Kinetis W MCUs: Over The Air Programming Thread Lab

Lab Hand Out

Contents

1 Lab Overview

This lab will use two FRDM-KW41Z boards to show how to do Over-The-Air-Programming (OTAP) updates. One board will be used as a server to send out the update, and the other board will be a client receiving the update.

2 Prerequisites

The following items are needed to complete this hands-on lab. This has already been installed on your computer:

- Boards
  - Two FRDM-KW41Z
- Software
  - MCUXpresso SDK for FRDM-KW41Z (includes Thread stack): http://mcuxpresso.nxp.com
  - MCUXpresso IDE: http://nxp.com/mcuxpresso/ide
  - Terminal Software (like TeraTerm or PuTTY)

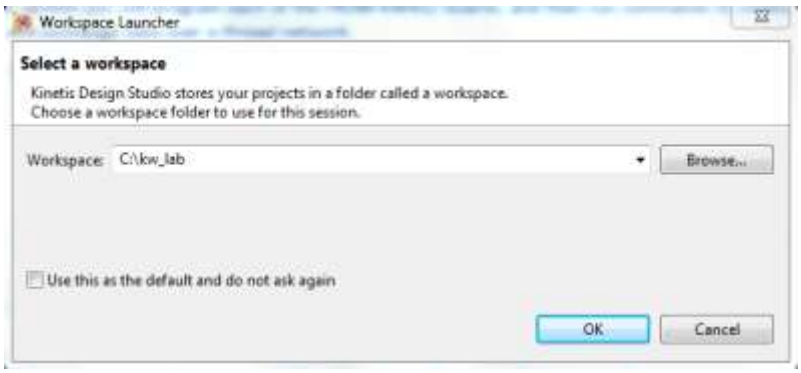
3 Programming the OTA Client

In this section you will program one of the FRDM-KW41Z boards with the OTA client firmware. This consists of two parts: First programming in the bootloader, and then programming in the router_eligible_device OTA firmware.

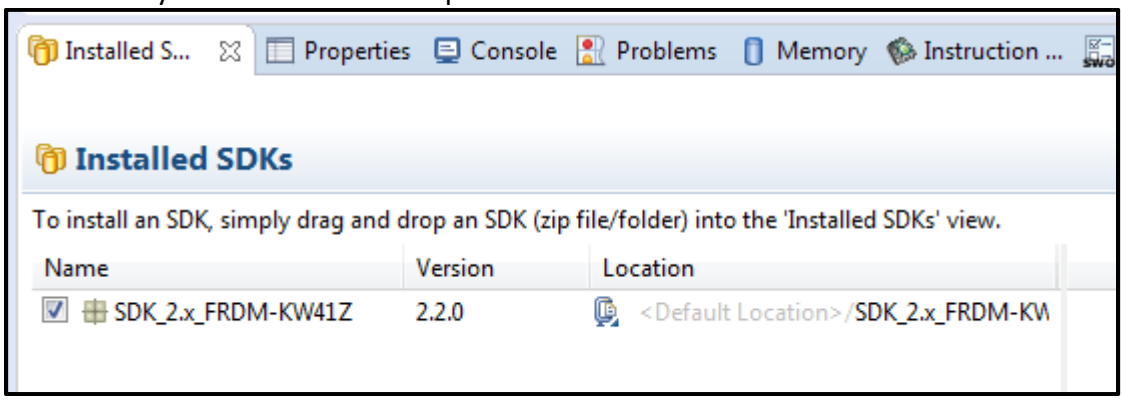
3.1 Programming the Bootloader

1. First we need to program in the bootloader firmware, which is what will be used to load the new image we receive over the air. This bootloader firmware is not part of the main Thread OTA client application, so it needs to be programmed into the flash separately first.
2. Open MCUXpresso IDE

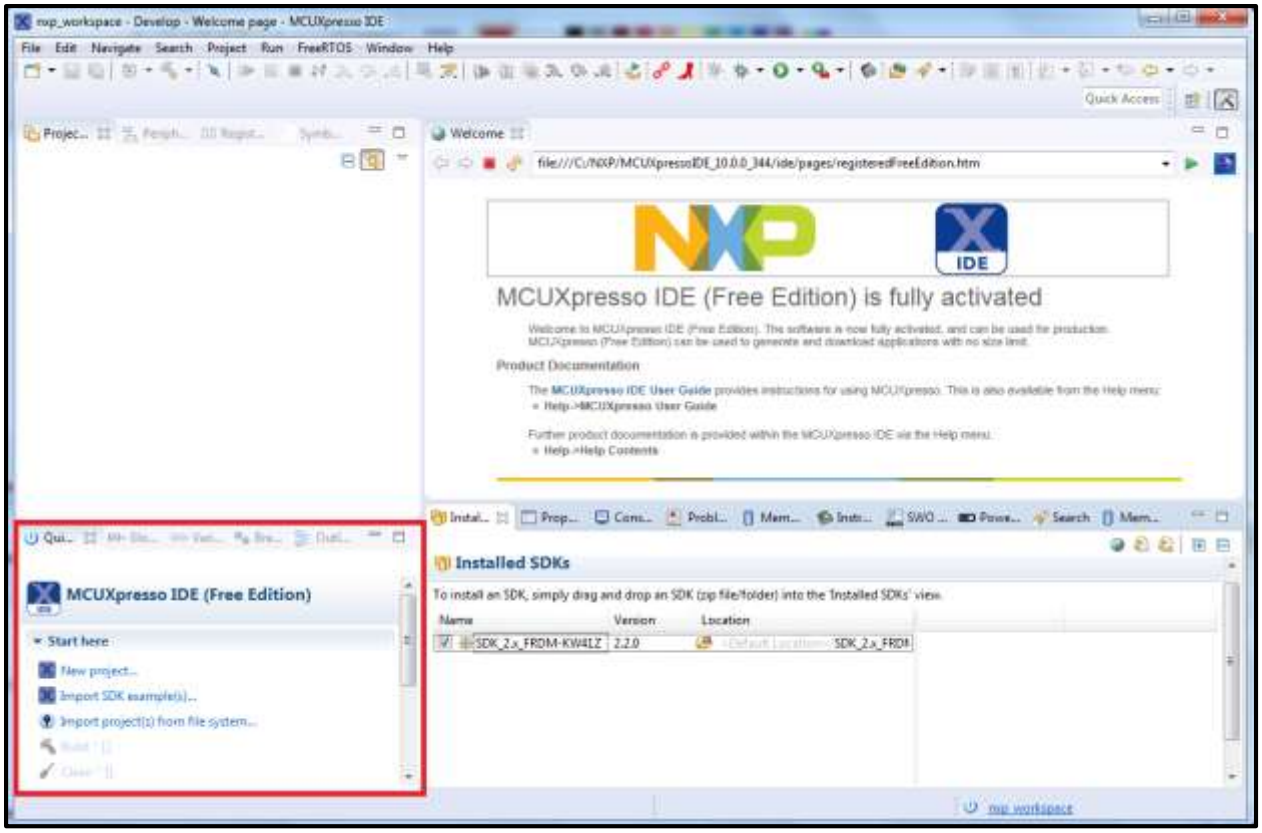
3. Set the workspace directory to **C:\kw_lab** (or your choice of any empty new directory) and click on OK.



4. If not done already, import the KW41Z MCUXpresso SDK into the MCUXpresso IDE. This should have already been done from the previous labs.

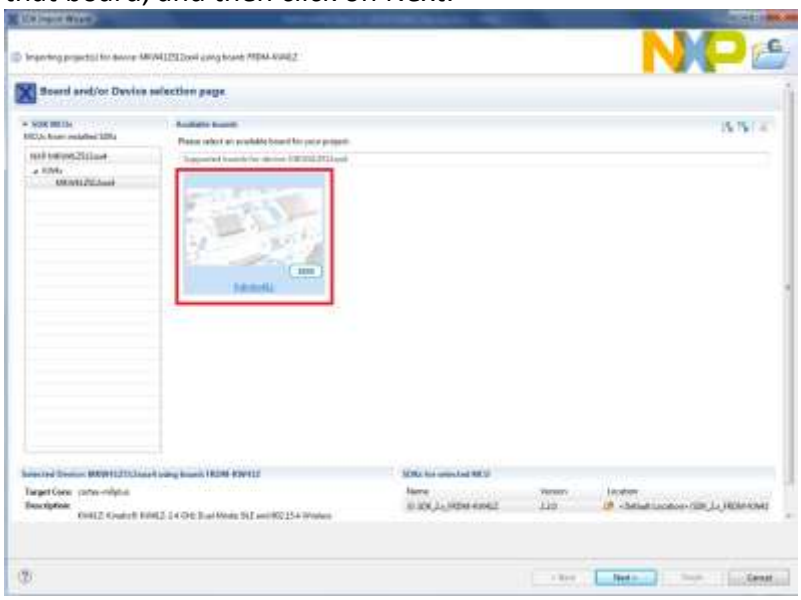


5. Next we will import the "bootloader_otap" project for the FRDM-KW41Z.
6. Find the Quickstart Panel in the lower left hand corner

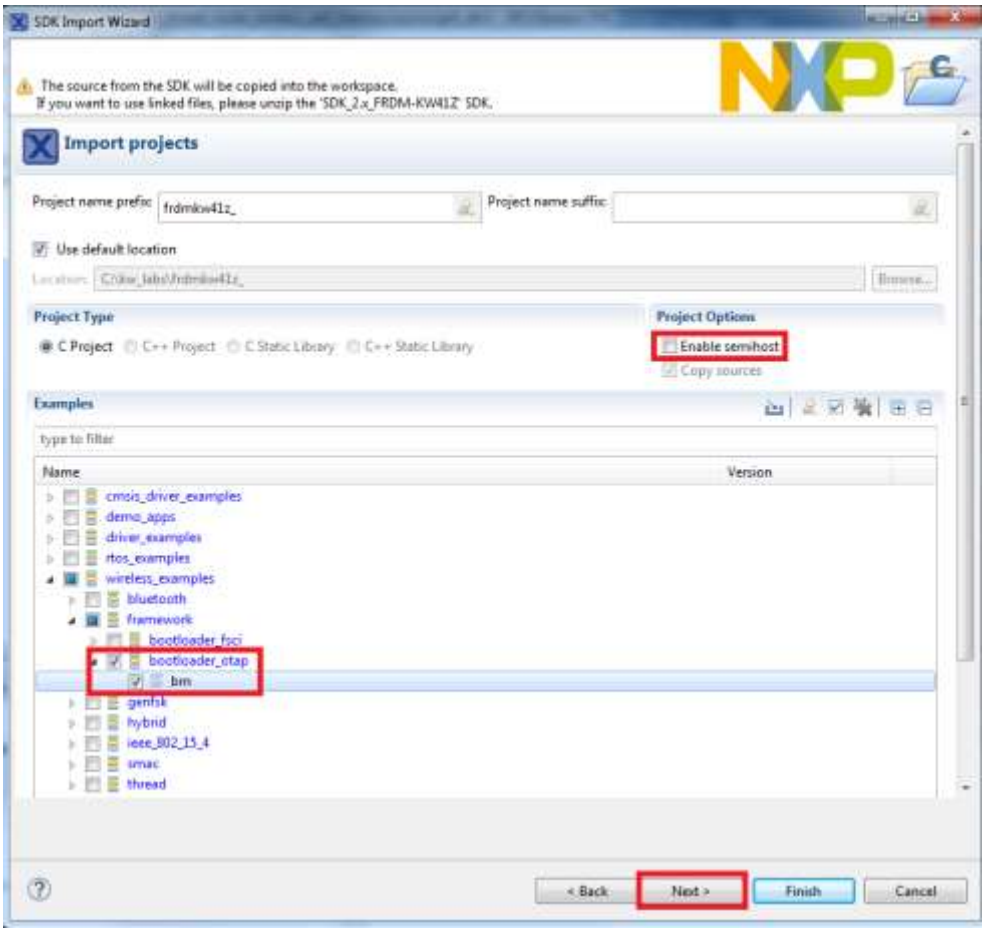


7. Then click on Import SDK examples(s)…

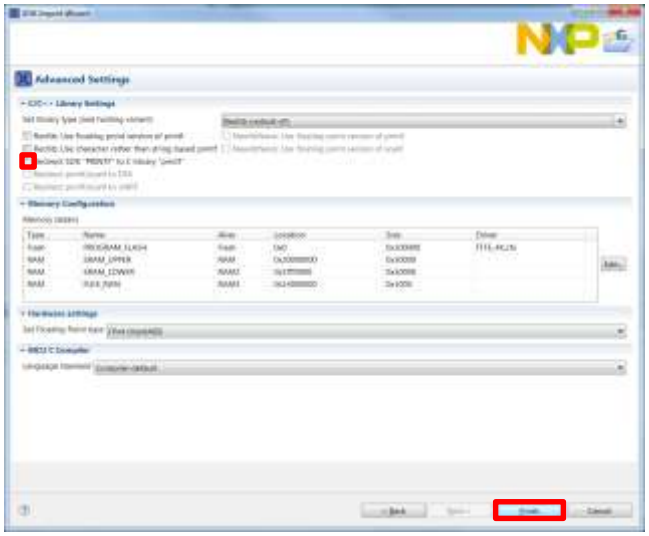
8. Click on the frdmKW41Z board to select that you want to import an example that can run on that board, and then click on Next.



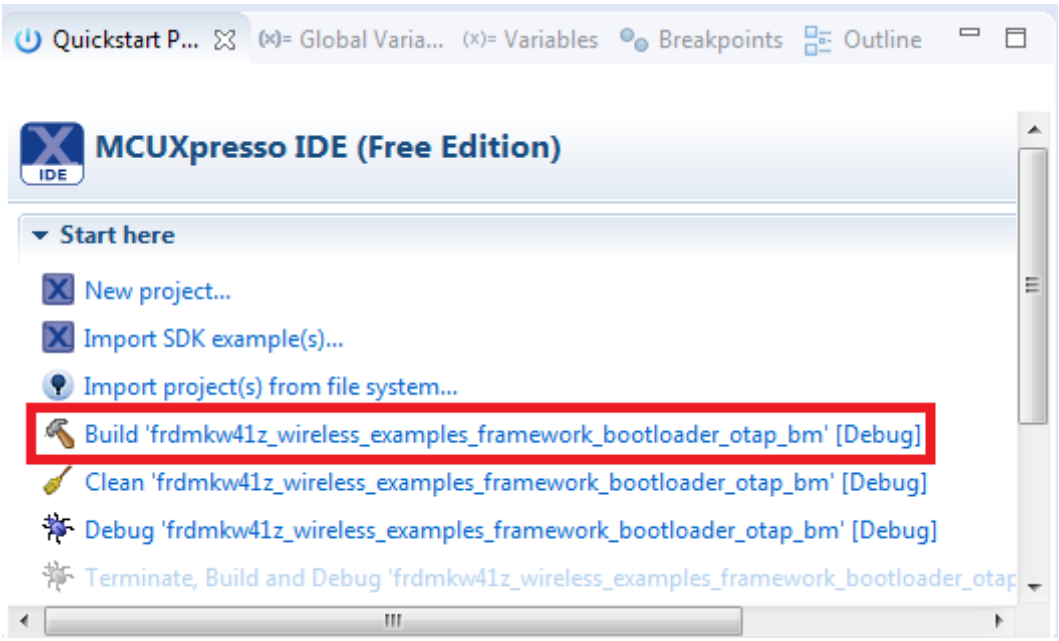
9. Use the arrow button to expand the **wireless_examples** category, and then under the **framework** category expand the **bootloader_otap** project and select the **bm** version of project. To use the UART for printing (instead of the default semihosting), clear the "**Enable semihost**" checkbox under the project options. Then, click on **Next**.



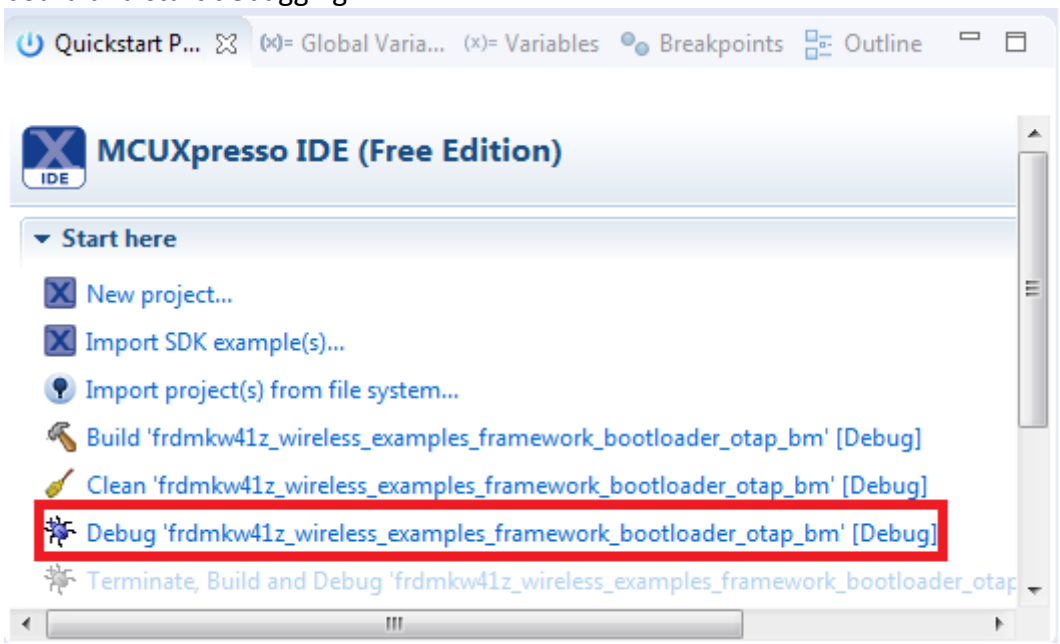
10. On the Advanced Settings wizard, clear the checkbox "**Redirect SDK "PRINTF" to C library "printf"**" in order to use the MCUXpresso SDK console functions for printing instead of generic C library ones. Then click on **Finish**.



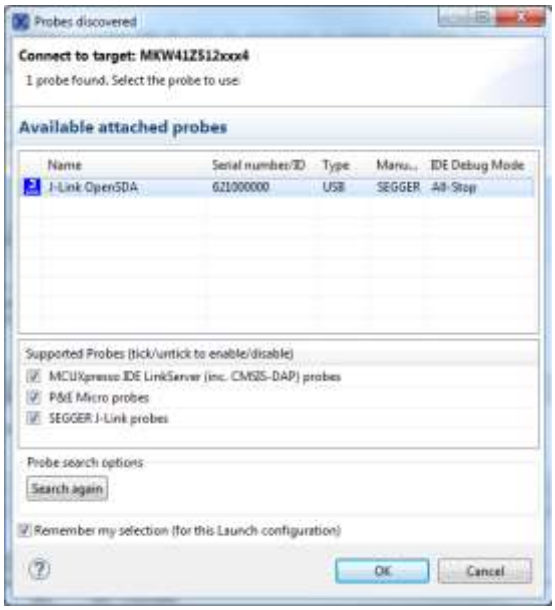
11. Now build the project by clicking on the project name and then in the Quickstart Panel click on Build.



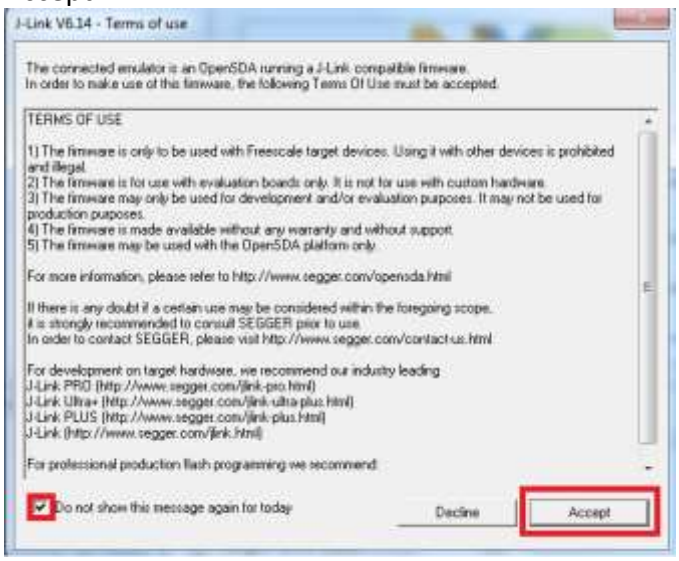
12. Download the firmware you just compiled to one of the FRDM-KW41Z boards
    a. Connect the micro-USB cable to the USB connector (J6) on the board.
    b. Windows may begin to install some drivers, this is normal and should be allowed to complete before downloading the application.
    c. Click on the Debug option in the Quickstart Panel to download the firmware to the board and start debugging:



d. MCUXpresso IDE will probe for connected boards and should find the JLink debug probe that is part of the integrated OpenSDA circuit on the FRDM-KW41Z. Click on OK to continue.



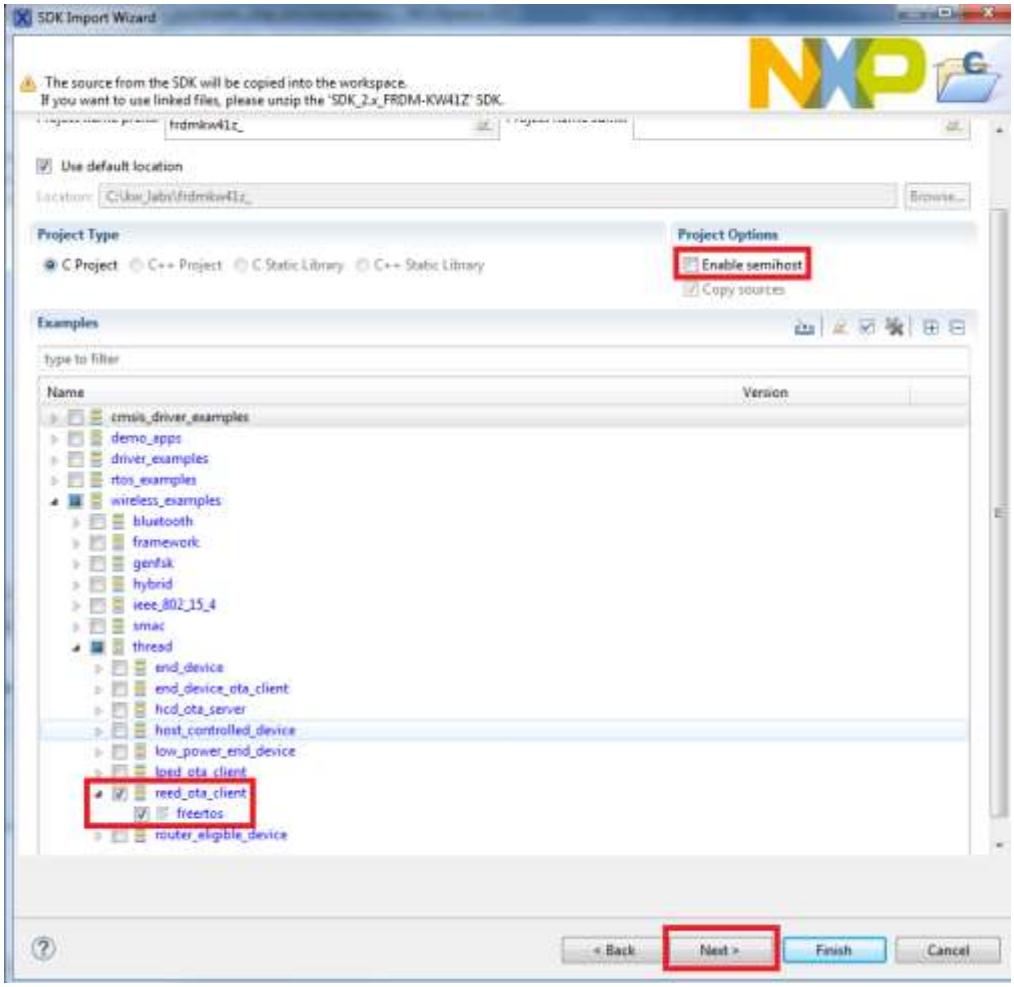
e. You may see the following message if this is your first debug with JLink of the day. Click on the checkbox at the bottom to not display the message again, and then click on Accept



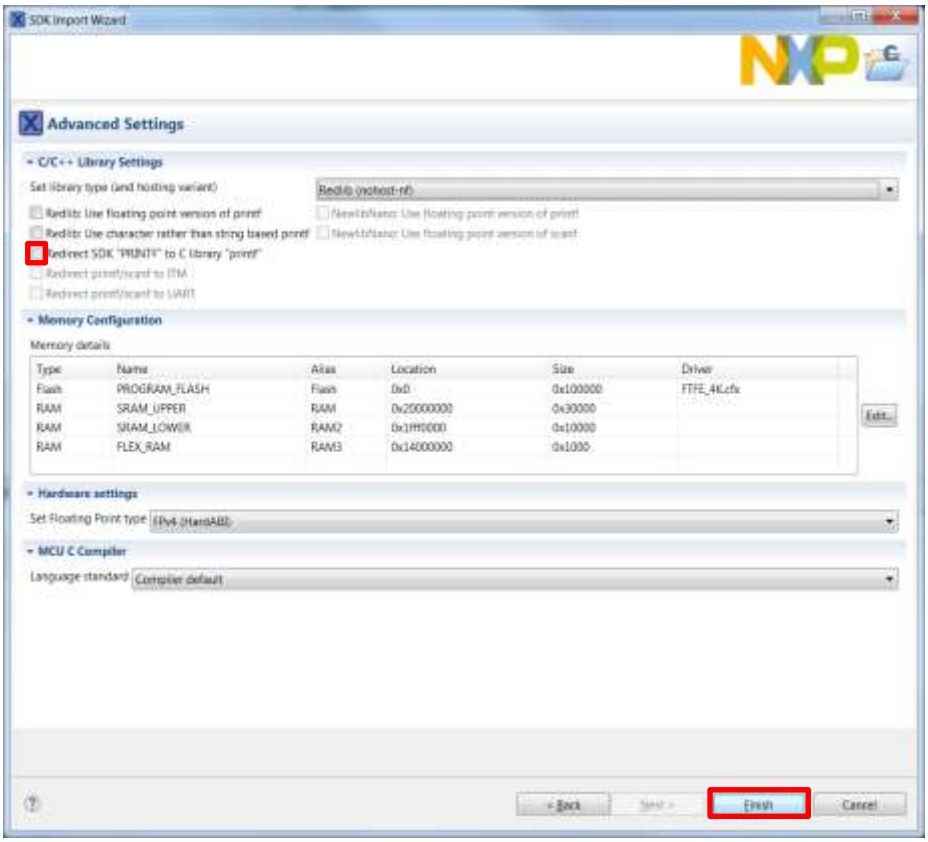
f. MCUXpresso IDE will now flash the board. Once it is finished, click on the Terminate icon to stop debugging.

3.2 Programming the OTA Client Firmware

13. Next we will import the "reed_ota_client" demo for the FRDM-KW41Z which is the main Thread Router Eligible Edge Device application.
14. Import the **reed_ota_client** project like you did previously for the other project. It can be found under **wireless_examples**->**thread** group with the name **reed_ota_client**. Make sure to uncheck **Enable Semihost** so it is cleared.

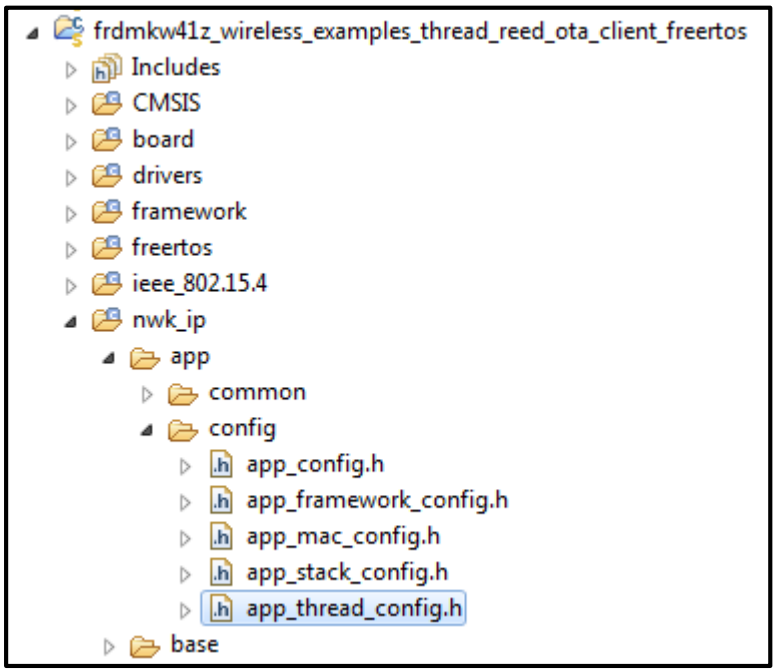


15. On the Advanced Settings wizard, clear the checkbox "**Redirect SDK "PRINTF" to C library "printf"**" in order to use the MCUXpresso SDK console functions for printing instead of generic C library ones. Then click on **Finish**.



16. Now we will change some configuration options to make your board unique for this lab, avoid wireless interference, and ensure you join your thread network, and not your neighbor's thread network.

17. Change the **THR_SCANCHANNEL_MASK** to something unique in **app_thread_config.h** so that your boards will not conflict with other boards in the class on the same channel. In a real system a channel scan would determine the least used channel, but we will hardcode it for this lab:
    a. File is located here:

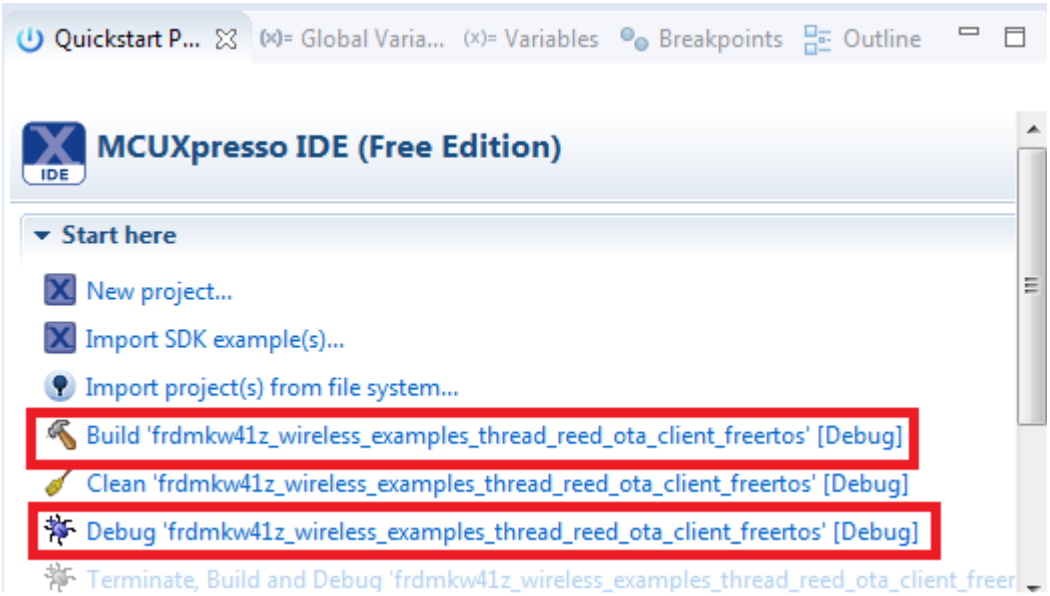


b. Edit the channel number (it's '25' by default) to match the **"Channel Number"** value that is printed on the sticker located on your FRDM-KW41Z box. After changing it, it would look similar to:
#define THR_SCANCHANNEL_MASK (0x00000001 << 19)

18. Also in that same **app_thread_config.h** file, change the **THR_PSK_D** to define to a unique PSKd for your particular set of boards, which is the password used when joining a Thread network.
    a. Append the value of the **"Unique Kit Number"** that can be found on the sticker on your FRDM-KW41Z box. Note you will also need to change the size
    b. When done it should look similar to the following:
    #define THR_PSK_D {16, "Kinetis_Thread33"}

19. Exploring the **app_thread_config.h** file further, you can see that this is where the master network key is set (THR_MASTER_KEY), where the PSKd password sent by a joiner node to the commissioner is set (THR_PSK_D), where the Extended PAN ID is set (THR_EXTENDED_PAN_ID), and many other Thread configuration options. You'll notice that the PAN ID is assigned to all FF's, which signals the stack to make it random, These values will not need to be changed for this lab.

20. Now build and flash the project like done previously for other projects using the Quickstart Panel window.

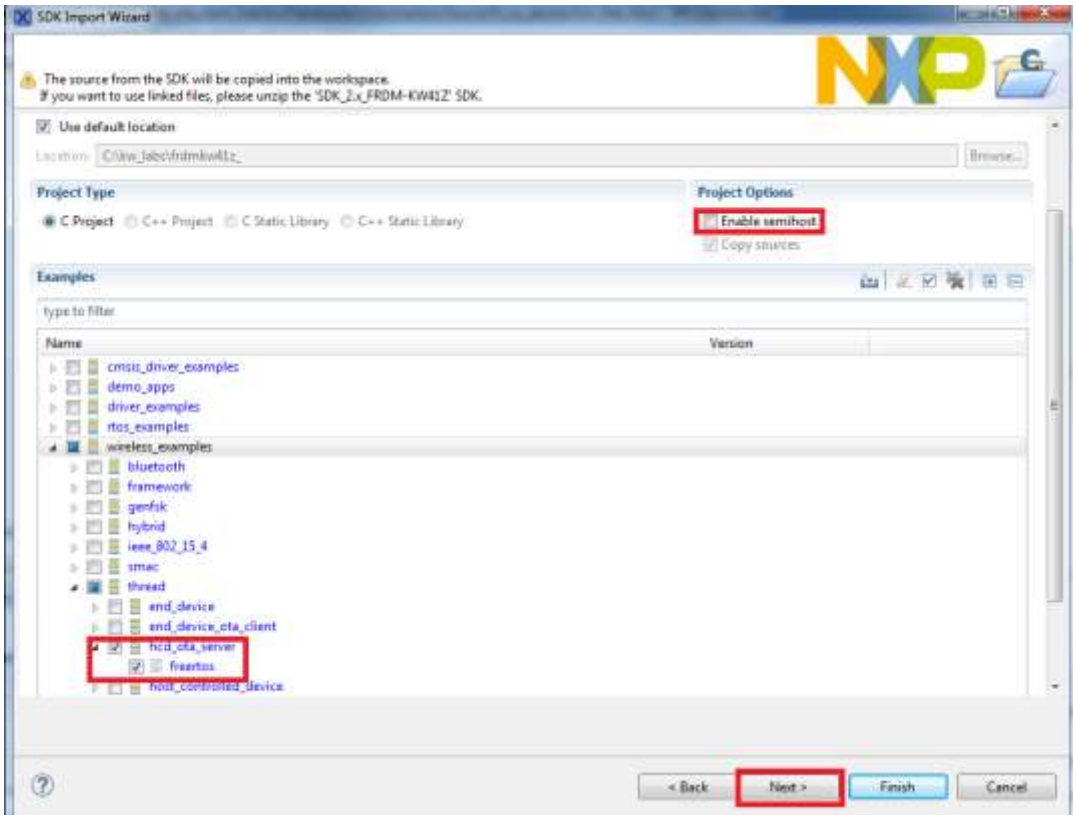


21. Verify that the firmware runs correctly by unplugging and plugging back in the board. Open up a TeraTerm command prompt, connect at 115200 baud, and hit Enter, and you should see the Thread command line.
22. Now unplug this client board, and plug in the other board to the computer (which will become the OTA server).

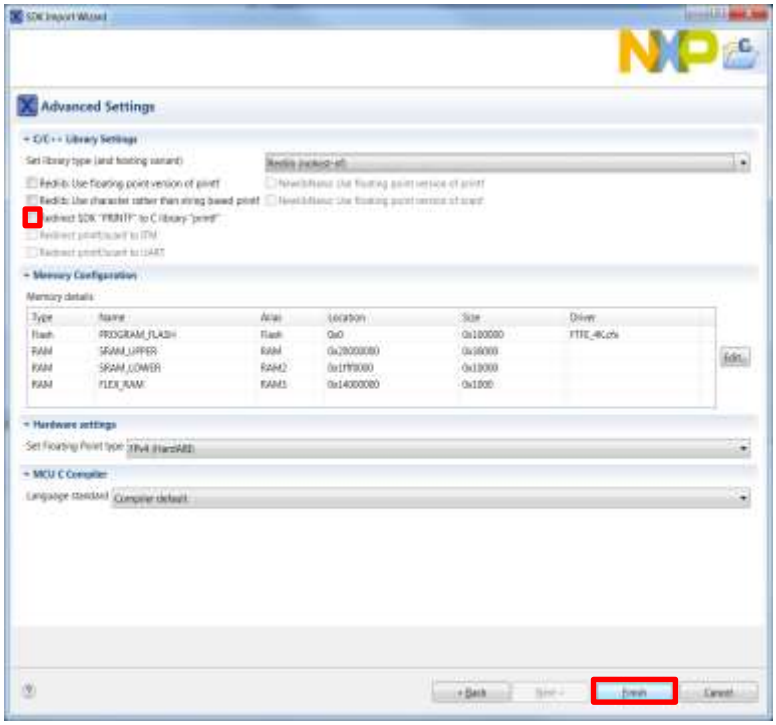
4 Programming the OTA Server

The following section will walk through programming the OTA server firmware into the other board.

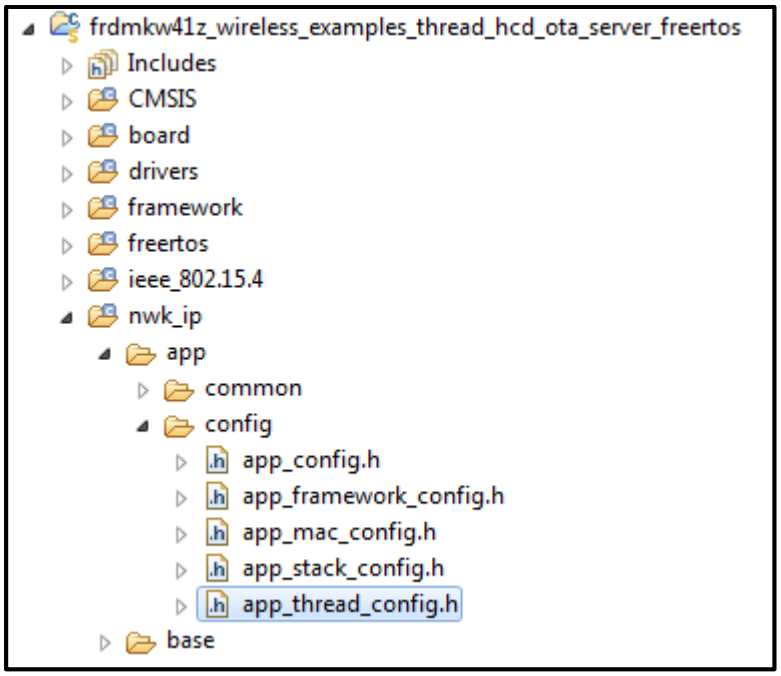
1. Ensure only the other board (that is not the client board) is connected to your computer.
2. We're now going to load the Host Controlled Device Over-the-Air server project, which is what will send out the over the air updates using a program called Test Tool on the PC. The Test Tool program communicates with the FRDM-KW41Z over a UART connection and sends it commands using the FSCI interface.
3. Import the **hcd_ota_server** project like you did previously for the other project. It can be found under **wireless_examples**->**thread** group with the name **hcd_ota_server**. Make sure to uncheck **Enable Semihost** so that it is cleared.



4. On the Advanced Settings wizard, clear the checkbox "**Redirect SDK "PRINTF" to C library "printf"**" in order to use the MCUXpresso SDK console functions for printing instead of generic C library ones. Then click on **Finish**.



5. Change the **THR_SCANCHANNEL_MASK** to something unique in **app_thread_config.h** so that your boards will not conflict with other boards in the class on the same channel. In a real system a channel scan would determine the least used channel, but we will hardcode it for this lab:
    a. File is located here:



    b. Edit the channel number (it's '25' by default) to match the **"Channel Number"** value that is printed on the sticker located on your FRDM-KW41Z box. After changing it, it would look similar to:
    #define THR_SCANCHANNEL_MASK (0x00000001 << 19)

6. Also in that same **app_thread_config.h** file, change the **THR_PSK_D** to define to a unique PSKd for your particular set of boards, which is the password used when joining a Thread network.
    a. Append the value of the **"Unique Kit Number"** that can be found on the sticker on your FRDM-KW41Z box. Note you will also need to change the size
    b. When done it should look similar to the following:
    #define THR_PSK_D {16, "Kinetis_Thread33"}
7. Now compile the project and flash the FRDM-KW41Z using the Quickstart Panel like done for the other projects.

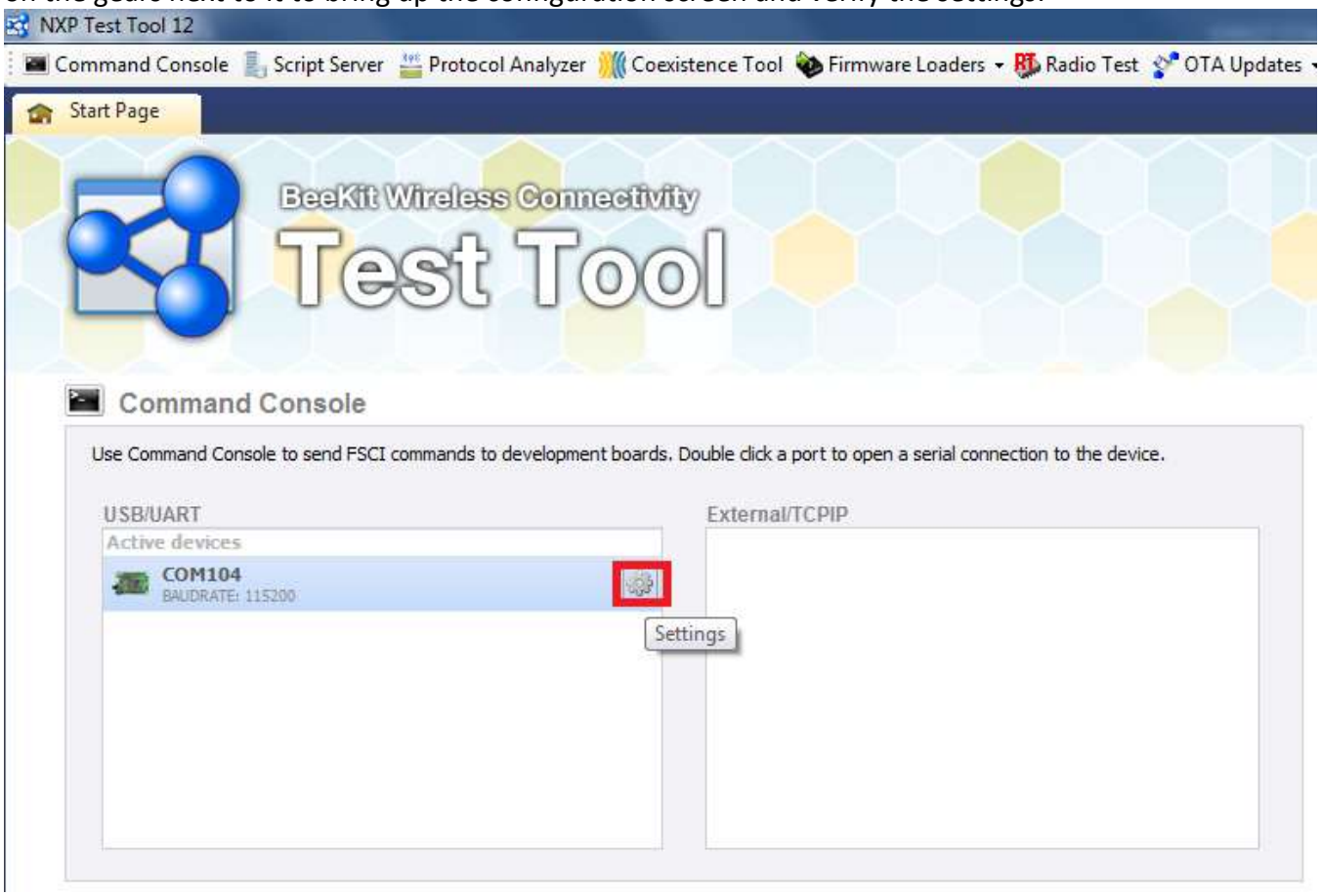
5 Test Tool and OTA Updates

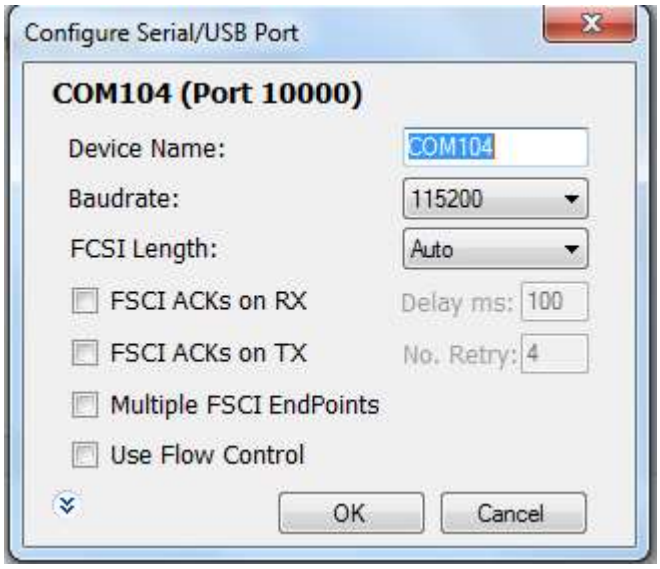
The following section will walk through using the Test Tool application with the OTA Server node to start a Thread network, then you will have the OTA client join that network, and then start the OTA process.

5.1 Test Tool with OTA Server

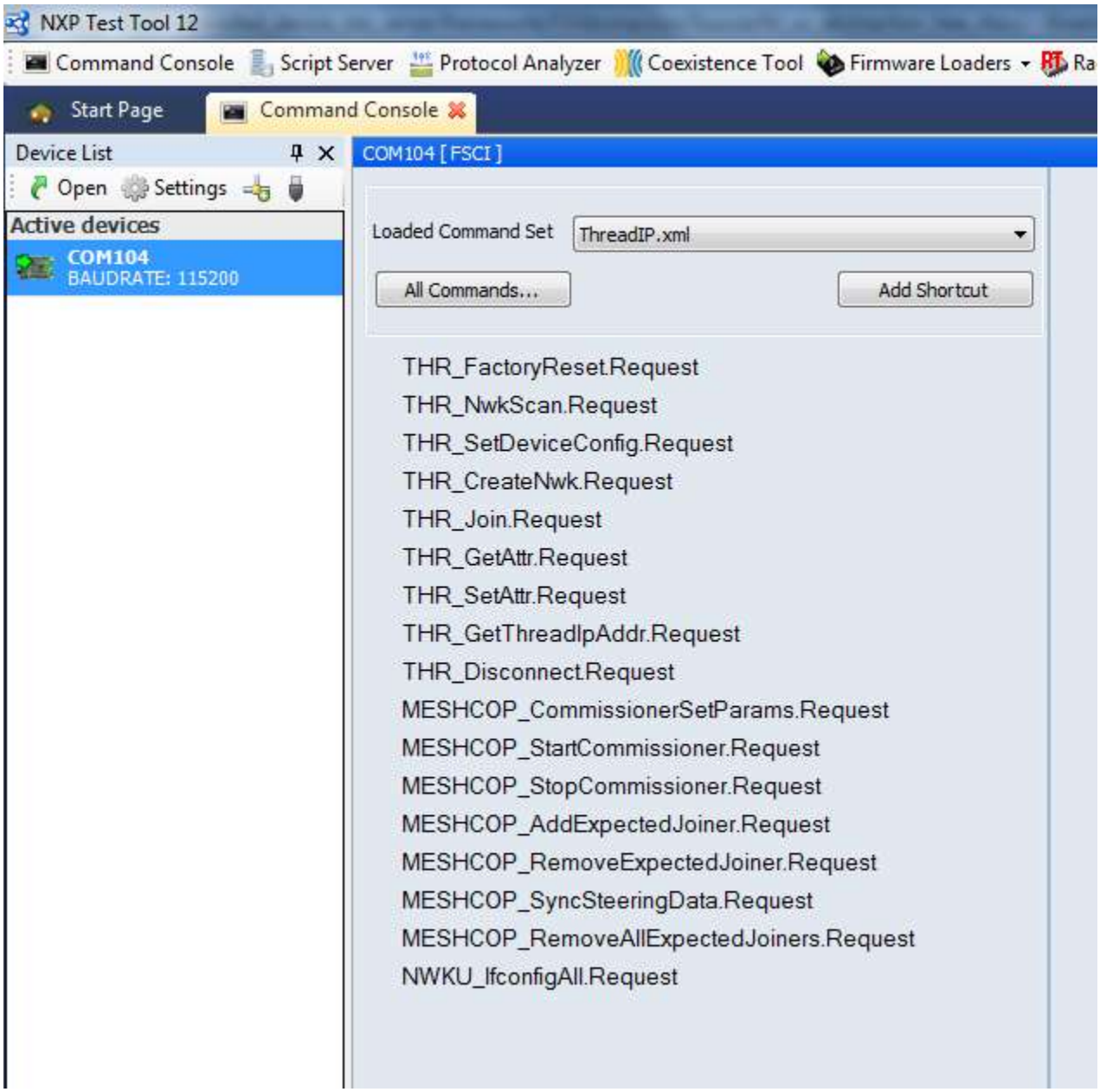
1. We will use the Test Tool application to tell the OTA Server node to create and configure a Thread network.
2. Open up the Test Tool application found on the desktop

3. You should see the OTA Server board enumerated as a COM port. Hover over it, and then click on the gears next to it to bring up the configuration screen and verify the settings:

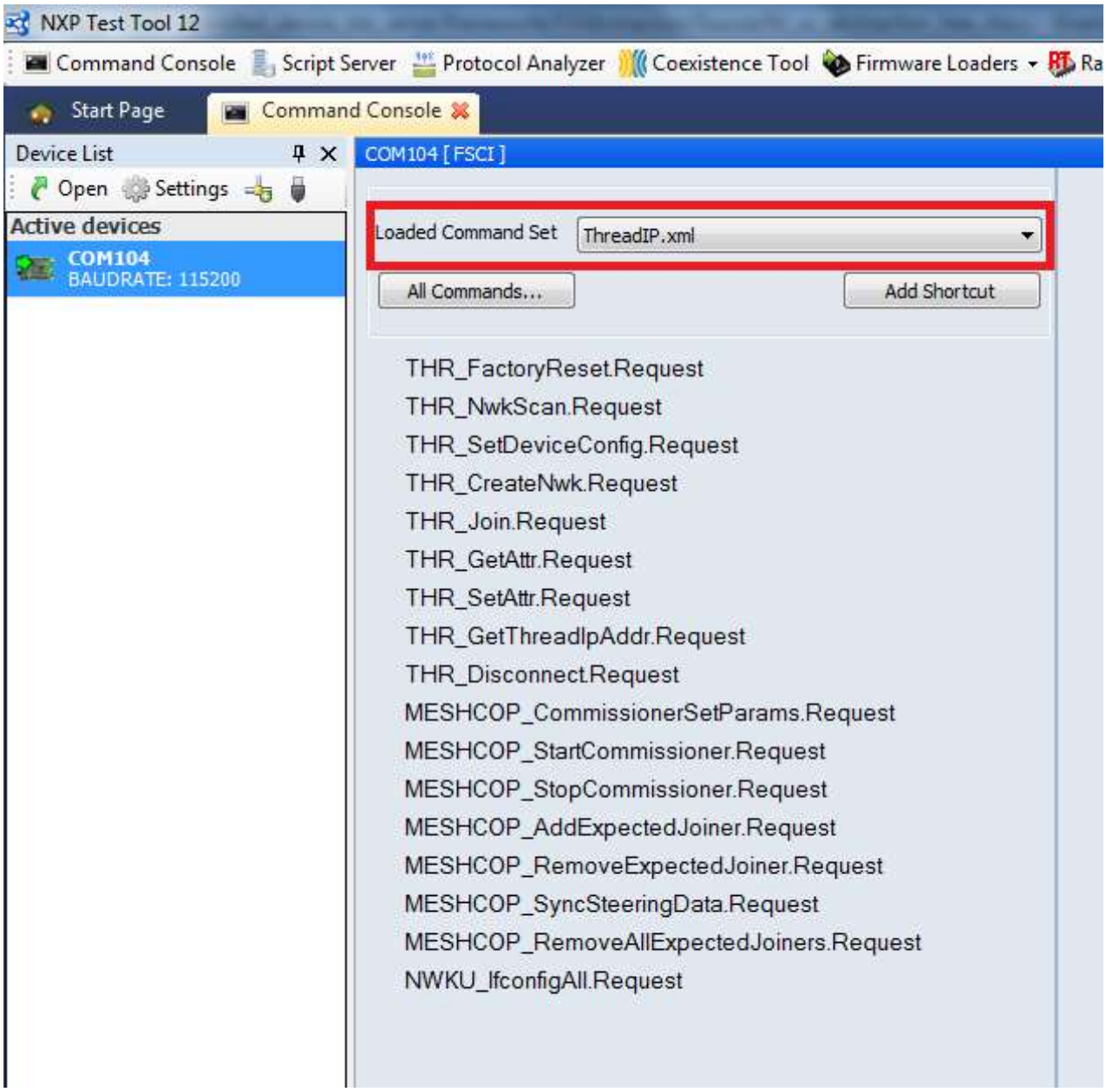




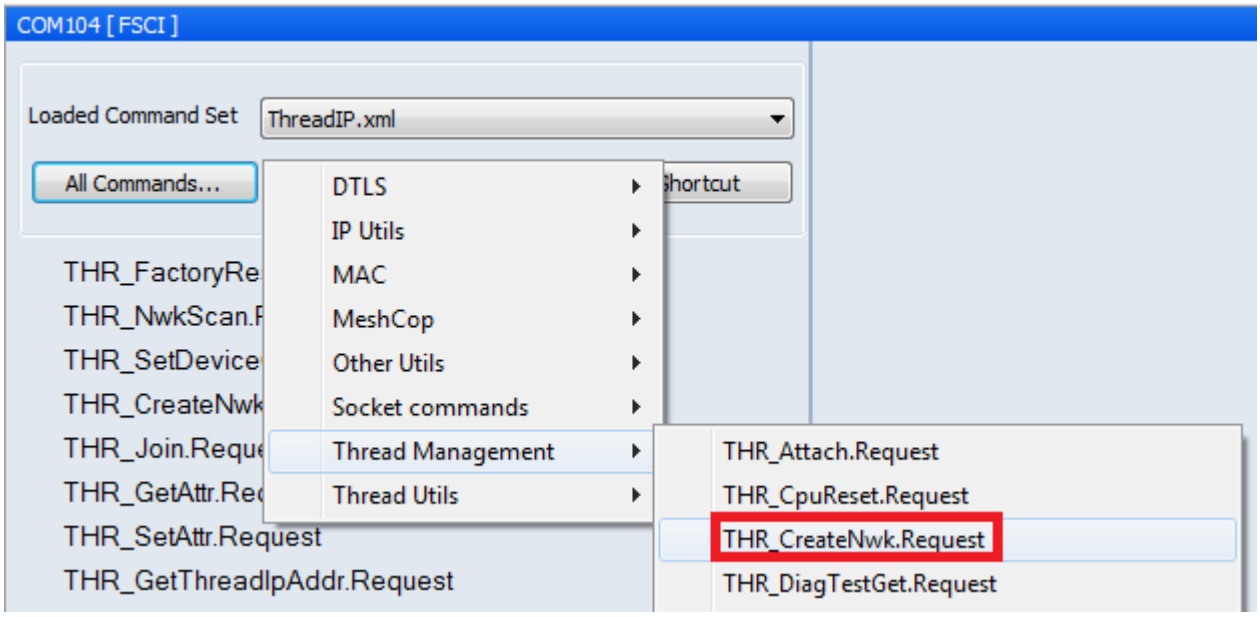
4. Now double click on the UART port, and it will take you to the Command Console.



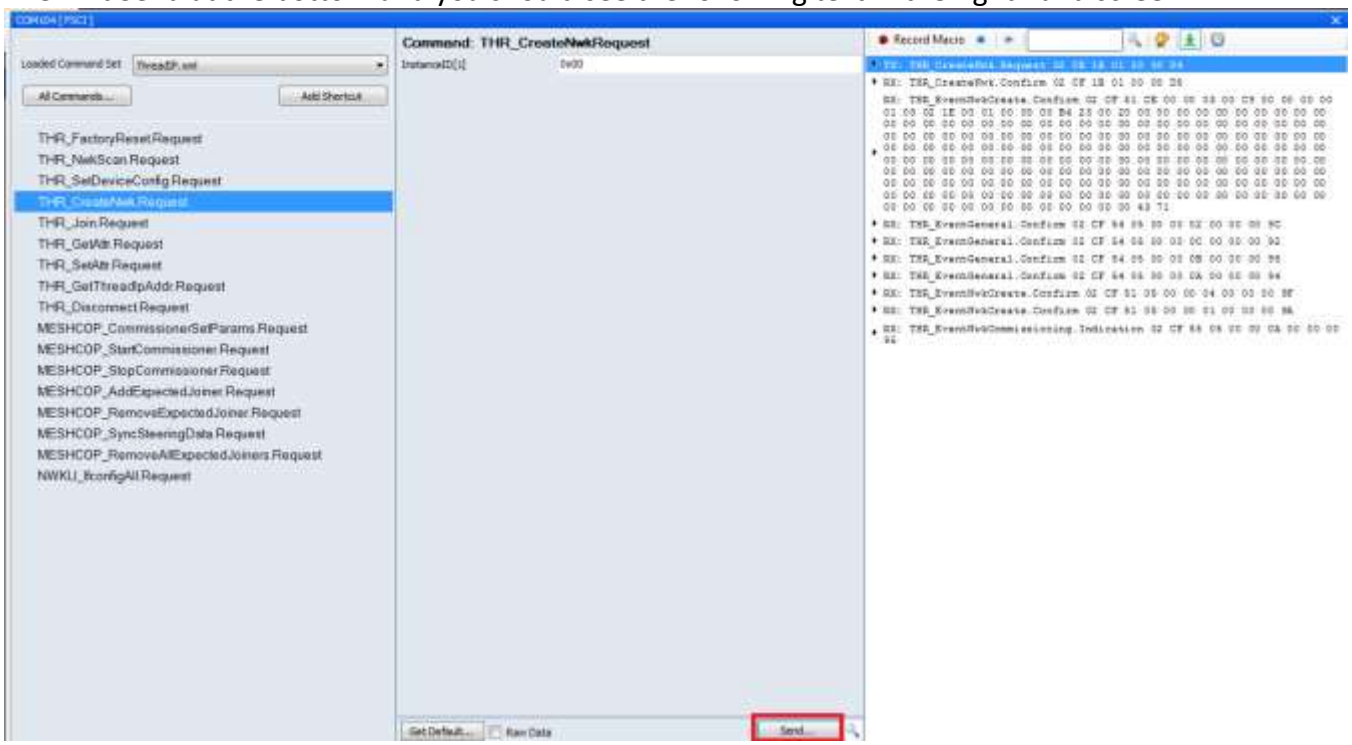
5. If not already set, change the “Loaded Command Set” to ThreadIP.xml, which allows you to select which Thread commands you want to send to the OTA Server node via THCI.



6. Now we are going to create a Thread network by using the THR_CreateNwk.Request command. Select it from All Commands…

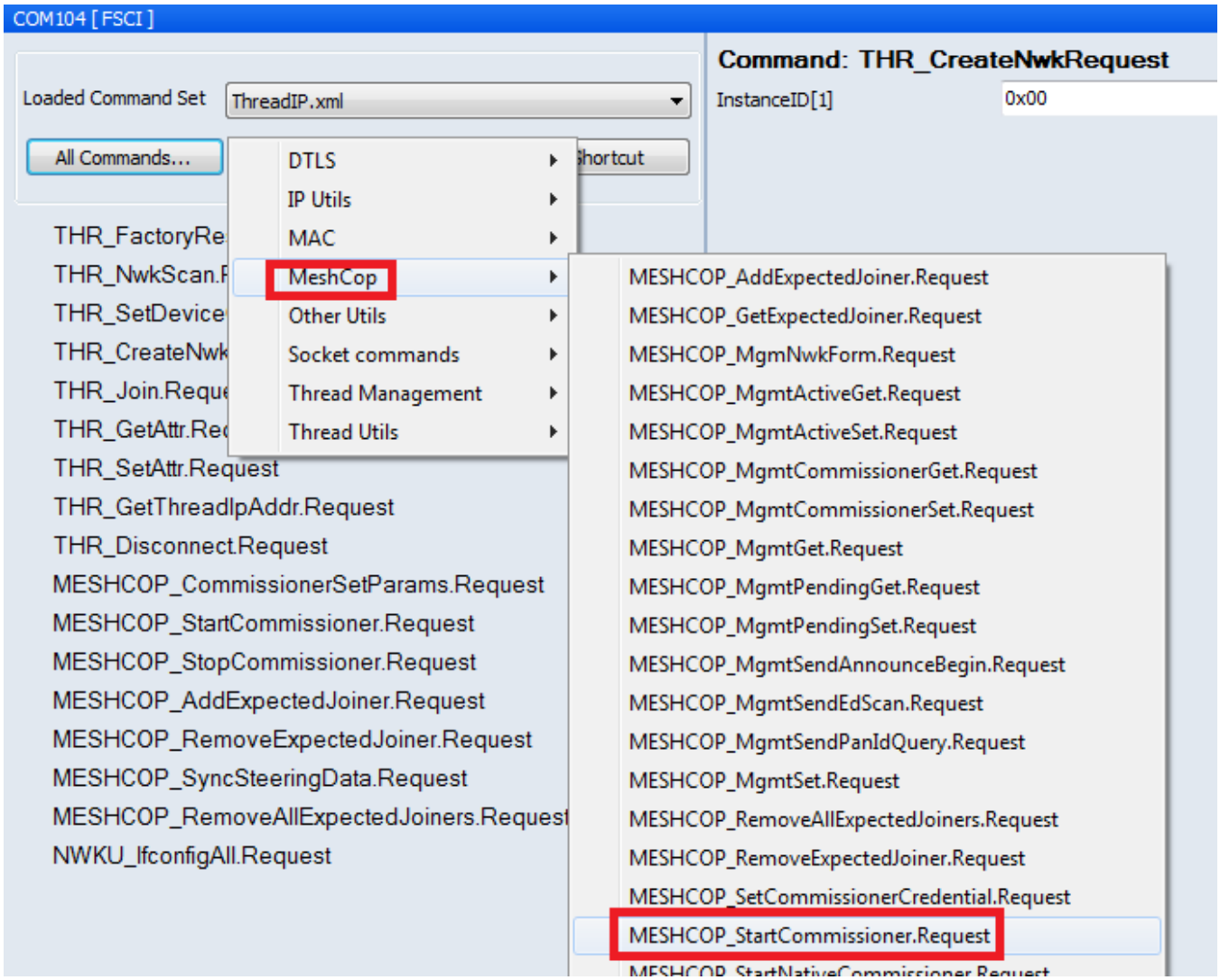


7. Then hit Send at the bottom and you should see the following text in the right hand screen:

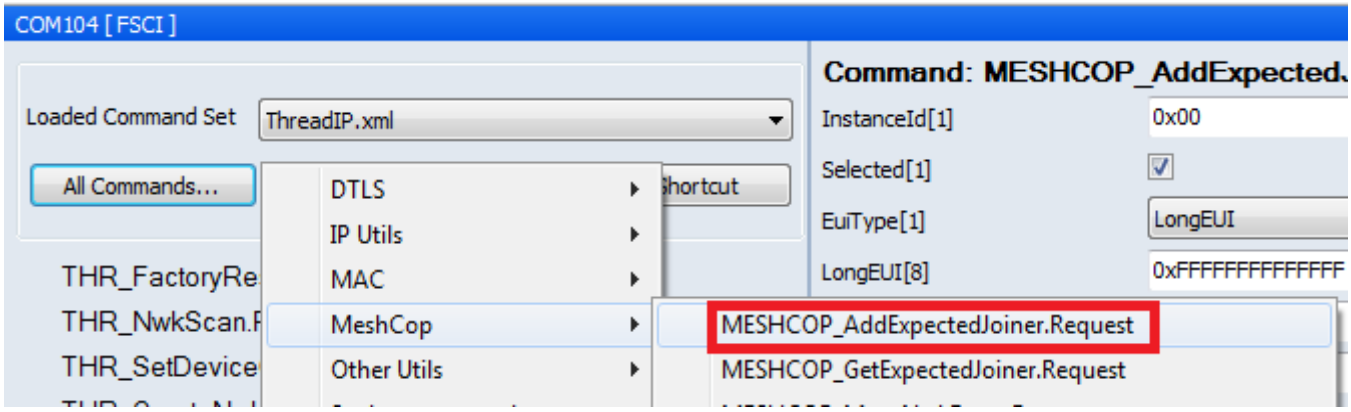


8. Next run the command MESHCOP_StartCommissioner.Request and hit Send to send it over to the OTA Server node and start the commissioner.

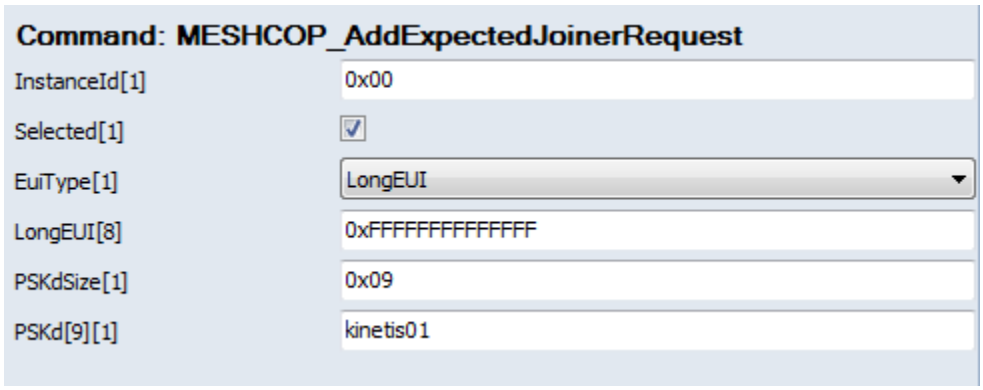
   As a note, when you type “thr create” in the command line like done in other labs, the commissioner was started automatically. But in this case, controlling it via these Thread THCI commands, the commissioner needs to be started explicitly.



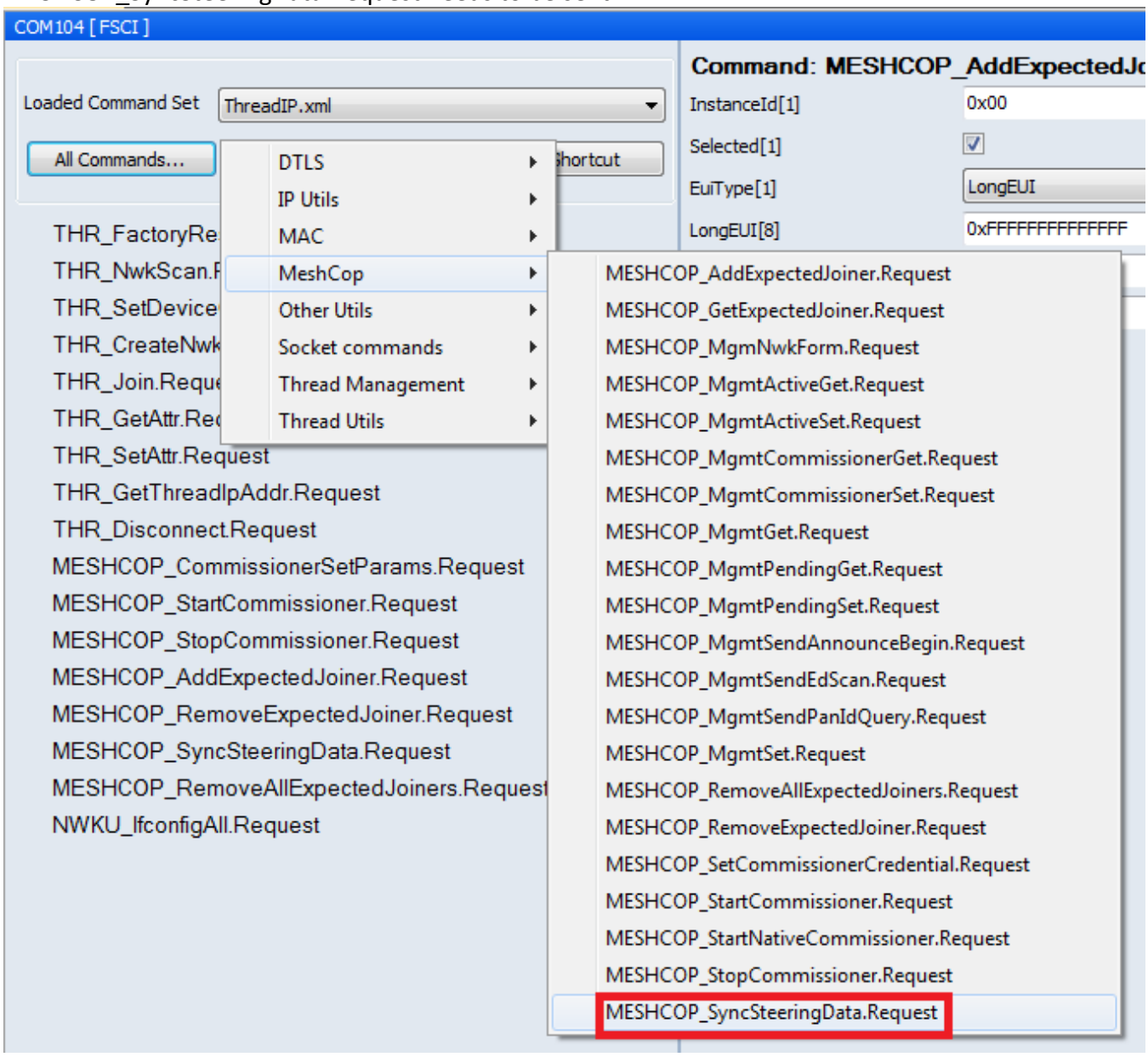
9. Next we'll use the MESHCOP_AddExpectedJoiner.Request command to tell the commissioner now running on this OTA Server node what type of boards can join its network.



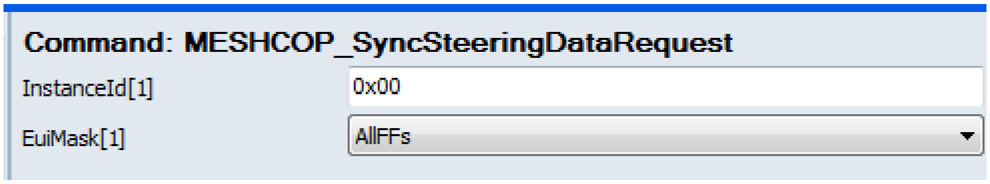
10. You will need to update the ExpectedJoiner fields to allow for a node with any EUI to be able to join, and then also adjust the PSDKdSize to increase it to 9, and then put in the unique password you had set in your project.



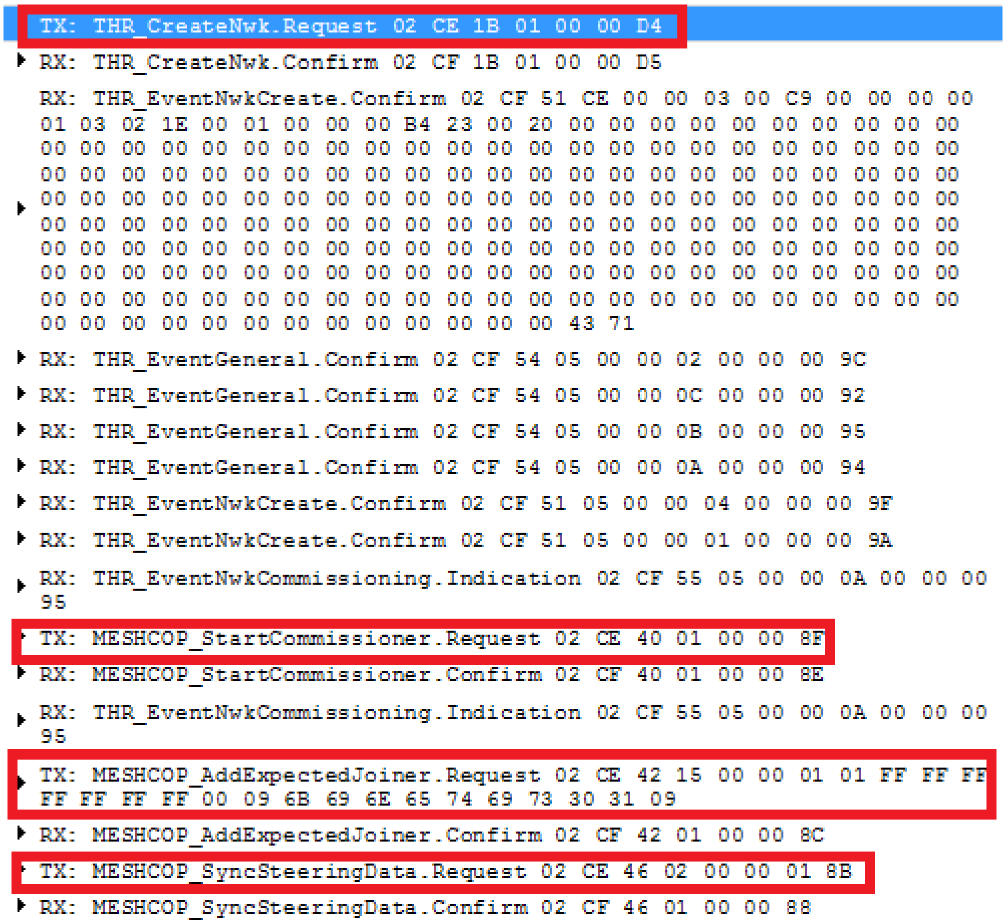
11. When the fields have been filled out, hit “Send” again to send the command to the OTA server.
12. Finally we need to sync up any other node in the network with this data, so a MESHCOP_SyncSteeringData.Request needs to be sent.



13. The EUI mask needs to be changed to AllFFs and then it can be sent using the Send button



14. In the log window on the right side you should now see the 4 commands sent to the OTA server and its responses:



15. Now close the Command Console window by pressing the X in the corner of the tab.

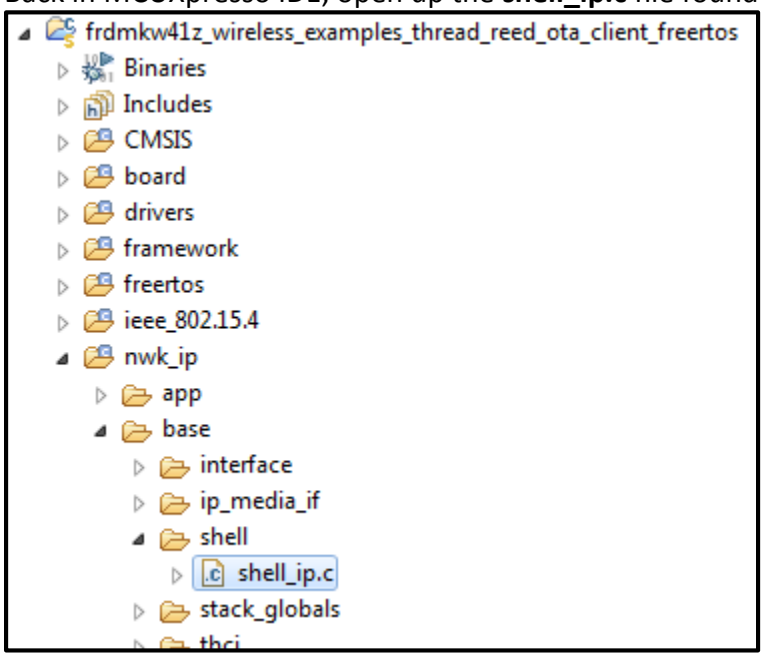
5.2 Join OTA Client to Network

16. Now we can join the client to the network that the OTA server has started.

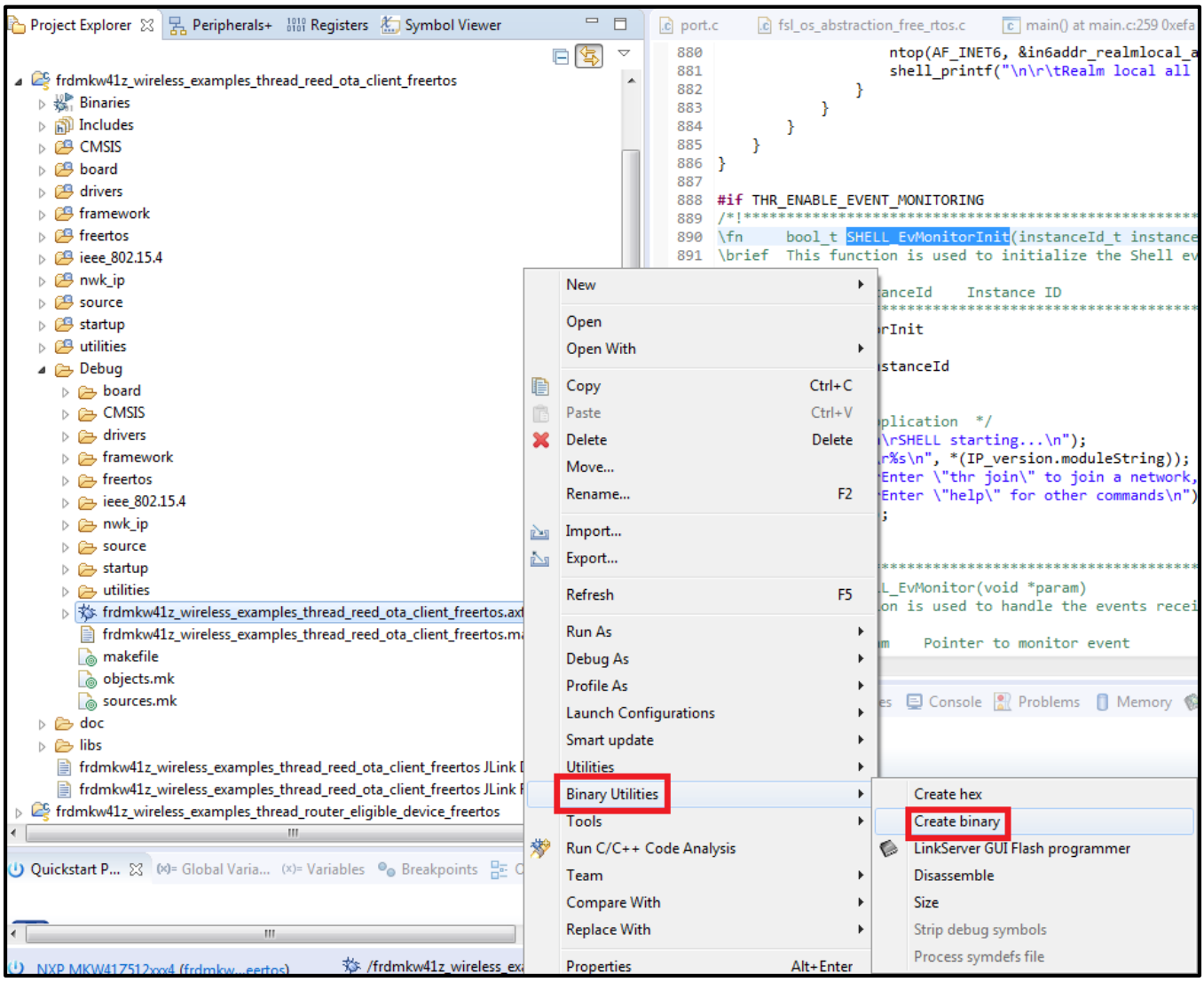
17. Reconnect the client board to your computer and open up a TeraTerm terminal. Connect to the COM port that the client board (and not the server board) is using:
18. Type "**thr join**" to join the existing Thread network.

5.3 Create New Client Project

19. Next we need to update the client side firmware so we can verify that the update took place.
20. Back in MCUXpresso IDE, open up the **shell_ip.c** file found here:



21. Inside SHELL_EvMonitorInit() function, add a new shell_write() command to add an extra line to the welcome message that comes up after a restart on a Thread node:
    shell_write("\rUpdated Firmware Version 1\n");
22. Compile the **reed_ota_client** project like done before.
23. Then we need to create a .bin file that will be sent out over the air. Open up the debug folder and right click on the **frdmkw41z_wireless_examples_thread_reed_ota_client_freertos.axf** file. Select the to create the .bin file.



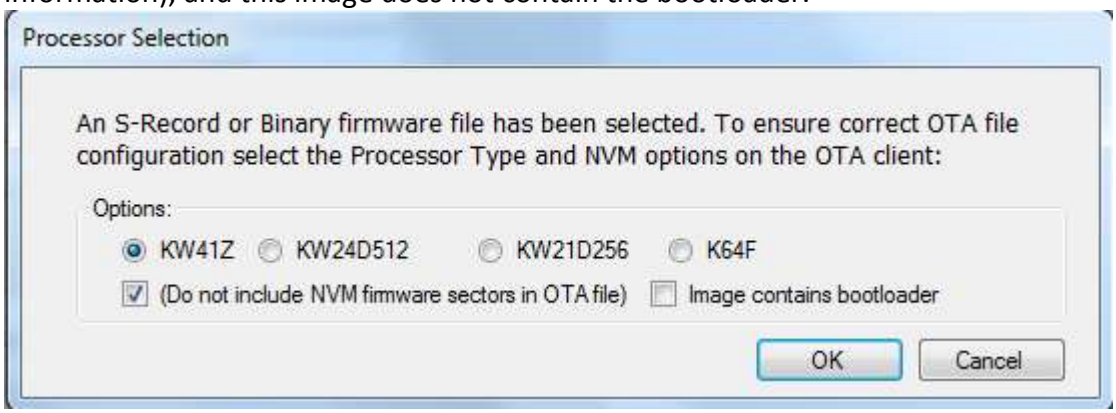
5.4 Run the Over The Air Update

24. Open the OTAP Thread window by going to the OTA Updates menu option and select OTAP Thread

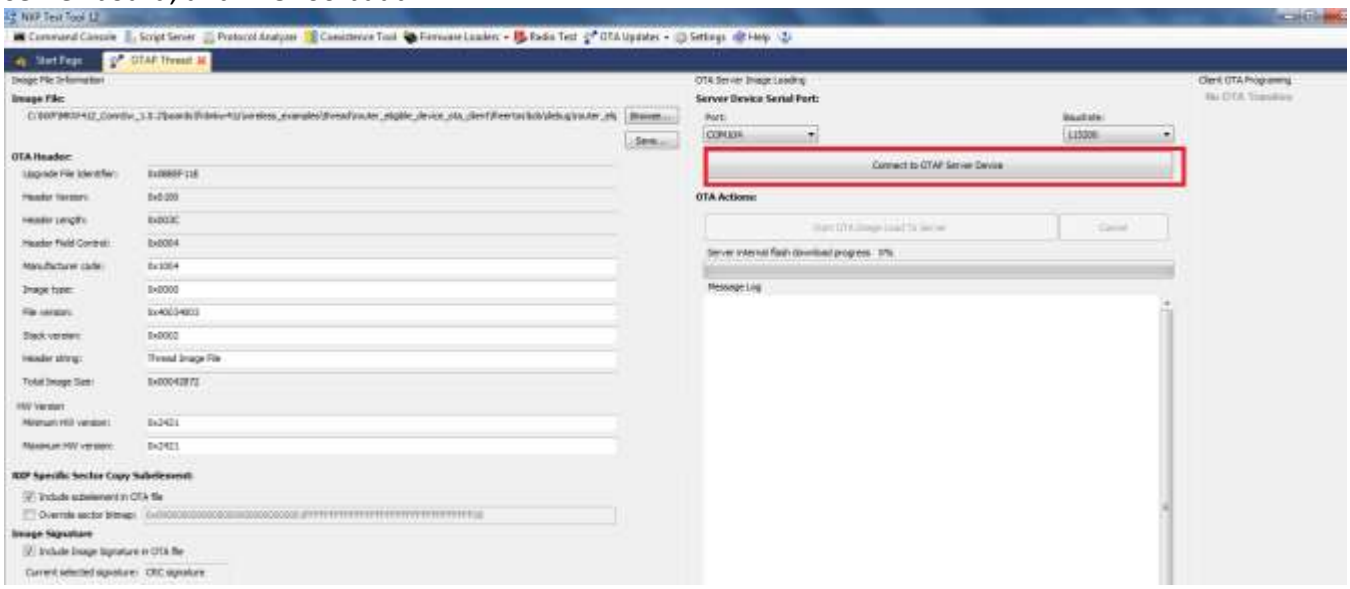
25. Select the firmware to update the client nodes with by going to Browse and selecting the .bin that was created in the last section, which would be found inside the workspace you selected

when opening up MCUXpresso IDE, and then the reed_ota_client project:
C:\kw_labs\frdmkw41z_wireless_examples_thread_reed_ota_client_freertos\Debug

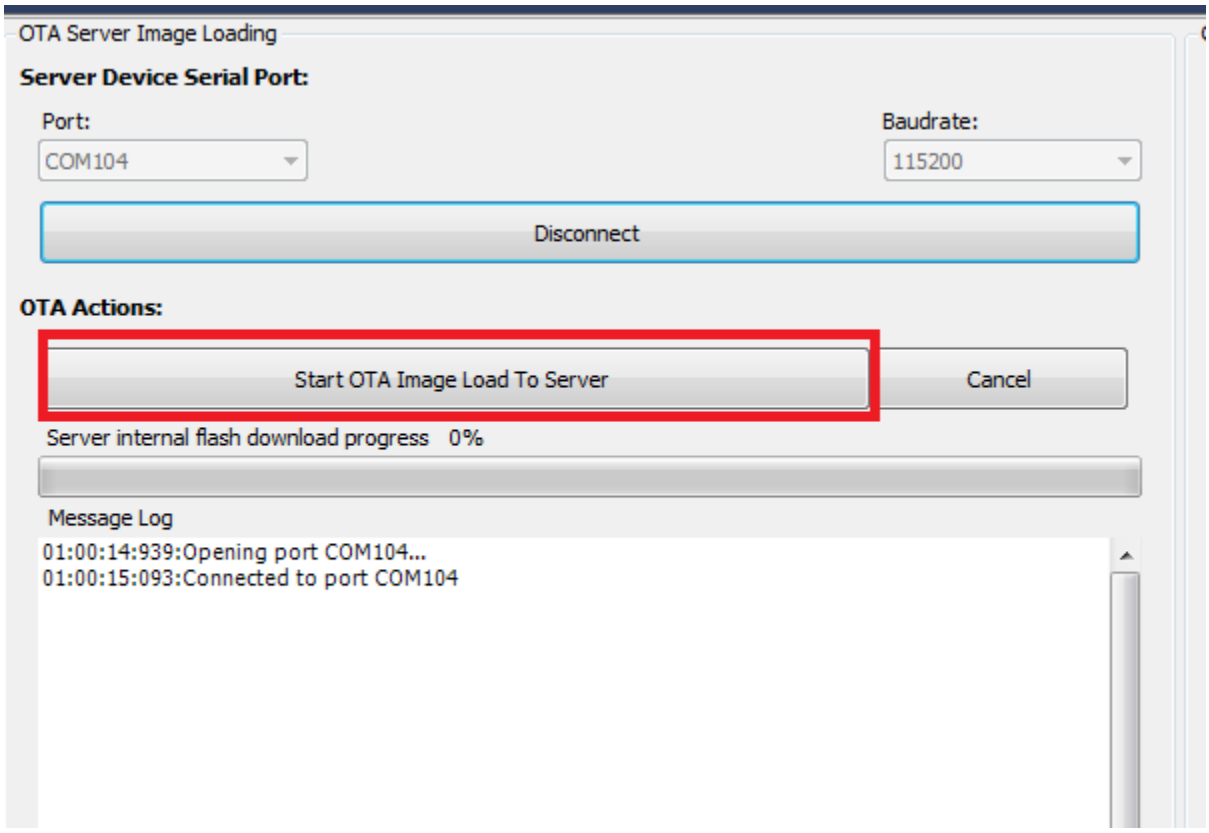
26. Select the **router_eligible_device_ota_client.bin** file
27. You will see the following dialog box come up. Leave the options set to default, which will select the KW41Z device, not overwrite the NVM settings (in other words, will keep all your network information), and this image does not contain the bootloader.



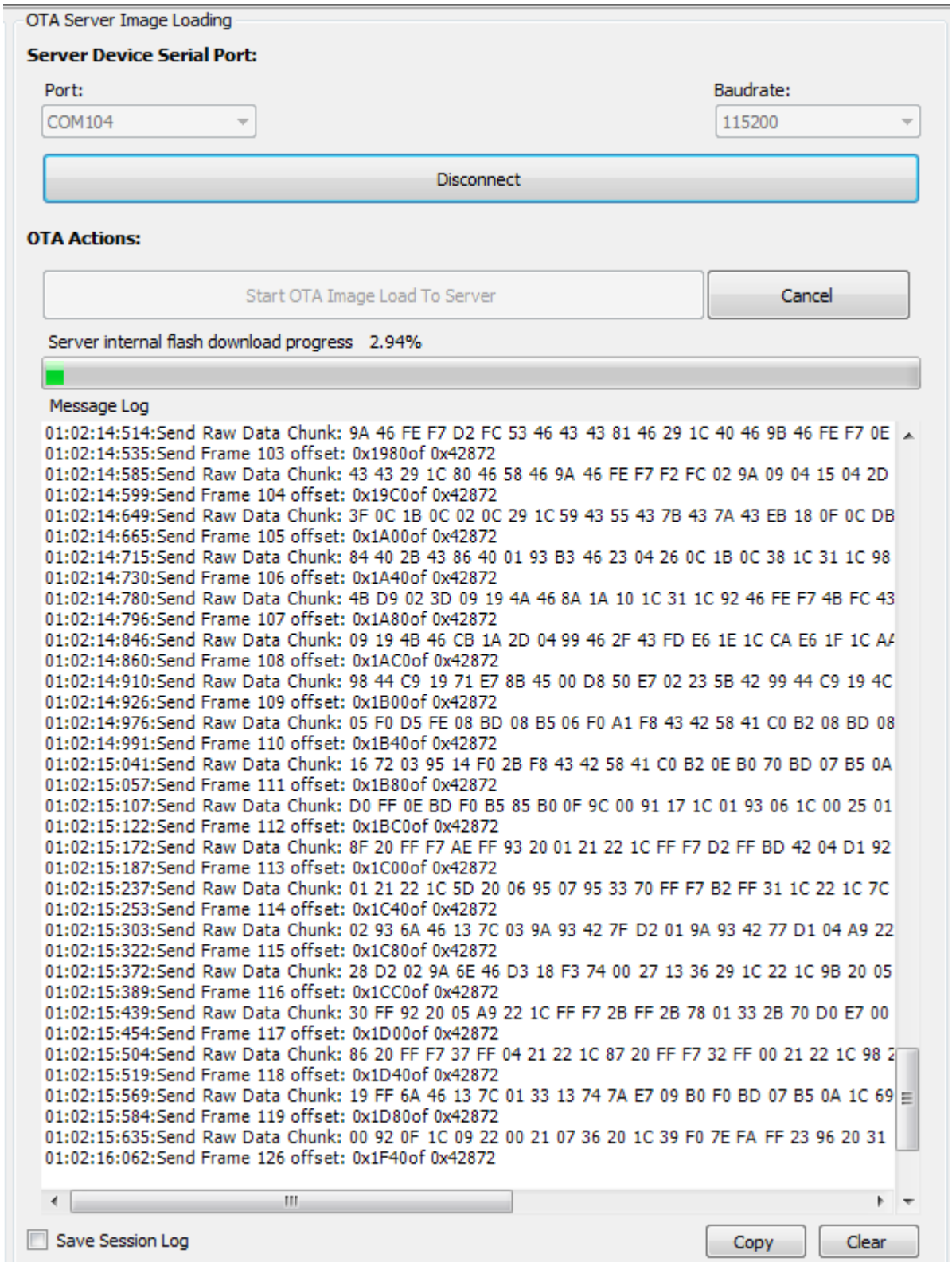
28. Then you'll want to click on "Connect to OTAP Server Device" using the COM port of the OTA server board, and 115200 baud.



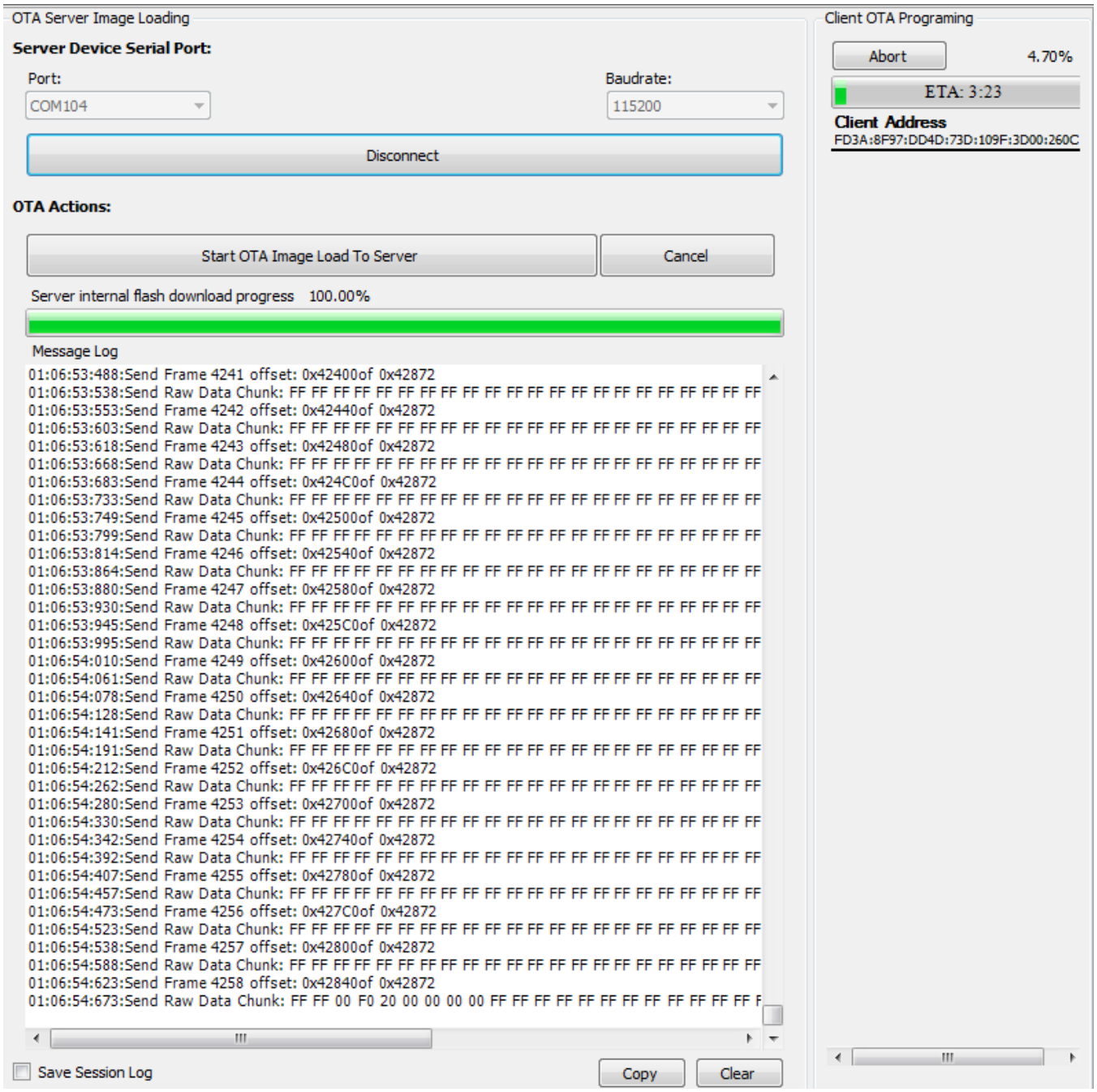
29. Then click on "Start OTA Image Load to Server" to begin the OTA update process.



30. You'll see the progress log of the firmware image being sent via FSCI to the OTA Server board:



31. Then you'll see the OTA progress as the firmware is sent over the air to the client node(s) in the right side window. A red LED should also begin flashing on the client board to signal it is receiving the update. The entire OTA update process will take approximately 10 minutes to complete.



32. When the Test Tool says the OTA has completed, the client board will reset and you should see the new updated message pop up. The OTA has completed successfully!