

# How to Enable Boot from QSPI Flash

## 1. Introduction

The i.MX RT Series is industry's first crossover processor provided by NXP. This document describes how to program a bootable image into the external storage device. For Information about Flashloader, MfgTool, refer to the application note [“How to Enable Boot from Octal SPIFlash and SD Card”AN12107](#).

The software used for example in this document are based on the i.MXRT1050 SDK 2.4.0. The development environment is IAR Embedded Workbench 8.22.2. The hardware development environment is IMXRT1050-EVKB Board. The version of Flashloader is V1.1.

## Contents

1.	Introduction.....	1
2.	MIMXRT1050 EVK board settings.....	2
2.1.	EVKA Settings .....	2
2.2.	EVKB Settings.....	3
3.	Program tools.....	5
3.1.	DAP-Link (OpenSDA MSD drag/drop) .....	5
3.2.	MFG tool .....	5
4.	Examples.....	5
4.1.	OpenSDA Drag/Drop and boot from QSPI Flash .....	5
4.2.	MFG Boot from QSPI Flash .....	10
4.3.	MFG Boot from QSPI Flash with DCD for SDRAM .....	21
5.	QSPI Flash support list .....	21
6.	Conclusion .....	22
7.	Revision history .....	23



## 2. MIMXRT1050 EVK board settings

### 2.1. EVKA Settings

In order to enable the onboard QSPI Flash features, EVK board (EVKA Board) settings need to be changed.

#### Step 1:

The onboard Hyper Flash should be removed, otherwise it will impact the QSPI Flash read and write timing.

#### Step 2:

- Weld 0  $\Omega$  resistor to the pad from R153 to R158.

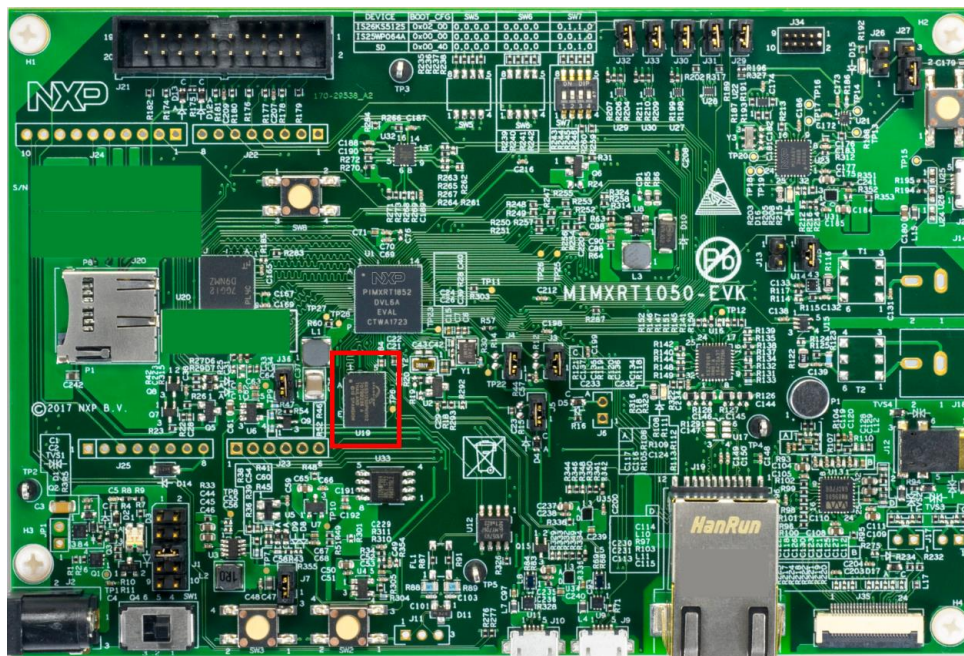


Figure 1. Hyper Flash

## 1V8 QSPI Flash

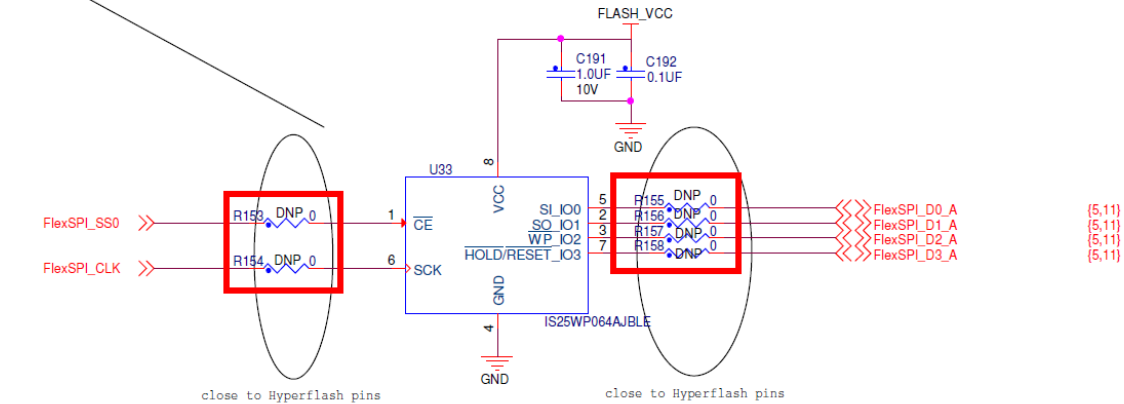


Figure 2. Weld 0  $\Omega$  resistor to the pad from R153 to R158

### Step 3:

- The firmware of OpenSDA needs to be replaced. The default firmware onboard is used to Hyper Flash, so that the firmware should be replaced to QSPI Flash. Both Hyper Flash and QSPI Flash's firmware can be downloaded from [NXP Website](#).

## 2.2. EVKB Settings

For EVKB board, the onboard Hyper Flash does not need to remove.

Removed resistors: R356, R361 - R366.

Weld 0  $\Omega$  resistors: R153 - R158.

Follow the Step3 of Section 2.1 to update the OpenSDA firmware.

After those steps, the onboard QSPI Flash is ready to use.

### NOTE

Even if QSPI flash itself doesn't have DQS pin, keep it to be floating and enable it to get a higher read/write frequency. Please refer to Table 35 and Table 36 in the [RT1050 datasheet](#). If DQS pin is not used, only 60 MHz frequency of operation is supported while could up to 133 MHz frequency of operation if DQS pin enabled for input timing.

**SDR mode with FlexSPIn\_MCR0[RXCLKSRC] = 0x0, 0x1**

**Table 35. FlexSPI input timing in SDR mode where FlexSPIn\_MCR0[RXCLKSRC]= 0X0**

Symbol	Parameter	Min	Max	Unit
	Frequency of operation	—	60	MHz
T <sub>IS</sub>	Setup time for incoming data	8.67	—	ns
T <sub>IH</sub>	Hold time for incoming data	0	—	ns

**Table 36. FlexSPI input timing in SDR mode where FlexSPIn\_MCR0[RXCLKSRC]= 0X1**

Symbol	Parameter	Min	Max	Unit
	Frequency of operation	—	133	MHz
T <sub>IS</sub>	Setup time for incoming data	2	—	ns
T <sub>IH</sub>	Hold time for incoming data	1	—	ns

**Figure 3. SDR mode input timing parameter**

**2.2.1. Macros for the boot header**

The [Table 1](#) shows three macros that are added in flexspi\_nor targets to support XIP:

**Table 1. Macros for the boot header**

<b>XIP_EXTERNAL_FLASH</b>	1: Exclude the code which will change the clock of flexspi. 0: make no changes.
<b>XIP_BOOT_HEADER_ENABLE</b>	1: Add flexspi configuration block, image vector table, boot data and device configuration data(optional) to the image by default. 0: Add nothing to the image by default.
<b>XIP_BOOT_HEADER_DCD_ENABLE</b>	1: Add device configuration data to the image. 0: Do <b>NOT</b> add device configuration data to the image.

The [Table 2](#) shows the different effect on the built image with different combination of these macros:

**Table 2. Different effect on the built image with difference macros**

		<b>XIP_BOOT_HEADER_DCD_ENABLE=1</b>	<b>XIP_BOOT_HEADER_DCD_ENABLE=0</b>
<b>XIP_EXTERNAL_FLASH=1</b>	<b>XIP_BOOT_HEADER_ENABLE=1</b>	Can be programmed to Hyper Flash by IDE and can run after POR reset if Hyper Flash is the boot source. SDRAM will be initialized.	Can be programmed to Hyper Flash by IDE and can run after POR reset if Hyper Flash is the boot source. SDRAM will <b>NOT</b> be initialized.
	<b>XIP_BOOT_HEADER_ENABLE=0</b>		

	<b>XIP_BOOT_HEADER_ENABLE=0</b>	Can <b>NOT</b> run after POR reset if it is programmed by IDE even if Hyper Flash is the boot source.
	<b>XIP_EXTERNAL_FLASH=0</b>	This image can <b>NOT</b> do XIP because when this macro is set to 1, it will exclude the code which will change the clock of flexspi.

## 3. Program tools

### 3.1. DAP-Link (OpenSDA MSD drag/drop)

- QSPI Flash on EVK only.
- Binary file supports only.

#### NOTE

The default firmware of DAP-Link on EVK supports Hyper Flash only. The firmware of DAP-Link should be replaced if the QSPI flash drag/drop is used. The firmware can be downloaded from [NXP Web](#).

### 3.2. MFG tool

The MfgTool supports I.MXRT BootROM and KBOOT based Flashloader, it can be used in factory production environment. The Mfgtool can detect the presence of BootROM devices connected to PC and invokes “blhost” to program the image on target memory devices connected to i.MX MCU device.

The blhost is a command-line host program used to interface with devices running KBOOT based Bootloader, part of MfgTool release .sb file support only.

## 4. Examples

### 4.1. OpenSDA Drag/Drop and boot from QSPI Flash

This chapter describes the steps needed that program an image to QSPI Flash by using OpenSDA Drag/Drop. The steps are as follows:

#### Step 1:

- Open the Hello world demo in the SDK and select the project configuration as flexspi\_nor\_debug.([Figure 4](#)).

## Examples

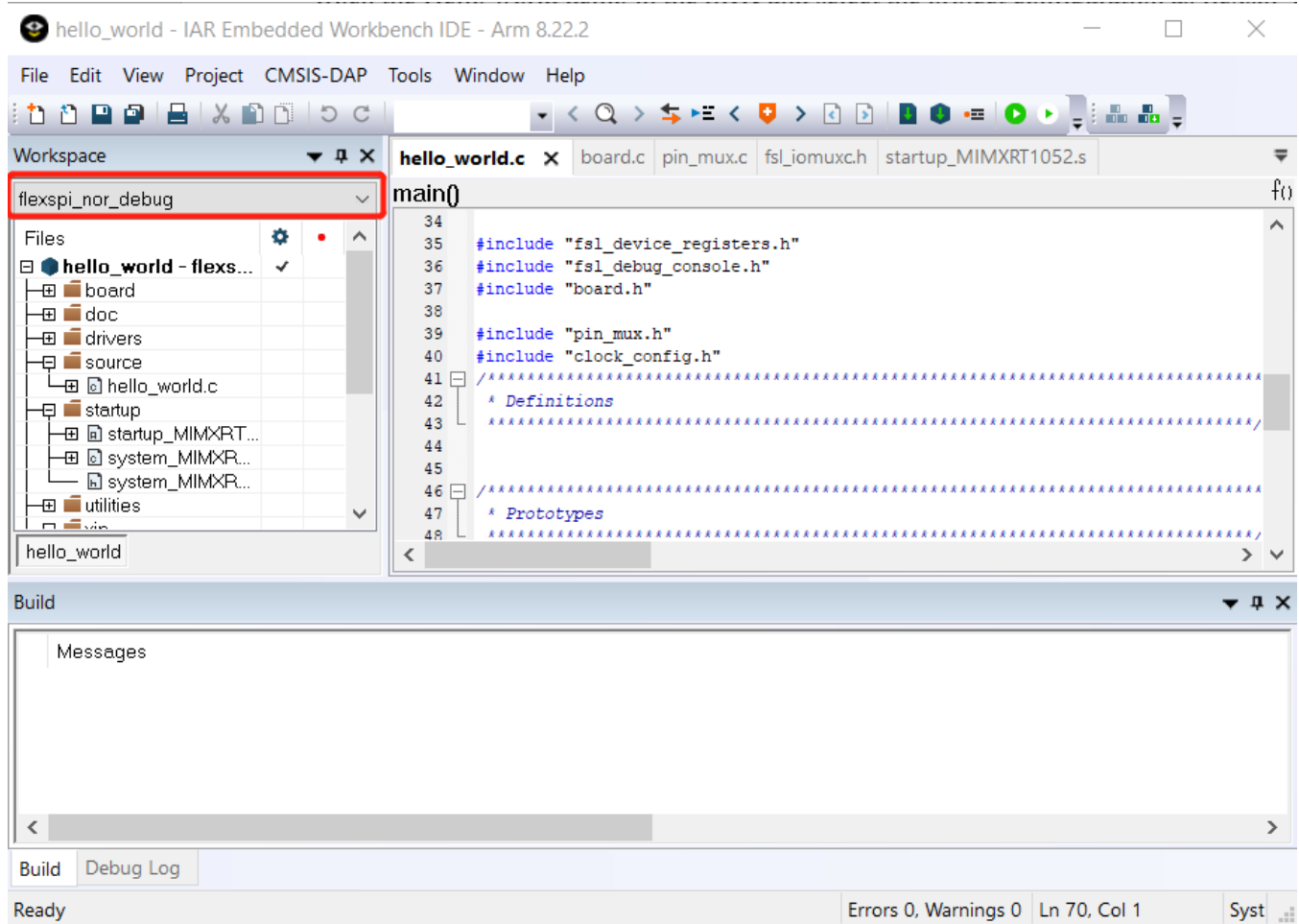


Figure 4. Select the project configuration as flexspi\_nor\_debug

### Step 2:

- Build the project and generate an image. You can find the hello\_world.bin at following location ([Figure 5](#)).

### NOTE

Before an image generate, flash configure parameters need to be changed. Please refer to [“How to Enable Debugging for FLEXSPI NOR Flash”](#), [AN12183](#)

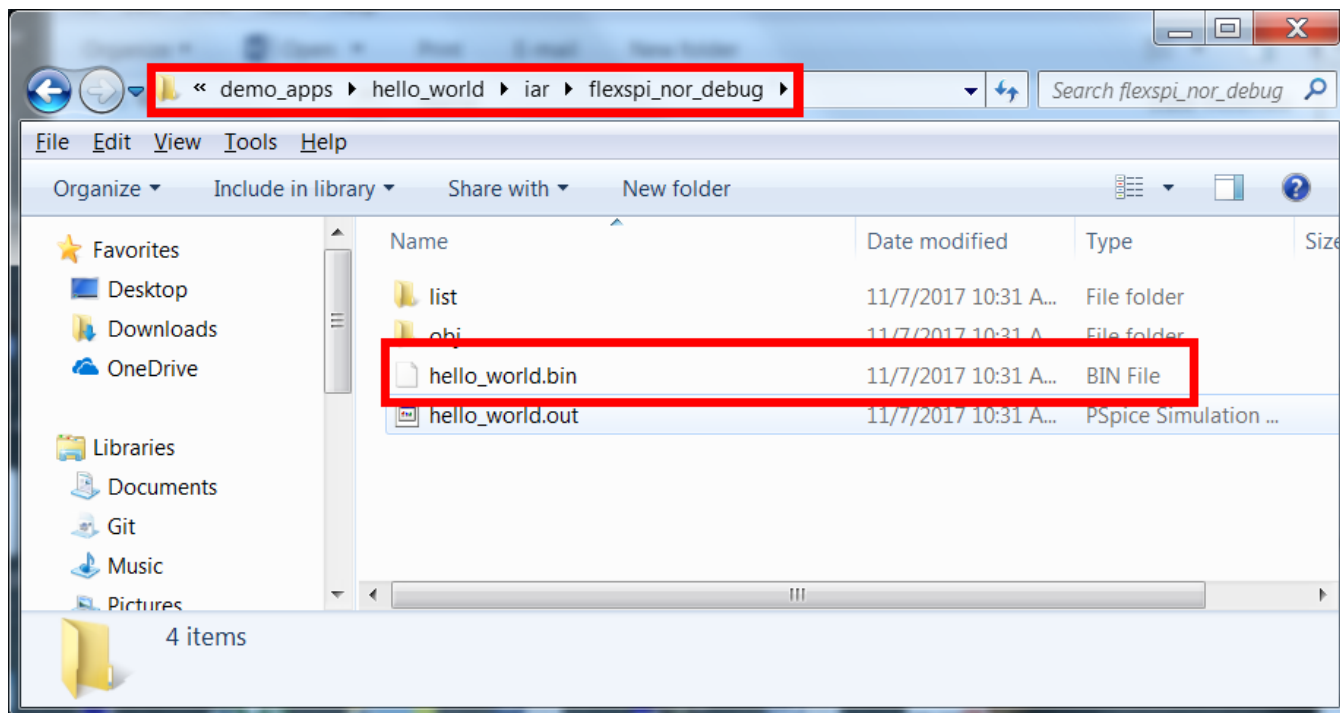


Figure 5. hello\_world.bin location

### Step 3:

- Configure the board to serial downloader mode and make sure the power supply is from the Debug USB. To achieve these, SW7-4 should pull-up others pull-down [Figure 6](#) and the J1-5, J1-6 should be connected [Figure 7](#).

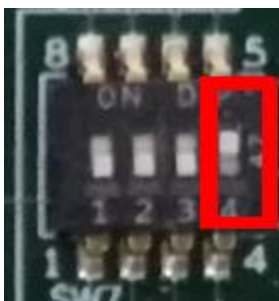


Figure 6. SW7-4 pull-up and others pull-down

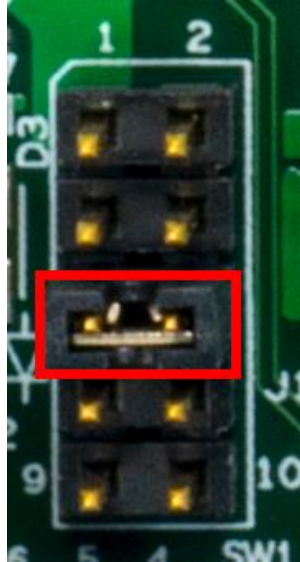


Figure 7. Power supply switch

**Step 4:**

- Power up the board by connecting USB Debug Cable to J28 and open windows explorer and confirm that a U-Disk appears as a drive like [Figure 8](#).

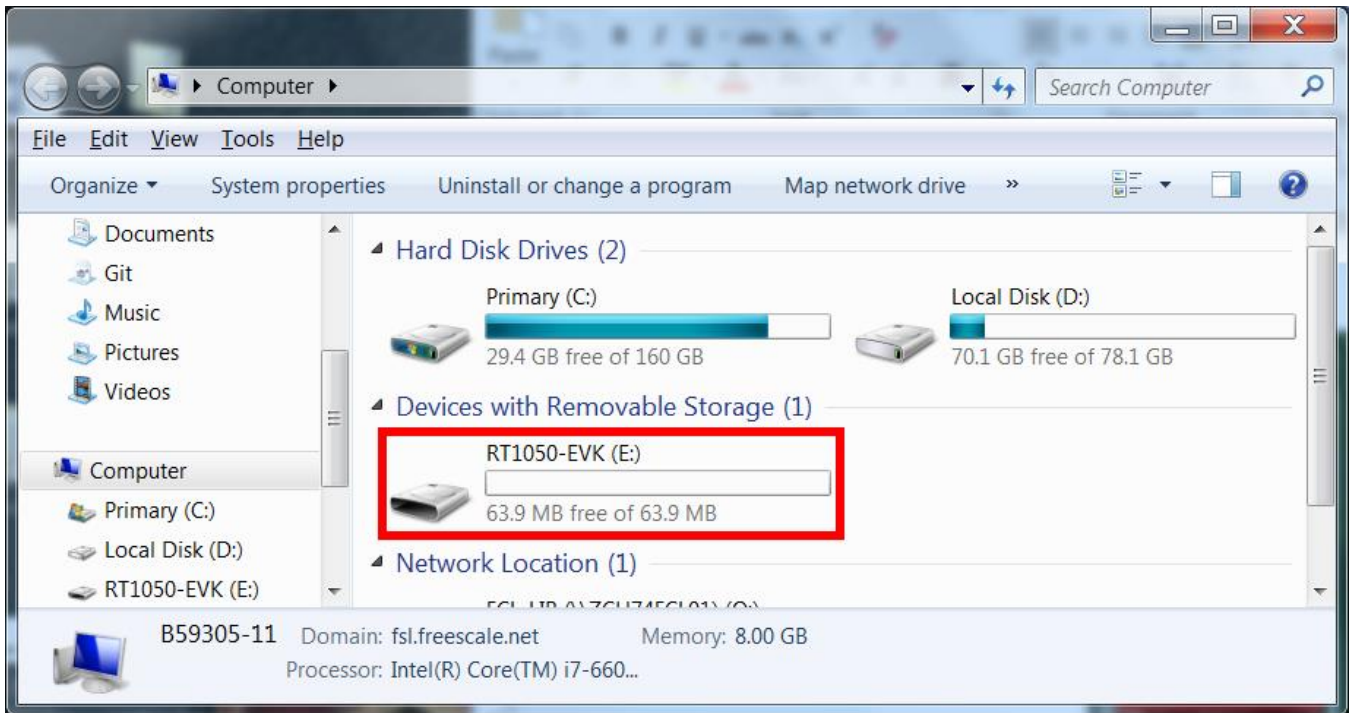


Figure 8. RT1050-EVK appeared

**NOTE**

The first time you connect the MBED USB to Host Computer Windows will ask to install the MBED serial driver.



**Step 5:**

- Drag/Drop the hello\_world.bin to RT1050-EVK. Then the RT1050-EVK disappears and after few seconds it will appear again.

**Step 6:**

- Disconnect the USB Debug Cable, and configure the board to QSPI Flash Boot Mode which means SW7-3 pull-up others pull-down [Figure 9](#).

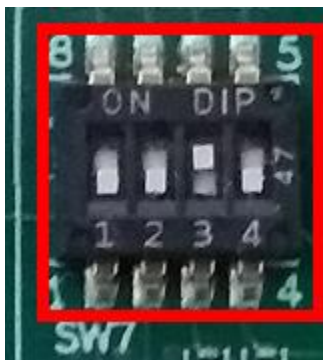


Figure 9. QSPI Flash Boot Mode Configuration

**Step 7:**

- Connect the USB Debug Cable again and configure the Terminal Window.
  - Baud rate: 115200
  - Data bits: 8
  - Stop bit: 1
  - Parity: None
  - Flow control: None

**Step 8:**

- Press SW3 to reset the EVK Board and “hello world” will be printed to the terminal. [Figure 10](#)

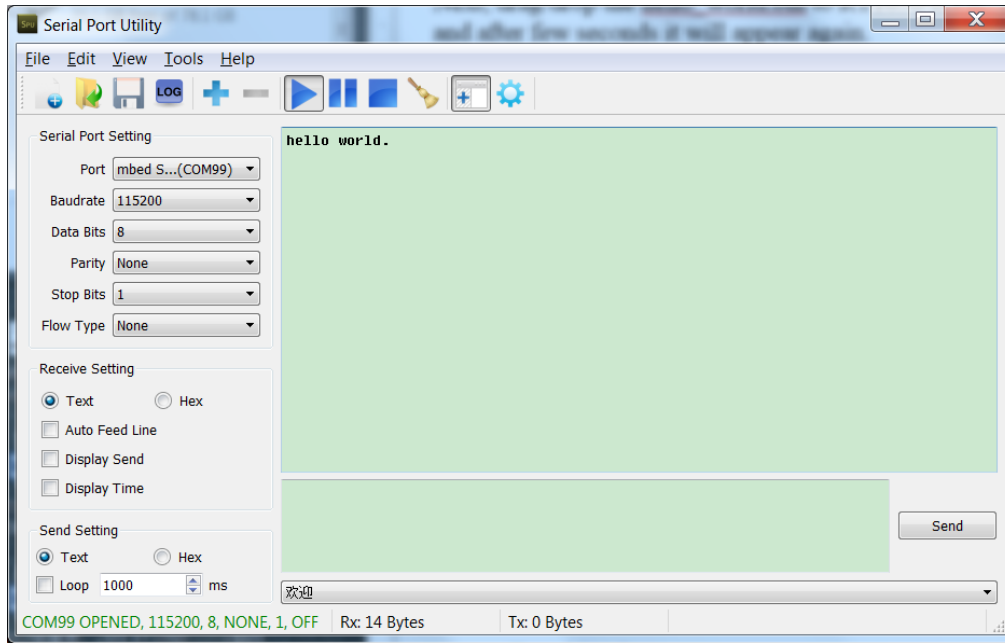


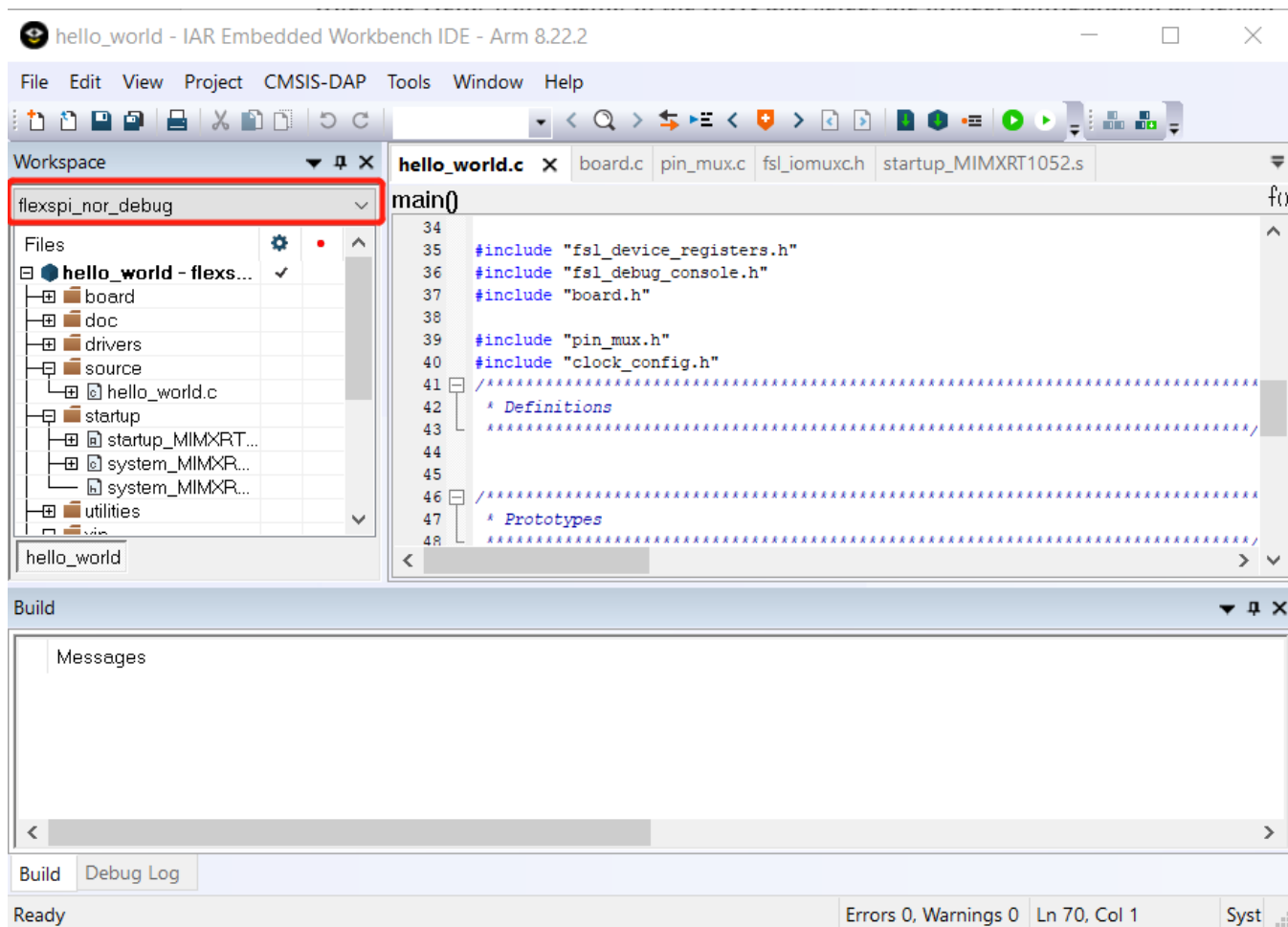
Figure 10. “hello world” be printed to the terminal

## 4.2. MFG Boot from QSPI Flash

This chapter describes the steps that using MFG tool to program an image to QSPI Flash and boot from the QSPI Flash.

**Step 1:**

- Open the Hello world demo in the SDK and select the project configuration as flexspi\_nor\_debug [Figure 11](#) and make sure the settings likes [Figure 12](#).



**Figure 11. Select the project configuration as flexspi\_nor\_debug**

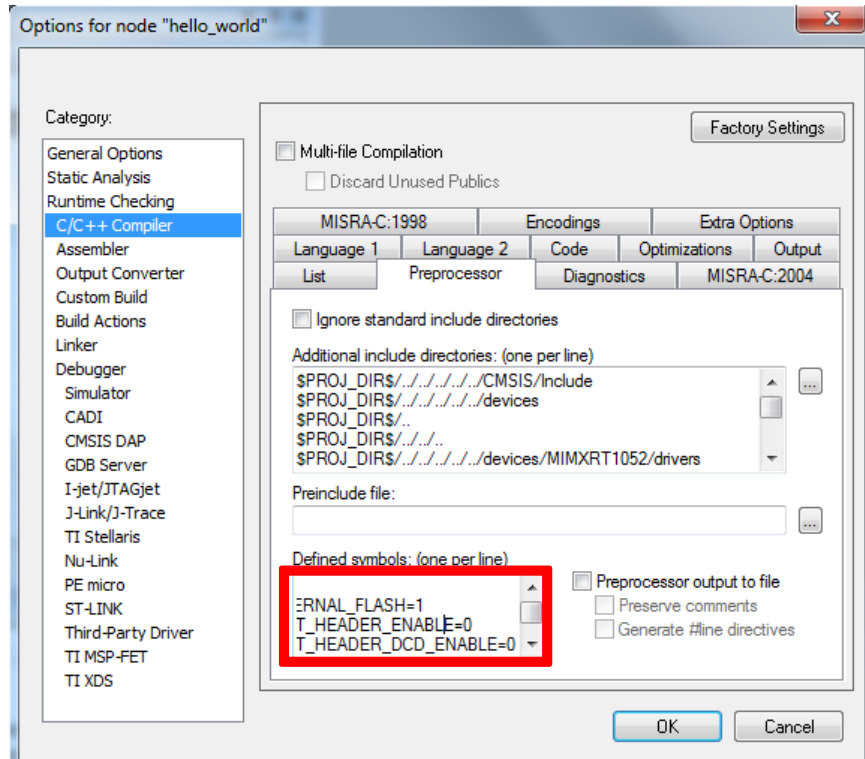


Figure 12. Defined Symbols for hello\_world

Step 2:

Change the default entry to Reset\_Handler likes following Figure.

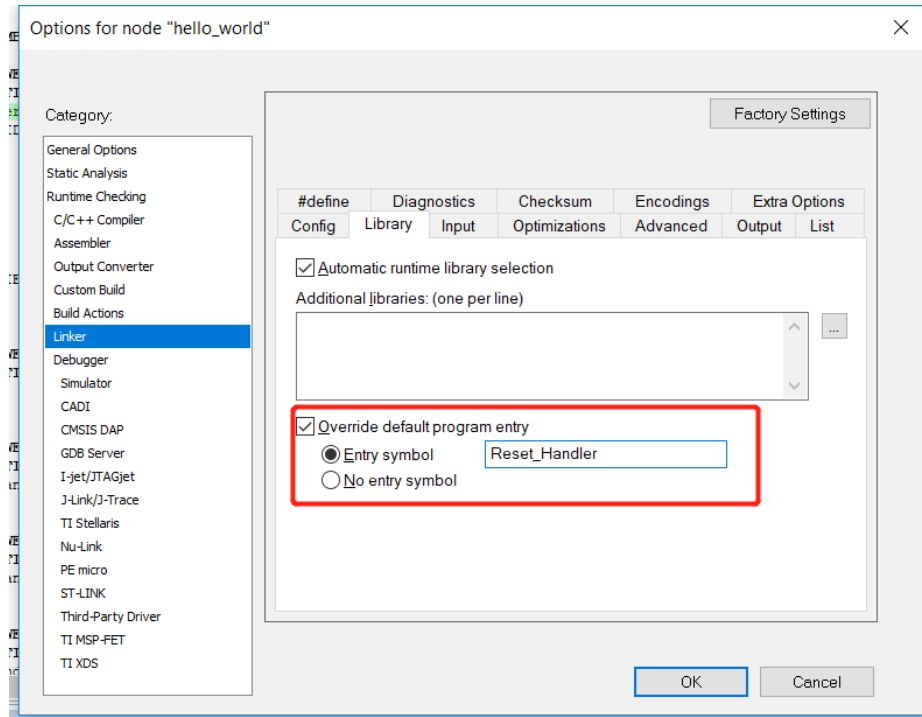


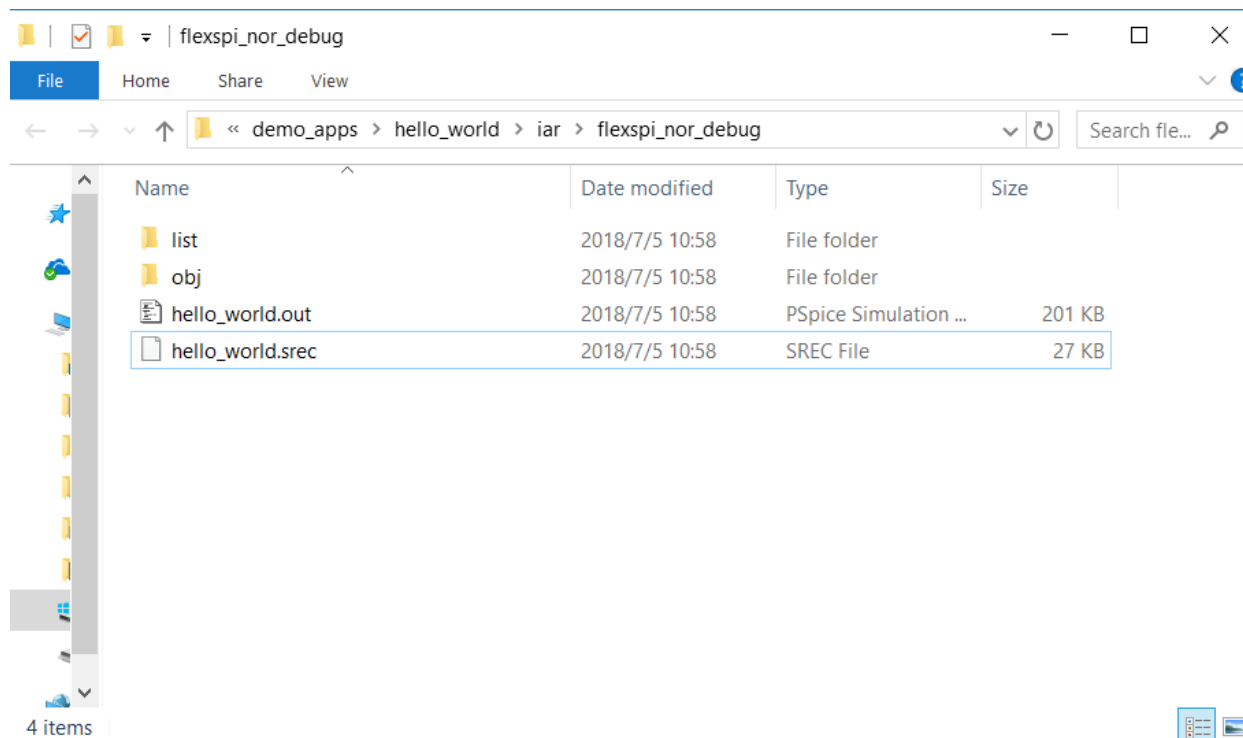
Figure 13. Change the default entry to Reset\_Handle

**NOTE**

Step 5 can be skipped if this step is set.

**Step 3:**

- Build the project and generate the image. You can find the *hello\_world.srec* at following location [Figure 14](#).



**Figure 14. hello\_world.srec location**

**Step 4:**

- Copy hello\_world.srec to the elftosb folder:

## Examples

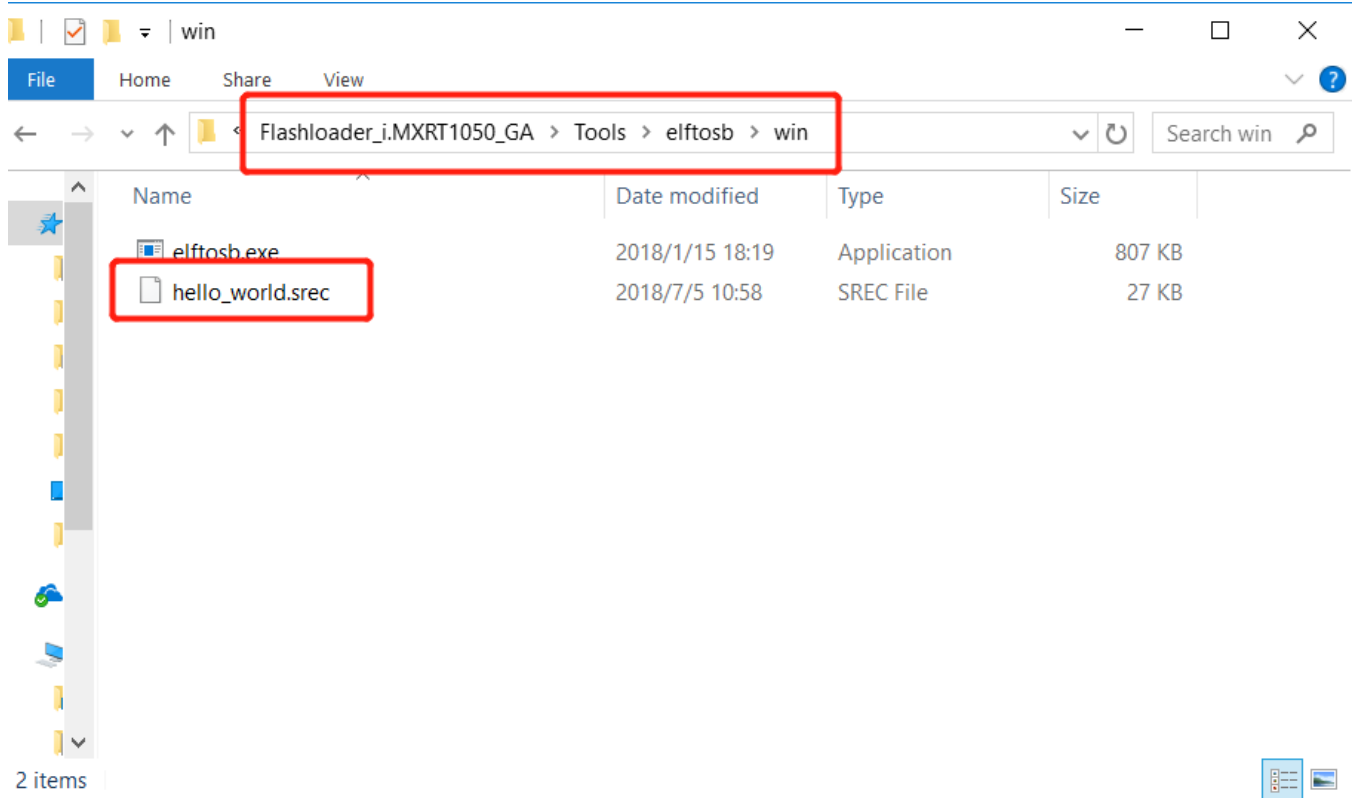


Figure 15. Copy hello\_world.srec

### Step 5:

Open the *imx-flexspinor-normal-unsigned.bd* under path `Flashloader_i.MXRT1050_GA\Tools\bd_file\imx10xx`. Open it and set the `entryPointAddress` to

0x60002000 likes following Figure.

```

1 options {
2     flags = 0x00;
3     startAddress = 0x60000000;
4     ivtOffset = 0x1000;
5     initialLoadSize = 0x2000;
6     # Note: This is required if the default entrypoint is not the
7     #       Please set the entryPointAddress to Reset_Handler add
8     entryPointAddress = 0x60002000;
9 }
10
11 sources {
12     elfFile = extern(0);
13 }
14
15 section (0)
16 {
17 }
18

```

Figure 16. Set the `entryPointAddress` to 0x60002000

#### NOTE

Step 2 can be skipped if this step is set.

#### Step 6:

- Now we can use command to generate the i.MX Bootable image using `elftosb` file. Open `cmd.exe` and type following command:

```

elftosb.exe -f imx -V -c ../bd_file/imx10xx/imx-flexspinor-normal-unsigned.bd -o
ivt_flexspi_nor_hello_world.bin hello_world.srec

```

```

Windows PowerShell
PS C:\Users\...\Desktop\Flashloader_i.MXRT1050_GA\Flashloader_i.MXRT1050_GA\Flashloa
n> .\elftosb.exe -f imx -V -c ../../bd_file/imx10xx/imx-flexspinor-normal-unsigned.bd -o
ivt_flexspi_nor_hello_world.bin hello_world.srec
Section: 0x0
iMX bootable image generated successfully
PS C:\Users\...\Desktop\Flashloader_i.MXRT1050_GA\Flashloader_i.MXRT1050_GA\Flashloa
der_RT1050_1.1\Tools\elftosb\win>

```

**Figure 17. Generate i.MX Bootable image**

After above command, two bootable images are generated:

- ivt\_flexspi\_nor\_hello\_world.bin
- ivt\_flexspi\_nor\_hello\_world\_nopadding.bin

ivt\_flexspi\_nor\_hello\_world.bin:

The memory regions from 0 to ivt\_offset are filled with padding bytes (all 0x00s).

ivt\_flexspi\_nor\_hello\_world\_nopadding.bin:

Starts from ivtdata directly without any padding before ivt.

The later one will be used to generate SB file for QSPI Flash programming in subsequent section.

### Step 7:

Open cmd.exe and type following command:

```

elftosb.exe -f kinetis -V -c ../../bd_file/imx10xx/program_flexspinor_image_qspinor.bd -o boot_image.sb
ivt_flexspi_nor_hello_world_nopadding.bin

```



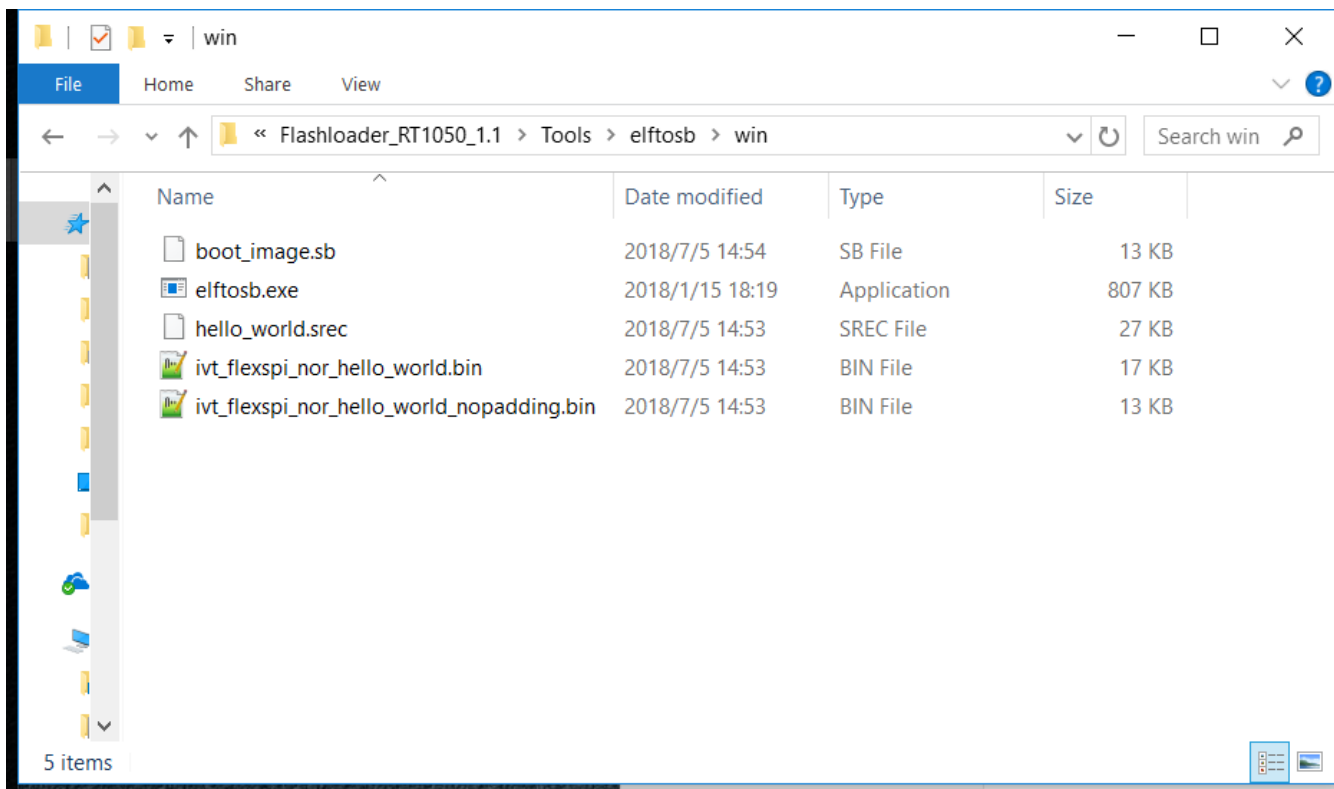
```

Windows PowerShell
PS C:\Users\...\Desktop\Flashloader_i.MXRT1050_GA\Flashloader_i.MXRT1050_GA\Flashloa
n> .\elftosb.exe -f imx -V -c ../../bd_file/imx10xx/imx-flexspinor-normal-unsigned.bd -o
ivt_flexspi_nor_hello_world.bin hello_world.srec
Section: 0x0
iMX bootable image generated successfully
PS C:\Users\...\Desktop\Flashloader_i.MXRT1050_GA\Flashloader_i.MXRT1050_GA\Flashloa
der_RT1050_1.1\Tools\elftosb\win> .\elftosb.exe -f kinetis -V -c ../../bd_file/imx10xx/pr
ogram_flexspinor_image_qspinor.bd -o boot_image.sb ivt_flexspi_nor_hello_world_nopadding.
bin
Boot Section 0x00000000:
FILL | adr=0x00002000 | len=0x00000004 | ptn=0xc0000006
ENA | adr=0x00002000 | cnt=0x00000004 | flg=0x0900
ERAS | adr=0x60000000 | cnt=0x00010000 | flg=0x0000
FILL | adr=0x00003000 | len=0x00000004 | ptn=0xf000000f
ENA | adr=0x00003000 | cnt=0x00000004 | flg=0x0900
LOAD | adr=0x60001000 | len=0x000032b4 | crc=0x9958d743 | flg=0x0000
PS C:\Users\...\Desktop\Flashloader_i.MXRT1050_GA\Flashloader_i.MXRT1050_GA\Flashloa
der_RT1050_1.1\Tools\elftosb\win>

```

Figure 18. Create a SB file for QSPI Flash programming

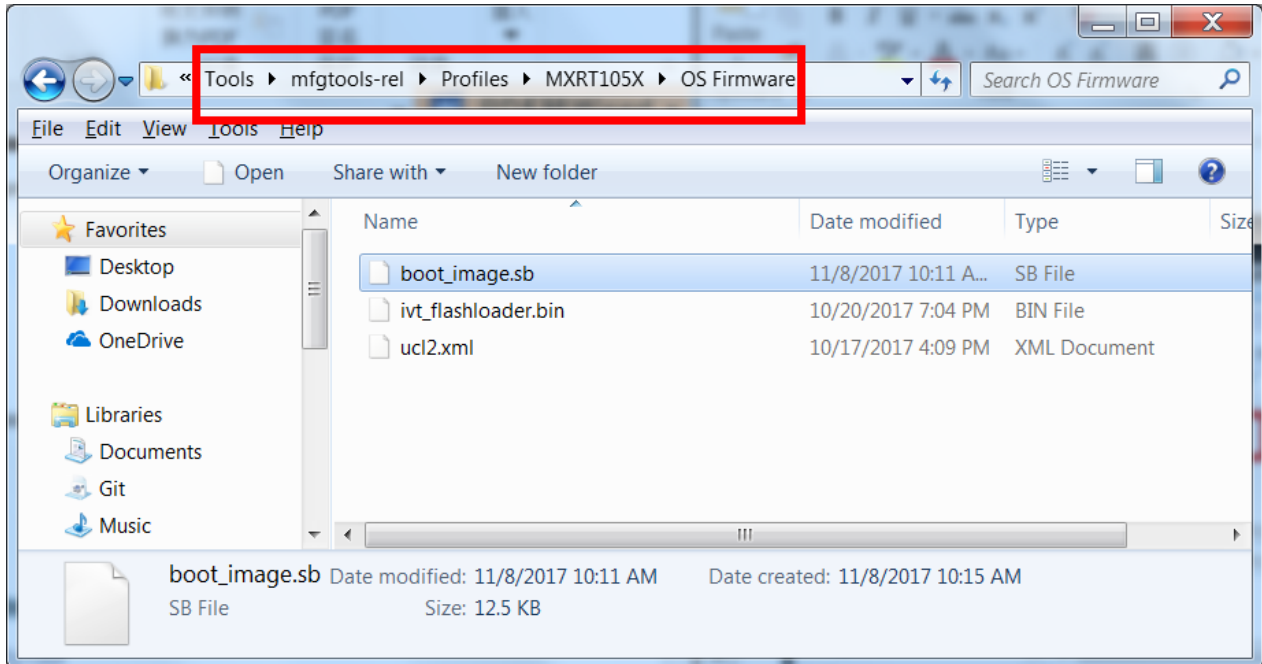
After performing above command, the boot\_image.sb is generated under elftosb folder [Figure 19](#).



**Figure 19. The boot\_image.sb is generated**

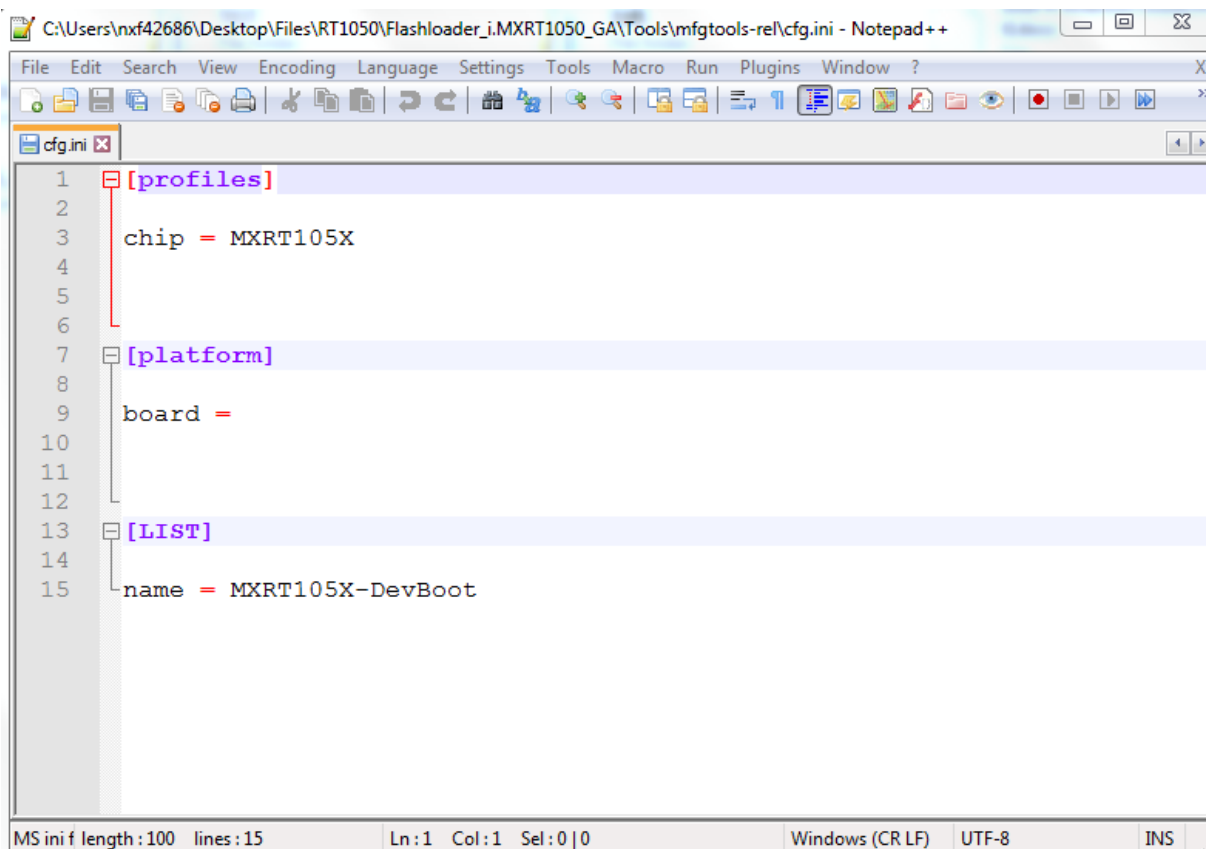
**Step 8:**

- Copy the boot\_image.sb file to OS Firmware folder:



**Figure 20. Copy the boot\_image.sb to OS Firmware folder**

Now, make sure the “name” under “[List]” to “**MXRT105x-DevBoot**” in *cfg.ini* file under `<mfgtool_root_dir>` folder.



```
1 [profiles]
2
3 chip = MXRT105X
4
5
6
7 [platform]
8
9 board =
10
11
12
13 [LIST]
14
15 name = MXRT105X-DevBoot
```

MS ini f length:100 lines:15 Ln:1 Col:1 Sel:0|0 Windows (CR LF) UTF-8 INS

**Figure 21. Make sure the name to “MXRT105x-DevBoot”**

Switch the EVK-Board to Serial Downloader mode by setting SW7 to “1-OFF, 2-OFF, 3-OFF, 4-ON”. Connect a UAB Cable to J9 and power on the EVK Board by inserting USB Cable to J28. Open MfgTool, it will show the detected device like [Figure 22](#).

#### NOTE

In some corner case, HID-compliant device is not recognized which is because the PC only have USB root device and no USB hub device, and this software limitation will be fixed in near future, the workaround at this moment is to use external USB hub as extension.

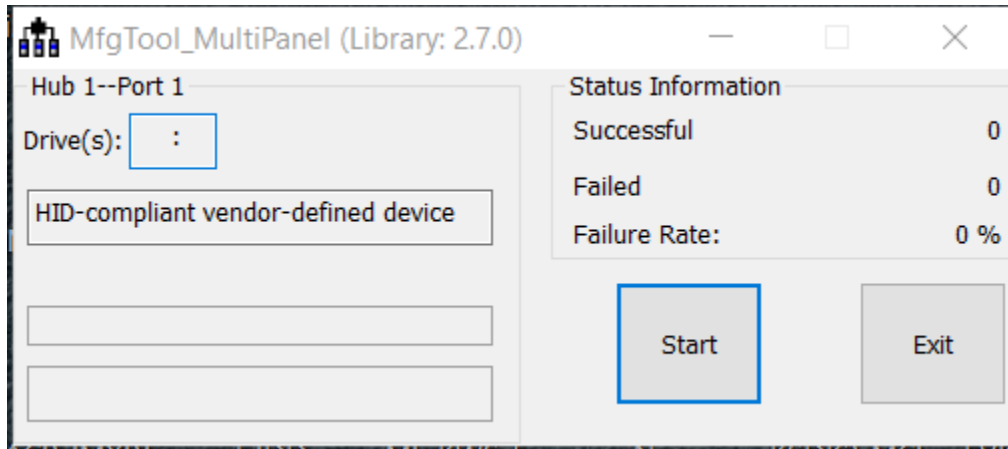


Figure 22. MfgTool GUI with device connected

Click **Start**. Mfgtool initiates and prompts the success status as shown in [Figure 23](#). Click **Stop** and close the Mfgtool.

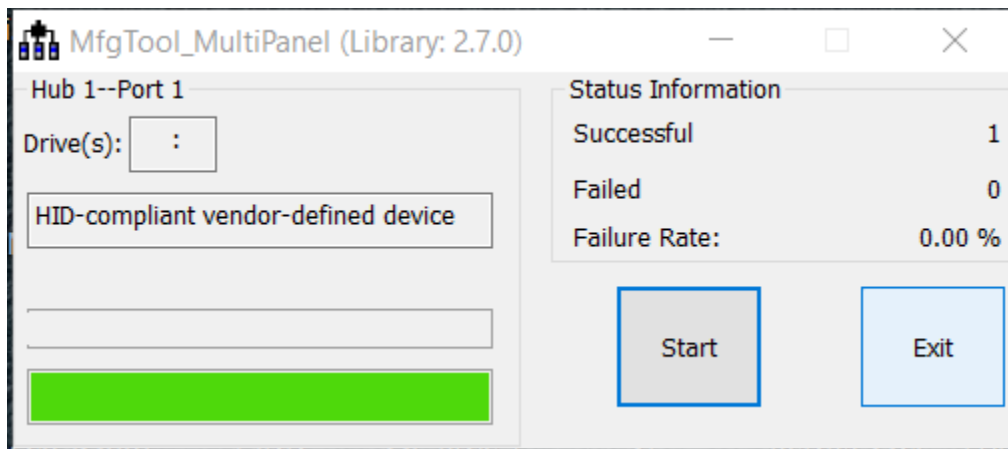


Figure 23. Successful Status

### Step 9:

- Switch the RT1050-EVK board to Internal boot mode and select QSPI Flash as boot device by setting SW7 to “1-OFF, 2-OFF, 3-ON, 4-OFF”. Connect the USB Cable to J28 and open a terminal, then reset the Board. We can see that “hello world” will be printed to the terminal.

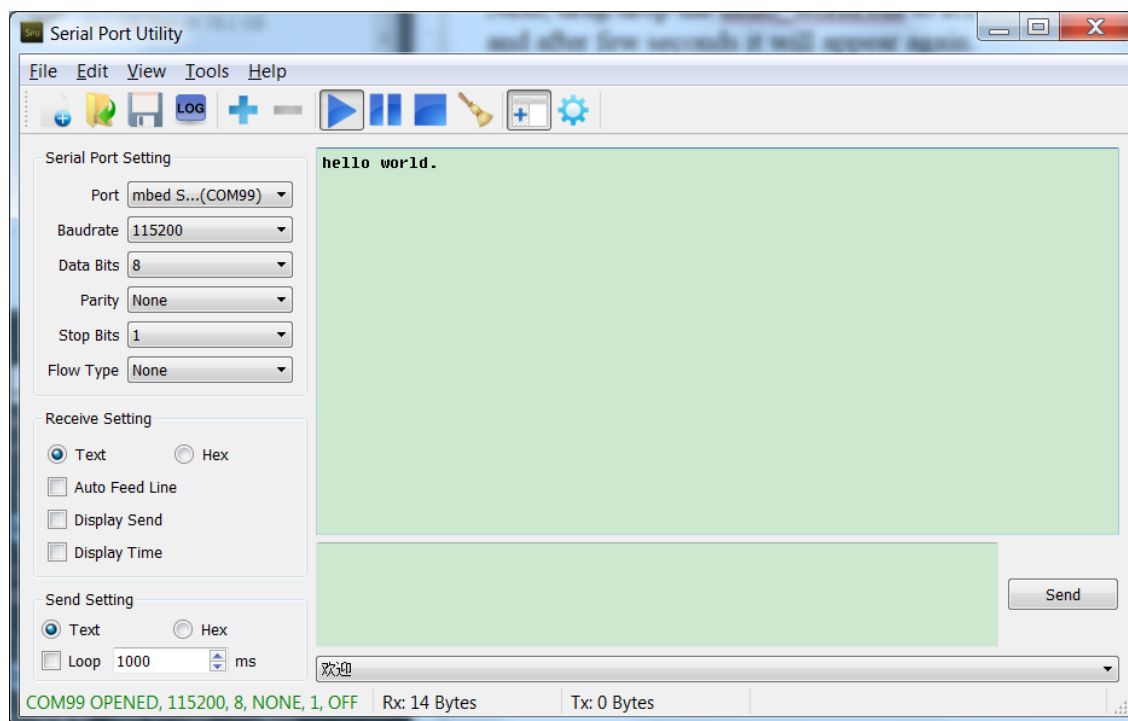


Figure 24. “hello world” be printed to the terminal

### 4.3. MFG Boot from QSPI Flash with DCD for SDRAM

For steps, please refer to [“How to Enable Boot from Octal SPIFlash and SD Card”, AN12107](#).

## 5. QSPI Flash support list

Besides the EVK onboard QSPI Flash, the following Flashes are also supported and please note those are just typical examples with those flash vendors, theoretically we could support all the flash memory that comply with JESD216/JESD216A/JESD216B.

At the same time, the RT1050EVK could support both 1.8 V and 3.3 V SPI flash device by switching the FLASH\_VCC power supply as below [Figure 25](#) shows. If 3.3 V SPI flash is mounted, you need mount R301 and DNP R49, otherwise you need mount R49 and DNP R301.

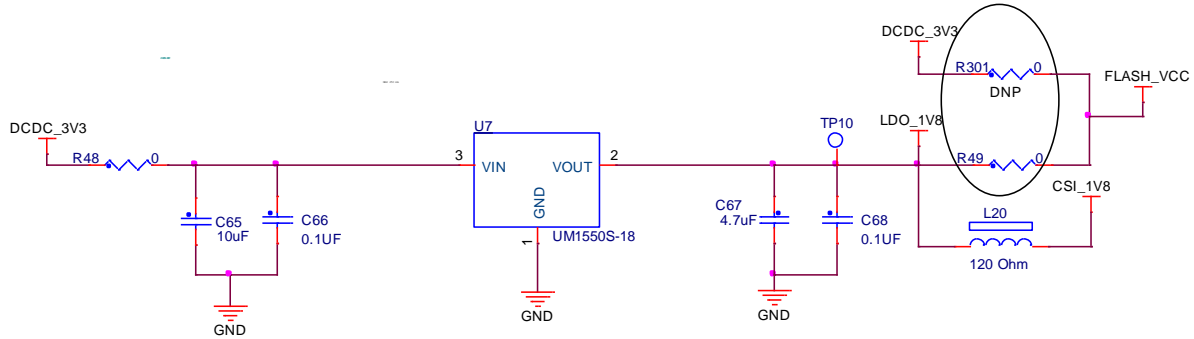


Figure 25. FLASH\_VCC switching

Table 3. QSPI Flash support list

Vendor	Flash Part Number	Voltage
Macronix	MX25L6433F	3.3 V
Macronix	MX25U6435E	1.8 V
ISSI	IS25LP064A-JBLE	3.3 V
ISSI	IS25WP064AJBLE	1.8 V
GigaDevice	GD25Q64C	3.3 V
GigaDevice	GD25LQ64C, GD25LT256E, GD25LB256E	1.8 V
WINBOND	W25Q64JV	3.3 V
WINBOND	W25Q64FW	1.8 V
Micron	MT25QL128ABA1ESE-0SIT	3.3 V
Micron	MT25QU128ABA1ESE-0SIT	1.8 V
Adesto	AT25QF641-SUB-T	3.3 V
Adesto	AT25QL641-SUE-T	1.8 V

## 6. Conclusion

This application note mainly describes how to use Flashloader step by step. For more information, refer to “i.MX MCU Manufacturing User's Guide” and [“How to Enable Boot from Octal SPIFlash and SD Card”](#).

## 7. Revision history

**Table 4. Revision history**

Revision number	Date	Substantive changes
0	12/2017	Initial release
1	06/2018	Adapted SDK version 2.3.1 and Flashloader version 1.1. In Table 1. QSPI Flash support list, changed MX25U6433F to MX25L6433F.
2	07/2018	<ul style="list-style-type: none"> <li>• Added steps to change the entry address.</li> <li>• Used srec file instead of .out file as the source file.</li> </ul>
3	09/2018	Updated Table 3 QSPI Flash support list.
4	07//2019	Updated Table 3 QSPI Flash support list.

**How to Reach Us:**

**Home Page:**  
[nxp.com](http://nxp.com)

**Web Support:**  
[nxp.com/support](http://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](http://nxp.com/SalesTermsandConditions).

While NXP has implemented advanced security features, all products may be subject to unidentified vulnerabilities. Customers are responsible for the design and operation of their applications and products to reduce the effect of these vulnerabilities on customer's applications and products, and NXP accepts no liability for any vulnerability that is discovered. Customers should implement appropriate design and operating safeguards to minimize the risks associated with their applications and products.

NXP, the NXP logo, NXP SECURE CONNECTIONS FOR A SMARTER WORLD, COOLFLUX, EMBRACE, GREENCHIP, HITAG, I2C BUS, ICODE, JCOP, LIFE VIBES, MIFARE, MIFARE CLASSIC, MIFARE DESFire, MIFARE PLUS, MIFARE FLEX, MANTIS, MIFARE ULTRALIGHT, MIFARE4MOBILE, MIGLO, NTAG, ROADLINK, SMARTLX, SMARTMX, STARPLUG, TOPFET, TRENCHMOS, UCODE, Freescale, the Freescale logo, AltiVec, C-5, CodeTEST, CodeWarrior, ColdFire, ColdFire+, C-Ware, the Energy Efficient Solutions logo, Kinetis, Layerscape, MagniV, mobileGT, PEG, PowerQUICC, Processor Expert, QorIQ, QorIQ Qonverge, Ready Play, SafeAssure, the SafeAssure logo, StarCore, Symphony, VortiQa, Vybrid, Airfast, BeeKit, BeeStack, CoreNet, Flexis, MXC, Platform in a Package, QUICC Engine, SMARTMOS, Tower, TurboLink, UMEMS, EdgeScale, EdgeLock, eIQ, and Immersive3D are trademarks of NXP B.V. All other product or service names are the property of their respective owners. AMBA, Arm, Arm7, Arm7TDMI, Arm9, Arm11, Artisan, big.LITTLE, Cordio, CoreLink, CoreSight, Cortex, DesignStart, DynamiQ, Jazelle, Keil, Mali, Mbed, Mbed Enabled, NEON, POP, RealView, SecurCore, Socrates, Thumb, TrustZone, ULINK, ULINK2, ULINK-ME, ULINK-PLUS, ULINKpro,  $\mu$ Vision, Versatile are trademarks or registered trademarks of Arm Limited (or its subsidiaries) in the US and/or elsewhere. The related technology may be protected by any or all of patents, copyrights, designs and trade secrets. All rights reserved. Oracle and Java are registered trademarks of Oracle and/or its affiliates. The Power Architecture and Power.org word marks and the Power and Power.org logos and related marks are trademarks and service marks licensed by Power.org.

Document Number: AN12108  
Rev. 4  
07/2019

