Over-The-Air (OTA) updates on i.MX RT1176 using NXP’s Secure Bootloader (SBL) and Secure Firmware (SFW)

Embedded Artists AB
Rundelsgatan 14
SE-211 36 Malmö
Sweden

http://www.EmbeddedArtists.com

Disclaimer

Embedded Artists AB makes no representation or warranties with respect to the contents hereof and specifically disclaim any implied warranties or merchantability or fitness for any particular purpose. Information in this publication is subject to change without notice and does not represent a commitment on the part of Embedded Artists AB.

Feedback

We appreciate any feedback you may have for improvements on this document. Send your comments by using the contact form: www.embeddedartists.com/contact.

Trademarks

All brand and product names mentioned herein are trademarks, services marks, registered trademarks, or registered service marks of their respective owners and should be treated as such.

Table of Contents

1 Document Revision History

Revision	*Date*	*Description*
A	2021-11-17	First release

2 Introduction

Over-the-air (OTA) updates, also sometimes known as remote updates, is a solution for remotely updating the software on a device. Remotely meaning that you don't need physical access to the device when updating its software.

This document focuses on NXP's software framework which they call *MCU SBL and SFW*. SBL stands for **S**ecure **B**oot**l**oader and SFW stands for **S**ecure **F**irm**w**are.

The main description and instructions for how to use the framework is available in NXP's document *MCU-OTA SBL and SFW User Guide.* It is recommended that you read through NXP's document to get a better understanding of their software before following the instructions in this document.

This specific document is providing instructions for how to use the software on Embedded Artists iMX RT1176 Developer's Kit and should be seen as a complement to NXP's document.

> **Note**: NXP's documentation describes remote OTA via both Amazon **AWS IoT** and Alibaba Aliyun, but we have only verified functionality on AWS IoT.

2.1 Additional Documentation

Additional recommended documentation:

- *iMX RT Developer's Kit Program Development Guide* – found on Embedded Artists website.
- *MCU-OTA SBL and SFW User Guide* – on NXP's website https://www.nxp.com/docs/en/user-guide/MCUOTASBLSFWUG.pdf

2.2 Conventions

A number of conventions have been used throughout to help the reader better understand the content of the document.

`Constant width text` – is used for file system paths and command, utility and tool names.

```
$ This field illustrates user input in a terminal running on the
development workstation, i.e., on the workstation where you edit,
configure and build Linux
```

```
# This field illustrates user input on the target hardware, i.e.,
input given to the terminal attached to the COM Board
```

```
This field is used to illustrate example code or excerpt from a
document.
```

> This field is used to highlight important information

3 Build the software

3.1 Secure Bootloader (SBL)

This section contains instructions for building SBL on a Windows host using GNU Arm Embedded Toolchain (GCC_ARM). NXP's documentation (chapter 2 in Rev 1.1.0) also describes how to use IAR and Keil MDK toolchain as well as building on a Linux host.

> **Note**: We will use the target evkmimxrt1170 in these instructions. For SBL, this target doesn't have to be modified to work with the Embedded Artists iMX RT1176 Developer's Kit.

3.1.1 Install Arm GCC toolchain

Go to the download section on the Arm Developer website and download a toolchain for your host.

https://developer.arm.com/tools-and-software/open-source-software/developer-tools/gnu-toolchain/gnu-rm

3.1.2 Get the source code

You must have git tools installed on your computer to follow these instructions. You can for example use https://gitforwindows.org/.

In this example we do the development in the directory mcu-ota-sbl.

```
$ cd <your path>/mcu-ota-sbl
$ git clone https://github.com/NXPmicro/sbl.git
```

3.1.3 Select toolchain

The path to the toolchain must be configured in sblprofile.py.

1. Open sbl\target\evkmimxrt1170\sblprofile.py
2. Locate the variable EXEC_PATH within the if CROSS_TOOL == 'gcc' statement

```
if  CROSS_TOOL == 'gcc':
    PLATFORM    = 'gcc'
    EXEC_PATH   =
```

3. Set EXEC_PATH to the path of the toolchain you installed in section 3.1.1 above. In this example it was set to:

```
r'E:\Program Files\GNU Arm Embedded Toolchain\10_2021.10\bin'
```

3.1.4 Build the project

The project is built with the help of SCons which is an open-source build system written in Python. This build system is included when getting the source code and is setup using the env.bat file.

1. Run sbl\target\evkmimxrt1170\env.bat and you will get a command prompt initialized to use SCons.
2. Within the command prompt run scons to build the project

```
> scons
```

3. If successfully built the sbl.bin file will be available in the sbl\target\evkmimxrt1170\build directory.

4. Follow the instructions in section 4.2 to deploy the image to the iMX RT1176 kit.

3.2 Secure Firmware (SFW)

This section contains instructions for building SFW on a Windows host using GNU Arm Embedded Toolchain (GCC_ARM). NXP's documentation (chapter 2 in Rev 1.1.0) also describes how to use IAR and Keil MDK toolchain as well as building on a Linux host.

> **Note 1**: We will use the target `evkmimxrt1170` in these instructions, but it must be patched to work with the Embedded Artists iMX RT1176 Developer's Kit since a different Ethernet PHY is used.
>
> **Note 2**: See section 4.1 if you want to use a pre-built image of SBL.

3.2.1 Install Arm GCC toolchain

Go to the download section on the Arm Developer website and download a toolchain for your host.

https://developer.arm.com/tools-and-software/open-source-software/developer-tools/gnu-toolchain/gnu-rm

> **Note**: If you followed the instructions in section 3.1.1 you already have the toolchain installed.

3.2.2 Get the source code

You must have git tools installed on your computer to follow these instructions. You can for example use https://gitforwindows.org/.

The instructions have been tested on version v1.1.0. The exact commit being tested was d7e76dbe1bf481bd7d4a7fbb167005bbfec8db6e.

In this example we do the development in the directory `mcu-ota-sbl`.

```
$ cd <your path>/mcu-ota-sbl
$ git clone https://github.com/NXPmicro/sfw.git
$ cd sfw
$ git checkout tags/v1.1.0
```

3.2.3 Patch the source code

The Embedded Artists iMX RT1176 uCOM board use a different Ethernet Phy than the NXP RT1170 EVK. Follow these instructions to apply a patch that changes the Ethernet Phy.

1. First go to your working directory for the project

```
$ cd <your path>/mcu-ota-sbl
```

2. Download the patch from http://imx.embeddedartists.com/imxrt/sfw_ea_rt1176_phy_ar8031.patch

3. Go to the `sfw` directory and apply the patch using git tools.

```
$ cd sfw
$ git apply ../sfw_ea_rt1176_phy_ar8031.patch
```

3.2.4 Setup AWS

Before continuing building SFW you must setup AWS as described in the AWS OTA section in NXP's documentation. In Rev 1.1.0 of the document the subsection was called 7.3.1.1 - AWS OTA Prerequisites. This section will create certificates, IAM user, permissions, and an IoT Thing.

When you have configured AWS you should follow the instructions in subsection 7.3.1.3 – Prepare the SFW. In that subsection you will include the certificate and key associated with your AWS IoT Thing into the SFW build. You will also configure SFW to enable AWS and to set the MQTT broker DNS name as well as the AWS IoT Thing name.

3.2.5 Build the project

The project is built with the help of SCons which is an open-source build system written in Python. This build system is included when getting the source code and is setup using the `env.bat` file.

1. Run `sfw\target\evkmimxrt1170\env.bat` and you will get a command prompt initialized to use SCons.
2. Within the command prompt run `scons` to build the project

```
> scons
```

3. If successfully built the `sfw.bin` file will be available in the `sfw\target\evkmimxrt1170\build` directory.
4. The image must be signed before deployed to target. Copy `sfw.bin` to `sbl\component\secure\mcuboot\scripts` and rename to `sfw_092.bin`. The appendix `092` indicates the version of the firmware. This version is specified in `sfw\firmware\aws_ota\main_enet.c` (`APP_VERSION_MAJOR`, `APP_VERSION_MINOR`, and `APP_VERSION_BUILD`).
5. Within the SCons command prompt sign the image. The signed image will have the file name `sfw092.bin` in this example.

```
> python imgtool.py sign --key sign-rsa2048-priv.pem --align 4 --
version "0.9.2" --header-size 0x400 --pad-header --slot-size
0x100000 --max-sectors 32 sfw_092.bin sfw092.bin
```

6. Follow the instructions in section 4.3 to deploy the `sfw092.bin` image to the iMX RT1176 kit. This first version of the firmware must be deployed locally. Once deployed new versions can be deployed using AWS IoT.

4 Deploy the software

4.1 Use pre-built software

Before deploying software to the target, it must be built. Instructions for building is available in chapter 3 above, but since **SBL** doesn't have to be configured to be used with SFW and/or AWS IoT we have provided a pre-built image. You can download it from the link below.

http://imx.embeddedartists.com/imxrt/ea_rt1176_ota_sbl.zip

The **SFW** software must however be configured to be used for your specific AWS account so a pre-built image cannot be provided.

4.2 SBL

We will use MCUXpresso to program the flash with the SBL image. You should have followed the instructions in the document *iMX RT Developer's Kit Program Development Guide* to install MCUXpresso as well as the SDK for Embedded Artists iMX RT1176 Developer's Kit.

4.2.1 Import hello world demo

The first step is to import an existing code example that has been setup for the iMX RT1176 Developer's Kit.

Click on Import SDK example in the QuickStart Panel as shown in Figure 1.

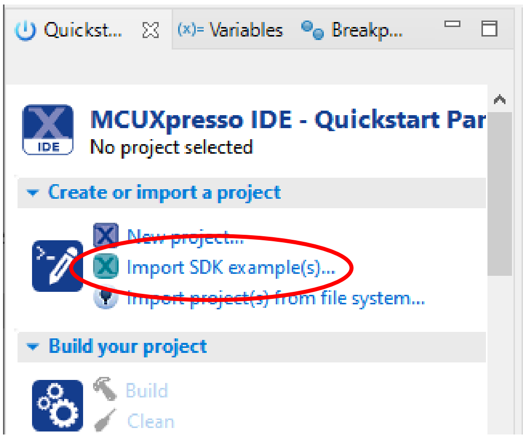
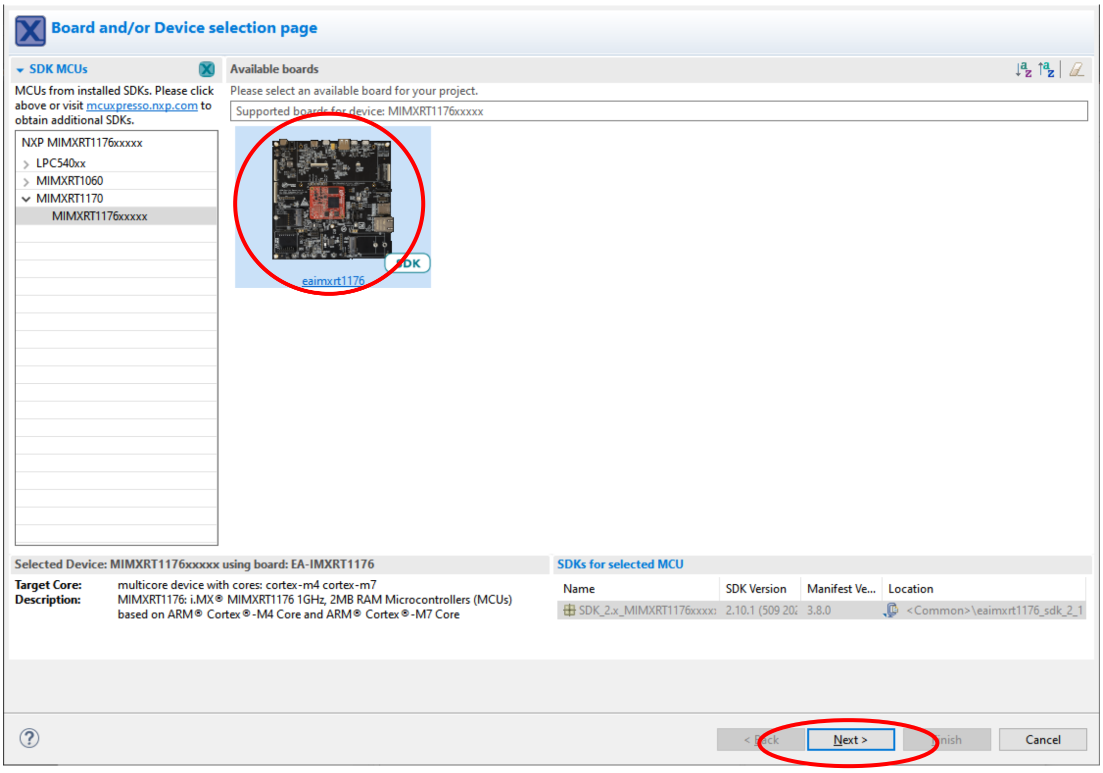


Figure 1 - Import SDK example

Select the eaimxrt1176 SDK from the Import Wizard and then click the Next button as shown in Figure 2 below.



Board and/or Device selection page

MCUs from installed SDKs. Please click above or visit mcuxpresso.nxp.com to obtain additional SDKs.

- NXP MIMXRT1176xxxxx
  - LPC540xx
  - MIMXRT1060
  - MIMXRT1170
    - MIMXRT1176xxxxx

Please select an available board for your project.

Supported boards for device: MIMXRT1176xxxxx

Selected Device: MIMXRT1176xxxxx using board: EA-IMXRT1176

Target Core: multicore device with cores: cortex-m4 cortex-m7

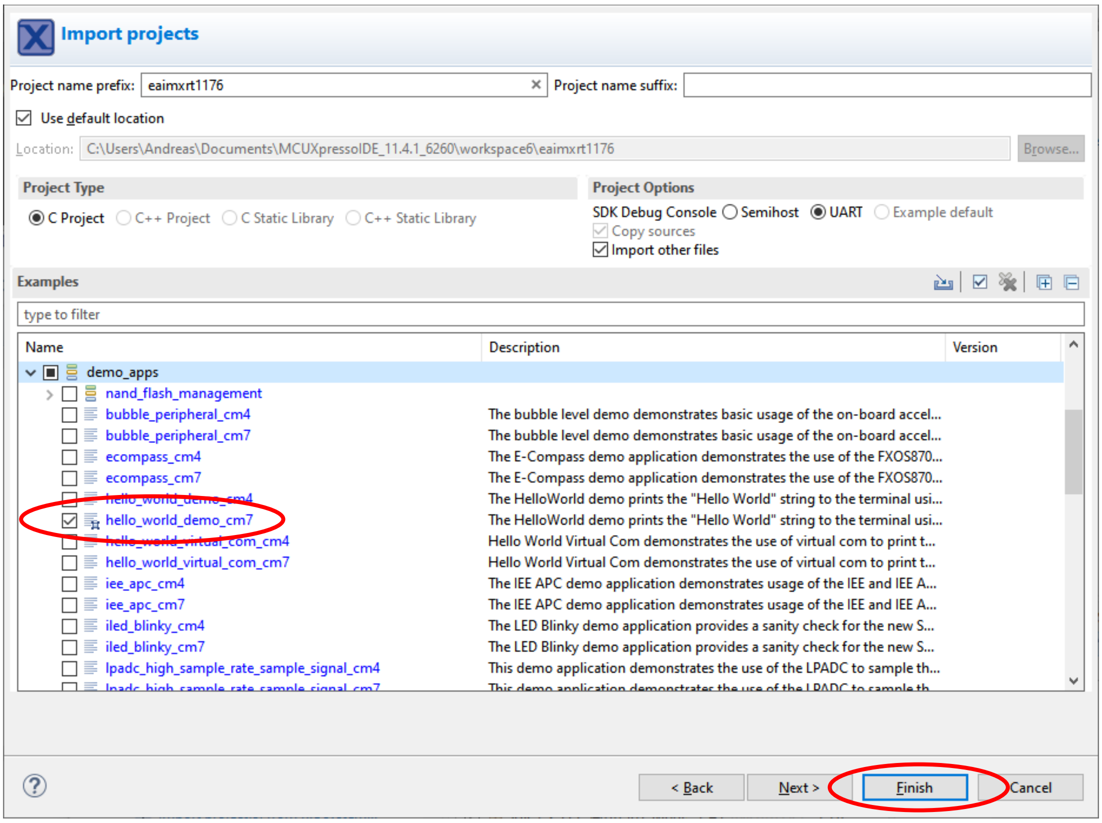
Description: MIMXRT1176: i.MX® MIMXRT1176 1GHz, 2MB RAM Microcontrollers (MCUs) based on ARM® Cortex®-M4 Core and ARM® Cortex®-M7 Core

SDKs for selected MCU

Name	SDK Version	Manifest Ve...	Location
SDK_2.x_MIMXRT1176xxxx:	2.10.1 (509 202	3.8.0	<Common>\eaimxrt1176_sdk_2_1

Figure 2 - The eaimxrt1176 SDK

Go to demo_apps and select hello_world_demo_cm7 as shown in Figure 3 and then click Finish.



Import projects

Project name prefix: eaimxrt1176

Project name suffix:

☑ Use default location

Location: C:\Users\Andreas\Documents\MCUXpressoIDE_11.4.1_6260\workspace6\eaimxrt1176

Project Type

● C Project ○ C++ Project ○ C Static Library ○ C++ Static Library

Project Options

SDK Debug Console ○ Semihost ● UART ○ Example default

☑ Copy sources

☑ Import other files

Examples

type to filter

Name	Description	Version
■ demo_apps		
□ nand_flash_management		
□ bubble_peripheral_cm4	The bubble level demo demonstrates basic usage of the on-board accel...	
□ bubble_peripheral_cm7	The bubble level demo demonstrates basic usage of the on-board accel...	
□ ecompass_cm4	The E-Compass demo application demonstrates the use of the FXOS870...	
□ ecompass_cm7	The E-Compass demo application demonstrates the use of the FXOS870...	
□ hello_world_demo_cm4	The HelloWorld demo prints the "Hello World" string to the terminal usi...	
☑ hello_world_demo_cm7	The HelloWorld demo prints the "Hello World" string to the terminal usi...	
□ hello_world_virtual_com_cm4	Hello World Virtual Com demonstrates the use of virtual com to print t...	
□ hello_world_virtual_com_cm7	Hello World Virtual Com demonstrates the use of virtual com to print t...	
□ iee_apc_cm4	The IEE APC demo application demonstrates usage of the IEE and IEE A...	
□ iee_apc_cm7	The IEE APC demo application demonstrates usage of the IEE and IEE A...	
□ iled_blinky_cm4	The LED Blinky demo application provides a sanity check for the new S...	
□ iled_blinky_cm7	The LED Blinky demo application provides a sanity check for the new S...	
□ lpadc_high_sample_rate_sample_signal_cm4	This demo application demonstrates the use of the LPADC to sample th...	
□ lpadc_high_sample_rate_sample_signal_cm7	This demo application demonstrates the use of the LPADC to sample th...	

Figure 3 - Hello World Demo

4.2.2 GUI Flash Tool

Once the hello world application is imported and built you can use the GUI Flash tool to program the board with SBL.

Click on the GUI Flash Tool icon in the toolbar as shown in Figure 4.

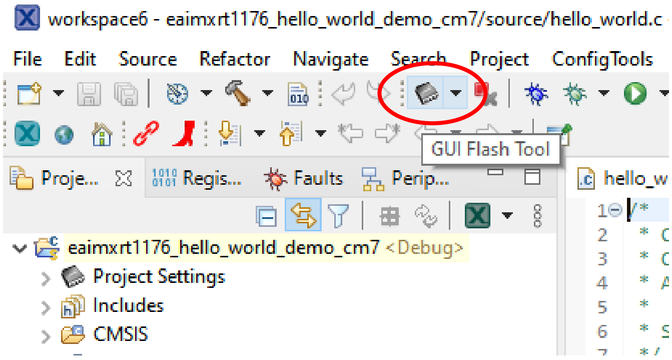

workspace6 - eaimxrt1176_hello_world_demo_cm7/source/hello_world.c
File Edit Source Refactor Navigate Search Project ConfigTools
GUI Flash Tool
Proje... Regis... Faults Perip...
hello_w
eaimxrt1176_hello_world_demo_cm7 <Debug>
Project Settings
Includes
CMSIS

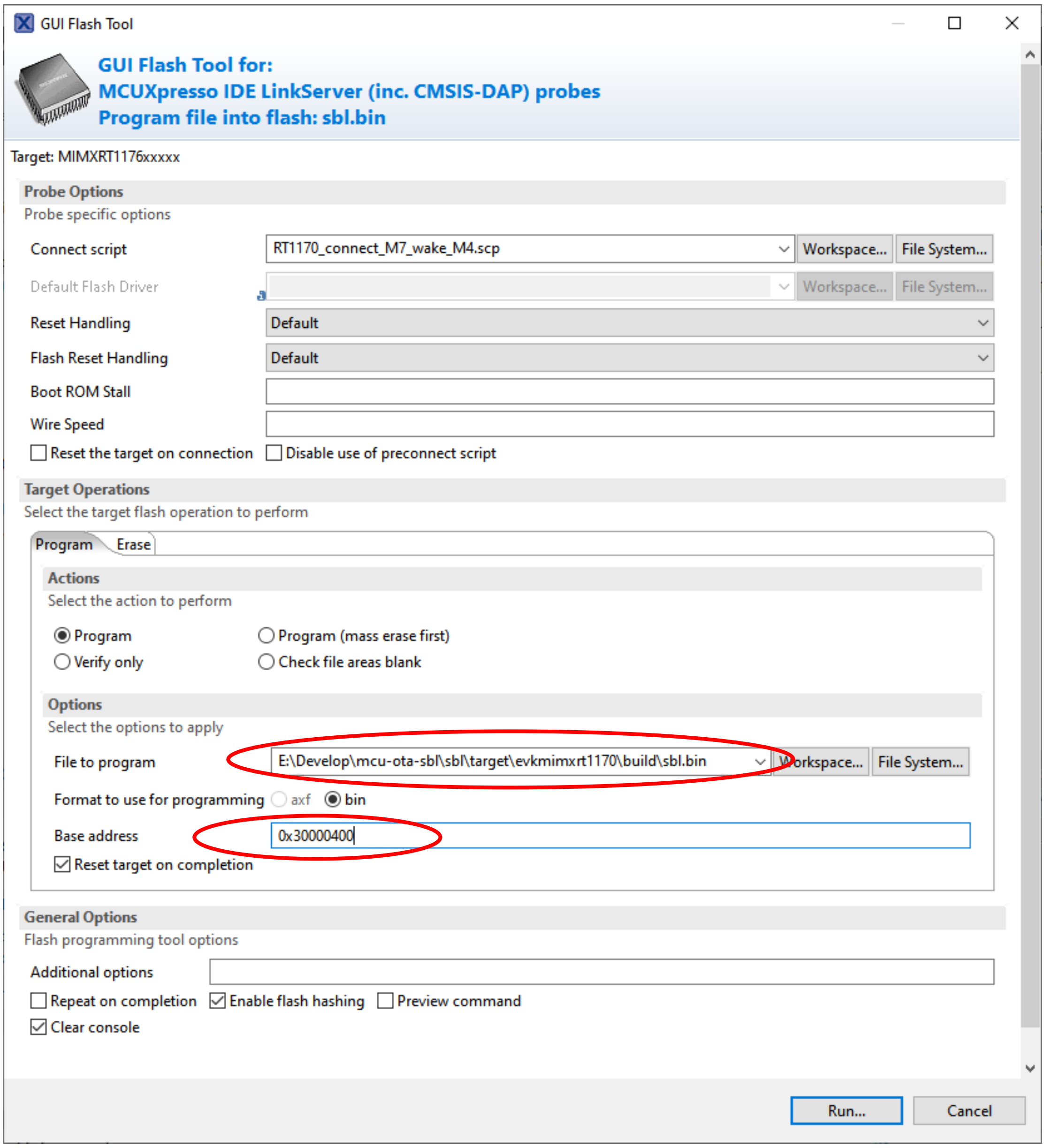
Figure 4 - GUI Flash Tool icon

This will open up the GUI Flash tool dialog window as shown in Figure 5. In the Target Operations → Program section click on the File System button and then navigate to the built `sbl.bin` file (use either the pre-built as described in section 4.1 or the one you built by following instructions in section 3.1 above).

Change the base address to **0x30000400** and then click the Run button to program the board.

Reset the board and SBL will start. In the console you will see the output below (after a few seconds).

```
hello sbl.
Bootloader Version 1.1.0
Remap type: none

Unable to find bootable image
```



GUI Flash Tool

GUI Flash Tool for:
MCUXpresso IDE LinkServer (inc. CMSIS-DAP) probes
Program file into flash: sbl.bin

Target: MIMXRT1176xxxxx

Probe Options
Probe specific options

Connect script: RT1170_connect_M7_wake_M4.scp

Default Flash Driver:

Reset Handling: Default

Flash Reset Handling: Default

Boot ROM Stall:

Wire Speed:

□ Reset the target on connection □ Disable use of preconnect script

Target Operations
Select the target flash operation to perform

Actions
Select the action to perform

● Program ○ Program (mass erase first)
○ Verify only ○ Check file areas blank

Options
Select the options to apply

File to program: E:\Develop\mcu-ota-sbl\sbl\target\evkmimxrt1170\build\sbl.bin

Format to use for programming ○ axf ● bin

Base address: 0x30000400

☑ Reset target on completion

General Options
Flash programming tool options

Additional options:

□ Repeat on completion ☑ Enable flash hashing □ Preview command
☑ Clear console

Figure 5 - GUI Flash tool

4.3 SFW

We will use the NXP MCU Boot Utility to download the SFW image to the board.

1. Download and install MCU Boot Utility from https://github.com/JayHeng/NXP-MCUBootUtility
2. Start the application and make sure i.MXRT117x is selected as MCU Device and FLEXSPI NOR as Boot Device. See top left part of Figure 6.
3. Chose UART as interface towards the board and select the COM port connected to the board (in this example COM4 is used with baudrate 115200). Click the Connect button. See bottom left part of Figure 6.
4. We are going to program the image to slot 1 which is available at an offset of 0x100000 from the start of the flash address. In this case that is 0x30100000. Set this address in the Start / Offset input box. Click the Browse button and select the `sfw092.bin` file created in section 3.2.5 above.
5. Click the Write (Auto Erase) button to program the image

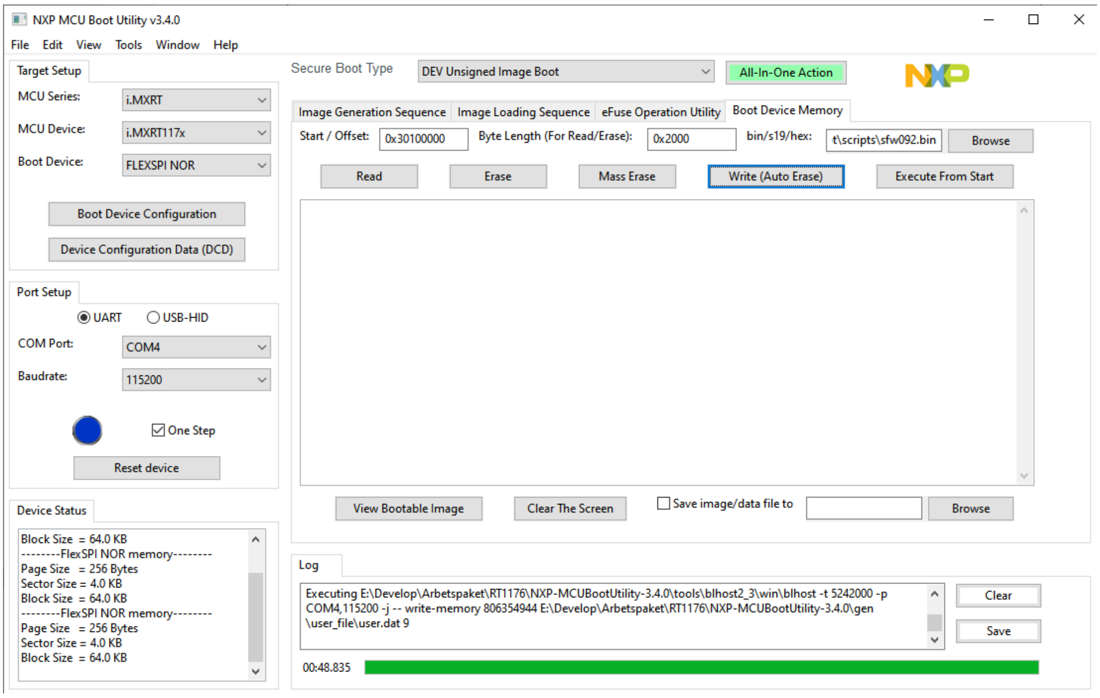
6. Once programmed reset the board and you will see output in the console similar to below (only first part of output is shown).

```
hello sbl.
Bootloader Version 1.1.0
Remap type: none

The image now in PRIMARY_SLOT slot

Bootloader chainload address offset: 0x100000
Reset_Handler address offset: 0x100400
Jumping to the image

hello sfw!
Current image verison: 0.9.2
```

7. When the application has started, initialized the network interface and established a connection to AWS IoT you will see the output below in the console.

```
Hello world1.
Hello world2.
54 25488 [iot_thread] State: WaitingForJob  Received: 1   Queued:
0   Processed: 0   Dropped: 0
```

8. You can now continue with the instructions in the AWS OTA section of NXP’s documentation. More specifically you should create `sfw093.bin` (a new version of SFW) as described in section 7.3.1.4, upload the file as described in section 7.3.1.5 – Upload new image to S3 bucket, and then create an OTA job as described in section 7.3.1.6 – Create OTA Job.



MCU Series: i.MXRT

MCU Device: i.MXRT117x

Boot Device: FLEXSPI NOR

● UART ○ USB-HID

COM Port: COM4

Baudrate: 115200

☑ One Step

```
Block Size  = 64.0 KB
--------FlexSPI NOR memory--------
Page Size   = 256 Bytes
Sector Size = 4.0 KB
Block Size  = 64.0 KB
--------FlexSPI NOR memory--------
Page Size   = 256 Bytes
Sector Size = 4.0 KB
Block Size  = 64.0 KB
```

Secure Boot Type: DEV Unsigned Image Boot

Start / Offset: 0x30100000 Byte Length (For Read/Erase): 0x2000 bin/s19/hex: t\scripts\sfw092.bin

□ Save image/data file to

```
Executing E:\Develop\Arbetspaket\RT1176\NXP-MCUBootUtility-3.4.0\tools\blhost2_3\win\blhost -t 5242000 -p
COM4,115200 -j -- write-memory 806354944 E:\Develop\Arbetspaket\RT1176\NXP-MCUBootUtility-3.4.0\gen
\user_file\user.dat 9
```

00:48.835

Figure 6 - MCU Boot Utility