# Kinetis Thread Stack Over-the-Air (OTA) Firmware Update User's Guide



## Contents

Chapter 1 About This Document	3
Chapter 2 OTA Firmware Updates for Kinetis Thread Stack	
2.1 OTA file format 2.2 OTA commands	
2.2 OTA commands	
2.2.2 gOtaCmd_QueryImageReq_c (/otaclient) OTA Query Image Request:	
2.2.3 gOtaCmd_QueryImageRsp_c (/otaserver) OTA Query Image Response	
2.2.4 gOtaCmd_BlockReq_c (/otaclient) OTA Block Request	
2.2.5 gOtaCmd_BlockRsp_c (/otaclient) OTA Block Response	
2.2.6 gOtaCmd_UpgradeEndReq_c (/otaclient) OTA Upgrade End Request 2.2.7 gOtaCmd_UpgradeEndRsp_c (/otaserver) OTA Upgrade End Response	
2.2.7 gOtaCind_OpgradeEndrsp_c (/otaserver) OTA Opgrade End Response	
Chapter 3 Software implementation	
Chapter 4 Test Tool OTA View	13
Chapter 5 Running a unicast OTA scenario	14
Chapter 6 Running a multicast OTA scenario	21
Chapter 7 Revision History	22
Chapter 8 Copyright	23

## Chapter 1 About This Document

This document provides an overview of the Over the Air Update (OTA) module, interface, and usage for the Kinetis Thread Stack. The transfer of an OTA image from the server to the client is done by unicast or multicast messages through the Thread network.

## Chapter 2 OTA Firmware Updates for Kinetis Thread Stack

The system setup consists of one OTA server and one or more clients, with potential intermediary multihop routers, as shown in Figure 1. The serial connections with PCs are for displaying status messages and for sending messages to the devices in the network. The OTA client serial connections are optional as the network nodes can also function in autonomous mode.

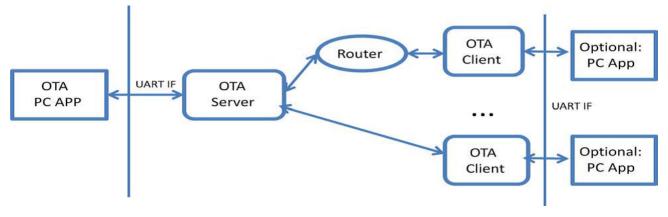


Figure 1. OTA Block Diagram

The download of an OTA image from server to client requires an application bootloader installed and sufficient memory (external or internal) to store the recently loaded image. The Thread framework includes an application interface in the OtaSupport header and module files for using the external storage modules while receiving the firmware image blocks over the air. The application can start to write a new image to the external storage, push one or more blocks to a storage location, or commit an image to indicate that the bootloader should use it for updating the internal flash at the next reboot.

## 2.1 OTA file format

The OTA file format is composed of a header followed by a number of sub-elements. The header describes general information about the file such as version, the manufacturer that created it, and the device it is intended for. Sub-elements in the file may contain upgrade data for the embedded device, certificates, configuration data, log messages, or other manufacturer-specific pieces.

Name	Header	ImageTag	Image	Bitmap Tag	Bitmap	CRC tag	CRC
Size	60 bytes	6 bytes	Variable	6 bytes	32 bytes	6 bytes	4 bytes

Table 1.	OTA file	format	using	CRC
----------	----------	--------	-------	-----

The bitmap sub-element of an OTA cluster image file indicates the sectors that should or should not be erased and reprogrammed in the internal flash. Setting a bit to 1 allows the erasure of the corresponding flash sector. If a bit is set to 0, the corresponding flash sector is protected

## 2.2 OTA commands

### 2.2.1 gOtaCmd\_ImageNotify\_c (/otaserver) OTA Image Notify:

- Sent unicast or multicast the Image Notify command represents the way by which the server alerts clients that a new image is available.
- URI-Path : NON POST coap://<dest>/otaserver
- · CoAP payload:

Table 2. OTA Image Notify CoAP payload	Table 2.	<b>OTA Image</b>	Notify CoAP	payload
--	----------	------------------	-------------	---------

Parameter	Size	Comments
CommandId	1	Command Identifier:
		gOtaCmd_ImageNotify_c = 0x00
TransferType	1	Transfer Type: Default Value = 0x00 (OtaUnicast)
ManufacturerCode	2	Manufacturer Code: Default Value = 0x1004
ImageType	2	Image Type: Default Value = 0x0000
ImageSize	4	ImageSize
FileSize	4	FileSize
FileVersion	4	New File version: Default Value = 0x40034005
ServerDownloadPort	2	ServerDownloadPort
FragmentSize	2	FragmentSize

### 2.2.2 gOtaCmd\_QueryImageReq\_c (/otaclient) OTA Query Image Request:

- Automatically sent when the client receives an Image Notify Command.
- URI-Path: NON GET coap://<dest>/otaclient
- CoAP payload:

Table 3.	Query	Image	Request	CoAP	payload
----------	-------	-------	---------	------	---------

Parameter	Size	Comments
CommandId	1	Command Identifier:
		gOtaCmd_ QueryImageReq _c = 0x01
ManufacturerCode	2	Manufacturer Code: Default Value = 0x1004
ImageType	2	Image Type: Default Value = 0x0000
FileVersion	4	Current File version
Hardware Version	2	Hardware version

### 2.2.3 gOtaCmd\_QueryImageRsp\_c (/otaserver) OTA Query Image Response

- The server response for Query Image Req command. Based on the status parameter, the OTA client decides whether or not to start downloading the new image.
- URI-Path : NON POST coap://<dest>/otaserver
- · CoAP payload:

Parameter	Size	Comments	
CommandId	1	Command Identifier:	
		gOtaCmd_ Queryli	mageRsp_c = 0x02
Status	1	Possible	e values:
		gOtaFileStatus_S	Success_c = 0x00
		gOtaFileStatus_Wa	aitForData_c = 0x97
		gOtaFileStatus_NoImageAvailable_c = 0x98	
Data	Variable	If status =	ManufacturerCode [2 bytes]
		gOtaFileStatus_Success_c	ImageType (2 bytes)
			FileVersion [4 bytes]
			FileSize [4 bytes]
			Server Download Port [2 bytes]
		If status !=	CurrentTime [4 bytes]
		gOtaFileStatus_Success_c	RequestTime [ 4 bytes]

#### Table 4. OTA Query Image Response CoAP payload

### 2.2.4 gOtaCmd\_BlockReq\_c (/otaclient) OTA Block Request

- This command is used to download the new image, by requesting a specific block. This command uses a socket open on a port that is announced by the Server on a ML\_EID address.
- Default port: 0xF0BE
- · Socket payload:

#### Table 5. OTA Block Request payload

Parameter	Size	Comments	
CommandId	1	Command Identifier: gOtaCmd_ BlockReq _c = 0x03	
ManufacturerCode	2	Manufacturer Code: Default Value = 0x1004	
Table continues on the next page			

ImageType	2	Image Type: Default Value = 0x0000
FileVersion	4	Downloaded File version
FileOffset	4	Image Offset
MaxDataSize	1	Maximum block length

#### Table 5. OTA Block Request payload (continued)

### 2.2.5 gOtaCmd\_BlockRsp\_c (/otaclient) OTA Block Response

- This is the server response for Block Req command. Based on the status parameter, the OTA client decides whether or not to continue the download procedure. This command uses a socket open on a port that is announced by the Server on a ML\_EID address. The response is sent on the port on which the command is received.
- Default port: 0xF0BE
- · Socket payload:

Parameter	Size	Com	nents			
CommandId	1	Command Identifier:				
		gOtaCmd_ Bloc	ckRsp_c = 0x04			
Status	1	Possible	e values:			
		gOtaFileStatus_S	Success_c = 0x00			
		gOtaFileStatus_	_Abort_c = 0x95			
		gOtaFileStatus_NotAuthorized_c = 0x7E				
		gOtaFileStatus_InvalidImage_c = 0x96				
		gOtaFileStatus_ServerBusy_c = 0x97				
		gOtaFileStatus_NoImageAvailable_c = 0x98				
Data	Variable	If status =	FileVersion [4 bytes]			
		gOtaFileStatus_Success_c	FileOffset [4 bytes]			
						DataSize [1 byte]
			Data [DataSize param bytes]			
		If status !=	CurrentTime [4 bytes]			
		gOtaFileStatus_Success_c	RequestTime [ 4 bytes]			

#### Table 6. OTA Block Response payload

### 2.2.6 gOtaCmd\_UpgradeEndReq\_c (/otaclient) OTA Upgrade End Request

- · This command is used to complete the transfer
- URI-Path: NON GET coap://<dest>/otaclient
- CoAP payload:

Parameter	Size	Comments
CommandId	1	Command Identifier:
		gOtaCmd_UpgradeEndReq_c = 0x05
Status	1	Possible values:
		gOtaFileStatus_Success_c = 0x00
		gOtaFileStatus_Abort_c = 0x95
		gOtaFileStatus_InvalidImage_c = 0x96
ManufacturerCode	2	Manufacturer Code: Default Value = 0x1004
ImageType	2	Image Type: Default Value = 0x0000
FileVersion	4	Downloaded File version

#### Table 7. OTA Upgrade End Request CoAP payload

### 2.2.7 gOtaCmd\_UpgradeEndRsp\_c (/otaserver) OTA Upgrade End Response

- This is the server response for Upgrade End Req command and contains the delay used by the client before a re-flash/ reboot to the new image (only if it detects the transfer procedure has been successful).
- URI-Path : NON POST coap://<dest>/otaserver
- CoAP payload:

Parameter	Size	Comments		
CommandId	1	Command	Identifier:	
		gOtaCmd_Upgrade	eEndRsp_c = 0x06	
Status	1	Possible	values:	
		gOtaFileStatus_S	Success_c = 0x00	
		gOtaFileStatus_	_Abort_c = 0x95	
		gOtaFileStatus_NotAuthorized_c = 0x7E		
		gOtaFileStatus_Inva	alidImage_c = 0x96	
		gOtaFileStatus_Se	rverBusy_c = 0x97	
		gOtaFileStatus_Nolma	ageAvailable_c = 0x98	
Data	12	If status =	Current Time [4 bytes]	
		gOtaFileStatus_Success_c	Upgrade Time [4 bytes]	
		File Version [4 by		
		If status != Current Time [4		
		gOtaFileStatus_Success_c	Request Time [4 bytes]	

#### Table 8. OTA Upgrade End Response CoAP payload

### 2.2.8 gOtaCmd\_ServerDiscovery\_c (/otaclient) OTA\_Server\_discovery

- This command is sent multicast and represents the way to discover a server by requesting a specific manufacturer and image type. A server will respond with an Image Notify unicast if success. The OTA Server discovery feature is disabled if the data regarding the OTA server (like the server's short address) is contained in a Service TLV in Network Data packets.
- URI-Path: NON POST coap://<dest>/otaclient
- CoAP payload:

Table 9.	ΟΤΑ	Server	discovery	CoAP	payload
----------	-----	--------	-----------	------	---------

Parameter	Size	Comments
CommandId	1	Command Identifier:
		gOtaCmd_ServerDiscovery_c = 0x07
ManufacturerCode	2	Manufacturer Code: Default Value = 0x1004
ImageType	2	Image Type: Default Value = 0x0000

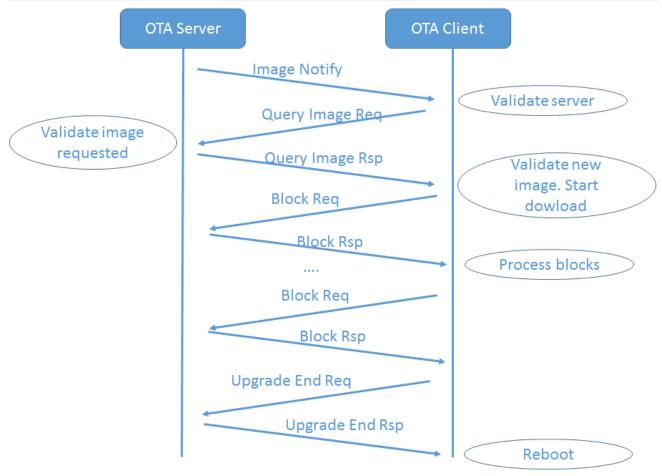


Figure 2. OTA Upgrade Diagram

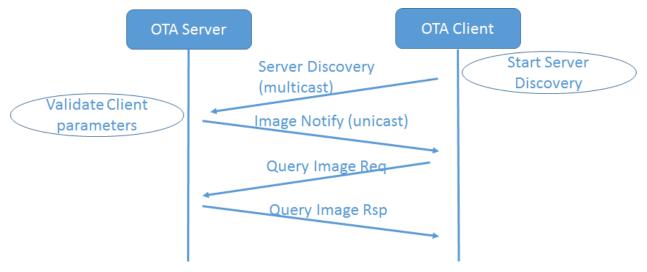


Figure 3. OTA Server Discovery Diagram

## Chapter 3 Software implementation

The files app\_ota.h, app\_ota\_client.c, app\_ota\_server.c, OtaSupport.c, OtaSupport.h comprise the OTA Upgrade functionality. All configurations are done at compile time. The configuration settings needed to enable the OTAServer and OTAClient functionalities are available in the config.h file of the corresponding example project and have already been set to the appropriate values.

- gEnableOTAServer\_d = TRUE enables the OTA Server functionality;
- gEnableOTAClient\_d = TRUE enables the OTA Client Functionality;
- gEepromType\_d = gEepromDevice\_ AT45DB041E\_c use this particular EEPROM device type to store the image in the external flash of the client for FRDM-KW41 boards. For other boards, check Eeprom.h;
- OTA\_USE\_NWK\_DATA = TRUE includes the OTA server data in a Service TLV in Network Data packets, so each board will always be aware of the server's short address.

OTA client/server initialization is performed by the following API:

- OtaClientInit(mpAppThreadMsgQueue);
- OtaServerInit(mpAppThreadMsgQueue);

OTA client project options;

- gUseBootloaderLink\_d=1 sets the linker configuration file to reserve the first flash sector for bootloader use with the
  application firmware following in the subsequent sectors
- gUseInternalStorageLink\_d=0 use external flash for temporary storing the image sent over the air; otherwise uses the internal flash.
- gEraseNVMLink\_d=1 includes NVM firmware sectors in OTA file, and will overwrite those sectors when performing OTA update. When using .bin format, gEraseNVMLink\_d must be set to 0 to preserve NVM data
- gNVMSectorCountLink\_d=32 allocates 32 sectors of NVM, each having 2 kb of memory
- gUseNVMLink\_d=1 sets the linker configuration to access the NVM
- \_\_ram\_vector\_table\_\_=1 places the vector table in RAM and allows dynamic insertion of an ISR handler

The following figures describe how to update the symbols definition of the Linker configuration files for IAR IDE.

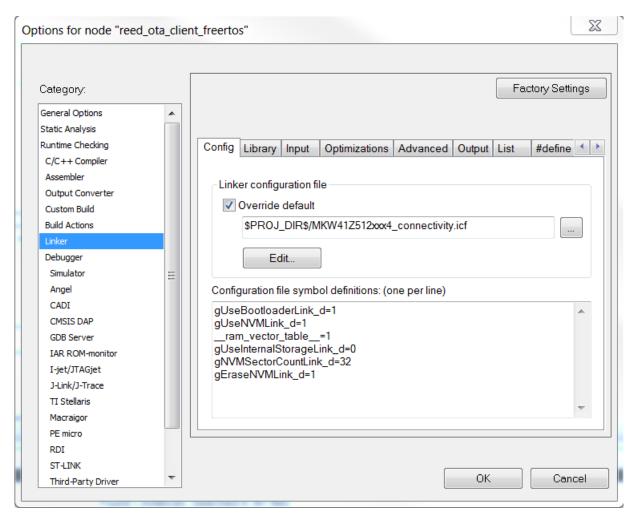


Figure 4. IAR Linker Configuration Options

In the case of MCU Xpresso IDE, the linker file for the ota client project is preconfigured with the appropriate settings.

## Chapter 4 Test Tool OTA View

The OTAP Thread view used for uploading binary firmware images (\*.srec and \*.bin) to the OTA server or directly to the OTA client is integrated into the Test Tool for Connectivity Products. Ensure that you use the latest Test Tool release available because it is updated for this Thread release.

Users can configure the Image Type, File Version, Min and Max HW Version, Sector Bitmap and Signature type (CRC) as shown below:

	😽 NXP Test Tool 12					
Image File Information   Image File:   OTA Header:   Upgrade File Identifier:   0x000   Header File (Control:   0x000   Header File (Control:   0x000   Header Length:   0x000   Header Length:   0x000   Header Kile (Control:   0x000   Header Kile (Control:   0x000   Header Length:   0x000   Header Kile (Control:   0x000   Manufacturer code:   0x000   Manufacturer   Manufacturer   Manufacturer   Manufacturer   Manufacturer   Manufacturer   Manufacturer <t< td=""><td>Command Console</td><td>🎩 Script Server 🚆 Protocol Analyzer 觽 Coexistence Tool 🛭 🏶 Firmware Loaders 🔹 🐺 Radi</td><td>io Test 😵 OTA Updates 🔹</td><td>🛿 Settings 🧇 Help 🔇</td><td></td><td></td></t<>	Command Console	🎩 Script Server 🚆 Protocol Analyzer 觽 Coexistence Tool 🛭 🏶 Firmware Loaders 🔹 🐺 Radi	io Test 😵 OTA Updates 🔹	🛿 Settings 🧇 Help 🔇		
Image File Information     Image Streat     Image Streat<	🍙 Start Page 🔗	OTAP Thread 🗱				
Network Save     Upgrade file identifie: 0.000     Header Version: 0.000     Header Length: 0.000     Header John 0.000     Header String: 0.000     Strice: 0.0000     Header String: 0.0000     Strice: 0.0000     Header String: 0.0000     Total Irage Strice: 0.0000     Note: 0.0000              Note: 0.0000 <th></th> <th></th> <th>Browse</th> <th>Server Device Serial Port:</th> <th>Productor</th> <th>Clent OTA Programing No OTA Transfers</th>			Browse	Server Device Serial Port:	Productor	Clent OTA Programing No OTA Transfers
Header Length: 0x000   Header Feld Control: 0x000   Mannicturer code: 0x1004   Mannicturer code: 0x1000   Mannicturer code: 0x1000   Mannicturer code: 0x1000   Rie version: 0x0002   Kask version: 0x0002   Header string: Tread Image Ffe   Total Image Serse: 0x00000   Houren HW version: 0x2035   NXXP Specific Sector Copy Subelement: 0x104de Image Signature in OTA file   Curnert selekt al syndur: (KC signature Change	Upgrade File Identifier:			Connect to OTAP Server Device		
Munde turer code: 0x1004   Smape type: 0x0000   File vention: 0x0000   Stack version: 0x0002   Heade string: 0reed findope File   Total large Signature in OTA file   Marrum MW version: 0x2004	Header Length:	0x0100		Start OTA Image Load To Server	Cancel	
Stack version: 0x0002   Header string: Treved Imoge File   Total Image Stack 0x00000   HW Version: 0x204   Marrum HW Version: 0x204   Marrum HW version: 0x204   Marrum HW version: 0x204   Image Stack befung: Treved Image Stack befung:   Image Stack befung: Treved Image Stack in OTA file   Current seleka di spatur (KC signature Change	Image type:	0x0000	Alleur come version unaradas			
HW Verson Merrum HW verson: 0x204 Marrum HW verson: 0x205 NXP Specific Sector Copy Subdement: Oncode subelement to TA fle Oncode subelement to TA fle Current sector behavio: "Terref" Current sector behavio: "Terref" Current sector behavio: "Terref" Current sector barbanic and the Company Current sector barbanic and the Current sector barbanic and the Cur	Stack version:	0x0002	<ul> <li>Allow same version upgrades</li> </ul>			
NXP Specific Sector Copy Subdement: V Include subdement in 0TA fie Overnde sector bitrage: TFFFFFF Image Signature V Include Image Signature in 0TA fie Current selected signatur CRC signature Change	HW Version Minimum HW version:	0x204				
Image Signature Signature in OTA file Current set-for disputur CC signature Change	NXP Specific Sector C	Copy Subelement: in OTA file				
Table OTA Muticast	Image Signature Include Image Signa Current selected signatu Multicast Option	ture in OTA file UT CRC sprature Change			E	

Figure 5. Test Tool overview

To start downloading the new image to the server, select the COM port on which the server board is connected, enter the baud rate (usually 115200 bps), and then press the *Connect to OTA Server Device*.

The transfer can be started by pressing Start OTA Image Load To Server button.

## Chapter 5 Running a unicast OTA scenario

- 1. Select the demo applications from <install folder >\boards\<board\_type >\wireless\_examples\thread :
  - a. OTA Server: hcd\_ota\_server
  - b. OTA Client:
    - i. reed\_ota\_client (frdmkw41z boards only)
    - ii. end\_device\_ota\_client
- 2. Validate configurations:
  - a. On the server side. Please make sure the following defines are set accordingly:
    - i. In config.h : #define gEnableOTAServer\_d 1
    - ii. In app\_thread\_config.h: #define THR\_SERIAL\_TUN\_ROUTER 0
  - b. On the client side.
    - i. Please open config.h make sure the following defines are set accordingly: #define gEnableOTAClient\_d 1
    - ii. For IAR IDE, right click project -> options -> linker -> config and add the appropriate settings as presented in Figure 4.
- 3. Load the *hcd\_ota\_server* and project onto one of the boards and make note of its COM port which will be used by the Test Tool in the steps below. Then, load the *reed\_ota\_client* project onto one or more of the other boards.
- 4. Load the OTAP Bootloader binary onto the client board. The linker files for the two project are configured so that the bootloader and wireless application will coexist on different areas of the flash. The process for writing the bootloader on the board is the following:
  - a. Go to <install\_folder >boards\frdmkw41z\wireless\_examples\framework\bootloader\_otap\bm\iar and open the project.
  - b. Compile it and download it on to the board.
- 5. Use the Test Tool application to communicate with the OTA server board via the THCI interface. Create a new Thread network, start the commissioner, and enable joiners (the client nodes) to start joining the network.
  - a. Connect the OTA server board to the Test Tool application. First, click on the Command Console button to open the Command Console tab. Then, select the COM port that the board enumerated as and open up the Settings dialog box. Set them to the values shown in the Figure below. Once set, click "Open", and for the "Loaded Command Set" drop down, select ThreadIP.xml to get access to the available Thread THCI commands.

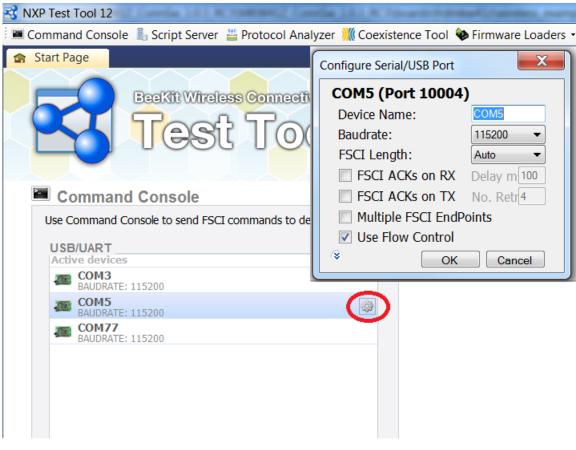


Figure 6.

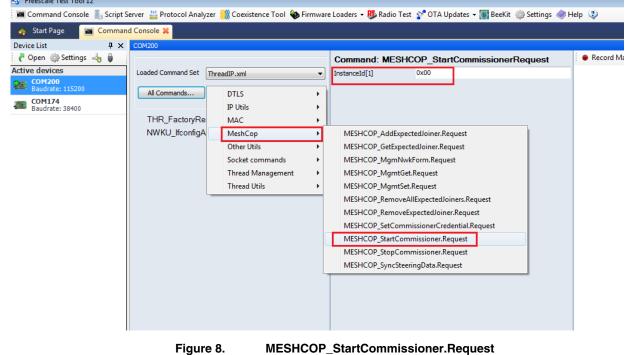
Test Tool Command Console

b. Use THR\_CreateNwk.Request command to create a new network (Instance ID parameter set to 0x00).

式 🛛 Freescale Test Tool 12				
🔳 Command Console 📳 Script S	erver 📲 Protocol Analyz	zer ) Coexistence Tool 😻 Fi	rmwar	e Loaders 👻 🐺 Radio Test 💸 OTA Updates 👻 🌆 BeeKit 🌼
👩 Start Page 🔄 Command	d Console 💢			
Device List 4 ×	COM200			
🦰 Open 🌼 Settings 👆 🏺				Command: THR_FactoryResetRequest
Active devices	Loaded Command Set	ThreadIP.xml	•	
COM200 Baudrate: 115200			_	
COM174	All Commands	DTLS	- 1	
Baudrate: 38400	l	IP Utils	-	
	THR_FactoryRe	MAC	-	
	NWKU_lfconfigA	MeshCop	- +	
		Other Utils	- +	
		Socket commands	→」	
		Thread Management	•	THR_CpuReset.Request
		Thread Utils	•	THR_CreateNwk.Request
			_	THR_Disconnect.Request
				THR_FactoryReset.Request
				THR_Join.Request
				THR_MgmtDiagnosticGet.Request
				THR_MgmtDiagnosticReset.Request
				THR_NwkScan.Request
				THR_Reattach.Reguest
				THR_SetDeviceConfig.Request
			L.,	



c. Use MESHCOP\_StartCommissioner.Requestcommandto start the commissioner:



d. Add expected joiner and synchronize steering data. In this scenario we will accept any EUI64 address.

```
TX: MESHCOP_AddExpectedJoiner.Request 02 CE 42 12 00 00
01 01 FF FF FF FF FF FF FF FF 06 54 48 52 45 41 44 96
                           = 02
               [1 byte]
    Sync
    OpGroup [1 byte]
                           = CE
    OpCode
              [1 byte]
                           = 42
    Length
               [2 bytes] = 00 12
    InstanceId [1 byte]
                           = 00
    Selected [1 byte]
                           = 01 (true)
    EuiType [1 byte]
                           = 01 (LongEUI)
    LongEUI [8 bytes] = FF FF FF FF FF FF FF FF
    PSKdSize [1 byte]
                           = 06
    PSKd
              [6 bytes] = 54 48 52 45 41 44 (THREAD)
    CRC
                           = 96
               [1 byte]
RX: MESHCOP_AddExpectedJoiner.Confirm 02 CF 42 01 00 00
 8C
    Sync [1 byte]
                        = 02
    OpGroup [1 byte]
                      = CF
    OpCode [1 byte]
                        = 42
    Length [2 bytes] = 00 01
    Status [1 byte]
                        = 00 (Success)
                       = 8C
    CRC
            [1 byte]
```

#### Figure 9. Add Expected Joiner Log

Use the following commands with the associated parameters.

MESHCOP\_AddExpectedJoiner.Request:

- InstanceId = 0x00
- Selected = TRUE
- EuiType = 0x01 (LongEUI)
- PSKdSize = 0x6
- PSKd = THREAD (the same used by OTA Client default configuration)

MESHCOP\_SyncSteeringDataRequest:

- InstanceId = 0x00
- EuiMask = 0x01 (AIIFFs)
- 6. After the server node has started a Thread network and a commissioner, the OTA client nodes can join the network by pressing any switch on the board. OTA client has the Shell enabled by default: #define THREAD\_USE\_SHELL
- 7. Start the OTA procedure:
  - a. Close the Command Console tab in Test Tool.
  - b. From TestTool choose OTA Updates -> OTAP Thread to launch the OTA Update View, as shown in the Figure below.

Command Console 📗 Se	rript Server 🚆 Protocol Analyzer ∭ Coexistence Tool 💊 Firmware Loaders 🔹 🐉 Radio Test 💸 OTA Updates 📼	🛞 Settings 🧼 Help 🔇 🕸			
🍖 Start Page 🛛 💅 OT.	AP Thread 🗱				
Image File Information			OTA Server Image Loading		Client OTA Programing No OTA Transfers
Image File:		Browse	Server Device Serial Port: Port:	Baudrate:	
			Porc	115200 ·	
OTA Header:		Save	Connect to OTAP Server Device		
Upgrade File Identifier:	0x08EEF11E			J	
Header Version:	0x0100		OTA Actions:		
Header Length:	0x0100		Start OTA Image Load To Server	Cancel	
Header Field Control:	0x0000		Server internal flash download progress 0%		
Manufacturer code:	0x1004				
Image type:	0x0000		Message Log		
File version:		Allow same version upgrades		Â	
Stack version:	0x0002				
Header string:	Thread Image File				
Total Image Size:	0x0000000				
HW Version					
Minimum HW version:	0x2204				
Maximum HW version:	0x2205				
NXP Specific Sector Copy S					
Include subelement in C					
Override sector bitmap:	OXFFFFFFF				
mage Signature	e in OTA file				
Current selected signature				E	
fulticast Option					
Enable OTA Multicast					
			<	)   +	
			Save Session Log	Copy Clear	

Figure 10. Thread OTAP view

c. In the OTA process the image file can be either i) stored on the OTA server in the external flash memory, called stand-alone mode, or ii) held by Test Tool with the OTA server polling for each chunk when it is required by the client, called dongle mode. By default, the Test Tool inquires the OTA Server about the extended memory support and adjusts the image storage location in accordance with it. The user may choose to keep the image in the Test Tool, regardless of the OTA server configuration, by selecting the "Thread OTA Server polls Test Tool for firmware file fragments" checkbox as displayed in the Figure below. It should be noted that USBKW41Z board does not have external flash and thus can work only in dongle mode.

Image File Information			OTA Server Image Loading			Client OTA Programing
						No OTA Transfers
(mage File:			Browse Port:		Baudrate:	
			Save		115200 -	
TA Header:			Save			
	0x08EEF11E	Test Tool 12 Settings		PTAP Server Device		
Header Version:	0x0100	General	OTA Update Settings			
Header Length:	0x0100	OTA Update	-			
Header Field Control:	0x0000		Enable OTA compatibility mode with BeeStack Codebases: 3.0.10, 3.0.11, HCS08 3.0.12	) Server	Cancel	
Manufacturer code:	0x1004		BeeStack Pro OTA Server polls Test Tool for			
Image type:	0x0000					
File version:	0x30103010		802.15.4 MAC OTA Server polls Test Tool for		*	
Stack version:	0x0002		Thread OTA Server polls Test Tool for firmware			
Header string:	Thread Image File					
Total Image Size:	0x0000000					
W Version Minimum HW version:	0x2204					
Maximum HW version:	0x2205					
IXP Specific Sector C	opy Subelement:					
Include subelement i						
Override sector bitm	ap: FFFFFFF					
mage Signature Include Image Signat	ture in OTA file					
Current selected signatu	r CRC signature Change				E	
ulticast Option						
Enable OTA Multicast						

Figure 11. OTA Update settings

- d. In the OTA Update View in the Image File Information area, click Browse... and navigate to the binary folder.
- e. Select Kinetis Image Files (\*.srec, \*.bin) in the *Files of type* drop down in the Open Window as shown in the Figure below.

3 NXP Test Tool 12													) d
🔳 Command Console 🚪	Script Server 🚆 Prot	ocol Analyzer 💥 Coexistence Ti	ool 🐞 Firmware Loaders 🔸 🚯 Radio Test	💕 OTA Updates 🔹 🎲 Settings 🦪	Help 🥨								
👩 Start Page 🔗	OTAP Thread 🙀					ويتحصر والمحمد والعرب والمرا						والمتحدين والمتحدين والم	
Image File Information					1	OTA Server Image Loa	ding					Client OTA Programing	_
Image File:						Server Device Seria	Port:					No OTA Transfers	
					Browse	Port:				Baudrate:			
					Save	COM3	-			115200	•		
OTA Header:								Connect	to OTAP Server Device				
Upgrade File Identifier:	0x0BEEF11E												
Header Version:	0x0100					OTA Actions:							
Header Length:	0x0100	🕄 Open						-	- Court	Cancel			
Header Field Control:	0x0000							-					
Manufacturer code:	0×1004	COO K examples	thread_router_eligible_device_ota_clien	t → kw4x → frdmkw41z → freerto	s iar i debug	a ►	Search debug	,	2				
Image type:	0x0000	Organize 🔻 New folde	87				811 <b>•</b>						
File version:	0x30103010	🔶 Favorites	Name		Date modifie	ed Type	Size				<b>^</b>		
Stack version:	0x0002	E Desktop	🎉 list		20.07.2016 1	0:52 File folder							
Header string:	Thread Image File	🔒 Downloads 😑	🁪 obj		22.07.2016 1								
Total Image Size:	0x0000000	Secent Places	thread_router_eligible_device_ota_cl	ient_frdmkw41z_freertos.srec	20.07.2016 1	0:52 SREC File	1.421 KB						
HW Version		libraries											
Minimum HW version:	0x2204	Documents											
Maximum HW version:	0x2205	J Music											
		Pictures											
NXP Specific Sector Copy		Videos											
Include subelement in t													
Cverride sector bitmap	C OXFFFFFFFF	Computer											
Image Signature		FRDM-KW41ZJ (E											
Current selected signature		😴 space (\\Zro04fil -											
Multicast Option		Filen	ame: thread_router_eligible_device_ota_cliv	nt frdmkw41z freertos.srec			Kinetis Image Files (*.a	srec:".bin) -			E		
Enable OTA Multicast								Cancel					
C CHAPTER CONTRACTOR							<u>O</u> pen	Cancel					

Figure 12. Selecting S-Record file type

NOTE -

The new OTA client image should have the same format as the original image, including the bootloader configuration presented in Figure 5.

- f. Select the reed\_ota\_client.srec file that was compiled previously for the client nodes. You may want to modify it slightly and recompile it to verify that the client nodes got updated using the OTA firmware update. The OTA headers in the OTA Update View for the file are filled in automatically. Configure the *Image Type, File Version, Sector Bitmap* and *Signature* (only CRC signature supported at this moment).
- g. Select the processor type of the OTA client. By default, the NVM check box is selected, which means that the NVM will be preserved.
- h. When deselecting the "Do not include NVM firmware sectors in OTA file" checkbox, the image used has to be compiled with gEraseNVMLink\_d=0 in the linker tab of the project's option for any build (bin, srec, etc.). Otherwise, it will cause an invalid length of the image and the OTAP process will fail.



By checking the "Image contains bootloader" checkbox, the user informs the test tool that the selected image contains a bootloader. For .bin files, this information is passed to the Test Tool by the user. For .srec files, the Test Tool is able to obtain this information automatically. Starting with Test Tool 12.5.4, the bootloader is no longer sent over the air during the OTA process, thus the OTA process duration has been decreased.

i. In a second level of configuration, the user can explicitly define which internal memory sectors to erase by selecting the "Override sector bitmap" checkbox, as shown in the Figure below. For more information, see the Connectivity Framework Reference Manual chapter 3.20.2. OTAP Bootloader.

NXP Specific Sector Copy Subelement:

Include subelement ir	1 OT	A file
-----------------------	------	--------

#### Figure 13. Override sector bitmap

j. Click Start Over the Air Programming. This uses the hcd\_ota\_server to initiate the OTA process by informing the clients that a new image is available, using a multicast ImageNotify command and in the same time to start pushing the reed\_ota\_client.srec to the OTA Client application and displaying the progress as shown in the Figure below.

Command Console 📗 Sc	ript Server 🚆 Protocol Analyzer 🍿 Coexistence Tool 💊 Firmware Loaders 🛛 🥦 Radio Test 😵 OTA Updates	s 🔹 🎲 Settings 🧼 Help 🔇 🥸			
🍖 Start Page 🛛 😵 OTA	IP Thread 🗱				-
Image File Information			OTA Server Image Loading		Client OTA Programing Abort 7.41%
Image File:			Server Device Serial Port:		PLOT 7.41.4
C: Users 815233 Documen	ts\Thread KW24BootboaderRemove\thread_router_eligible_device_ota_client_frdmkw24_freertos.srec	Browse	Port: COM67 v	Baudrate: 115200 v	Client Address
OTA Header:		Save			FD26:BECC:6EB1:0:BC89:4689:88A2:4300
Upgrade File Identifier:	0x08EEF11E		Disconnect		
Header Version:	0x0100		OTA Actions:		
Header Length:	0x003C		Start OTA Image Load To Server	Cancel	
Header Field Control:	0x0004		Server internal flash download progress 100.00%		
Manufacturer code:	0x1004				
Image type:	0x0000		Message Log 18:21:07:026:Send Frame 5372 offset: 0x4EAD4of 0x4F071		
File version:	0x40034003	Allow same version upgrades	18:21:07:026:Send Raw Data Chunk: 0x00 00 0A 00 00 00 0A 00 00 00 E8 03 00 00 E8 03 00 01 18:21:07:055:Send Frame 5373 offset: 0x4EB100f 0x4F071		
Stack version:	0x0002		18:21:07:055:Send Raw Data Chunk: 0x0C 00 4D 4D 0A 00 04 03 0C 00 4D 4D 0A 00 4D 4D 0A 18:21:07:064:Send Frame 5374 offset: 0x4EB4Cof 0x4F071		
Header string:	Thread Image File		18:21:07:05:5:5end Raw Data Chunk: 0x0A 00 4D 4D 0A 00 00 50 06 40 00 90 02 40 00 00 00 18:21:07:073:5end Frame 5375 offset: 0x4E8886 f 0x4071 18:21:07:073:5end Raw Data Chunk: 0x6E 3C 3A F5 4F A5 7F 52 0E 51 8C 68 05 9B AB D9 83 1		
Total Image Size:	0x0004F071		18:21107/07/354hd Raw Data Chunk: 0xbe 3C 3A F5 4F A5 7F 52 0E 51 8C 68 05 9E AB D9 85 1 18:21:07:081:Send Frame 5376 offset: 0x4EBC4of 0x4F071 18:21:07:081:Send Raw Data Chunk: 0x83 12 8E 85 31 24 C3 7D 0C 55 74 5D 8E 72 FE B1 DE 6		
HW Version			18:21:07:098:Send Frame 5377 offset: 0x4EC00of 0x4F071		
Minimum HW version:	0x2421		18:21:07:098:Send Raw Data Chunk: 0x51 3E 98 6D C6 31 A8 C8 27 03 B0 C7 7F 59 BF F3 0B E 18:21:07:110:Send Frame 5378 offset: 0x4EC3Cof 0x4F071		
Maximum HW version:	0x2421		18:21:07:110:Send Raw Data Chunk: 0x2C 72 92 A1 E8 BF A2 4B 66 1A A8 70 8B 4B C2 A3 51 1 18:21:07:114:Send Frame 5379 offset: 0x4EC78of 0x4F071		
NXP Specific Sector Copy S	ubelement:		18:21:07:114:Send Raw Data Chunk: 0xCA 9C 58 F3 6F 2D 2E 68 EE 82 8F 74 6F 63 A5 78 14 7 18:21:07:120:Send Raw Data Chunk: 0x00 57 A9 08 1B 08 E0 88 10 8E 0 48 18 08 E0 00 10 08 18:21:07:120:Send Frame 5380 offset: 0x4E0E4of 0x4F071	8 C8 84 08 02 C7 8C PA PP BI 8 E0 F8 00 17 B4 AA 15 BC 00	
Include subelement in O	TA file 0x00000000000000000000000000000000000		18:21:07:123:Send Frame 5381 offset: 0x4ECF0of 0x4071 18:21:07:123:Send Frame 5381 offset: 0x4ECF0of 0x4071 18:21:07:123:Send Raw Data Chunk: 0x00 08 FD 00 00 10 FD 00 00 20 FD 00 00 40 FD 00 00 8	0 ED 00 00 18 ED 00 00 26 E	
			18:21:07:165:Send Frame 5382 offset: 0x4ED2Cof 0x4F071		
Image Signature Include Image Signature	in OTA file		18:21:07:165:Send Raw Data Chunk: 0x04 44 00 10 00 01 FD 00 04 A0 00 10 00 01 FD 00 04 0 18:21:07:180:Send Frame 5383 offset: 0x4ED68of 0x4F071		
Current selected signature:			18:21:07:180:5end Raw Data Chunk: 0x00 0A FD 00 08 03 04 00 00 01 01 00 00 0A FD 00 08 0 18:21:07:185:5end Frame 5384 offset: 0x4EDA460 0x4F071 18:21:07:185:5end Raw Data Chunk: 0x03 00 BF 18 03 F7 00 00 FF D7 00 01 02 40 FA 00 00 05		
Multicast Option			18:21:07:195:56h0 Raw Data Chunk: 0x03 00 8F 16 03 F7 00 00 FF D7 00 01 02 40 FA 00 00 05 18:21:07:194:56h0 Frame 5385 offset: 0x4EDE0x6 0x4F071 18:21:07:194:56h0 Raw Data Chunk: 0x20 05 FB 00 16 70 1D 05 00 78 19 05 00 70 22 05 00 9-		
Enable OTA Multicast			18:21:07:201:Send Frame 5386 offset: 0x4EELCof 0x4F071 18:21:07:202:Send Raw Data Churk: 0x4F0 00 F0 1C 01 FE 80 E9 00 04 09 3D 00 00 80 FA 00 FI		
			18:21:07:208:Send Frame 5387 offset: 0x4EE58of 0x4F071 18:21:07:208:Send Raw Data Chunk: 0x00 08 5C 24 05 00 E0 0F 05 00 01 FD 00 03 58 EF 04 00		
			18:21:07:224:Sand Frame 5388 offset: 0x4EE94of 0x4E971 18:21:07:224:Sand Raw Data Chunk: 0xFC FF 08 D4 14 05 00 EF 8E AD DE 1F 21 23 00 F8 FF F		
			18:21:07:233:Send Frame 5389 offset: 0x4EED0of 0x4F071 18:21:07:233:Send Raw Data Chunk: 0xFF FF		
			18:21:07:244:Send Frame 5390 offset: 0x4EF0Cof 0x4F071 18:21:07:244:Send Raw Data Chunk: 0x9F FF F		
			18:21:07:251:Send Frame 5391 offset: 0xHEF48of 0xHF071 18:21:07:251:Send Raw Data Chunk: 0xFE FE F		
			18:21:07:261:Send Frame 5392 offset: 0x4EF84of 0x4F071 18:21:07:261:Send Raw Data Chunk: 0xFF FF	F FF FF FF FF FF FF FF FF FF	
			18:21:07:270:Send Frame 5393 offset: 0x4EFC0of 0x4F071 18:21:07:270:Send Raw Data Chunk: 0x4F FF F		
			18:21:07:274:Send Frame 5394 offset: 0x4EFFCof 0x4F071 18:21:07:274:Send Raw Data Chunk: 0xFF FF	F FF FF FF FF FF FF FF FF FF	
			18:21:07:310:Send Raw Data Chunk: 0xFF FF 00 F0 1F 00 00 00 FF FF		
			<	+	
			Save Session Log	Copy Clear	

Figure 14.Running OTA procedure

k. Wait until the process reaches 100%. The client board resets itself when it has received the full image. The bootloader on the client board then runs and reprograms the internal flash of the client node. The client node reboots when the programming is finished and starts running the new image.

## Chapter 6 Running a multicast OTA scenario

To run a multicast OTA scenario follow the same steps as for a unicast scenario. Select the "Enable OTA multicast" checkbox before starting the over the air programing, as shown in the figure below.

The multicast process begins and finishes when the main progress bar reaches 100%. After the multicast process finishes, each client board starts a unicast OTA process with the server, requesting the image chunks that failed to reach them during the multicast process. The status of the unicast transfers can be seen in the right side of the screen, as seen in the unicast OTA scenario. The client board resets itself when it has received the full image. The bootloader on the client board then runs and reprograms the internal flash of the client node. The client node reboots when the programming is finished and starts running the new image.

🔳 Command Console 👢 Sc	ript Server 🔚 Protocol Analyzer 🇰 Coexistence Tool 💊 Firmware Loaders 🛛 🚯 Radio Test 🔗 OTA Updates 🕫 🎲 Settings 🤞	🥏 Help 🔇 😲			
🍙 Start Page 🔗 🕅	AP Thread 🙀				•
Image File Information			OTA Server Image Loading		Client OTA Programing
Image File:			Server Device Serial Port:		No OTA Transfers
	ts\Thread KW24BootloaderRemove\thread router eligible device ota client frdmkv24 freertos.srec	Browse	Port:	Baudrate:	
			COM67 v	115200 -	
OTA Header:		Save			
Upgrade File Identifier:	0x08EEF11E		Disconnect		
Header Version:	0x0100		OTA Actions:		
Header Length:	0x003C		UTA ACIDIIS:		
Header Eield Control:	0x0004		Start OTA Image Load To Server	Cancel	
			Multicast OTA Update progress 11.12%		
Manufacturer code:	0x1004				
Image type:	0x0000		Message Log 18:37:40:155:Send Frame 5372 offset: 0x4EAD4of 0x4E071		
File version:	0x40034003	version upgrades	18:37:40:156:Send Raw Data Chunk: 0x00 00 0A 00 00 00 0A 00 00 00 E8 03 00 00 E8 03 00 00 80 18:37:40:161:Send Frame 5373 offset: 0x4E810of 0x4F071		
Stack version:	0x0002		18:37:40:162:Send Raw Data Chunk: 0x0C 00 4D 4D 0A 00 04 03 0C 00 4D 4D 0A 00 4D 4D 0A 00 0 18:37:40:167:Send Frame 5374 offset: 0x4E84Cof 0x4F071		
Header string:	Thread Image File		18:37:40:167:Send Raw Data Chunk: 0x0A 00 4D 4D 0A 00 00 50 06 40 00 90 02 40 00 00 00 0FF 18:37:40:173:Send Frame 5375 offset: 0x4EB886f 0x4F071		
Total Image Size:	0x0004F071		18:37:40:173:Send Raw Data Chunk: 0x6E 3C 3A F5 4F A5 7F 52 0E 51 8C 68 05 9B AB D9 83 1F 19 18:37:40:177:Send Frame 5376 offset: 0x4EBC4of 0x4F071		
HW Version			18:37:40:177:Send Raw Data Chunk: 0x83 12 BE 85 31 24 C3 7D 0C 55 74 5D BE 72 FE B1 DE 80 A3 18:37:40:186:Send Frame 5377 offset: 0x4EC00of 0x4F071	/ 06 DC 9B 74 F1 9B C1 C	
Minimum HW version:	0x2421		18:37:40:186:Send Raw Data Chunk: 0x51 3E 98 6D C6 31 A8 C8 27 03 B0 C7 7F 59 BF F3 0B E0 C6 18:37:40:200:Send Frame 5378 offset: 0x4EC3Cof 0x4F071	47 91 A7 D5 51 63 CA 0	
Maximum HW version:	0x2421		18:37:40:200:Send Raw Data Chunk: 0x2C 72 92 A1 E8 BF A2 4B 66 1A A8 70 8B 4B C2 A3 51 6C C 18:37:40:209:Send Frame 5379 offset: 0x4EC78of 0x4F071	7 19 E8 92 D1 24 06 99 D	
NXP Specific Sector Copy S	ubelement:		18:37:40:209:Send Raw Data Chunk: 0xCA 9C 5B F3 6F 2D 2E 68 EE 82 8F 74 6F 63 A5 78 14 78 C8 18:37:40:224:Send Frame 5380 offset: 0x4ECB4of 0x4F071	84 08 02 C7 8C FA FF BE	
Include subelement in O	TA fie		18:37:40:224:Send Raw Data Chunk: 0x00 57 A9 08 18 08 E0 88 18 08 E0 48 18 08 E0 00 10 08 E0 18:37:40:238:Send Raw Data Chunk: 0x00 08 FD 00 00 10 FD 00 00 20 FD 00 00 40 FD 00 00 80 FD		
Override sector bitmap:	0x000000000000000000000000000000000000		18:37:40:238:Send Frame 5381 offset: 0x4ECF0of 0x4F071 18:37:40:244:Send Frame 5381 offset: 0x4ECF0of 0x4F071	00 00 10 10 00 00 30 11	
Image Signature			18:37:40:244:Send Raw Data Chunk: 0x04 44 00 10 00 01 FD 00 04 A0 00 10 00 01 FD 00 04 04 01	06 00 01 FD 00 04 1C 02	
Include Image Signature	in OTA file		18:37:40:257:Send Frame 5383 offset: 0x4ED68of 0x4F071 18:37:40:257:Send Raw Data Chunk: 0x00 0A FD 00 08 03 04 00 00 01 01 00 00 0A FD 00 08 01 03	00 00 01 01 00 00 0A FD	
Current selected signature:	CRC signature Change Signature		18:37:40:269:Send Frame 5384 offset: 0x4EDA4of 0x4F071 18:37:40:269:Send Raw Data Chunk: 0x03 00 BF 18 03 F7 00 00 FF D7 00 01 02 40 FA 00 00 08 F4	00 00 14 F8 00 FC FF 0B	
Multicast Option			18:37:40:282:Send Frame 5385 offset: 0x4EDE0of 0x4F071 18:37:40:282:Send Raw Data Chunk: 0x20 05 FB 00 16 70 1D 05 00 78 19 05 00 70 22 05 00 94 25	05 00 7C 22 05 00 88 22	
Enable OTA Multicast			18:37:40:288:Send Frame 5386 offset: 0x4EE1Cof 0x4F071 18:37:40:288:Send Raw Data Chunk: 0xF9 00 F0 1C 01 FE 80 E9 00 04 09 3D 00 00 80 FA 00 FB FF	ED 00 0E 64 22 05 00 EC	
			18:37:40:292:Send Frame 5387 offset: 0x4EE58of 0x4F071		
			18:37:40:292:Send Raw Data Chunk: 0x00 08 SC 24 05 00 E0 0F 05 00 01 FD 00 03 58 EF 04 00 FC 18:37:40:296:Send Frame 5388 offset: 0x4EE94of 0x4F071		
			18:37:40:296:Send Raw Data Chunk: 0xFC FF 0B D4 14 05 00 EF BE AD DE 1F 21 23 00 F8 FF F8 00 18:37:40:308:Send Frame 5389 offset: 0x4EED0of 0x4F071		
			18:37:40:308:Send Raw Data Chunk: 0xFF FF	FF FF FF FF FF FF FF FF	
			18:37:40:319:Send Raw Data Chunk: 0xFF FF	FF FF FF FF FF FF FF	
			18:37:40:333:Send Raw Data Chunk: 0xFF FF	FF FF FF FF FF FF FF FF	
			18:37:40:339:Send Frame 5392 offset: 0x4EF84of 0x4F071 18:37:40:339:Send Raw Data Chunk: 0xFF FF	FF FF FF FF FF FF FF FF	
			18:37:40:352:Send Frame 5393 offset: 0x4EFC0of 0x4F071 18:37:40:352:Send Raw Data Chunk: 0xFE FE F	FE FE FE FE FE FE FE FE	
			18:37:40:361:Send Frame 5394 offset: 0x4EFFCof 0x4F071 18:37:40:361:Send Raw Data Chunk: 0x4EFFCof 0x4F071		
			18:37:40:402:Send Frame 5395 offset: 0x4F038of 0x4F071		
			18:37:40:402:Send Raw Data Chunk: 0xFF FF 00 F0 1F 00 00 00 FF FF FF FF	FF FF FF FF FF FF FF FF	
			Save Session Log		
			_ sure season coy	Copy Clear	

Figure 15. Running OTA procedure multicast

## Chapter 7 Revision History

This table summarizes revisions to this document.

	Revision history							
Revision number	Date	Substantive changes						
0	10/2015	Initial release						
1	03/2016	Removed ambiguous references to FSCI length field						
2	06/2016	Updates related to the Thread OTA Multicast implementation and its corresponding Test Tool updates						
3	08/2016	Updates for Thread KW41 Beta Release						
4	09/2016	Updates for Thread KW41 GA Release						
5	12/2016	Updates for Thread KW24D GA Release						
6	3/2017	Updates for Thread KW41 MCUX Release						
7	01/2018	Updates for Thread KW41 Maintenance Release						

## Chapter 8 Copyright

How To Reach Us Home Page: nxp.com Web Support: nxp.com/support Information in this document is provided solely to enable system and software implementers to use NXP products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits based on the information in this document. NXP reserves the right to make changes without further notice to any products herein.

NXP makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does NXP assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters that may be provided in NXP data sheets and/or specifications can and do vary in different applications, and actual performance may vary over time. All operating parameters, including "typicals," must be validated for each customer application by customer's technical experts. NXP does not convey any license under its patent rights nor the rights of others. NXP sells products pursuant to standard terms and conditions of sale, which can be found at the following address: nxp.com/SalesTermsandConditions.

NXP, the NXP logo, NXP SECURE CONNECTIONS FOR A SMARTER WORLD, All other product or service names are the property of their respective owners. ARM, AMBA, ARM Powered, are registered trademarks of ARM Limited (or its subsidiaries) in the EU and/or elsewhere. All rights reserved.

© 2017 NXP B.V.

