuccess_c* value is returned.

2.1.30 Synkro_PollDevice Request

The Synkro_PollDevice service makes a request for the SynkroRF layer to perform the setup of one node from the Pair Table for periodic poll request. It must be called whenever the application needs to perform a periodic poll request to a certain device. Before calling this function, the _PollConfig function must be called in order for the poll interval and the receiver enabled time interval to be set. This service is available only on controller node.

2.1.30.1 Service Request Semantics

The semantics of the Synkro_PollDevice Request service are as follows:

```
Synkro_PollDevice(
        deviceId
        bPollEnable
        bPollNow
        )
```

Table 2-40 specifies the parameters for the Synkro_PollDevice Request service.

Table 2-40. Synkro_PollDevice Request Service Parameters

Name	Type	Dir	Valid range	Description
deviceId	uint8_t	IN	0 – (SOPT 1)0xFF	Position in the Pair Table where the information about the destination node should be found.
bPollEnable	bool_t	IN	{TRUE,FALSE}	Enables / disables the periodic poll request.
bPollNow	bool_t	IN	{TRUE,FALSE}	If bPollNow is TRUE then a poll request to the node corresponding to deviceId in Pair Table will start as soon as possible. If bPollNow is FALSE, a poll request will start after the first expiration of the poll interval.

The possible return values for the Synkro_PollDevice Request service API call are shown in the following table.

Table 2-41. Synkro_PollDevice Request API Call Return Values

Type	Possible values	Description
uint8_t	gNWUnavailable_c gNWNodeNotStarted_c gNWDeviceIdNotPaired_c gNWInvalidParam_c gNWNoTimers_c gNWSuccess_c gNWPollReqNotConfigured_c	All possible return values are fully described in Section 2.1.30.3, “Effect on Receipt”

2.1.30.2 When Generated

The Synkro_PollDevice service is generated by the application layer when it needs to start a periodic poll request.

2.1.30.3 Effect on Receipt

On receipt of Synkro_PollDevice Request service, the SynkroRF layer verifies if all the conditions to begin a periodic poll process are met.

If the request is received on a controlled node, the function exits with the *gNWUnavailable_c* error code.The SynkroRF node should already be started before calling this function, so if the node is not yet started, the function exits with *gNWNodeNotStarted_c* error code.If there is no information in the Pair Table at the position indicated by deviceId parameter, then the function exits with the

gNWDeviceIdNotPaired_c value. If the deviceId parameter exceeds the boundaries of the Pair Table, but is different from 0xFF, the function returns the *gNWInvalidParameter_c* value. If the deviceId parameter is equal to 0xFF and there is no device in the poll map then the *gNWInvalidParameter_c* status is returned. If the poll interval and the receiver enabled interval were not already set, the function returns the *gNWPollReqNotConfigured_c* value.For SynkroRF periodic poll requests, a number of platform timers must be allocated. In case at least one of these timers could not be allocated, the function exits with *gNWNoTimers_c* error code.

After the validity of the parameters has been verified, the SynkroRF layer accepts the Synkro_PollDevice Request for processing, and the *gNWSuccess_c* value is returned.

The behavior of the SynkroRF layer given the various of parameter values is described in the following table.

Table 2-42. SynkroRF Layer Behavior

deviceId is the first device configured to be polled	deviceId	bPollEnable	bPollNow	Functionality
TRUE	Valid deviceId(not 0xFF)	TRUE	TRUE	Start poll request as soon as possible, the only device to be polled is deviceId.
TRUE	Valid deviceId(not 0xFF)	TRUE	FALSE	Start poll request after the first expiration of the poll interval, the only device to be polled is deviceId.
TRUE	Valid deviceId(not 0xFF)	FALSE	TRUE	Stop poll request on deviceId. In this case, deviceId being the only device configured to be polled, the periodic poll request will be stopped.
TRUE	Valid deviceId(not 0xFF)	FALSE	FALSE	Stop poll request on deviceId. In this case, deviceId being the only device configured to be polled, the periodic poll request will be stopped.
TRUE	0xFF	TRUE	TRUE	Because there aren't any devices to be polled and first deviceId is 0xFF, `gNWInvalidParameter_c` status is returned.
TRUE	0xFF	TRUE	FALSE	Because there aren't any devices to be polled and first deviceId is 0xFF, `gNWInvalidParameter_c` status is returned.
TRUE	0xFF	FALSE	TRUE	Because there aren't any devices to be polled and first deviceId is 0xFF, `gNWInvalidParameter_c` status is returned.
TRUE	0xFF	FALSE	FALSE	Because there aren't any devices to be polled and first deviceId is 0xFF, `gNWInvalidParameter_c` status is returned.

Table 2-42. SynkroRF Layer Behavior

FALSE	Valid deviceId(not 0xFF)	TRUE	TRUE	Start poll request as soon as possible, the first device to be polled is deviceId. When the poll request on deviceId is finished, the next device to be polled is the next device in Pair Table configured to be polled. If there is no other device configured to be polled after deviceId in Pair Table, when the poll interval expires, the first device to be polled is the smallest deviceId from the Pair Table configured to be polled.
FALSE	Valid deviceId(not 0xFF)	TRUE	FALSE	deviceId is added to the poll map. When the next poll interval expires, a poll request will be send also to deviceId.
FALSE	Valid deviceId(not 0xFF)	FALSE	TRUE	deviceId is deleted from the poll map. When the next poll interval expires, a poll request will not be send to deviceId.
FALSE	Valid deviceId(not 0xFF)	FALSE	FALSE	deviceId is deleted from the poll map. When the next poll interval expires, a poll request will not be send to deviceId.
FALSE	0xFF	TRUE	TRUE	Start poll request as soon as possible, the first device to be polled is the first device from the poll map.
FALSE	0xFF	TRUE	FALSE	The poll map will remain the same and `gNWSuccess_c value is returned.`
FALSE	0xFF	FALSE	TRUE	The poll map will be cleared, the poll request will be stopped because there are no devices in the map and `gNWSuccess_c value is returned.`
FALSE	0xFF	FALSE	FALSE	The poll map will be cleared, the poll request will be stopped because there are no devices in the map and `gNWSuccess_c value is returned.`

When a poll request process to all devices in the poll map will complete, the network will inform the application about the status of the operation using a Synkro_Poll Confirm message that will be sent to the application layer through the SynkroRF SAP.

2.1.31 Synkro_Poll Confirm

The Synkro_Poll Confirm message indicates the application layer that a previous Synkro_PollDevice Request service has been processed and informs it about the status of the operation. This message can be received by the application layer of the controller node. The Synkro_Poll Confirm message is sent to the application layer trough the SynkroRF SAP.

2.1.31.1 Service Confirmation Semantics

The semantics of the Synkro_Poll.confirm message is as follows:

```
synkroPollCnf(
        deviceId
        status
        )
```

Table 2-43 specifies the fields available in the Synkro_Poll Confirm message structure.

Table 2-43. Synkro_Poll Confirm message structure

Field	Type	Possible values	Description
deviceId	uint8_t	0 – (SOPT-)0xFF	Position in the table where information about the node can be found when the no memory status is returned; 0xFF is received in case the Poll process completes successfully
status	uint8_t	gNWNoMemory_c gNWSuccess_c	Specifies the result of a Synkro_Poll processed request

2.1.31.2 When Generated

The Synkro_Poll Confirm message is generated by the SynkroRF layer entity in response to a Synkro_PollDevice service request.

When SynkroRF is processing a Synkro_Poll request, the following steps are observed:

- Send a poll request to the first device to poll.
- Wait for a SynkroRF poll response from the current polled device.
- Check the poll response status.

If the current polled device has data: if rxOnInterval was configured to be different from 0, enable the receiver for `rxOnInterval` ms.During this interval, the controlled node can send the data it has to transmit. When the `rxOnInterval` interval expires, pass to the next device in the poll list. If there are no other devices to poll, send a Synkro_Poll Confirm with the *gNWSuccess_c* status and 0xFF as device identifier to the application, informing that the poll of all devices in the poll list has completed.If rxOnInterval was configured to be 0, SynkroRF layer will send a Synkro_Poll Confirm with the *gNWSuccess_c* status and the device identifier of the device the Poll Response with pending data status was received from to the application. Then, it will pass to the next device in the poll list. If there are no other devices to poll, send a Synkro_Poll Confirm with the *gNWSuccess_c* status and 0xFF as device identifier to the application, informing it that the poll of all devices in the poll list has completed.

If the current device does not have data:regardless of whether rxOnInterval was configured to be different from 0 or not, pass to the next device in the poll list If there are no other devices to poll, send a Synkro_Poll Confirm with the *gNWSuccess_c* status and 0xFF as device identifier to the application, informing it that the poll of all devices in the poll list has completed.

2.1.31.3 Effect on Receipt

On receipt of the Synkro_Poll Confirm message, the application layer of the initiating device is notified of the result of its request to send a poll request to one or many of its paired nodes.

2.1.32 Synkro_Poll Indication

The Synkro_Poll Indication message informs the application layer about the arrival of a poll request from one of the controller nodes in the Pair Table. This message is received by the application layer of a controlled node. The Synkro_Poll Indication message is sent to the application layer trough the SynkroRF SAP.

2.1.32.1 Service Confirmation Semantics

The semantics of the Synkro_Poll Indication message is as follows:

```
synkroPollInd(
        deviceId
        rxOnTime
        )
```

Table 2-44 specifies the fields available in the Synkro_Poll Indication message structure.

Table 2-44. Synkro_Poll Indication message structure

Field	Type	Possible values	Description
deviceId	uint8_t	0 – (SOPT 1)	Specifies the position in the Pair Table of the node the poll request is received from
rxOnTime	uint16_t	1 – 65535	Specifies the receiver enabled interval on the node that sent the poll request. During this interval, data can be sent to the controller node.

2.1.32.2 When Generated

At the arrival of a poll request on a controlled node, the SynkroRF layer checks if the application has data available for the node which generated the poll request. A poll response will be sent to the polling device, informing it about the availability of application data. In the case when application data is available, a Synkro_Poll Indication is sent to the application.

The Synkro_Poll Indication message is generated by the SynkroRF layer asynchronously, without any connection to a previously service request, when a poll request from one of the controller nodes in the Pair Table is received.

2.1.32.3 Effect on Receipt

On receipt of the Synkro_Poll Indication message, the application layer on the polled device is notified about the arrival of a poll request from one of the controller nodes in the Pair Table that the application has specified it has data to transmit to. During the polling device's receiver enabled interval (specified in the indication message), the application layer can send data to the node where the poll request was received from with increased probability of success.

2.1.33 Synkro_DataAvailable Request

The Synkro_DataAvailable service is used by the application layer to inform the network layer that it has data to send to a controller device already in the pairing table. It must be called whenever the application needs to send data to an already paired controller device. This service is available only on controlled node.

2.1.33.1 Service Request Semantics

The semantics of the Synkro_DataAvailable Request service are as follows:

```
Synkro_DataAvailable(
        deviceId
        bDataAvailable
        )
```

Table 2-45 specifies the parameters for the Synkro_DataAvailable Request service.

Table 2-45. Synkro_DataAvailable Request Service Parameters

Name	Type	Dir	Valid range	Description
deviceId	uint8_t	IN	0 – (SOPT 1)	Specifies the position in the Pair Table of the node which needs to be informed about the presence of application data destined to it
bDataAvailable	bool_t	IN	{TRUE, FALSE}	bDataAvailable is TRUE then the application has data to be transmitted to deviceId after a poll request will be received from that device. If bDataAvailable is FALSE, data will not be transmitted to deviceId after the receiving of a poll request.

The possible return values for the Synkro_DataAvailable service API call are shown in the following table.

Table 2-46. Synkro_DataAvailable Request API Call Return Values

Type	Possible Values	Description
uint8_t	gNWNodeNotStarted_c gNWUnavailable_ c gNWDeviceIdNotPaired_c gNWInvalidParam_c gNWSuccess_c	All possible return values are fully described in Section 2.1.33.3, "Effect on Receipt"

2.1.33.2 When Generated

The Synkro_DataAvailable Request service is generated by the application layer when it needs to inform the network layer that it has data to send to a controller device already in the pairing table. The Synkro_DataAvailable request is transmitted to the network layer by calling the Synkro_DataAvailable() function in the SynkroRF API.

2.1.33.3 Effect on Receipt

On receipt of Synkro_DataAvailable Request service, the SynkroRF layer verifies if all the conditions to inform the network layer about data availability are met.

The SynkroRF node should already be started before calling this function, so if the node is not yet started, the function exits with *gNWNodeNotStarted_c* error code.If the deviceId parameter exceeds the boundaries of the Pair Table, the function returns the *gNWInvalidParameter_c* value.If there is no information in the Pair Table at the position indicated by deviceId parameter, then the function exits with the *gNWDeviceIdNotPaired_c* value.If the request is received on a controller node, the function exits with the *gNWUnavailable_c* error code.

Once all the parameters are proved to be valid, the SynkroRF layer returns *gNWSuccess_c*to the application.

2.1.34 Synkro_UpdateCapabilities Request

The Synkro_UpdateCapabilities service makes a request for the SynkroRF layer to send a message to each node in the Pair Table containing its supported application command sets (capabilities). This service is available only on controlled nodes.

2.1.34.1 Service Request Semantics

The semantics of the Synkro_UpdateCapabilities Request service are as follows:

```
Synkro_UpdateCapabilities(
        cmdCapabilities[MAX_DEVICE_CAPABILITIES]
        )
```

Table 2-47 specifies the parameters for the Synkro_UpdateCapabilities Request service.

Table 2-47. Synkro_UpdateCapabilities Request Service Parameters

Name	Type	Dir	Valid range	Description
cmdCapabilities	uint8_t*	IN	-	40 bit array having each bit set or clear depending if the corresponding application command set (capability) is supported or not

The possible return values for the Synkro_UpdateCapabilities Request service API call are shown in the following table.

Table 2-48. Synkro_UpdateCapabilities Request API Call Return Values

Type	Possible values	Description
int8_t	gNWNodeNotStarted_c gNWUnavailable_c gNWDenied_c gNWSuccess_c	All possible return values are fully described in Section 2.1.34.3, “Effect on Receipt”

2.1.34.2 When Generated

The Synkro_UpdateCapabilities service is generated by the application layer when it needs to inform all the nodes in its Pair Table that the application command sets (capabilities) it supports have changed.

2.1.34.3 Effect on Receipt

On receipt of Synkro_UpdateCapabilities Request service, the SynkroRF layer verifies if all the conditions to begin a SendCommand with a reserved *gCmdUpdateCapabilities_c* command id process are met.

SynkroRF first checks its internal state machine and status variables, to determine if it can accept or not the Synkro_UpdateCapabilities Request. The SynkroRF node should already be started before calling this function, so if the node is not yet started, the function exits with *gNWNodeNotStarted_c* error code. If

another SynkroRF process is already in place, the function exits with the *gNWDenied_c* error code. If the request is received on a controller node, the function exits with the *gNWUnavailable_c* error code.

If the SynkroRF layer accepts the Synkro_UpdateCapabilities Request for processing, the *gNWSuccess_c* value is returned.

When the process of sending this special command completes, the network will inform the application about the status of the operation using a Synkro_ UpdateCapabilities Confirm message that will be sent to the application layer through the SynkroRF SAP. Because the UpdateCapabilities command is sent to all the paired devices, the network will inform the application about the status of command transmission to each device by setting (for success) or clearing (for failure) a bit in a map of32 bits, corresponding to the node's position in the Pair Table.For more details about how the SendCommand process takes place inside the SynkroRF layer see Section 2.1.20, "Synkro_SendCommand Request".

The nodes in the Pair Table having their radio Rx modules open will receive a SendCommand Indication message, having the command Id set to *gCmdUpdateCapabilities_c* reserved value and the payload set to the sender's supported capabilities. The payload will be automatically copied by the network into the NodeData Database, in the location corresponding to the node that sent the command. For more details about SendCommand Indication messages, see Section 2.1.22, "Synkro_Command Indication".

2.1.35 Synkro_UpdateCapabilities Confirm

The Synkro_UpdateCapabilities Confirm message indicates the application layer that a previous Synkro_UpdateCapabilities Request service has been processed and informs it about the status of the operation. This message can be received only by the application layer of controlled nodes. The Synkro_UpdateCapabilities Confirm message is sent to the application layer trough the SynkroRF SAP.

2.1.35.1 Service Confirmation Semantics

The semantics of the Synkro_UpdateCapabilities.confirm message is as follows:

```
synkroUpdateCapabilitiesCnf(
        result
        map
        )
```

Table 2-49 specifies the fields available in the Synkro_UpdateCapabilities Confirm message structure.

Table 2-49. Synkro_UpdateCapabilities Confirm message structure

Field	Type	Possible values	Description
result	uint8_t	gNWFailed gNWPartialSuccess_c gNWSuccess_c	Specifies the result of a Synkro_UpdateCapabilities processed request
map	uint8_t*	-	32 bit array. Each bit indicates if the transmission of update capabilities command to the node on the corresponding position in the Pair Table was successful (bit set to 1) or has failed (bit set to 0)

2.1.35.2 When Generated

The Synkro_UpdateCapabilities Confirm message is generated by the SynkroRF layer entity in response to a Synkro_UpdateCapabilities service request.

When SynkroRF processes a Synkro_ UpdateCapabilities request, it follow these steps:

- Reserves a 32 bit array in the RAM memory and sets all the bits inside to 0. This is a map in which each bit corresponds to a location in the Pair Table.
- SynkroRF sends the update capabilities command to all nodes in the pair table, one at a time. After every successful transmission, the corresponding bit in the map array is set to one (1). After the update capabilities command is sent to all paired nodes, the map array is computed, considering only the bits corresponding to positions in the Pair Table where pair information exists. If all the values are set to 1, the result in the Confirm message is set to *gNWSuccess_c*. If all the values in the array are set to 0, the result in the Confirm message is set to *gNWFailed_c*. In the other cases, the result in the Confirm message is set to *gNwPartialSuccess_c*. The map array is also transmitted in the Confirm message, for the application to identify the nodes where the update capabilities command has been successfully transmitted to from the nodes where the update capabilities command failed to be transmitted.

2.1.35.3 Effect on Receipt

On receipt of the Synkro_UpdateCapabilities Confirm message, the application layer of the initiating device is notified about the result of its request to send an update capabilities command to all of its paired nodes.

2.1.36 Synkro_UpdateCapabilities Indication

The Synkro_UpdateCapabilities Indication message informs the application layer about the arrival of an update capabilities command from one of the controlled nodes in the Pair Table. This message can be received by the application layer of both a controller and a controlled node. The Synkro_UpdateCapabilities message is sent to the application layer trough the SynkroRF SAP.

2.1.36.1 Service Indication Semantics

The semantics of the Synkro_UpdateCapabilities Indication message is as follows:

```
synkroUpdateCapabilitiesInd(
        deviceId
        cmdCapabilities
        )
```

Table 2-50 specifies the fields available in the Synkro_UpdateCapabilities Indication message structure.

Table 2-50. Synkro_UpdateCapabilities Indication message structure

Field	Type	Possible values	Description
deviceId	uint8_t	0 – (SOPT 1)	Specifies the position in the Pair Table of the node the update capabilities command is received from
cmdCapabilities	uint8_t*	-	Address in memory where the update capabilities command payload information (the new capabilities) starts

2.1.36.2 When Generated

The Synkro_UpdateCapabilities Indication message is generated by the SynkroRF layer asynchronously, without any connection to a previously service request, when an update capabilities command from one of the controlled nodes in the Pair Table is received.

2.1.36.3 Effect on Receipt

On receipt of the Synkro_UpdateCapabilities Indication message, the application layer is notified about the arrival of an update capabilities command from one of the controlled nodes in the Pair Table. The update capabilities payload contains the new capabilities (application command sets supported) of the controlled device the command is received from

2.1.37 Synkro_RefreshCapabilities Request

The Synkro_RefreshCapabilities service makes a request for the SynkroRF layer to ask for the latest application command sets supported by one of the controlled nodes in the Pair Table. This service is available only on controller nodes.

2.1.37.1 Service Request Semantics

The semantics of the Synkro_RefreshCapabilities Request service are as follows:

```
Synkro_RefreshCapabilities(
        deviceId
        )
```

Table 2-51 specifies the parameters for the Synkro_RefreshCapabilities Request service.

Table 2-51. Synkro_RefreshCapabilities Request Service Parameters

Name	Type	Dir	Valid range	Description
deviceId	int8_t	IN	0 – (SOPT_c 1)	Identifies the node in the NodeData Database for which the updated information is requested.

The possible return values for the Synkro_RefreshCapabilities Request service API call are shown in the following table.

Table 2-52. Synkro_RefreshCapabilities Request API Call Return Values

Type	Possible values	Description
uint8_t	gNWNodeNotStarted_c gNWUnavailable_c gNWDenied_c gNWSuccess_c	All possible return values are fully described in Section 2.1.37.3, "Effect on Receipt"

2.1.37.2 When Generated

The Synkro_RefreshCapabilities service is generated by the application layer when it needs to get the latest application command sets supported by one of the controlled nodes in the Pair Table.

2.1.37.3 Effect on Receipt

On receipt of Synkro_RefreshCapabilities Request service, the SynkroRF layer verifies if all the conditions to begin a SendCommand with a reserved *gCmdRefreshCapabilities_c* command id process are met.

SynkroRF first checks its internal state machine and status variables, to determine if it can accept or not the Synkro_RefreshCapabilities Request. The SynkroRF node should already be started before calling this function, so if the node is not yet started, the function exits with *gNWNodeNotStarted_c* error code. If another SynkroRF process is already in place, the function exits with the *gNWDenied_c* error code. If the request is received on a controlled node, the function exits with the *gNWUnavailable_c* error code.

Once the SynkroRF layer accepts the Synkro_RefreshCapabilities Request for processing, the *gNWSuccess_c* value is returned.

When the processing of this special command completes, the network will inform the application about the status of the operation using a Synkro_SendCommand Confirm message that will be sent to the application layer through the SynkroRF SAP. For more details about how the SendCommand process takes place inside the SynkroRF layer see Section 2.1.20, "Synkro_SendCommand Request".

The targeted node will receive a SendCommand Indication message, having the command id set to *gCmdRefreshCapabilities_c* reserved value. It is the application layer running on that node's choice to send back its supported capabilities. For more details about SendCommand Indication messages, see Section 2.1.22, "Synkro_Command Indication".

2.1.38 Synkro_RefreshCapabilities Confirm

The Synkro_RefreshCapabilities Confirm message indicates the application layer that a previous Synkro_RefreshCapabilities Request service has been processed and informs it about the status of the operation. This message can be received only by the application layer of controller nodes. The Synkro_UpdateCapabilities Confirm message is sent to the application layer trough the SynkroRF SAP.

2.1.38.1 Service Confirmation Semantics

The semantics of the Synkro_RefreshCapabilities.confirm message is as follows:

```
synkroRefreshCapabilitiesCnf(
        result
        )
```

Table 2-53-specifies the fields available in the Synkro_RefreshCapabilities Confirm message structure.

Table 2-53. Synkro_UpdateCapabilities Confirm message structure

Field	Type	Possible values	Description
result	uint8_t	gNWFailed gNWSuccess_c	Specifies the result of a Synkro_RefreshCapabilities processed request

2.1.38.2 When Generated

The Synkro_RefreshCapabilities Confirm message is generated by the SynkroRF layer entity in response to a Synkro_RefreshCapabilities service request.

When SynkroRF processes the Synkro_RefreshCapabilities request, it will send the refresh capabilities command to the specified controlled node from the pair table. If the transmission was successful, the result in the Confirm message is set to *gNWSuccess_c*. Otherwise, the result in the Confirm message is set to *gNWFailed_c.*

2.1.38.3 Effect on Receipt

On receipt of the Synkro_RefreshCapabilities Confirm message, the application layer of the initiating device is notified about the result of its request to send a refresh capabilities command to one of its controlled paired nodes.

2.1.39 Synkro_RefreshCapabilities Indication

The Synkro_RefreshCapabilities Indication message informs the application layer about the arrival of a refresh capabilities command from one of the controller nodes in the Pair Table. This message can be received only by the application layer of a controlled node. The Synkro_RefreshCapabilities message is sent to the application layer trough the SynkroRF SAP.

2.1.39.1 Service Indication Semantics

The semantics of the Synkro_RefreshCapabilities Indication message is as follows:

```
synkroRefreshCapabilitiesInd(
        deviceId
        )
```

Table 2-54 specifies the fields available in the Synkro_RefreshCapabilities Indication message structure.

Table 2-54. Synkro_RefreshCapabilities Indication message structure

Field	Type	Possible values	Description
deviceId	uint8_t	0 – (SOPT 1)	Specifies the position in the Pair Table of the node the refresh capabilities command is received from

2.1.39.2 When Generated

The Synkro_RefreshCapabilities Indication message is generated by the controlled SynkroRF layer asynchronously, without any connection to a previously service request, when a refresh capabilities command from one of the controller nodes in the Pair Table is received.

2.1.39.3 Effect on Receipt

On receipt of the Synkro_RefreshCapabilities Indication message, the application layer is notified about the arrival of a refresh capabilities command from one of the controlled nodes in the Pair Table.The application can choose to send to the requester device latest capabilities by initiating a SendCommand with commandId set to *gCmdUpdateCapabilities_c* and payload set to the capabilities array of its own nodeDescriptor database.

2.1.40 Synkro_ClearPairingInformation Request

The Synkro_ClearPairingInformation service makes a request for the SynkroRF layer to clear pair information from one or all position in the Pair Table of NodeData Database. This service is available both on controller and controlled nodes.

2.1.40.1 Service Request Semantics

The semantics of the Synkro_ClearPairingInformation Request service are as follows:

```
Synkro_ClearPairingInformation(
        deviceId
        )
```

Table 2-55 specifies the parameters for the Synkro_ClearPairingInformation Request service.

Table 2-55. Synkro_ClearPairingInformation Request Service Parameters

Name	Type	Dir	Valid range	Description
deviceId	uint8_t	IN	0 – (SOPT – 1), 0xFF	Identifies the node to clear in the Pair Table of NodeData Database. A value of 0xFF specifies that the entire Pair Table should be cleared

The possible return values for the Synkro_ClearPairingInformation Request service API call are shown in the following table.

Table 2-56. Synkro_ClearPairingInformationRequest API Call Return Values

Type	Possible values	Description
uint8_t	gNWNodeNotStarted_c gNWInvalidParam gNWSuccess_c	All possible return values are fully described in Section 2.1.40.3, "Effect on Receipt"

2.1.40.2 When Generated

The Synkro_ClearPairingInformation Request service is generated by the application layer when it needs to delete one or all nodes information from the Pair Table.

2.1.40.3 Effect on Receipt

First thing the SynkroRF layer performs is the validation of the parameters. If deviceId parameter is different from 0xFF but exceeds the boundaries of the Pair Table in NodeData Database, *gNWInvalidParam_c* error code is returned. The SynkroRF node should already be started before calling this function, so if the node is not started, the function exits with the *gNwNodeNotStarted_c* error code.

The function then behaves different depending on the value of the deviceId parameter and on the type of node where the call was made.

If the node is of type controlled and the deviceId parameter is set to 0xFF (which means clearing the whole Pair Table), the SynkroRF checks its internal state machine and status variables, to determine if it can accept or not the Synkro_ClearPairingInformation request. If another SynkroRF process is already in place, the function exits with the *gNWDenied_c* error code. If no error conditions are met, the SynkroRF layer clears the whole Pair Table information and restarts the controlled node, choosing in the same time new PanId and ShortAddress, different from the ones before. When the restart process completes, the network will inform the application about the status of the operation using an Synkro_ClearPairInformation Confirm message that will be sent to the application layer through the SynkroRF SAP.

Else, the SynkroRF clears the Pair Table according to the value of the deviceId parameter. If this value is inside the boundaries of the Pair Table, the information at that specific position will be erased. If the deviceId is set to 0xFF, the information in the whole Pair Table will be erased. The function exits with the *gNWSuccess_c* status value. The Synkro_ClearPairingInformation Request is a blocking function. The whole function's task are accomplished inside its body. No further Confirm messages are associated with this function.

2.1.41 Synkro_SetNewMACAddress Request

The Synkro_SetNewMACAddress service makes a request for an SynkroRF node change its IEEE 802.15.4 64 bit MAC address. This service is available on both controller and controlled nodes. The Synkro_SetNewMACAddress service allows the application to replace the device's address with a desired valid value.

2.1.41.1 Service Request Semantics

The semantics of the Synkro_SetNewMACAddress Request service are as follows:

```
Synkro_SetNewMACAddress(
        pMacAddr
        )
```

Table 2-57-specifies the parameters for the Synkro_SetNewMACAddress Request service.

Table 2-57. Synkro_SetNewMACAddress Request Service Parameters

Name	Type	Dir	Valid range	Description
pMacAddr	uint8_t*	IN	-	64 bit array containing the new 802.15.4 MAC address

The possible return values for the Synkro_SetNewMACAddress Request service API call are shown in the following table.

Table 2-58. Synkro_SetNewMACAddress Request API Call Return Values

Type	Possible values	Description
uint8_t	gNWInvalidParam_c gNWNodeNotStarted_c gNWDenied_c gNWSuccess_c	All possible return values are fully described in Section 2.1.41.3, “Effect on Receipt”

2.1.41.2 When Generated

The Synkro_SetNewMacAddress service makes a request for the SynkroRF layer to change the IEEE 802.15.4 64 bit MAC address.

2.1.41.3 Effect on Receipt

On receipt of Synkro_SetNewMACAddress Request service, the SynkroRF layer verifies if all the conditions to begin a SetNewMACAddress process are met.

- SynkroRF first checks the validity of the parameter. If all the bits in the 64 bit array have the same value, the function exits with the *gNWInvalidParam_c* error code.SynkroRF then checks its internal state machine and status variables, to determine if it can accept or not the Synkro_SetNewMACAddress Request. The node should already be started before calling this function, so if the node is not yet started, the function exits with *gNWNodeNotStarted_c* error code. If another SynkroRF process is already in place, the function exits with the *gNWDenied_c* error code.
- If the SynkroRF layer accepts the Synkro_SetNewMACAddress Request for processing, the *gNWSuccess_c* value is returned.

When the processing of the service request completes, the network will inform the application about the status of the operation using an Synkro_SetNewMACAddress Confirm message that will be sent to the application layer through the SynkroRF SAP. For more details about how the SetNewMACAddress

process takes place inside the SynkroRF layer see Section 2.1.42, “Synkro_SetNewMACAddress Confirm”.

2.1.42 Synkro_SetNewMACAddress Confirm

The Synkro_SetNewMACAddress Confirm message indicates the application layer that a previous Synkro_SetNewMACAddress Request service has been processed and informs it about the status of the operation. This message can be received by the application layer of both controller and controlled nodes. The Synkro_SetNewMACAddress Confirm message is sent to the application layer trough the SynkroRF SAP.

2.1.42.1 Service Confirmation Semantics

The semantics of the Synkro_SetNewMACAddress Confirm message is as follows:

```
synkroSetNewMACAddressCnf(
        result
        )
```

Table 2-59 specifies the fields available in the Synkro_SetNewMACAddress Confirm message structure.

Table 2-59. Synkro_SetNewMACAddress Confirm message structure

Field	Type	Possible values	Description
Result	uint8_t	gNWFailed_c gNWSuccess_c	Specifies the result of an Synkro_SetNewMACAddress processed request.

2.1.42.2 When Generated

The Synkro_SetNewMACAddress Confirm message is generated by the SynkroRF layer entity in response to a Synkro_SetNewMACAddress service request that has been completed.

When SynkroRF is processing a Synkro_SetNewMACAddress Request, it set a flag to inform the network layer that a SetNewMACAddress Request is pending.First time the SynkroRF layer task enters its Idle sub-task, the above mentioned flag triggers the MAC address changing procedure. The procedure involves closing the node’s radio receiver, setting the new MAC value and opening the radio receiver if it was previously open. If one of these operations fail, a confirm message is sent to the application, having the result field set to *gNWFailed_c* value. If all operations succeed, a confirm message is sent to the application, having the result field set to *gNWSuccess_c* value. If the MAC address of a controller node is changed, the node will not be able to receive future application commands coming from any of the nodes in it’s pair table.

2.1.42.3 Effect on Receipt

On receipt of the Synkro_SetNewMACAddress Confirm message, the application layer of the initiating device is notified of the result of its request to change the node’s IEEE 802.15.4 MAC address.

2.1.43 Synkro_GetMACAddress Request

The Synkro_GetMACAddress service makes a request for a SynkroRF node to return its IEEE 802.15.4 64 bit MAC address. This service is available on both controller and controlled nodes.

2.1.43.1 Service Request Semantics

The semantics of the Synkro_GetMACAddress Request service are as follows:

```
Synkro_GetMACAddress(
        pMacAddr
        )
```

Table 2-60 specifies the parameters for the Synkro_GetMACAddress Request service.

Table 2-60. Synkro_GetMACAddress Request Service Parameters

Name	Type	Dir	Valid range	Description
pMacAddr	uint8_t*	OUT	-	64 bit array that will be filled with the 802.15.4 MAC address

2.1.43.2 When Generated

The Synkro_GetMacAddress service makes a request for the SynkroRF layer to return its IEEE 802.15.4 64 bits MAC address.

2.1.43.3 Effect on Receipt

On receipt of Synkro_GetMACAddress Request service, the SynkroRF layer obtains the 802.15.4 extended address of the MAC layer running beneath and copies this address in the first 8 bytes pointed by pMacAddr parameter.

2.1.44 Synkro_Sleep Request

The Synkro_Sleep service makes a request for a SynkroRF node to pass to sleep functioning mode. This service is available on both controller and controlled nodes.

2.1.44.1 Service Request Semantics

The semantics of the Synkro_Sleep Request service are as follows:

```
Synkro_Sleep(
        )
```

Synkro_Sleep Request does not pass any parameters to SynkroRF layer.

The possible return values for the Synkro_Sleep Request service API call are shown in the following table.

Table 2-61. Synkro_Sleep Request API Call Return Values

Type	Possible values	Description
uint8_t	gNWNodeNotStarted_c gNWDenied_c gNWSuccess_c	All possible return values are fully described in Section 2.1.44.3, "Effect on Receipt"

2.1.44.2 When Generated

The Synkro_Wake service makes a request for the SynkroRF layer to enter the Sleep functioning mode of the SynkroRF layer.

2.1.44.3 Effect on Receipt

On receipt of Synkro_Sleep Request service, the SynkroRF layer verifies if all the conditions to begin a Sleep process are met.

SynkroRF first checks its internal state machine and status variables, to determine if it can accept or not the Synkro_Sleep Request. The SynkroRF node should already be started before calling this function, so if the node is not yet started, the function exits with *gNWNodeNotStarted_c* error code. If the node is already in Sleep mode or if another SynkroRF process is already in place, the function exits with the *gNWDenied_c* error code.

Once the SynkroRF accepts the Synkro_Sleep Request for processing, the *gNWSuccess_c* value is returned.

When the processing of the service request completes, the network will inform the application about the status of the operation using an Synkro_Sleep Confirm message that will be sent to the application layer through the SynkroRF SAP. For more details about how the Sleep process takes place inside the SynkroRF layer see Section 2.1.45, "Synkro_Sleep Confirm".

2.1.45 Synkro_Sleep Confirm

The Synkro_Sleep Confirm message indicates the application layer that a previous Synkro_Sleep Request service has been processed and informs it about the status of the operation. This message can be received by the application layer of both controller and controlled nodes. The Synkro_Sleep Confirm message is sent to the application layer trough the SynkroRF SAP.

2.1.45.1 Service Confirmation Semantics

The semantics of the Synkro_Sleep Confirm message is as follows:

```
synkroSleepCnf(
        result
        )
```

Table 2-62 specifies the fields available in the Synkro_Sleep Confirm message structure.

Table 2-62. Synkro_Sleep Confirm message structure

Field	Type	Possible values	Description
Result	uint8_t	gNWFailed_c gNWSuccess_c	Specifies the result of an Synkro_Sleep processed request.

2.1.45.2 When Generated

The Synkro_Sleep Confirm message is generated by the SynkroRF layer entity in response to a Synkro_Sleep service request that has been completed.

A flag is set to inform the network layer that a Sleep Request is pending. First time the SynkroRF layer task enters its Idle sub-task, the above mentioned flag triggers the Sleep mode enter procedure. The procedure involves disabling the Channel Agility mechanism and closing the node's radio receiver. If one of these operations fails, a confirm message is sent to the application, having the result field set to *gNWFailed_c* value. If both operations succeed, a confirm message is sent to the application, having the result field set to *gNWSuccess_c* value.

2.1.45.3 Effect on Receipt

On receipt of the Synkro_Sleep Confirm message, the application layer of the initiating device is notified of the result of its request to pass the node in Sleep mode.

2.1.46 Synkro_Wake Request

The Synkro_Wake service makes a request for the SynkroRF layer to exit the Sleep functioning mode of the SynkroRF layer. This service is available on both controller and controlled nodes.

2.1.46.1 Service Request Semantics

The semantics of the Synkro_Wake Request service are as follows:

```
Synkro_Wake(
         )
```

Synkro_Wake Request does not pass any parameters to SynkroRF layer.

The possible return values for the Synkro_Wake Request service API call are shown in the following table.

Table 2-63. Synkro_Wake Request API Call Return Values

Type	Possible values	Description
uint8_t	gNWNodeNotStarted_c gNWDenied_c gNWSuccess_c	All possible return values are fully described in Section 2.1.46.3, "Effect on Receipt"

2.1.46.2 When Generated

The Synkro_Wake Request service is generated by the application layer when it needs to exit the SynkroRF Sleep mode functioning. The Synkro_Wake request is transmitted to the network layer by calling the Synkro_Wake() function in the SynkroRF API.

2.1.46.3 Effect on Receipt

First step is to check its internal state machine and status variables, to determine if it can accept or not the Synkro_Wake request. The SynkroRF node should already be started before calling this function, so if the node is not started, the function exits with *gNWNodeNotStarted_c* error code. If the is not in Sleep mode, the function exits with the *gNWDenied_c* error code.

If all above conditions are fulfilled, SynkroRF will try to place the network back into normal functioning mode, by enabling the Channel Agility mechanism and by updating the radio receiver module state to the one before the sleep request. If one of these two operations fails, the function will exit with the error code *gNWDenied_c*. Otherwise, the function will return the *gNWSuccess_c* value, informing the calling layer that the service request has been completed successfully.

2.1.47 Synkro_SetReceiveMode Request

The Synkro_SetReceiveMode service makes a request for the SynkroRF layer to open or close the receive module of the node's IEEE 802.15.4 radio transceiver. This service is available on both controller and controlled nodes.

2.1.47.1 Service Request Semantics

The semantics of the Synkro_SetReceiveMode Request service are as follows:

```
Synkro_SetReceiveMode(
        rxMode
        )
```

Table 2-64 specifies the parameters for the Synkro_SetReceiveMode.request service.

Table 2-64. Synkro_SetReceiveMode Request Service Parameters

Name	Type	Dir	Valid range	Description
rxMode	uint8_t	IN	{TRUE,FALSE}	Specifies if the Rx module of the radio transceiver shall be enabled or disabled

The possible return values for the Synkro_SetReceiveMode Request service API call are shown in the following table.

Table 2-65. Synkro_SetReceiveMode Request API Call Return Values

Type	Possible values	Description
uint8_t	gNWDenied_c gNWFailed_c gNWSuccess_c	All possible return values are fully described in Section 2.1.47.3, “Effect on Receipt”

2.1.47.2 When Generated

The Synkro_SetReceiveMode Request service is generated by the application layer when it needs to enable or disable the receive module of the radio transceiver. The Synkro_SetReceiveMode request is transmitted to the network layer by calling the Synkro_SetReceiveMode() function in the SynkroRF API.

2.1.47.3 Effect on Receipt

On receipt of Synkro_SetReceiveMode Request service, the SynkroRF layer first checks if the node is not in Sleep mode. In this case, the function exits with *gNWDenied_c* error code.

If the node is not in sleep mode, SynkroRF tries to update the state of the radio receiver module, as specified by the parameter in the call. If the update succeeds, the function returns *gNWSuccess_c* value. Otherwise, the function returns *gNWFailed_c* error code, to inform the calling entity layer that the service request could not be successfully completed.

2.1.48 Synkro_SetPowerLevel Request

The Synkro_SetPowerLevel service makes a request for the SynkroRF layer to set the emitting power level of the node’s IEEE 802.15.4 radio transceiver. This service is available on both controller and controlled nodes.

2.1.48.1 Service Request Semantics

The semantics of the Synkro_SetPowerLevel Request service are as follows:

```
Synkro_SetPowerLevel(
         level
         )
```

Table 2-66 specifies the parameters for the Synkro_SetPowerLevel.request service on MC1321x.

NOTE

As of this release, the values for the MC1323x are yet to be determined.

Table 2-66. Synkro_SetPowerLevel Request Service Parameters on MC1321x

Name	Type	Dir	Valid range	Description
Level	uint8_t	IN	0 - 15	Specifies a value that corresponds to a desired power level.

The relationship between the level parameter and the desired transmit power is not perfectly linear (or exponential). This relationship is shown in the following table for the MC1321x device:

Table 2-67. Relationship Between the Level Parameter and the Desired Transmit Power on MC1321x

Level	Pout
0x00	-30 dB
0x01	-26 dB
0x02	-22 dB
0x03	-18 dB
0x04	-14 dB
0x05	-10 dB
0x06	-8 dB
0x07	-6 dB
0x08	-4 dB
0x09	-2 dB
0x0A	0 dB
0x0B	2 dB
0x0C	4 dB
0x0D	6 dB
0x0E	8 dB
0x0F	10 dB

Table 2-68 specifies the parameters for the Synkro_SetPowerLevel.request service on MC1322x .

Table 2-68. Synkro_SetPowerLevel Request Service Parameters on MC1322x

Name	Type	Dir	Valid range	Description
Level	uint8_t	IN	gAspPowerLevel_m30dBm_c (-30 dBm) gAspPowerLevel_4d5dBm_c (4.5 dBm)	Specifies a value that corresponds to a desired power level.

For the application to be able to call the Synkro_SetPowerLevel() API, which has a parameter with one of the enums in the gAspPowerLevel_m30dBm_c - gAspPowerLevel_4d5dBm_c range, include the `AppAspInterface.h` interface file.

The possible return values for the Synkro_SetPowerLevel Request service API call are shown in the following table.

Table 2-69. Synkro_SetPowerLevel Request API Call Return Values

Type	Possible values	Description
uint8_t	gNWInvalidParam_c gNWSuccess_c	All possible return values are fully described in Section 2.1.48.3, “Effect on Receipt”

2.1.48.2 When Generated

The Synkro_SetPowerLevel Request service is generated by the application layer when it needs to set the radio transceiver emitting power level to a desired value. The Synkro_SetPowerLevel request is transmitted to the network layer by calling the Synkro_SetPowerLevel() function in the SynkroRF API.

2.1.48.3 Effect on Receipt

On receipt of Synkro_SetPowerLevel Request service, the SynkroRF layer first checks if the desired value is between 0x00 and 0x0F. If not, the function exits with *gNWInvalidParam_c* error code.

If the parameter has a valid value, the SynkroRF tries to set the radio emitting power to the desired value and returns *gNWSuccess_c* value.

2.1.49 Synkro_IsFeatureSetAvailable Request

The Synkro_IsFeatureSetAvailable service makes a request for the SynkroRF layer to confirm if the application command set of a specified application command is supported or not by one of the nodes in the Pair Table. This service is available on both controller and controlled nodes.

2.1.49.1 Service Request Semantics

The semantics of the Synkro_IsFeatureSetAvailable Request service are as follows:

```
Synkro_IsFeatureSetAvailable(
         deviceId
         cmdIdx
         )
```

Table 2-70 specifies the parameters for the Synkro_IsFeatureSetAvailable.request service.

Table 2-70. Synkro_IsFeatureSetAvailable Request Service Parameters

Name	Type	Dir	Valid range	Description
deviceId	uint8_t	IN	0 – (SOPT 1)	Identifies the node in the Pair Table of NodeData Database for which information is requested.
cmdIdx	uint16_t	IN	1 32767	Identifier of the command the application command set it belongs to is checked to see if supported on targeted node.

The possible return values for the Synkro_IsFeatureSetAvailable Request service API call are shown in the following table.

Table 2-71. Synkro_IsFeatureSetAvailable Request API Call Return Values

Type	Possible values	Description
uint8_t	gNWInvalidParam_c gNWNodeNotStarted_c gNWDeviceIdNotPaired_c gNWUnavailable_c gNWSuccess_c	All possible return values are fully described in Section 2.1.49.3, “Effect on Receipt”

2.1.49.2 When Generated

The Synkro_IsFeatureSetAvailable Request service is generated by the application layer when it wants to find out if the application command set to which a specific application command belongs is supported or not by one of the nodes in the Pair Table of NodeData Database. The Synkro_IsFeatureSetAvailable request is transmitted to the network layer by calling the Synkro_IsFeatureSetAvailable() function in the SynkroRF API.

2.1.49.3 Effect on Receipt

First thing the SynkroRF layer performs is the validation of the parameters. If deviceId parameter exceeds the boundaries of the Pair Table in NodeData Database, *gNWInvalidParam_c* error code is returned.

Second step for SynkroRF is to check its internal state machine and status variables, to determine if it can accept or not the Synkro_IsFeatureSetAvailable request. The SynkroRF node should already be started before calling this function, so if the node is not started, the function exits with *gNWNodeNotStarted_c* error code. If in the Pair Table of NodeData Database, at the specified by deviceId parameter position, there is no information related to a paired node, the function exits with the *gNWDeviceIdNotPaired_c* error code.

If all above conditions are fulfilled, the SynkroRF will return *gNWSuccess_c* or *gNWUnavailable_c* value, depending if the application command set to which the given application command belongs is supported or not by the target node.

2.1.50 Synkro_GetPairedDeviceCapabilities Request

The Synkro_GetPairedDeviceCapabilities service makes a request for the SynkroRF layer to retrieve the application command sets supported by one of the nodes in the pair table. This service is available on both controller and controlled nodes.

2.1.50.1 Service Request Semantics

The semantics of the Synkro_GetPairedDeviceCapabilities Request service are as follows:

```
Synkro_GetPairedDeviceCapabilities(
         deviceId
featureCapabilities[MAX_DEVICE_CAPABILITIES]
         )
```

Table 2-72 specifies the parameters for the Synkro_GetPairedDeviceCapabilities Request service.

Table 2-72. Synkro_GetPairedDeviceCapabilities Request Service Parameters

Name	Type	Dir	Valid Range	Description
deviceId	uint8_t	IN	0 – (SOPT 1)	Identifies the node in the NodeData database for which information is requested.
featureCapabilities	uint8_t*	OUT	-	Address in memory of a 40 bit array where the supported application command sets of the targeted node will be copied.

The possible return values for the Synkro_GetPairedDeviceCapabilities Request service API call are shown in the following table.

Table 2-73. Synkro_GetPairedDeviceCapabilities Request API Call Return Values

Type	Possible values	Description
uint8_t	gNWInvalidParam_c gNWNodeNotStarted_c gNWDeviceIdNotPaired_c gNWSuccess_c	All possible return values are fully described in Section 2.1.50.3, "Effect on Receipt"

For more details about the how the featureCapabilities array is defined, see Section 4.2.1.1, "Pair Request Command Frame".

2.1.50.2 When Generated

The Synkro_GetPairedDeviceCapabilities Request service is generated by the application layer when information about supported application command sets of one of the nodes in the NodeData database is needed. The Synkro_GetPairedDeviceCapabilities request is transmitted to the network layer by calling the Synkro_GetPairedDeviceCapabilities () function in the SynkroRF API.

2.1.50.3 Effect on Receipt

First thing the SynkroRF layer performs is the validation of the parameters. If deviceId parameter exceeds the boundaries of the Pair Table in NodeData Database, *gNWInvalidParam_c* error code is returned.

Second step for SynkroRF is to check its internal state machine and status variables, to determine if it can accept or not the Synkro_GetPairedDeviceCapabilities request. The SynkroRF node should already be started before calling this function, so if the node is not started, the function exits with *gNWNodeNotStarted_c* error code. If in the Pair Table of NodeData Database, at the specified by deviceId parameter position, there is no information related to a paired node, the function exits with the *gNWDeviceIdNotPaired_c* error code.

If all above conditions are fulfilled, the SynkroRF copies the requested information in the location specified as parameter and returns the *gNWSuccess_c* value.

2.1.51 Synkro_GetPairedDeviceInfo Request

The Synkro_GetPairedDeviceInfo service makes a request for the SynkroRF layer to retrieve specific information about one of the nodes in the Pair Table. This service is available on both controller and controlled nodes.

2.1.51.1 Service Request Semantics

The semantics of the Synkro_GetPairedDeviceInfo Request service are as follows:

```
Synkro_GetPairedDeviceInfo(
        deviceId
        nwkVersion
        vendorId
        productId
        productVersion
        supportedConnections
        )
```

Table 2-74 specifies the parameters for the Synkro_GetPairedDeviceInfo Request service.

Table 2-74. Synkro_GetPairedDeviceInfo Request Service Parameters

Name	Type	Dir	Valid Range	Description
deviceId	uint8_t	IN	0 – (SOPT_c 1)	Identifies the node in the NodeData database for which information is requested.
nwkVersion	uint8_t*	OUT	-	Address in memory of a 16 bit variable where the SynkroRF version of the targeted node will be copied.
vendorId	uint8_t*	OUT	-	Address in memory of a 16 bit variable where the vendorId field of the targeted node will be copied.
productId	uint8_t*	OUT	-	Address in memory of a 16 bit variable where the productId field of the targeted node will be copied.
productVersion	uint8_t*	OUT	-	Address in memory of a 8 bit variable where the productVersion field of the targeted node will be copied.
supportedConnections	uint8_t*	OUT	-	Address in memory of a 8 bit variable which will be set with the maximum number of connections the targeted supports.

The possible return values for the Synkro_GetPairedDeviceInfo Request service API call are shown in the following table.

Table 2-75. Synkro_GetPairedDeviceInfo Request API Call Return Values

Type	Possible values	Description
uint8_t	gNWInvalidParam_c gNWNodeNotStarted_c gNWDeviceIdNotPaired_c gNWSuccess_c	All possible return values are fully described in Section 2.1.51.3, “Effect on Receipt”

2.1.51.2 When Generated

The Synkro_GetPairedDeviceInfo Request service is generated by the application layer when complete information about one of the nodes in the NodeData database is needed. The Synkro_GetPairedDeviceInfo request is transmitted to the network layer by calling the Synkro_GetPairedDeviceInfo() function in the SynkroRF API.

2.1.51.3 Effect on Receipt

First thing the SynkroRF layer performs is the validation of the parameters. If deviceId parameter exceeds the boundaries of the pair table in NodeData database, *gNWInvalidParam_c* error code is returned.

Second step for SynkroRF is to check its internal state machine and status variables, to determine if it can accept or not the Synkro_GetPairedDeviceInfo request. The SynkroRF node should already be started before calling this function, so if the node is not started, the function exits with *gNWNodeNotStarted_c* error code. If in the pair table of NodeData database, at the specified by deviceId parameter position there is no information related to a paired node, the function exits with the *gNWDeviceIdNotPaired_c* error code.

If all above conditions are fulfilled, the SynkroRF copies the requested information in the locations specified as parameters and returns the *gNWSuccess_c* value.

2.1.52 Synkro_GetLocalNodeInfo Request

The Synkro_GetLocalNodeInfo service makes a request for the SynkroRF layer to retrieve specific network information of the current node. This service is available on both controller and controlled nodes.

2.1.52.1 Service Request Semantics

The semantics of the Synkro_GetLocalNodeInfo Request service are as follows:

```
Synkro_GetLocalNodeInfo(
        nwkVersion
        panId
        shortAddress
        )
```

Table 2-76 specifies the parameters for the Synkro_GetLocalNodeInfo Request service.

Table 2-76. Synkro_GetLocalNodeInfo Request Service Parameters

Name	Type	Dir	Valid range	Description
nwkVersion	uint8_t*	OUT	-	Memory address of an application provided 2 byte array where the SynkroRF version of the local node will be copied.
panId	uint8_t*	OUT	-	Memory address of an application provided 2 byte array where the IEEE 802.15.4 PanId of the local node will be copied.
shortAddress	uint8_t*	OUT	-	Memory address of an application provided 2 byte array where the IEEE 802.15.4 ShortAddress of the local node will be copied.

The possible return values for the Synkro_GetLocalNodeInfo Request service API call are shown in the following table.

Table 2-77. Synkro_GetLocalNodeInfo Request API Call Return Values

Type	Possible values	Description
uint8_t	gNWNodeNotStarted_c gNWSuccess_c	All possible return values are fully described inSection 2.1.52.3, "Effect on Receipt"

2.1.52.2 When Generated

The Synkro_GetLocalNodeInfo Request service is generated by the application layer when network information about local node is needed. The Synkro_GetLocalNodeInfo request is transmitted to the network layer by calling the Synkro_GetLocalNodeInfo() function in the SynkroRF API.

2.1.52.3 Effect on Receipt

First step for SynkroRF is to check its internal state machine and status variables, to determine if it can accept or not the SynkroRF GetLocalNodeInfo request. The SynkroRF node should already be started before calling this function, so if the node is not started, the function exits with *gNWNodeNotStarted_c* error code.

If the above condition is fulfilled, the SynkroRF copies the requested information in the locations specified as parameters and returns the *gNWSuccess_c* value. If this service is called on a controller node, the returned values for the local PanId and ShortAddress will be both set to 0x0000, as a controller node does start its own PAN.

2.1.53 Synkro_GenerateNewShortAddress Request

The Synkro_GenerateNewShortAddress service makes a request for the SynkroRF layer to retrieve a randomly generated IEEE 802.15.4 short address value. This service is available only on controlled nodes.

2.1.53.1 Service Request Semantics

The semantics of the Synkro_GenerateNewShortAddress Request service are as follows:

```
Synkro_GenerateNewShortAddress(
        newShortAddress
        )
```

Table 2-78 specifies the parameters for the Synkro_GenerateNewShortAddress Request service.

Table 2-78. Synkro_GenerateNewShortAddress Request Service Parameters

Name	Type	Dir	Valid range	Description
newShortAddress	uint8_t*	OUT	-	Memory address of an application provided 2 byte array where the generated short address will be copied.

The possible return values for the Synkro_GenerateNewShortAddress Request service API call are shown in the following table.

Table 2-79. Synkro_GenerateNewShortAddress Request API Call Return Values

Type	Possible values	Description
uint8_t	gNWNodeNotStarted_c gNWSuccess_c	All possible return values are fully described in Section 2.1.53.3, "Effect on Receipt"

2.1.53.2 When Generated

The Synkro_GenerateNewShortAddress Request service is called by the application layer when it needs a random IEEE 802.15.4 shortAddress value. The Synkro_GenerateNewShortAddress request is transmitted to the network layer by calling the Synkro_GenerateNewShortAddress() function in the SynkroRF API.

2.1.53.3 Effect on Receipt

First step for SynkroRF is to check its internal state machine and status variables, to determine if it can accept or not the SynkroRF GenerateNewShortAddress request. The SynkroRF node should already be started before calling this function, so if the node is not started, the function exits with *gNWNodeNotStarted_c* error code.

If the above condition is fulfilled, the SynkroRF copies the generated information in the location specified as parameter and returns the *gNWSuccess_c* value. The generated shortAddress is random and is unique among the shortAddresses of the nodes that already exist in the pair table.

2.1.54 Synkro_GenerateNewSecurityKey Request

The Synkro_GenerateNewSecurityKey service makes a request for the SynkroRF layer to retrieve a randomly generated 128 bit value that can be used further for configuring the security mechanism of the SynkroRF layer. This service is available only on controlled nodes.

2.1.54.1 Service Request Semantics

The semantics of the Synkro_GenerateNewSecurityKey Request service are as follows:

```
Synkro_GenerateNewSecurityKey(
        newSecurityKey
        )
```

Table 2-80 specifies the parameters for the Synkro_GenerateNewSecurityKey Request service.

Table 2-80. Synkro_GenerateNewSecurityKey Request Service Parameters

Name	Type	Dir	Valid range	Description
newSecurityKey	uint8_t*	OUT	-	Memory address of an application provided 128 bit array where the generated security key will be copied.

The possible return values for the Synkro_GenerateNewSecurityKey Request service API call are shown in the following table.

Table 2-81. Synkro_GenerateNewSecurityKey Request API Call Return Values

Type	Possible values	Description
uint8_t	gNWNodeNotStarted_c gNWSuccess_c	All possible return values are fully described in Section 2.1.54.3, “Effect on Receipt”

2.1.54.2 When Generated

The Synkro_GenerateNewSecurityKey Request service is called by the application layer when it needs a random 128 bit value. The Synkro_GenerateNewSecurityKey request is transmitted to the network layer by calling the Synkro_GenerateNewSecurityKey() function in the SynkroRF API.

2.1.54.3 Effect on Receipt

First step for the SynkroRF is to check its internal state machine and status variables, to determine if it can accept or not the SynkroRF GenerateNewSecurityKey request. The SynkroRF node should already be started before calling this function, so if the node is not started, the function exits with *gNWNodeNotStarted_c* error code.

If the above condition is fulfilled, the SynkroRF copies the generated key in the location specified as parameter and returns the *gNWSuccess_c* value. The generated securityKey is random and is unique among the securityKeys of the nodes that already exist in the pair table.

2.1.55 Synkro_AddEntryInControllerPairTable Request

The Synkro_AddEntryInControllerPairTable service makes a request for the SynkroRF layer to add in the pair table of the local node, at a specified location, application provided pair information of a node. This service is available only on controller nodes.

2.1.55.1 Service Request Semantics

The semantics of the Synkro_AddEntryInControllerPairTable Request service are as follows:

```
Synkro_AddEntryInControllerPairTable(
        deviceId
        nodeEntry
        )
```

Table 2-82 specifies the parameters for the Synkro_AddEntryInControllerPairTable Request service.

Table 2-82. Synkro_AddEntryInControllerPairTable Request Service Parameters

Name	Type	Dir	Valid range	Description
deviceId	uint8_t	IN	0– (gMaxPairingTableEntries_c -1)	Identifies the position the NodeData database where the pair information should be added
nodeEntry	controllerNodeEntry_t*	IN	-	Pointer to a structure in the application's memory space containing the pair information that should be added in the pair table of the NodeData database

The possible return values for the Synkro_AddEntryInControllerPairTable Request service API call are shown in the following table.

Table 2-83. Synkro_AddEntryInControllerPairTable Request API Call Return Values

Type	Possible values	Description
int8_t	gNWNodeNotStarted_c gNWUnavailable_c gNWInvalidParam_c gNWSuccess_c	All possible return values are fully described in Section 2.1.55.3, "Effect on Receipt"

2.1.55.2 When Generated

The Synkro_AddEntryInControllerPairTable Request service is called by the application layer when it needs to add pair information of a node in the local node's pair table, without performing a pair process. The Synkro_AddEntryInControllerPairTable request is transmitted to the network layer by calling the Synkro_AddEntryInControllerPairTable() function in the SynkroRF API.

2.1.55.3 Effect on Receipt

First thing the SynkroRF layer performs is the validation of the parameters. If deviceId parameter exceeds the boundaries of the pair table in NodeData database, *gNWInvalidParam_c* error code is returned. If the function is called on a controlled node, the error code *gNWUnavailable_c* error code is returned.

Second step for SynkroRF is to check its internal state machine and status variables, to determine if it can accept or not the SynkroAddEntryInControllerPairTable request. The SynkroRF node should already be started before calling this function, so if the node is not started, the function exits with *gNWNodeNotStarted_c* error code

If all above conditions are fulfilled, the SynkroRF copies the application provided information at the specified positon in the pair table and returns the *gNWSuccess_c* value.

2.1.56 Synkro_AddEntryInControlledPairTable Request

The Synkro_AddEntryInControlledPairTable service makes a request for the SynkroRF layer to add in the pair table of the local node, at a specified location, application provided pair information of a node. This service is available only on controlled nodes.

2.1.56.1 Service Request Semantics

The semantics of the Synkro_AddEntryInControlledPairTable Request service are as follows:

```
Synkro_AddEntryInControlledPairTable(
        deviceId
        nodeEntry
        )
```

Table 2-84-specifies the parameters for the Synkro_AddEntryInControlledPairTable Request service.

Table 2-84. Synkro_AddEntryInControlledPairTable Request Service Parameters

Name	Type	Dir	Valid range	Description
deviceId	uint8_t	IN	0– (gMaxPairingTableEntries_c -1)	Identifies the position the NodeData database where the pair information should be added
nodeEntry	controlledNodeEntry_t*	IN	-	Pointer to a structure in the application's memory space containing the pair information that should be added in the pair table of the NodeData database

The possible return values for the Synkro_AddEntryInControlledPairTable Request service API call are shown in the following table.

Table 2-85. Synkro_AddEntryInControlledPairTable Request API Call Return Values

Type	Possible values	Description
uint8_t	gNWNodeNotStarted_c gNWUnavailable_c gNWInvalidParam_c gNWSuccess_c	All possible return values are fully described in Section 2.1.56.3, "Effect on Receipt"

2.1.56.2 When Generated

The Synkro_AddEntryInControlledPairTable Request service is called by the application layer when it needs to add pair information of a node in the local node's pair table, without performing a pair process. The Synkro_AddEntryInControlledPairTable request is transmitted to the network layer by calling the Synkro_AddEntryInControlledPairTable() function in the SynkroRF API.

2.1.56.3 Effect on Receipt

First thing the SynkroRF layer performs is the validation of the parameters. If deviceId parameter exceeds the boundaries of the pair table in NodeData database, *gNWInvalidParam_c* error code is returned. If the function is called on a controller node, the error code *gNWUnavailable_c* error code is returned.

Second step for SynkroRF is to check its internal state machine and status variables, to determine if it can accept or not the SynkroAddEntryInControlledPairTable request. The SynkroRF node should already be started before calling this function, so if the node is not started, the function exits with *gNWNodeNotStarted_c* error code

If all above conditions are fulfilled, the SynkroRF copies the application provided information at the specified positon in the pair table and returns the *gNWSuccess_c* value.

2.1.57 Synkro_SavePersistentData Request

The Synkro_SavePersistentData service makes a request for a SynkroRF node to update in non-volatile memory all the network sensitive information that needs to be kept between MCU resets. This service is available on both controller and controlled nodes.

2.1.57.1 Service Request Semantics

The semantics of the Synkro_SavePersistentData Request service are as follows:

```
Synkro_SavePersistentData(
        )
```

The Synkro_SavePersistentData Request does not return any value to the calling entity layer, as it always completes successfully.The Synkro_SavePersistentData Request is generated by the application layer when it needs to force the saving of the network persistent information in the non volatile memory. The Synkro_SavePersistentData Request is transmitted to the network layer by calling the Synkro_SavePersistentData() function in the SynkroRF API.

2.1.57.2 Effect on Receipt

On receipt of Synkro_SavePersistentData Request service, the SynkroRF layer entity will backup in the non volatile memory all the sensitive information that should not be lost between CPU resets.

2.1.58 Synkro_GetLastLQI Request

The Synkro_GetLastLQI service makes a request for the SynkroRF layer to retrieve the LQI of the last received packet. This service is available on both controller and controlled nodes.

2.1.58.1 Service Request Semantics

The semantics of the Synkro_GetLastLQI Request service are as follows:

```
Synkro_GetLastLQI(
        )
```

The Synkro_GetLastLQI Request returns the LQI value of the last received IEEE 802.15.4 packet.

2.1.58.2 When Generated

The Synkro_GetLastLQI Request is generated by the application layer when it needs to get the value of the LQI for the last received IEEE 802.15.4 packet. The Synkro_GetLastLQI Request is transmitted to the network layer by calling the Synkro_GetLastLQI() function in the SynkroRF API.

2.1.58.3 Effect on Receipt

On receipt of Synkro_GetLastLQI Request service, the SynkroRF layer entity will retrieve the LQI value of the last received IEEE 802.15.4 packet.

2.1.59 Synkro_SetSearchThreshold Request

The Synkro_SetSearchThreshold service makes a request for a SynkroRF node to set the minimum LQI level a future received Search Request should have to be forwarded by the network to the application. This service is available only on controlled nodes.

2.1.59.1 Service Request Semantics

The semantics of the Synkro_SetSearchThreshold Request service are as follows:

```
Synkro_SetSearchThreshold(
        threshold
        )
```

Table 2-86-specifies the parameters for the Synkro_SetSearchThreshold Request service.

Table 2-86. Synkro_SetSearchThreshold Service Parameters

Name	Type	Dir	Valid range	Description
Threshold	uint8_t	IN	0x00 -0xFF	Specifies the minimum LQI for a received Search Request SynkroRF internal command to be forward by the network to the application

The Synkro_SetSearchThreshold Request does not return any value to the calling entity layer, as it always completes successfully.

2.1.59.2 When Generated

The Synkro_SetSearchThreshold Request is generated by the application layer when the minimum LQI for the Pair Requests that should be forwarded to the application needs to be updated. The Synkro_SetSearchThreshold Request is transmitted to the network layer by calling the Synkro_SetSearchThreshold() function in the SynkroRF API.

2.1.59.3 Effect on Receipt

On receipt of Synkro_SetSearchThreshold Request service, the SynkroRF layer entity will set one internal variale used during the Search Process to the value of the API function parameter.

2.1.60 Synkro_SetPairingThreshold Request

The Synkro_SetPairingThreshold service makes a request for an SynkroRF node to set the minimum LQI level a future received Pair Request should have to be forward by the network to the application. This service is available only on controlled nodes.

2.1.60.1 Service Request Semantics

The semantics of the Synkro_SetPairingThreshold Request service are as follows:

```
Synkro_SetPairingThreshold(
        threshold
        )
```

Table 2-87 specifies the parameters for the Synkro_SetPairingThreshold Request service.

Table 2-87. Synkro_SetPairingThreshold Service Parameters

Name	Type	Dir	Valid range	Description
Threshold	uint8_t	IN	0x000xFF	Specifies the minimum LQI for a received Pair Request SynkroRF internal command to be forward by the network to the application

The Synkro_SetPairingThreshold Request does not return any value to the calling entity layer, as it always completes successfully.

2.1.60.2 When Generated

The Synkro_SetPairingThreshold Request is generated by the application layer when the minimum LQI for the Pair Requests that should be forwarded to the application needs to be updated. The Synkro_SetPairingThreshold Request is transmitted to the network layer by calling the Synkro_SetPairingThreshold() function in the SynkroRF API.

2.1.60.3 Effect on Receipt

On receipt of Synkro_SetPairingThreshold Request service, the SynkroRF layer entity will set one internal variable used during the Pair Process to the value of the API function parameter.

2.1.61 Synkro_SetCloningThreshold Request

The Synkro_SetCloningThreshold service makes a request for an SynkroRF node to set the minimum LQI level a future received Clone Request should have to be forward by the network to the application. This service is available only on controller nodes.

2.1.61.1 Service Request Semantics

The semantics of the Synkro_SetCloningThreshold Request service are as follows:

```
Synkro_SetCloningThreshold(
        threshold
        )
```

Table 2-88 specifies the parameters for the Synkro_SetCloningThreshold Request service.

Table 2-88. Synkro_SetCloningThreshold Service Parameters

Name	Type	Dir	Valid range	Description
threshold	uint8_t	IN	0x00 -0xFF	Specifies the minimum LQI for a received Clone Request SynkroRF internal command to be forward by the network to the application

The Synkro_SetCloningThreshold Request does not return any value to the calling entity layer, as it always completes successfully.

2.1.61.2 When Generated

The Synkro_SetCloningThreshold Request is generated by the application layer when the minimum LQI for the Clone Requests that should be forwarded to the application needs to be updated. The Synkro_SetCloningThreshold Request is transmitted to the network layer by calling the Synkro_SetCloningThreshold() function in the SynkroRF API.

2.1.61.3 Effect on Receipt

On receipt of Synkro_SetCloningThreshold Request service, the SynkroRF layer entity will set one internal variable used during the Clone Process to the value of the API function parameter.

2.1.62 Synkro_GetNwkStatus Request

The Synkro_GetNwkStatus service makes a request for the SynkroRF layer to communicate its functioning status to the calling entity layer. This service is available on both controller and controlled nodes.

2.1.62.1 Service Request Semantics

The semantics of the Synkro_GetNwkStatus Request service are as follows:

```
Synkro_GetNwkStatus(
        bOutOfMemory
        )
```

Table 2-89 specifies the parameters for the Synkro_GetNwkStatus Request service.

Table 2-89. Synkro_GetNwkStatus Service Parameters

Name	Type	Dir	Valid range	Description
bOutOfMemory	bool_t*	OUT	-	Specifies the location in memory of a Boolean variable the SynkroRF will set to TRUE if it has stopped functioning because of lack of memory or to FALSE if it is running in normal conditions

The Synkro_GetNwkStatus Request does not return any value to the calling entity layer, as it always completes successfully.

2.1.62.2 When Generated

The Synkro_GetNwkStatus Request is generated by the application layer when a check for the memory status of the network layer is needed. The Synkro_ GetNwkStatus Request is transmitted to the network layer by calling the Synkro_GetNwkStatus() function in the SynkroRF API.

2.1.62.3 Effect on Receipt

On receipt of Synkro_GetNwkStatus Request service, the SynkroRF layer entity will set the Boolean variable whose address in memory is provided as parameter to TRUE or FALSE, depending if it has stopped functioning from a lack of memory reason or it is functioning perfectly normal.

2.1.63 Synkro_IsIdle Request

The Synkro_IsIdle service makes a request for the SynkroRF layer to communicate if its current state is Idle or not. This service is available on both controller and controlled nodes.

2.1.63.1 Service Request Semantics

The semantics of the Synkro_IsIdle Request service are as follows:

```
Synkro_IsIdle(
        )
```

Synkro_IsIdle Request service does not pass any parameters to SynkroRF layer.

The possible return values for the Synkro_IsIdle Request service API call are shown in the following table.

Table 2-90. Synkro_IsIdle Request API Call Return Values

Type	Possible values	Description
bool_t	TRUE, FALSE	All possible return values are fully described in Section 2.1.63.3, “Effect on Receipt”

2.1.63.2 When Generated

The Synkro_IsIdle Request is generated by the application layer when it wants to see if the SynkroRF layer is in Idle state. The Synkro_IsIdle Request is transmitted to the network layer by calling the Synkro_IsIdle() function in the SynkroRF API.

2.1.63.3 Effect on Receipt

On receipt of Synkro_IsIdle Request service, the SynkroRF layer entity will verify its state. If this state is Idle, the function will return TRUE, otherwise it will return FALSE.

Chapter 3
SynkroRF Data Definitions

3.1 NodeData Database

NodeData Database is a structure defined in RAM memory that keeps information about the current node, but mostly information about the nodes that are paired with it. The information in this table is modified during some of the SynkroRF processes such as Start, Pair, RemotePair, Clone, ClearPairInformation, UpdateCapabilities or RefreshCapabilities and each time it is changed, the whole structure is saved to NVM. When a node is started, the application can choose if it wants to restore the NodeData Database structure from NVM, or if it wants have a clean version.

The NodeData Database structure is shown in Figure 3-1 and Figure 3-2 along with the byte sizes of the fields they are built into. Figure 3-1 shows the Controller Node database structure.

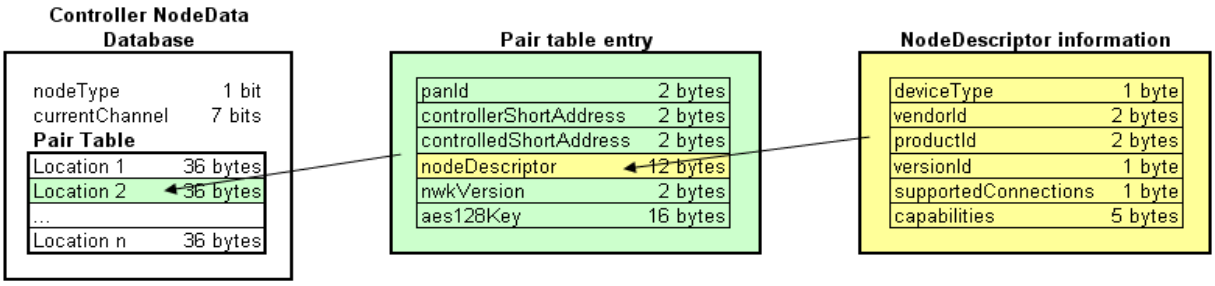

Figure 3-1. NodeData Database Structure on Controller Node

Figure 3-2 shows the Controlled Node database structure.

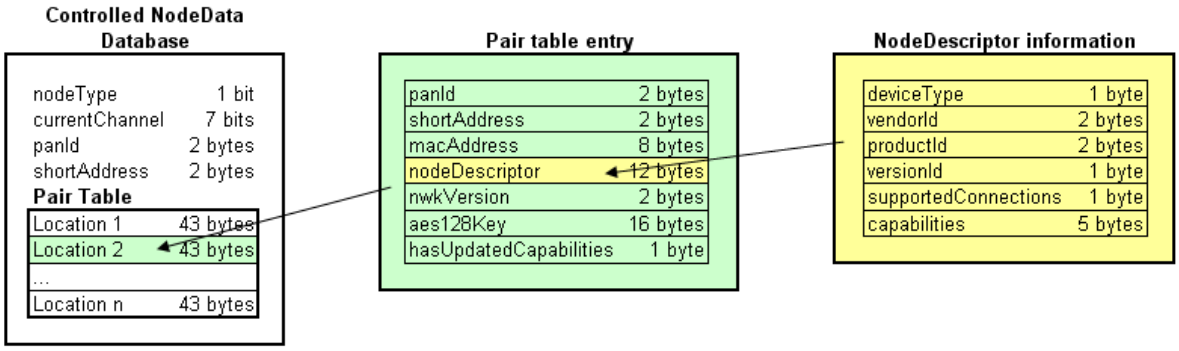

Figure 3-2. NodeData Database Structure on Controlled Node

3.2 NodeDescriptor Database

The NodeDescriptor Database structure is defined in the ROM memory which keeps information about particular characteristics of the current node. This information is defined at compile time by the application. During some of the SynkroRF processes, such as Pair, RemotePair or Clone, the information in this structure is exchanged between the nodes involved in the process.

The structure of the NodeDescriptor Database is shown in Figure 3-3, along with the byte sizes of the fields they are built into.

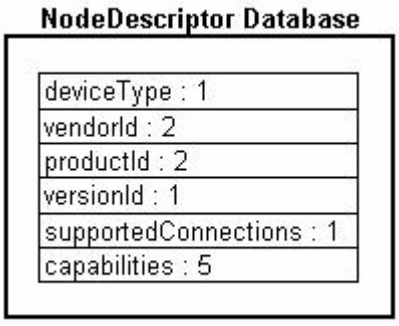

Figure 3-3. NodeDescriptor Database Structure

3.3 Constants Database

The Constants Database is a collection of SynkroRF internal defined values used for implementing network functionality. Table 3-1 lists the database values:

Table 3-1. SynkroRF Functionality Defining Constant Values

Constant	Description	Value
channelList	802.15.4 channels the SynkroRF will operate on.	152025
gChangeChannelInterval_c	Interval in ms the Channel Agility sublayer has to wait before performing a new Fast ED Scan, to determine if the current SynkroRF channel has to be changed, as result of high radio activity on it.	5000
gChangeChannelRxDelay_c	Interval in ms the Channel Agility sublayer has to wait for a packet receive indication before starting an already programmed Fast ED Scan. When the Channel Agility sublayer receives the command to start a new Fast ED Scan, it will not execute it immediately, in order not to loose a packet that is currently being received. Instead, it will wait for a packet receive indication and will start the Scan immediately after it. If no packet receive indication appears during gChangeChannelRxDelay_c interval, the Channel Agility sublayer will start the Fast ED Scan. This mechanism improves the rate of successfully received packets when receiving data at high speed rates.	200

Table 3-1. SynkroRF Functionality Defining Constant Values

gPairReqRetryIntervalTxN_c	Interval in ms to wait before retransmit a pair request transmitted in normal (not fragmented) mode, in case that no pair response was received.	50
gPairReqRetryIntervalTxFrag_c	Interval in ms to wait before retransmit a pair request transmitted in fragmented mode, in case that no pair response was received.	150
gCloneReqRetryInterval_c	Interval in ms to wait before retransmit a clone request in case that no clone response was received.	50
gCloneReceiveTimeout_c	Interval in ms to wait before canceling a clone process when clone entries are no longer received.	500
gMaxSearchPayload_c	Maximum size in bytes of the payload of a search request command.	32
gMaxReportedSearchDescriptors_c	Maximum number of nodes that can be found by a controller during a search process	4
gMaxPairCommandPayload_c	Maximum size in bytes of the payload of a pair request command.	64
gMaxAppCommandPayload_c	Maximum size in bytes of the payload of an application defined command.	90
gMaxFragmentSize_c	Maximum number of payload bytes an SynkroRF command frame can have when is sent using fragmentation.	10
gFragRxTimeout_c	Interval in ms to wait before canceling a fragmented receive session when fragments are no longer received.	50
gActiveScanDuration_c	Duration in ms of the Active Scan that is performed at the starting of a controlled SynkroRF node, to detect the other PAN coordinators active on the SynkroRF channels.	800
gEdScanDuration_c	Duration in ms of the ED Scan that is performed at the starting of a controlled SynkroRF node, to initialize the seed of the internal random number generator.	100
gFastEdScanDuration_c	Duration in ms of the Fast ED Scan the Channel Agility sublayer performs to detect if current channel needs to be changed because of the presence of a high radio activity.	12

Chapter 4
SynkroRF Frame Format

4.1 SynkroRF General Frame Format

This section describes the SynkroRF data frame format used to communicate over the air between devices in the same network. Each SynkroRF frame consists of the following basic components:

- An NHR which is comprised of frame control, sequence number and fragmentation information.
- A SynkroRF payload (variable length), which contains information specific to the command frame type.

The frames in the SynkroRF sub-layer are described as a sequence of fields in a specific order. All frame formats in this section are depicted in the order in which they are transmitted by the PHY, from left to right, where the left most bit is transmitted first in time. Bits within each field are numbered from 0 (left most and least significant) to “n” (right most and most significant), where the length of the field is “n” bits. Fields that are longer than a single octet are sent to the PHY in the order from the octet containing the lowest numbered bits to the octet containing the highest numbered bits.

The general SynkroRF frame is formatted as shown in Figure 4-1.

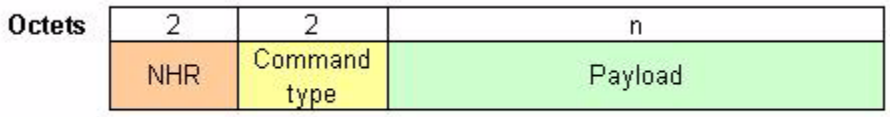

Figure 4-1. General SynkroRF Frame Format

NHR Field

The NHR field is 16 bits in length and contains information defining the frame type, fragment type and sequence number of the packet. The NHR field is formatted as shown in Figure 4-2.

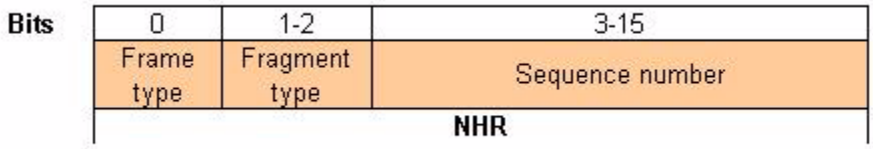

Figure 4-2. NHR Field Structure

Frame Type Field

The frame type subfield is 1 bit in length and it is always set to 1.

Fragment Type Field

The fragment type subfield is 2 bits in length and it can be set to one of the values listed in Table 4-1.

Table 4-1. Fragment Field Values

Fragment type value b1-b0	Description
00	Current frame represents a whole SynkroRF command packet
01	Current frame is the first fragment from an SynkroRF fragmented command packet
10	Current frame is a middle fragment from an SynkroRF fragmented command packet
11	Current frame is the last fragment from an SynkroRF fragmented command packet

Sequence Number Field

The sequence number field is 13 bit in length and it represents a unique identifier for the frame being sent.

The sequence number is increased by the transmitting device each time a new frame is transmitted. The field is used by the receiving device, to discard a frame that has been retransmitted by the MAC layer of the sending device.

Command Type Field

The command type field is 16 bits in length and contains information defining the command the receiving device has to execute. The SynkroRF defines three types of commands:

- SynkroRF internal commands, used during the Pair, RemotePair and Clone processes
- SynkroRF application public commands, provided by the network for the application use. These are defined in the NwkCommand.h file.
- SynkroRF application proprietary commands, that can be defined by the application. These commands must be declared in the Commands.h file and defined in the Commands.c file, in a special reserved section.

Table 4-2 contains a list of the SynkroRF supported commands.

Table 4-2. SynkroRF command types list

CommandID	Description
1 - 16384	SynkroRF application public commands
16385 - 32767	SynkroRF application proprietary commands
32768	Internal SynkroRF command - Pair request
32769	Internal SynkroRF command - Pair response
32770	Internal SynkroRF command - Remote pair request
32771	Internal SynkroRF command - Remote pair response
32772	Internal SynkroRF command - Clone request

Table 4-2. SynkroRF command types list

32773	Internal SynkroRF command - Clone response
32774	Internal SynkroRF command - Clone entry request
32775	Internal SynkroRF command - Clone entry response
32776	Internal SynkroRF command – Search request
32777	Internal SynkroRF command – Search response
32778 - 65535	Reserved

Payload Field

The payload field has a variable length, depending on the type of command frame. Considering the MAC data request maximum payload is 102 bytes, the SynkroRF command payload cannot be greater than 98 bytes.

4.2 SynkroRF Command Frames

This section describes all of the SynkroRF Command Frames and the general format.

4.2.1 SynkroRF Internal Command Frame Format

The command frames defined by the SynkroRF layer are listed in Table 4-3.

Table 4-3. SynkroRF Internal Command List

Command Id	Command name	Section
0x8000	Pair request	4.2.1.1
0x8001	Pair response	4.2.1.2
0x8002	Remote pair request	4.2.1.3
0x8003	Remote pair response	4.2.1.4
0x8004	Clone request	4.2.1.5
0x8005	Clone response	4.2.1.6
0x8006	Clone entry request	4.2.1.7
0x8007	Clone entry response	4.2.1.8
0x8008	Search request	4.2.1.9
0x8009	Search response	4.2.1.10

4.2.1.1 Pair Request Command Frame

The SynkroRF Pair Request command is composed of a CHR and a pair request command payload. Both these fields overlay the payload of the general SynkroRF frame command, as shown in Figure 4-3.

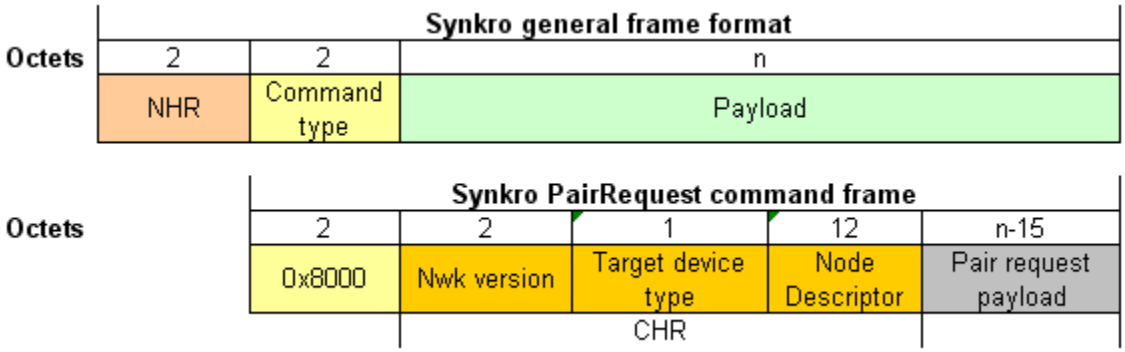

Figure 4-3. SynkroRF Pair Request Command Frame Format

SynkroRF Version Field

The SynkroRF version field is 16 bits in length and contains information defining version of SynkroRF running on the device that has transmitted the pair request. The SynkroRF version is defined as a combination of a major and a minor version. The major version is placed in the left most byte of the field, while the minor version is placed in the right most byte.

Target Device Type Field

The target device type field is 8 bits in length and contains information defining the type of device that is searched during the pairing process. The SynkroRF provides a number of public device types,that are listed in Table 4-4.

Table 4-4. SynkroRF Public Device Types List

Target device Type	Description
0	TV
1	Projector
2	DVD
3	CD
4	VCR
5	Reserved
6	AV
7	Reserved
8	DVR
9	SetTopBox
10	MediaCenter
11	GameConsole
12	SatRadio
13	OnOffLight

Table 4-4. SynkroRF Public Device Types List

14	DimmableLight
15	PortablePlayer
16	TVDVDCombo
17	HVAC
18	WindowBlind
19	RemoteControl
20	Pager
21	RFIRBridge

Node Descriptor Field

The node descriptor field is 12 bytes in length and contains information that describes the device that is requesting the pairing. The structure of the node descriptor field is detailed in Figure 4-4.

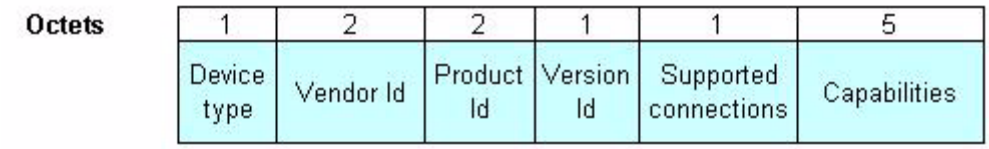

Figure 4-4. Node Descriptor Field Structure

Device Type Field

The device type field is 8 bits in length and contains information defining the device type of the sender. The device type field is set to one of the values in listed in Table 4-4 or can be an application proprietary defined value, in the range 22-255.

Vendor ID Field

The vendor id field is 16 bits in length and contains information about the vendor of the device that is sending the packet.

Product Id Field

The product id field is 16 bits in length and contains information about the product on the device that is sending the packet. It can be any value in the range 0x0000 – 0xFFFF.

Version Id Field

The version id field is 8 bits in length and contains information about the version of the product on the device that is sending the packet. It can be any value in the range 0x00 – 0xFF.

Supported Connections Field

The supported connection field is 8 bits in length and specifies how many devices the node transmitting the frame has space to pair with. The maximum number of supported connections is limited to 40, so this field can have any value in the range 0 – 40.

Capabilities Field

The capabilities field is 40 bits in length and specifies the application command sets that are supported the sending device. In the current version of SynkroRF, a number of 20 public command sets are already defined in the NwkCommands.h file. Also, an application proprietary defined command set is available and it is placed on the last position in the capabilities map.The command sets defined in the current version of SynkroRF are listed in Table 4-5.

Table 4-5. SynkroRF Command Set List

Bit number in capabilities map	Command set name
1	PublicNumbers
2	PublicKeyboard
3	PublicChannels
4	PublicAudioSettings
5	PublicVideoSettings
6	PublicDeviceSettings
7	PublicNavigation
8	PublicDeviceInputs
9	PublicDeviceGeneral
10	DTVProjectorsVideoSetings
11	DTVProjectorsPictureInPicture
12	DTVProjectorDeviceSettings
13	DvdCdVcrCasseteVideoSettings
14	DvdCdVcrCasseteMediaSettings
15	AVReceiverAudiosettings
16	AVReceiverSpeakerSettings
17	AVReceiverDevicesettings
18	AVReceiverPortableDeviceSettings
19	PvrDvrSetTopBoxMediaCenter
20	PublicGeneralController
21 - 39	Reserved
40	UserDefinedCommand

The SynkroRF public application commands and the command sets they belong to are listed in Table 4-6.

Table 4-6. SynkroRF Public Application Command List

Command Set	Command ID	Command name	From controller to controlled	From controlled to controller	From controlled to controlled	Payload
Public Numbers	1	gCmdDot_c	x			None
	2	gCmdNum0_c	x			None
	3	gCmdNum1_c	x			None
	4	gCmdNum2_c	x			None
	5	gCmdNum3_c	x			None
	6	gCmdNum4_c	x			None
	7	gCmdNum5_c	x			None
	8	gCmdNum6_c	x			None
	9	gCmdNum7_c	x			None
	10	gCmdNum8_c	x			None
	11	gCmdNum9_c	x			None
	12	gCmdNumN_c	x			None
	13	gCmdNumGreater10_c	x			None
	14	gCmdNumGreater100_c	x			None
	15	gCmdNumGreater1000_c	x			None
	16	gCmdNumNDotNumN_c	x			None
	17	gCmdNumEnt	x			None
Public Keyboard	30	gCmdAscii_c	x			None
	31	gCmdFunctionKey_c	x			None
	32	gCmdControlKey_c	x			None
	33	gCmdAltKey_c	x			None

Table 4-6. SynkroRF Public Application Command List (continued)

	34	gCmdInsertKey_c	x			None
	35	gCmdDeleteKey_c	x			None
	36	gCmdHomeKey_c	x			None
	37	gCmdPageUpKey_c	x			None
	38	gCmdPageDownKey_c	x			None
	39	gCmdWindowsKey_c	x			None
	40	gCmdAppleKey_c	x			None
	41	gCmdPrintScreenKey_c	x			None
	42	gCmdSystemRequestKey_c	x			None
	43	gCmdScrollLockKey_c	x			None
	44	gCmdNumberLockKey_c	x			None
	45	gCmdPauseKey_c	x			None
	46	gCmdBreakKey_c	x			None
Public Channels	50	gCmdAddChannel_c	x			None
	51	gCmdDelChannel_c	x			None
	52	gCmdAutoPgm_c	x			None
	53	gCmdChangeChannel_c	x			BinaryMax5Bytes
	54	gCmdChannelDown_c	x			None
	55	gCmdChannelUp_c	x			None
	56	gCmdLastChannel_c	x			None
Public Audio Settings	60	gCmdAudio_c	x			None
	61	cCmdClosedCaption_c	x			None
	62	gCmdMute_c	x			None
	63	gCmdSap_c	x			None
	64	gCmdSurroundMode_c	x			None
	65	gCmdVolumeDown_c	x			None
	66	gCmdVolumeUp_c	x			None
Public Video Settings	70	gCmdFastForward_c	x			None

Table 4-6. SynkroRF Public Application Command List (continued)

	71	gCmdFastForwardN_c	x			None
	72	gCmdPause_c	x			None
	73	gCmdPauseN_c	x			None
	74	gCmdPlay_c	x			None
	75	gCmdPlayN_c	x			None
	76	gCmdPlayPause_c	x			None
	77	gCmdRecord_c	x			None
	78	gCmdRecordN_c	x			Binary1ByteMaxValue255
	79	gCmdReplay_c	x			None
	80	gCmdRewind_c	x			None
	81	gCmdRewindN_c	x			None
	82	gCmdSlow_c	x			None
	83	gCmdSlowBack_c	x			None
	84	gCmdSlowForward_c	x			None
	85	gCmdSlowPlay_c	x			None
	86	gCmdStop_c	x			None
	87	gCmdStopN_c	x			None
	88	gCmdZoomIn_c	x			None
	89	gCmdZoomOut_c	x			None
Public Device Settings	100	gCmdAspectRatio_c	x			None
	101	gCmdBlue_c	x			None
	102	gCmdGreen_c	x			None
	103	gCmdPowerOn_c	x			None
	104	gCmdPowerOff_c	x			None
	105	gCmdPowerToggle_c	x			None
	106	gCmdRed_c	x			None
	107	gCmdReset_c	x			None
	108	gCmdResolution_c	x			Binary1ByteMaxValue9

Table 4-6. SynkroRF Public Application Command List (continued)

	109	gCmdSetup_c	x			None
	110	gCmdTools_c	x			None
	111	gCmdYellow_c	x			None
Public Navigation	120	gCmdDirectionDown_c	x			None
	121	gCmdDirectionLeft_c	x			None
	122	gCmdDirectionRight_c	x			None
	123	gCmdDirectionUp_c	x			None
	124	gCmdDirection_c	x			Binary1ByteMaxValue4
	125	gCmdDirectionDown2_c	x			None
	126	gCmdDirectionLeft2_c	x			None
	127	gCmdDirectionRight2_c	x			None
	128	gCmdDirectionUp2_c	x			None
	129	gCmdDirection2_c	x			Binary1ByteMaxValue4
	130	gCmdMenuDown_c	x			None
	131	gCmdMenuLeft_c	x			None
	132	gCmdMenuRight_c	x			None
	133	gCmdMenuUp_c	x			None
	134	gCmdMenuOff_c	x			None
	135	gCmdMenuOn_c	x			None
	136	gCmdMenuToggle_c	x			None
	137	gCmdMenuSelect_c	x			None
	138	gCmdSelectOk_c	x			None
Public Device Inputs	150	gCmdInputNext_c	x			None
	151	gCmdInputPrev_c	x			None
	152	gCmdInputSelectN_c	x			Binary1ByteMaxValue255
	153	gCmdInputAm_c	x			None
	154	gCmdInputAmN_c	x			Binary1ByteMaxValue255
	155	gCmdInputAnalog_c	x			None

Table 4-6. SynkroRF Public Application Command List (continued)

	156	gCmdInputAnalogN_c	x			Binary1ByteMaxValue255
	157	gCmdInputAntenna_c	x			None
	158	gCmdInputAntennaN_c	x			Binary1ByteMaxValue255
	159	gCmdInputAux_c	x			None
	160	gCmdInputAuxN_c	x			Binary1ByteMaxValue255
	161	gCmdInputAuto_c	x			None
	162	gCmdInputCable_c	x			None
	163	gCmdInputCableN_c	x			Binary1ByteMaxValue255
	164	gCmdInputCard_c	x			None
	165	gCmdInputCardN_c	x			Binary1ByteMaxValue255
	166	gCmdInputCd_c	x			None
	167	gCmdInputCdN_c	x			Binary1ByteMaxValue255
	168	gCmdInputCdr_c	x			None
	169	gCmdInputCdrN_c	x			Binary1ByteMaxValue255
	170	gCmdInputComponent_c	x			None
	171	gCmdInputComponentN_c	x			Binary1ByteMaxValue255
	172	gCmdInputDigital_c	x			None
	173	gCmdInputDigitalN_c	x			Binary1ByteMaxValue255
	174	gCmdInputDvd_c	x			None
	175	gCmdInputDvdN_c	x			Binary1ByteMaxValue255
	176	gCmdInputDvi_c	x			None
	177	gCmdInputDviN_c	x			Binary1ByteMaxValue255
	178	gCmdInputFm_c	x			None
	179	gCmdInputFmN_c	x			Binary1ByteMaxValue255
	180	gCmdInputFront_c	x			None
	181	gCmdInputFrontN_c	x			Binary1ByteMaxValue255
	182	gCmdInputHdd_c	x			None
	183	gCmdInputHddN_c	x			Binary1ByteMaxValue255
	184	gCmdInputHdmi_c	x			None

Table 4-6. SynkroRF Public Application Command List (continued)

	185	gCmdInputHdmiN_c	x			Binary1ByteMaxValue255
	186	gCmdInputPortable_c	x			None
	187	gCmdInputPortableN_c	x			Binary1ByteMaxValue255
	188	gCmdInputLd_c	x			None
	189	gCmdInputLdN_c	x			Binary1ByteMaxValue255
	190	gCmdInputMd_c	x			None
	191	gCmdInputMdN_c	x			Binary1ByteMaxValue255
	192	gCmdInputMemoryCard_c	x			None
	193	gCmdInputMemoryCardN_c	x			Binary1ByteMaxValue255
	194	gCmdInputPc_c	x			None
	195	gCmdInputPcN_c	x			Binary1ByteMaxValue255
	196	gCmdInputPhone_c	x			None
	197	gCmdInputPhono	x			None
	198	gCmdInputSatRadio_c	x			None
	199	gCmdInputSatRadioN_c	x			Binary1ByteMaxValue255
	200	gCmdInputSatTv_c	x			None
	201	gCmdInputSatTvN_c	x			Binary1ByteMaxValue255
	202	gCmdInputSVideo_c	x			None
	203	gCmdInputSVideoN_c	x			Binary1ByteMaxValue255
	204	gCmdInputTapeDat_c	x			None
	205	gCmdInputTapeDatN_c	x			Binary1ByteMaxValue255
	206	gCmdInputTuner_c	x			None
	207	gCmdInputTunerN_c	x			Binary1ByteMaxValue255
	208	gCmdInputTv_c	x			None
	209	gCmdInputTvN_c	x			Binary1ByteMaxValue255
	210	gCmdInputUsb_c	x			None
	211	gCmdInputUsbN_c	x			Binary1ByteMaxValue255
	212	gCmdInputVcr_c	x			None
	213	gCmdInputVcrN_c	x			Binary1ByteMaxValue255

Table 4-6. SynkroRF Public Application Command List (continued)

	214	gCmdInputVideo_c	x			None
	215	gCmdInputVideoN_c	x			Binary1ByteMaxValue255
	216	gCmdTvVideo_c	x			None
Public Device General	250	gCmdCancel_c	x			None
	251	gCmdClear_c	x			None
	252	gCmdDisplay_c	x			None
	253	gCmdEnter_c	x			None
	254	gCmdExit_c	x			None
	255	gCmdHelp_c	x			None
	256	gCmdInfo_c	x			None
	257	gCmdTime_c	x			None
	258	gCmdDimmerDown_c	x			None
	259	gCmdDimmerUp_c	x			None
	260	gCmdDimmer_c	x			None
	261	gCmdSleepTimer_c	x			Binary1ByteMaxValue255
	262	gCmdSleepTimerDown_c	x			None
	263	gCmdSleepTimerUp_c	x			None
	264	gCmdGetStatus_c	x			None
DTV Projectors Video Settings	300	gCmd16x9_c	x			None
	301	gCmd4x3_c	x			None
	302	gCmdTeleText_c	x			None
DTV Projectors Picture In Picture	310	gCmdPiP_c	x			None
	311	gCmdPiPChannelDown_c	x			None
	312	gCmdPiPChannelUp_c	x			None
	313	gCmdPiPInputDown_c	x			None
	314	gCmdPiPInputUp_c	x			None
	315	gCmdPiPZoomIn_c	x			None

Table 4-6. SynkroRF Public Application Command List (continued)

	316	gCmdPiPZoomOut_c	x			None
	317	gCmdPiPSwap_c	x			None
	318	gCmdPiPToggle_c	x			None
DTV Projectors Device Settings	330	gCmdBrightnessDown_c	x			None
	331	gCmdBrightnessUp_c	x			None
	332	gCmdColorDown_c	x			None
	333	gCmdColorUp_c	x			None
	334	gCmdContrastDown_c	x			None
	335	gCmdContractUp_c	x			None
	336	gCmdKeystone_c	x			Binary1ByteMaxValue4
	337	gCmdSharpnessDown_c	x			None
	338	gCmdSharpnessUp_c	x			None
Dvd Cd Vcr Cassete Video Settings	400	gCmdAngle_c	x			None
	401	gCmdAutoTrack_c	x			None
	402	gCmdBookmark_c	x			None
	403	gCmdChapterNext_c	x			None
	404	gCmdChapterPrev_c	x			None
	405	gCmdChapterMark_c	x			None
	406	gCmdCommercialAdv_c	x			None
	407	gCmdCopy_c	x			None
	408	gCmdCounter_c	x			None
	409	gCmdCounterReset_c	x			None
	410	gCmdFrameNext_c	x			None
	411	gCmdFramePrev_c	x			None
	412	gCmdFocusDown_c	x			None
	413	gCmdFocusUp_c	x			None
	414	gCmdJogLeft_c	x			None
	415	gCmdJogRight_c	x			None

Table 4-6. SynkroRF Public Application Command List (continued)

	416	gCmdIndexDown	x			None
	417	gCmdIndexUp	x			None
	418	gCmdManTrackDown_c	x			None
	419	gCmdManTrackUp_c	x			None
	420	gCmdOneTouchCopy_c	x			None
	421	gCmdOneTouchRecord_c	x			None
	422	gCmdProgressiveToggle_c	x			None
	423	gCmdRecordSpeed_c	x			Binary1ByteMaxValue3
	424	gCmdSearch_c	x			None
	425	gCmdSearchMode_c	x			None
	426	gCmdSet_c	x			None
	427	gCmdSkipBlank_c	x			None
	428	gCmdStepBack_c	x			None
	429	gCmdStepForward_c	x			None
	430	gCmdTimerRecord_c	x			None
	431	gCmdUpconvert_c	x			None
	432	gCmdVcrPlus_c	x			None
Dvd Cd Vcr Cassete Media Settings	445	gCmdDisc1_c	x			None
	446	gCmdDisc2_c	x			None
	447	gCmdDisc3_c	x			None
	448	gCmdDisc4_c	x			None
	449	gCmdDisc5_c	x			None
	450	gCmdDisc6_c	x			None
	451	gCmdDiscN_C	x			Binary1ByteMaxValue255
	452	gCmdDiscNext_c	x			None
	453	gCmdDiscPrev_c	x			None
	454	gCmdMagazineSelect_c	x			Binary1ByteMaxValue255
	455	gCmdOpenClose_c	x			None

Table 4-6. SynkroRF Public Application Command List (continued)

	456	gCmdOpenCloseN_c	x			None
	457	gCmdProgram_c	x			None
	458	gCmdRandom_c	x			None
	459	gCmdRepeat_c	x			None
	460	gCmdRepeatA_B_c	x			None
	461	gCmdShuffle_c	x			None
	462	gCmdShuttleNext_c	x			None
	463	gCmdShuttlePrev_c	x			None
	464	gCmdSubTitle_c	x			None
	465	gCmdTitleList_c	x			None
	466	gCmdTopMenu_c	x			None
	467	gCmdTrackNext_c	x			None
	468	gCmdTrackPrev_c	x			None
AV Receiver Audio Settings	486	gCmdEqualizer_c	x			None
	487	gCmdBassDown_c	x			None
	488	gCmdBassUp_c	x			None
	489	gCmdLoudness_c	x			None
	490	gCmdTrebleDown_c	x			None
	491	gCmdTrebleUp_c	x			None
	492	gCmdMode3ChLogic_c	x			None
	493	gCmdMode5ChStereo_c	x			None
	494	gCmdModeAC3_c	x			None
	495	gCmdModeAnalogDigital_c	x			None
	496	gCmdModeAutoSurround_c	x			None
	497	gCmdModeDirect_c	x			None
	498	gCmdDolbyDts_c	x			None
	499	gCmdModeDolbySurround_c	x			None
	500	gCmdModeDts_c	x			None

Table 4-6. SynkroRF Public Application Command List (continued)

	501	gCmdModeProLogic_c	x			None
	502	gCmdModeStereo_c	x			None
	503	gCmdModeThx_c	x			None
	504	gCmdModeToggle_c	x			None
	505	gCmdNoisereduction_c	x			None
	506	gCmdSoundEffectOff_c	x			None
	507	gCmdSoundEffectOn_c	x			None
	508	gCmdSoundEffectNext_c	x			None
	509	gCmdSoundEffectPrev_c	x			None
	510	gCmdChurchEffect_c	x			None
	511	gCmdCinemaEffect_c	x			None
	512	gCmdConcertEffect_c	x			None
	513	gCmdConcertHallEffect_c	x			None
	514	gCmdJazzEffect_c	x			None
	515	gCmdMovieEffect_c	x			None
	516	gCmdMusicEffect_c	x			None
	517	gCmdMusicalEffect_c	x			None
	518	gCmdRockEffect_c	x			None
	519	gCmdVideoEffect_c	x			None
AV Receiver Speaker Settings	550	gCmdBalanceLeft_c	x			None
	551	gCmdBalanceRight_c	x			None
	552	gCmdCenterLevelDown_c	x			None
	553	gCmdCenterLevelUp_c	x			None
	554	gCmdCenterDelayDown_c	x			None
	555	gCmdCenterDelayUp_c	x			None
	556	gCmdFrontLevelDown_c	x			None
	557	gCmdFrontLevelUp_c	x			None
	558	gCmdFrontDelayDown_c	x			None

Table 4-6. SynkroRF Public Application Command List (continued)

	559	gCmdFrontDelayUp_c	x			None
	560	gCmdRear1LevelDown_c	x			None
	561	gCmdRear1LevelUp_c	x			None
	562	gCmdRear1DelayDown_c	x			None
	563	gCmdRear1DelayUp_c	x			None
	564	gCmdRearNLevelDown_c	x			None
	565	gCmdRearNLevelUp_c	x			None
	566	gCmdRearNDelayDown_c	x			None
	567	gCmdRearNDelayUp_c	x			None
	568	gCmdSubLevelDown_c	x			None
	569	gCmdSubLevelUp_c	x			None
	570	gCmdChLevelDown_c	x			Binary1ByteMaxValue5
	571	gCmdChLevelUp_c	x			Binary1ByteMaxValue5
	572	gCmdSpeakerA_c	x			None
	573	gCmdSpeakerB_c	x			None
	574	gCmdSpeakerN_c	x			Binary1ByteMaxValue2
	575	gCmdSpeakerToggle_c	x			None
	576	gCmdTestTone_c	x			None
	577	gCmdTestToneFront_c	x			None
	578	gCmdTestToneCenter_c	x			None
	579	gCmdTestToneRear_c	x			None
	580	gCmdTestToneRearN_c	x			None
	581	gCmdTestToneSub_c	x			None
AV Receiver Device Settings	600	gCmdAmFmToggle_c	x			None
	601	gCmdMultiZone_c	x			None
	602	gCmdMultiZoneN_c	x			Binary1ByteMaxValue255
	603	gCmdPresetSelect_c	x			Binary1ByteMaxValue255
	604	gCmdPresetDown_c	x			None

Table 4-6. SynkroRF Public Application Command List (continued)

	605	gCmdPresetUp_c	x			None
	606	gCmdTuneDown_c	x			None
	607	gCmdTuneUp_c	x			None
AV Receiver Portable Device Settings	620	gCmdPortableDirectionDown_c	x			None
	621	gCmdPortableDirectionLeft_c	x			None
	622	gCmdPortableDirectionRight_c	x			None
	623	gCmdPortableDirectionUp_c	x			None
	624	gCmdPortableFastForward_c	x			None
	625	gCmdPortableMenu_c	x			None
	626	gCmdPortablePause_c	x			None
	627	gCmdPortablePlay_c	x			None
	628	gCmdPortableRewind_c	x			None
	629	gCmdPortableSkipBack_c	x			None
	630	gCmdPortableSkipForward_c	x			None
	631	gCmdPortableStop_c	x			None
	632	gCmdPortableVolumeDown_c	x			None
	633	gCmdPortableVolumeUp_c	x			None
	634	gCmdPortableGetPlaylistNum_c	x			None
	635	gCmdPortableGetPlaylistName_c	x			BinaryMax5Bytes
	636	gCmdPortableSelectPlaylistNum_c	x			BinaryMax5Bytes
	637	gCmdPortableGetCurrentPlaylistNum_c	x			None
Pvr Dvr Set Top Box Media Center	640	gCmdAdvance_c	x			None
	641	gCmdForward_c	x			None
	642	gCmdGuide_c	x			None
	643	gCmdLiveTv_c	x			None
	644	gCmdMedia_c	x			None

Table 4-6. SynkroRF Public Application Command List (continued)

	645	gCmdMyMusic_c	x			None
	646	gCmdMyPictures_c	x			None
	647	gCmdMyRadio_c	x			None
	648	gCmdMyTv_c	x			None
	649	gCmdMyVideos_c	x			None
	650	gCmdOK_c	x			None
	651	gCmdRecordedTv_c	x			None
	652	gCmdSkip_c	x			None
	653	gCmdVod_c	x			None
	654	gCmdPpv_c	x			None
	655	gCmdFav_c	x			None
	656	gCmdPageDown_c	x			None
	657	gCmdPageUp_c	x			None
	658	gCmdThumbsDown_c	x			None
	659	gCmdThumbsUp_c	x			None
	660	gCmdActive_c	x			None
Public General Controlled	700	gCmdCategory_c	x			BinaryMax90Ascii
	701	gCmdChName_c	x			BinaryMax90Ascii
	702	gCmdChNum_c	x			Binary2Bytes
	703	gCmdArtist_c	x			BinaryMax90Ascii
	704	gCmdSong_c	x			BinaryMax90Ascii
	705	gCmdVol_c	x			Binary1ByteMaxValue255
	706	gCmdSource_c	x			BinaryMax90Ascii
	707	gCmdMovie_c	x			BinaryMax90Ascii
	708	gCmdPower_c	x			BinaryMax90Ascii
	709	gCmdStatus_c	x			BinaryMax90Ascii
	710	gCmdDisplayString_c	x		x	BinaryMax90Ascii
	711	gCmdPlaylistNum_c	x		x	BinaryMax5Bytes

Table 4-6. SynkroRF Public Application Command List (continued)

	712	gCmdSetPlaylistName_c	x		x	BinaryMax90Ascii
	713	gCmdCurrentPlaylist_c	x		x	BinaryMax5Bytes
Always Implemented	16368	gCmdUpdateCapabilities_c	x		x	BinaryCapabilitiesSizeInBytes
	16369	gCmdRefreshCapabilities_c	x			None
	16370	gCmdPollReq_c	x			Binary2Bytes
	16371	gCmdPollResp_c		x		Binary1ByteMaxValue2
	16372	gCmdBulkDataStartReq_c	x	x	x	gPayloadTypeBinaryMax5Bytes_c
	16373	gCmdBulkDataStartResp_c	x	x	x	gPayloadTypeBinaryMax5Bytes_c
	16374	gCmdBulkDataReq_c	x	x	x	gPayloadTypeBinaryMaxAppBulk_c
	16375	gCmdBulkDataResp_c	x	x	x	gPayloadTypeNone_c

Even if the payload types of the commands have self describing names, the following is a list and a brief description of each what each value means.

None: no payload bytes

Binary1ByteMaxValue2 1 byte of payload, data payload interval: [0..1], binary encoded.

Binary1ByteMaxValue3 1 byte of payload, data payload interval: [0..2], binary encoded.

Binary1ByteMaxValue4 1 byte of payload, data payload interval: [0..3], binary encoded.

Binary1ByteMaxValue5 1 byte of payload, data payload interval: [0..4], binary encoded.

Binary1ByteMaxValue9 1 byte of payload, data payload interval: [0..8], binary encoded.

Binary1ByteMaxValue12 1 byte of payload, data payload interval: [0..11], binary encoded.

Binary1ByteMaxValue255 1 byte of payload, data payload interval: [0..254], binary encoded.

Binary2Bytes 2 bytes of payload, binary encoded.

BinaryMax5Bytes Up to 5 bytes of payload, binary encoded.

BinaryCapabilitiesSizeInBytes 5 bytes of payload, binary encoded.

Binary1Ascii 1 byte of payload, ASCII encoded.

BinaryMax90Ascii Up to 90 bytes of payload, ASCII encoded.

Payload Field

The payload field has a variable length. Considering that an SynkroRF command frame maximum payload is 98 bytes, and that the size of the Pair Request command header is 15, the maximum value of the Pair Request payload is limited to 83 bytes.

4.2.1.2 Pair Response Command Frame

The SynkroRF Pair Response command is composed of a CHR and a pair response command payload. Both of these fields overlay the payload of the general SynkroRF frame command, as shown in Figure 4-5.

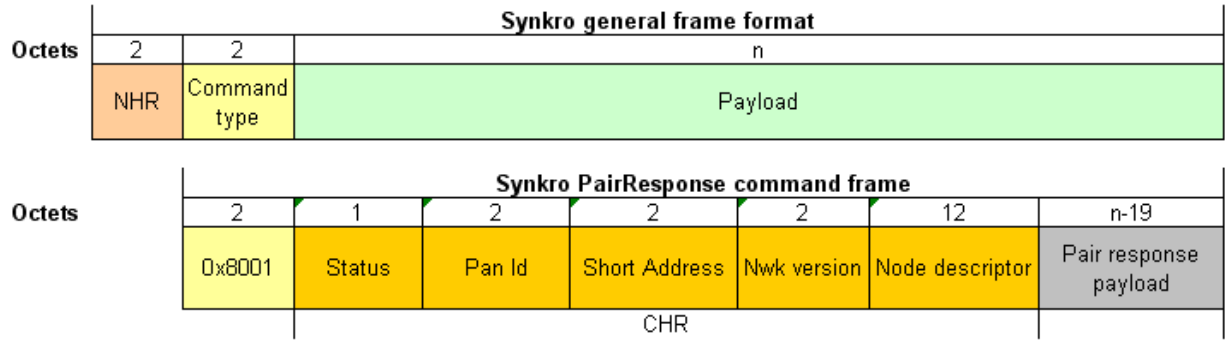

Figure 4-5. SynkroRF Pair Response Command Frame Format

Status Field

The status field is 8 bits in length and specifies if the device that has send the pair response is accepting the pairing or is refusing it. A value of 0x01 in this field means the pair request is accepted, while a value of 0x00 means that the pair request is rejected by the addressed device.

Pan Id Field

The Pan Id field is 16 bits in length and contains the Pan Id of the device sending the pair response (the controlled device).This is the Pan Id the pair requesting device should use after the pairing process to communicate with the paired device.

Short Address Field

The Short Address field is 16 bits in length and contains the short address the pair requesting device will have after the pairing process.

SynkroRF Version Field

The SynkroRF version field is 16 bits in length and contains information defining version of SynkroRF running on the device that has transmitted the pair response. The SynkroRF version is defined as a combination of a major and a minor version. The major version is placed in the left most byte of the field, while the minor version is placed in the right most byte.

Node Descriptor Field

The node descriptor field is 12 bytes in length and contains information that describes the device that is responding the pairing request. The structure of the node descriptor field is detailed in Figure 4-4.

Payload Field

The payload field has a variable length. Considering that an SynkroRF command frame maximum payload is 98 bytes, and that the size of the Pair Response command header is 19, the maximum value of the Pair Request payload is limited to 79 bytes.

4.2.1.3 Remote Pair Request Command Frame

The SynkroRF Remote Pair Request command is composed only by a CHR. The CHR field overlays the payload of the general SynkroRF frame command, as shown in Figure 4-6.

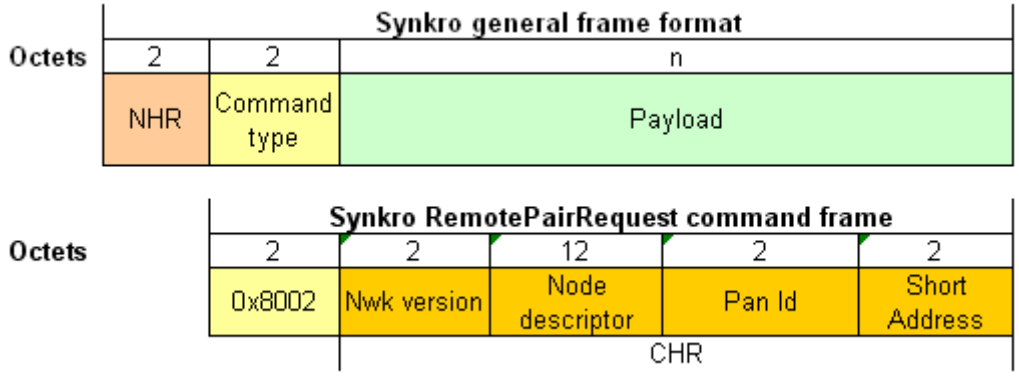

Figure 4-6. SynkroRF Remote Pair Request Command Frame Format

SynkroRF Version Field

The SynkroRF version field is 16 bits in length and contains information defining version of SynkroRF running on the device that has transmitted the remote pair request. The SynkroRF version is defined as a combination of a major and a minor version. The major version is placed in the left most byte of the field, while the minor version is placed in the right most byte.

Node Descriptor Field

The node descriptor field is 12 bytes in length and contains information that describes the device that is intended to be paired with the device the command is addressed to. The structure of the node descriptor field is detailed in Figure 4-4.

Pan Id Field

The Pan Id field is 16 bits in length and contains the Pan Id of the device that is intended to be paired with the device the command is addressed to.

Short Address Field

The Short Address field is 16 bits in length and contains the short address of the device that is intended to be paired with the device the command is addressed to.

The remote Pair Request command does not support any payload.

4.2.1.4 Remote Pair Response Command Frame

The SynkroRF Remote Pair Response command is composed only by a CHR. The CHR field overlays the payload of the general SynkroRF frame command, as shown in Figure 4-7.

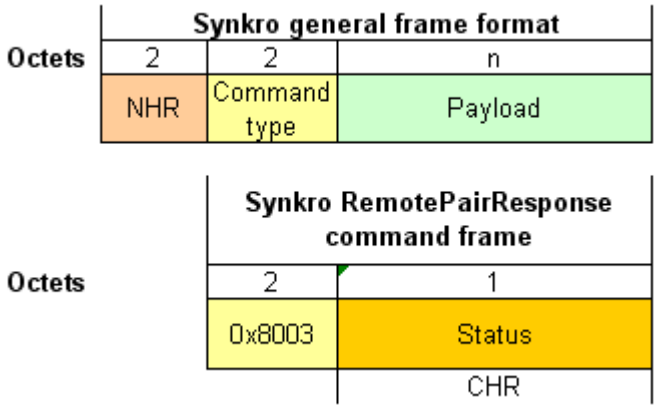

Figure 4-7. SynkroRF Remote Pair Response Command Frame Format

Status Field

The status field is 8 bits in length and specifies if the device that has send the remote pair response is accepting the pairing or is refusing it. A value of 0x01 in this field means the remote pair request was accepted, while a value of 0x00 means that the remote pair request was rejected by the addressed device.

NOTE

The Remote Pair Response command does not support any payload.

4.2.1.5 Clone Request Command Frame

The SynkroRF Clone Request command is composed only by a CHR. The CHR field overlays the payload of the general SynkroRF frame command, as shown in Figure 4-8.

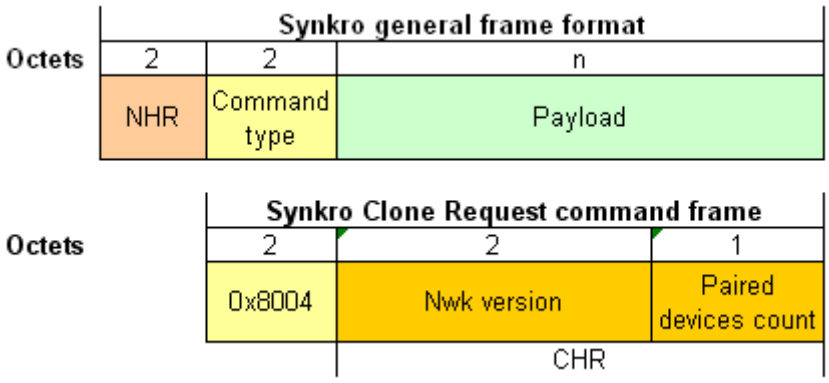

Figure 4-8. SynkroRF Clone Request command frame format

SynkroRF Version Field

The SynkroRF version field is 16 bits in length and contains information defining version of SynkroRF running on the device that has transmitted the clone request. The SynkroRF version is defined as a combination of a major and a minor version. The major version is placed in the left most byte of the field, while the minor version is placed in the right most byte.

Paired Devices Count Field

The paired devices count field is 8 bits in length and it contains the number of devices the clone request transmitting device is paired with.

NOTENOTE

The Clone Request command does not support any payload.

4.2.1.6 Clone Response Command Frame

The SynkroRF Clone Response command is composed only by a CHR. The CHR field overlays the payload of the general SynkroRF frame command, as shown in Figure 4-9.

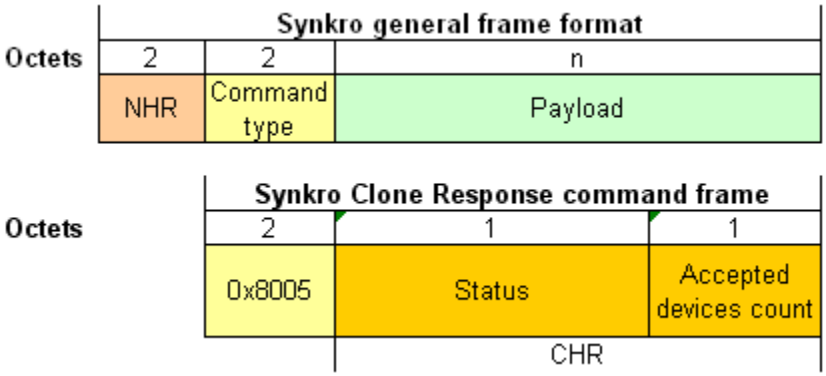

Figure 4-9. SynkroRF Clone Response Command Frame Format

Status Field

The status field is 8 bits in length and specifies if the device that has send the clone response is accepting the cloning or is refusing it. A value of 0x01 in this field means the clone request was accepted, while a value of 0x00 means that the clone request was rejected by the addressed device.

Accepted Devices Count Field

The accepted devices count field is 8 bits in length and it contains the number of devices the clone response transmitting device is ready to accept during a clone process.

NOTE

The Clone Response command does not support any payload.

4.2.1.7 Clone Entry Request Command Frame

The SynkroRF Clone Entry request command is composed only by a CHR. The CHR field overlays the payload of the general SynkroRF frame command, as shown in Figure 4-10.

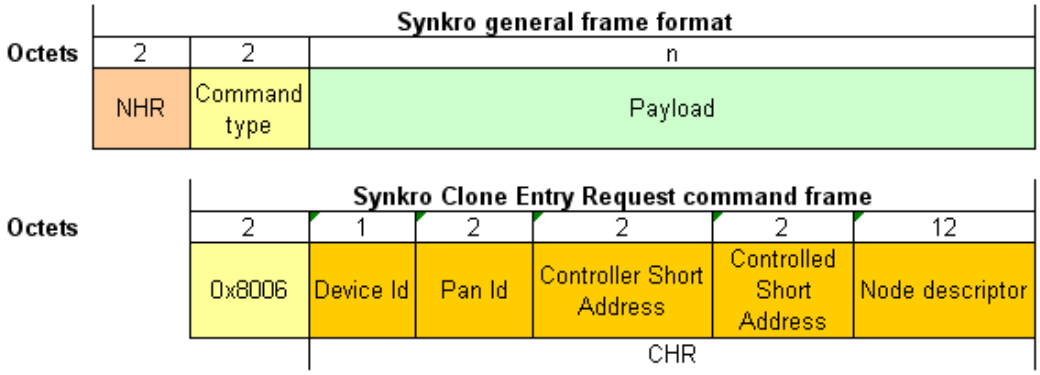

Figure 4-10. SynkroRF Clone Entry Request Command Frame Format

Device Id Field

The Device Id field is 8 bits in length and contains the position in the pair table of the clone requesting device where the pair information about the device the current clone entry request packet is referring is stored

Pan Id Field

The Pan Id field is 16 bits in length and contains the Pan Id of the controlled device the current clone entry request packet is referring.

Controller Short Address Field

The Controller Short Address field is 16 bits in length and contains the short address the remote device initiating the clone request has received during the pairing with the controlled device the current clone entry request packet is referring.

Controlled Short Address Field

The Controlled Short Address field is 16 bits in length and contains the short address of the controlled device the current clone entry request packet is referring.

Node Descriptor Field

The node descriptor field is 12 bytes in length and contains information that describes the device the current clone entry request packet is referring. The structure of the node descriptor field is detailed in Figure 4-11.

NOTE

The Clone Entry Request command does not support any payload.

4.2.1.8 Clone Entry Response Command Frame

The SynkroRF Clone Entry Response command is composed only by a CHR. The CHR field overlays the payload of the general SynkroRF frame command, as shown in Figure 4-11.

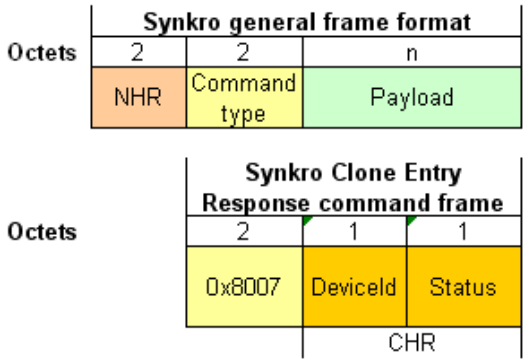

Figure 4-11. SynkroRF Clone Entry Response command frame format

Device Id Field

The Device Id field is 8 bits in length and contains the same device Id the remote device receiving the clone request has received in the Clone Entry Request command.

Status Field

The Status field is 8 bits in length and specifies if the device that has send the clone entry response is accepting the clone entry or is refusing it. A value of 0x01 in this field means the clone entry request was accepted.

NOTE

The Clone Entry Response command does not support any payload.

4.2.1.9 Search Request Command Frame

The SynkroRF Search Request command is composed of a CHR and a search request command payload. Both these fields overlay the payload of the general SynkroRF frame command, as shown in Figure 4-12.

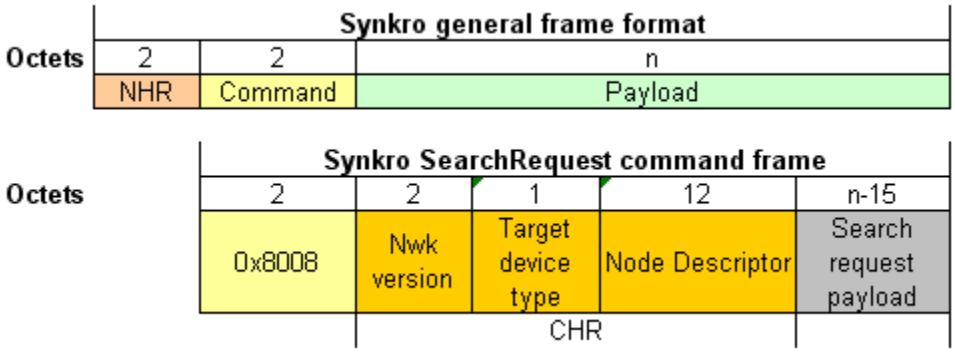

Figure 4-12. SynkroRF Search Response Command Frame Format

SynkroRF Version Field

The SynkroRF version field is 16 bits in length and contains information defining version of SynkroRF running on the device that has transmitted the search request. The SynkroRF version is defined as a combination of a major and a minor version. The major version is placed in the left most byte of the field, while the minor version is placed in the right most byte.

Target Device Type Field

The target device type field is 1 byte in length and contains information defining the type of device that is searched during the search process. The SynkroRF provides a number of public device types,that are listed in Table 4-4. The application can define its own proprietary deviceTypes, using values in the range 20 – 254. The 255 value is used as wildcard, specifying that the device sending the search request is trying to find controlled devices of any device type.

Node Descriptor Field

The node descriptor field is 12 bytes in length and contains information that describes the device that is sending the search request. The structure of the node descriptor field is detailed in Figure 4-4.

Payload Field

The payload field has a variable length. The Search Request payload is limited to 32 bytes.

4.2.1.10 Search Response Command Frame

The SynkroRF Search Response command is composed of a CHR and a search response command payload. Bothh these fields overlay the payload of the general SynkroRF frame command, as shown in Figure 4-13.

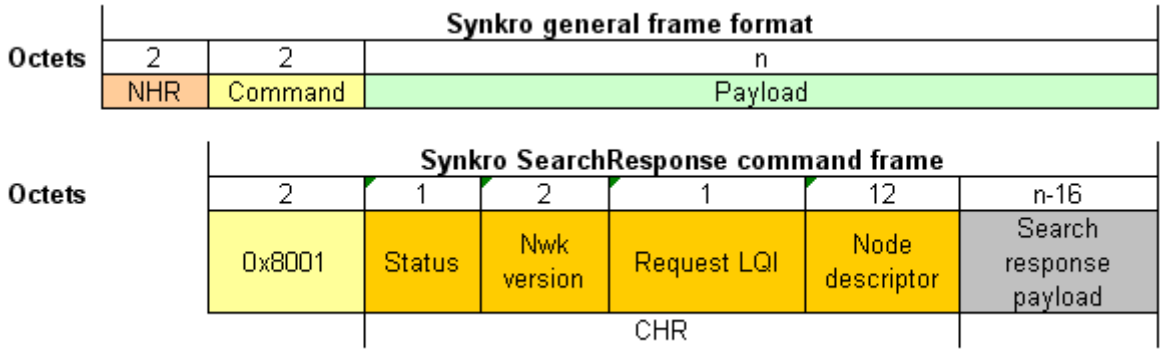

Figure 4-13. SynkroRF Search Response Command Frame Format

Status Field

The status field is 1 byte in length is always set to *gNWSuccess_c*.

Request LQI Field

The request field is 1 byte in length and contains the LQI of the search request received by the node issuing the search response.

SynkroRF Version Field

The SynkroRF version field is 16 bits in length and contains information defining version of SynkroRF running on the device that has transmitted the search response. The SynkroRF version is defined as a combination of a major and a minor version. The major version is placed in the left most byte of the field, while the minor version is placed in the right most byte.

Node Descriptor Field

The node descriptor field is 12 bytes in length and contains information that describes the device that is sending the search response. The structure of the node descriptor field is detailed in Figure 4-4.

Payload Field

The payload field has a variable length. The Search Response payload is limited to 32 bytes.

4.2.2 SynkroRF Application Command Frame Format

The application command frame defined by the SynkroRF layer in both public or proprietary cases is represented in Figure 4-14.

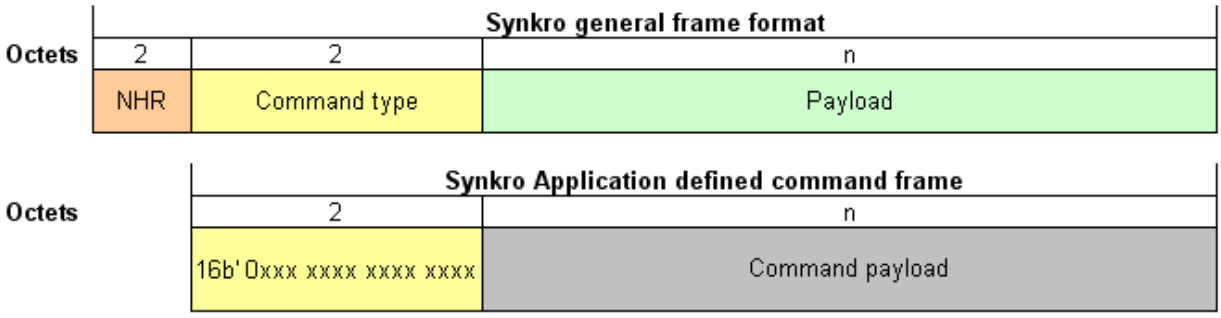

Figure 4-14. SynkroRF Application Command Frame Format

The payload of the application defined command frame overlays the general SynkroRF command frame payload.