AN14589

Boot the i.MX RT1180 and Test the Boot Time from Different Devices

Rev. 1.0 — 4 March 2025 Application note

Document information

Information	Content
Keywords	AN14589, i.MX RT1180, boot ROM, boot time, FlexSPI
Abstract	This document describes the i.MX RT1180 system boot, i.MX RT1180 booting from different boot devices, use of the MCUXpresso secure provisioning tool, and the method to test the boot time from different boot devices.

1 Introduction

The i.MX RT1180 crossover MCU of NXP is a dual-core device that features the Arm Cortex-M7 (CM7) and Cortex-M33 (CM33) cores. The CM7 and CM33 cores can operate at a speed of up to 800 MHz and 240 MHz respectively with 1.5 MB on-chip random-access memory (RAM).

The i.MX RT1180 supports the following features:

- Multiple protocols
- Bridging communications between the real-time Ethernet and the Industry 4.0 systems
- Advanced security with the integrated EdgeLock Secure Enclave

The i.MX RT1180 MCU includes an integrated Gbit/s time sensitive networking (TSN) switch and EtherCAT slave controller (ESC). These features make it suitable for industrial and automotive communication applications. The MCU supports the use of the MCUXpresso ecosystem, which has the following features:

- Software development kit (SDK)
- A choice of IDEs
- Secure provisioning and configuration tools to enable rapid development

The i.MX RT1180 supports booting from various boot devices. For more details, see Section 2.8.

This document outlines the following details:

- i.MX RT1180 system boot
- i.MX RT1180 booting from different booting devices
- Use of the MCUXpresso secure provisioning tool
- Method to test the boot time from different booting devices

2 i.MX RT1180 boot overview

2.1 Boot feature

The boot process starts with any reset when the hardware reset logic triggers the CM33 boot core to execute from the on-chip boot ROM. To determine the boot flow behavior of the device, the boot ROM uses the state of the internal register bit field `SBMR2[BOOT_MODE]` and various fuse settings. This register reflects the status of the Boot mode pins.

2.2 Serial downloader

The boot ROM code allows the downloading of the programs that the device runs. For example, the provisioning program can use a serial connection to provide a new image to the boot device. Usually, the provisioning program is downloaded to the internal RAM and allows programming of the boot devices, such as the FlexSPI NOR flash. The boot ROM serial downloader uses a low-power serial peripheral interface (LPSPI), low-power universal asynchronous receiver/transmitter (LPUART), or a high-speed USB in a non-stream mode connection. Table 1 lists the serial downloader peripherals PinMux on MIMXRT1180-EVK.

Table 1. Serial downloader peripherals PinMux

Peripheral	Instance	Port (IO function)	PAD
LPUART	1	LPUART1_TX	GPIO_AON_08
		LPUART1_RX	GPIO_AON_09
LPSPI	1	LPSPI1_SCK	GPIO_AON_04
		LPSPI1_PCS0	GPIO_AON_05

Table 1. Serial downloader peripherals PinMux*...continued*

Peripheral	Instance	Port (IO function)	PAD
		LPSPI1_SDO	GPIO_AON_06
		LPSPI1_SDI	GPIO_AON_07
USB	1	USB1_DN	USB1_DN
		USB1_DP	USB1_DP

2.3 External memory configuration data

The external memory configuration data (XMCD) allows the boot ROM code to configure the following memory:

- Synchronous dynamic random-access memory (SDRAM) connected to the smart external memory controller (SEMC) controller
- HyperRAM/APMemory pseudostatic random-access memory (PSRAM) via the FlexSPI controller, from an external program image residing on the boot device

The XMCD aims to simplify the external RAM enablement, and it is a replacement for the legacy device configuration data (DCD).

2.4 Secure boot

A key feature of the i.MX RT1180 boot ROM is to perform a secure boot. The signing process can be done through a secure provisioning tool, which NXP provides. The signatures are then included as a part of the final program image. If configured, the signatures are verified via the public keys included in the program image.

In addition to support the digital signature verification to authenticate the program images, the encrypted boot is also supported. The encrypted boot can be used to prevent the cloning of the program image directly off the boot device. It can be performed on all the boot devices supported on the chip.

2.5 Boot ROM features

The following are the main features of the boot ROM:

- Booting from CM33
- Booting from various boot devices
- Recovery boot once primary boot failed
- External RAM expansion using XMCD
- Serial downloader via USB, universal asynchronous receiver/transmitter (UART), and LPSPI
- Encrypted boot via advanced encryption standard-cipher block chaining (AES-CBC)
- Digital signature based advanced high assurance boot (AHAB)
- Encrypted execute-in-place (XIP) booting on serial NOR via FlexSPI interface powered by on-the-Fly AES decryption (OTFAD) and inline encryption engine (IEE)

The boot ROM supports the following primary boot devices:

- Serial NOR flash via FlexSPI
- Serial NAND flash via FlexSPI
- SLC RAWNAND flash via SEMC
- SD or embedded multi-media card (eMMC) via uSDHC

2.6 Boot mode selection

The combination of the following settings decides the Boot mode:

- ROM API
- Fuse
- BOOT_MODE pins

Figure 1 shows the Boot mode determination.

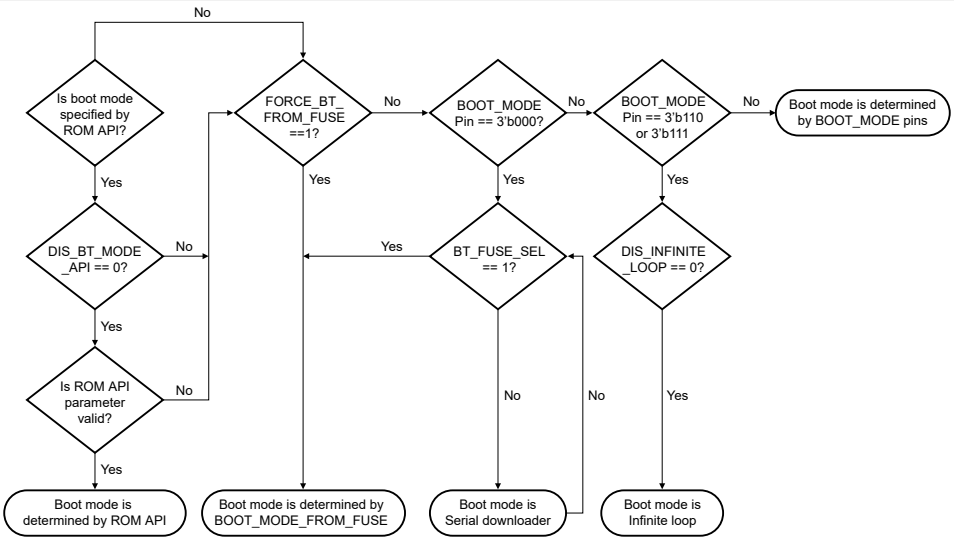

Figure 1. Boot mode determination

2.7 Boot mode pin settings

The i.MX RT1180 has three Boot mode pins. The selection of the Boot mode depends on the binary value stored in the internal register field `SBMR2[BOOT_MODE]`. Sampling the BOOT_MODE[2:0] pin inputs on the rising edge of the POR_B initializes the Boot mode. After sampling these inputs, their subsequent state does not affect the contents of `SBMR2[BOOT_MODE]`. Table 2 shows the Boot mode pin definitions.

Table 2. Boot mode pin definitions

BOOT_MODE[2:0]	Boot type
000	Boot from internal fuses
001	Serial downloader
010	eMMC 8 bits via an uSDHC2
011	SD 4 bits via an uSDHC1
100	Serial NOR flash with JESD216 via FlexSPI1, primary group, PortA
101	Serial NAND flash with 2K page via FlexSPI1, primary group, PortA
110	Infinite loop modes
111	Test mode

Note: *Burn the BOOT_CFG fuse for different boot devices.*

2.8 Boot devices

The i.MX RT1180 supports various boot devices:

- Serial NOR flash via FlexSPI interface
- Serial NAND flash via FlexSPI interface
- NAND flash with SEMC interface on CS0 with bus width 8 bits or 16 bits
- SD/eMMC via uSDHC interface supporting high capacity cards
- Serial NOR/electrically erasable programmable read-only memory (EEPROM) via LPSPI

2.9 Boot image

The i.MX RT1180 boot image can also be called a container-based image. Table 3 shows the top-level view of the boot image.

Table 3. The Top-level view of the boot image

Memory configuration block	
Container 1	Container header
	Image array entry
	Signature block
	Padding to 1 kB (`0x400`) alignment
Container 2 (optional)	Container header
	Image array entry
	Signature block
	Padding to 1 kB (`0x400`) alignment
Padding to 8 kB (`0x2000`) alignment	
NXP images (optional)	NXP firmware
User or OEM images	Application data (optional)
	CM7 image (optional)
	CM33 image

The memory configuration block is a data region from the beginning of the boot device memory, to the start of the container-based image. The memory configuration block has the different size and layout for the different boot devices.

The container consists of the following items:

- Container header
- Image array entry
- Signature block

The container header contains the following essential information about the entire container image, which the ROM uses:

- Size of the container
- Number of images
- Fuse version
- SW version, and so on

The image array entry contains the following information:

- Image offset
- Image size
- Load address, and so on

The signature block contains the following information:

- SRK table offset
- Certificate offset
- Signature offset, and so on

2.10 MCUXpresso secure provisioning tool

The MCUXpresso secure provisioning tool is a GUI-based application, which simplifies the generation and provisioning of bootable executables on NXP MCU devices. The graphical interface provides a streamlined development flow, making it simpler to prepare, flash and fuse images, while using and providing access to the existing utilities.

To achieve the advance scripting, use the command-line interface. To achieve a more advanced secure provisioning flow, modify the tool-generated scripts. Figure 2 shows the MCUXpresso secure provisioning tool block diagram.

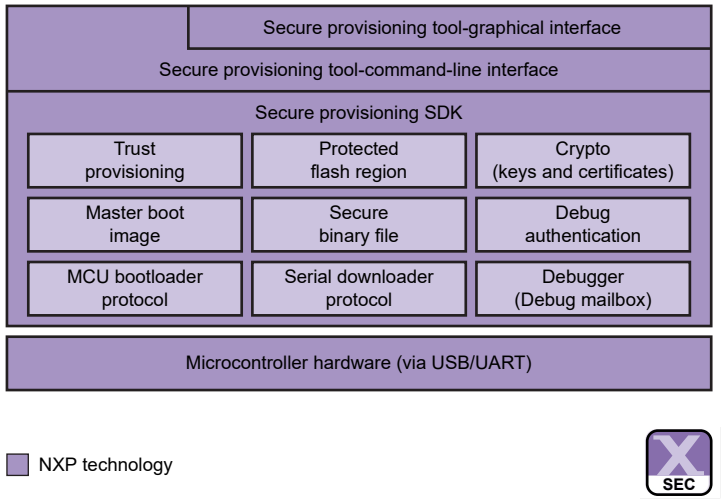

Figure 2. MCUXpresso secure provisioning tool block diagram

3 Boot i.MX RT1180 from different boot devices

This section explains the process to use the MCUXpresso secure provisioning tool to generate the bootable image, which can boot from different boot devices. These boot devices can be QSPI flash, NAND flash, eMMC, SD card, and SPI flash.

3.1 Boot from Quad SPI flash via FlexSPI

1. Open the SDK demo and build the XIP image via IAR. Figure 3 shows the IAR configuration.

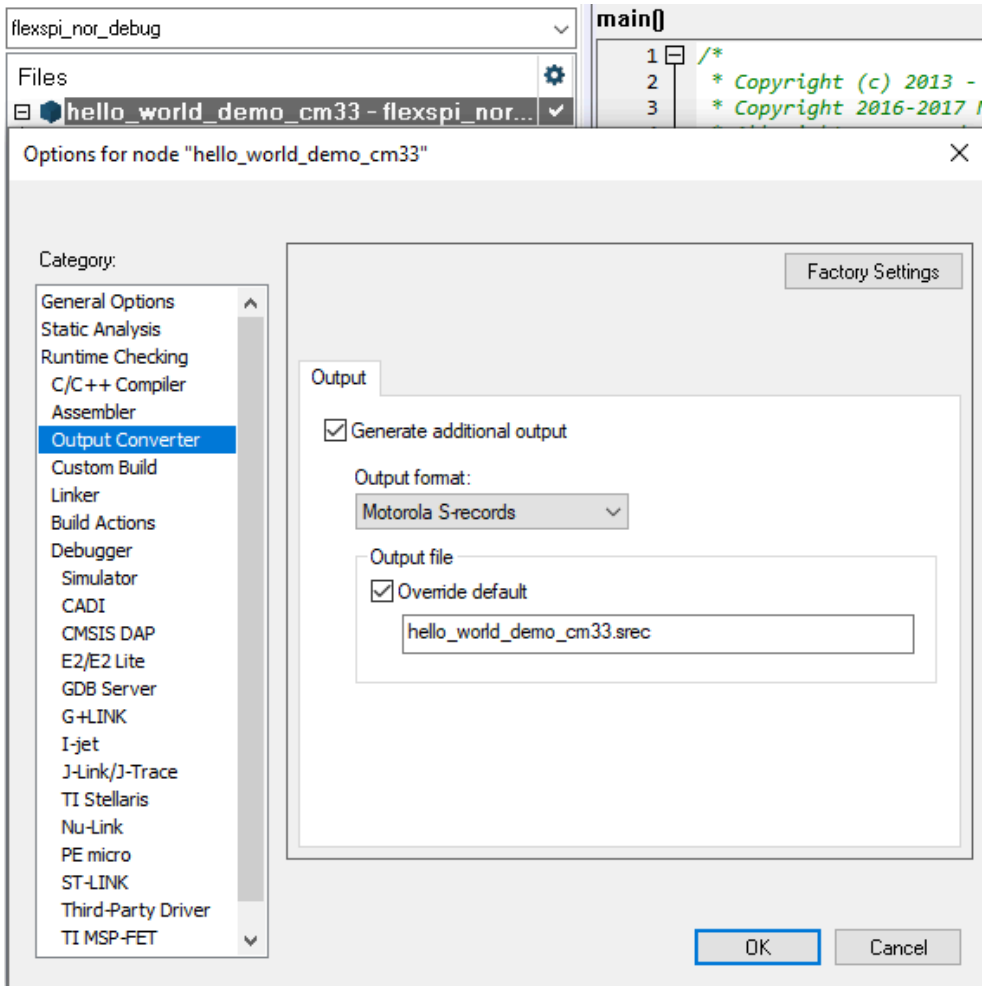

Figure 3. XIP image IAR configuration

- To build an image for booting on FlexSPI instance2, ensure that you link file that requires modification. Change the address for `m_flash_start` to FlexSPI2 address (`0x40000000`).

```
/* CM33 use last 2M bytes space of sdram and hyperram for
define symbol m_sdram_start             = isdefinedsymbol
define symbol m_sdram_size              = isdefinedsymbol
define symbol m_hyperram_start          = isdefinedsymbol
define symbol m_hyperram_size           = isdefinedsymbol
define symbol m_flash_start             = 0x04000000;
define symbol m_flash_size              = 0x00800000;
```

Figure 4. Flash start address change

- Burn the FlexSPI instance2 boot fuse.

BOOT_CFG2[7]	XSPI_INSTANCE	0 - FlexSPI1 1 - FlexSPI2	0

Figure 5. FlexSPI instance fuse

2. To create a “new workspace” in the MCUXpresso secure provisioning tool, perform the following steps:
   a. Open the MCUXpresso secure provisioning tool.

b. Select “choose file” -> “new workspace”.
c. Select the **MIMXRT1189** option button.

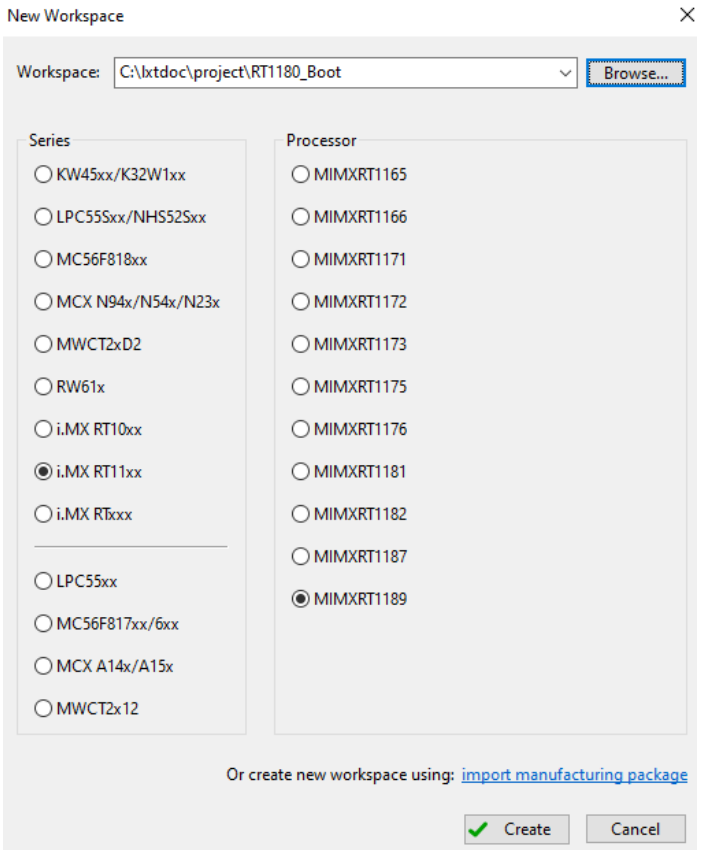

Figure 6. Create a “new workspace” in the MCUXpresso secure provisioning tool

3. To build the image via the MCUXpresso secure provisioning tool, perform the following steps:
   a. Choose the image generated by IAR.
   b. Click the **Build image** button.

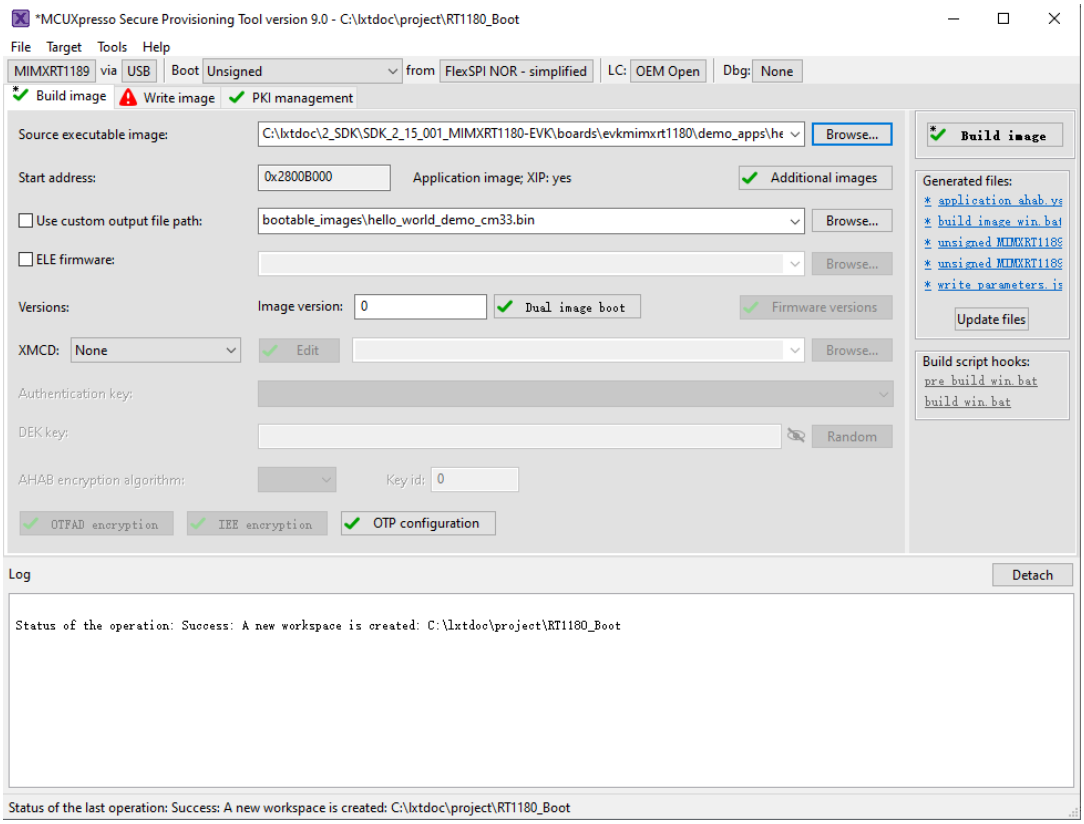

Figure 7. Build image via MCUXpresso secure provisioning tool

4. To write the image via the MCUXpresso secure provisioning tool, perform the following steps:
   a. Ensure that the MIMXRT1180-EVK board is in the serial Download mode SW5 (0001).
   b. Click the **Write image** button.

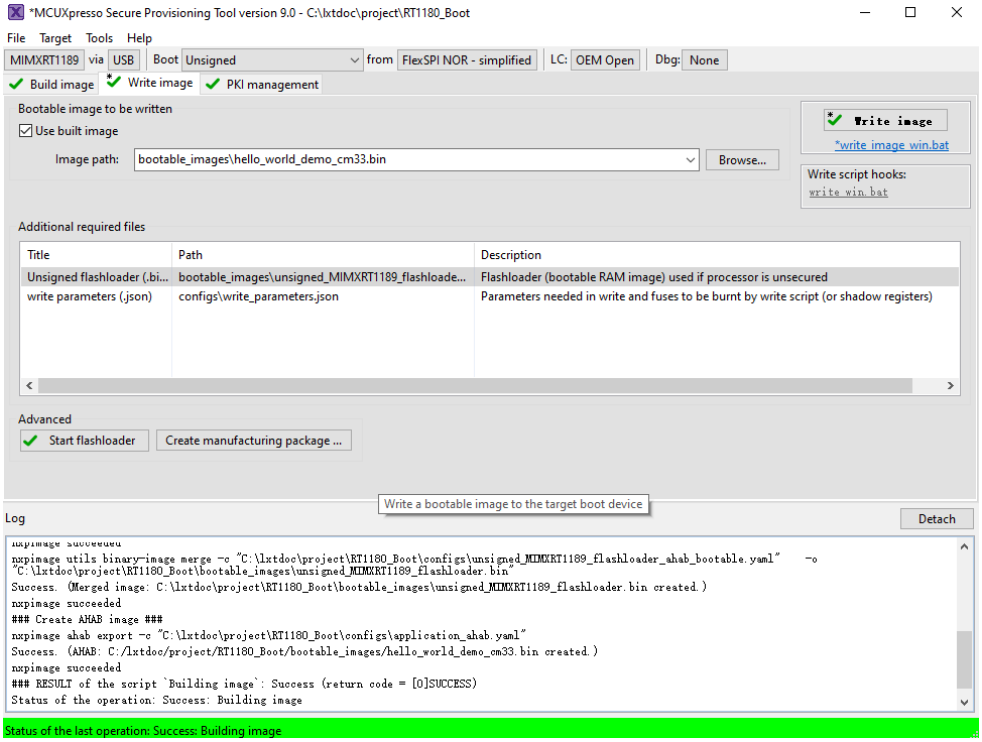

Figure 8. Write image via MCUXpresso secure provisioning tool

5. To debug message on the debug console, perform the following steps:
   a. Switch to the internal Boot mode SW5 (0100).
   b. Press the power reset (POR) button on the MIMXRT1180-EVK, “hello world” can be seen on the serial debug console.

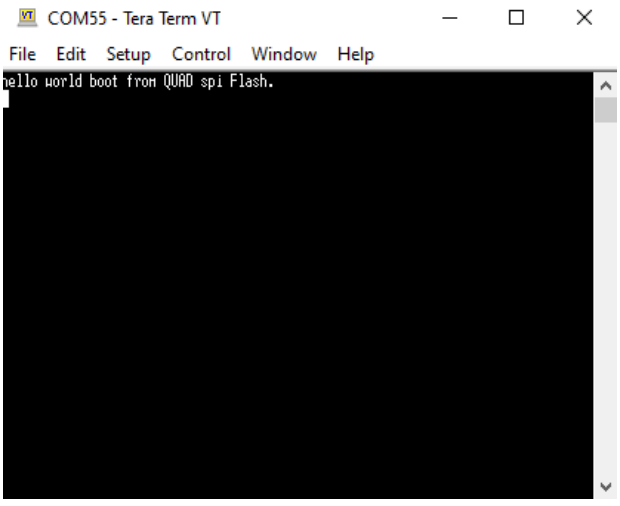

Figure 9. debug message on debug console

3.2 Boot from FlexSPI NAND flash

1. Because the FlexSPI NAND flash cannot support the XIP directly, to build the non-XIP image, perform the following steps:
   a. Build the image.

b. Change flexspi_nor_debug to debug.
c. Figure 10 shows the settings of the **Output Converter**.

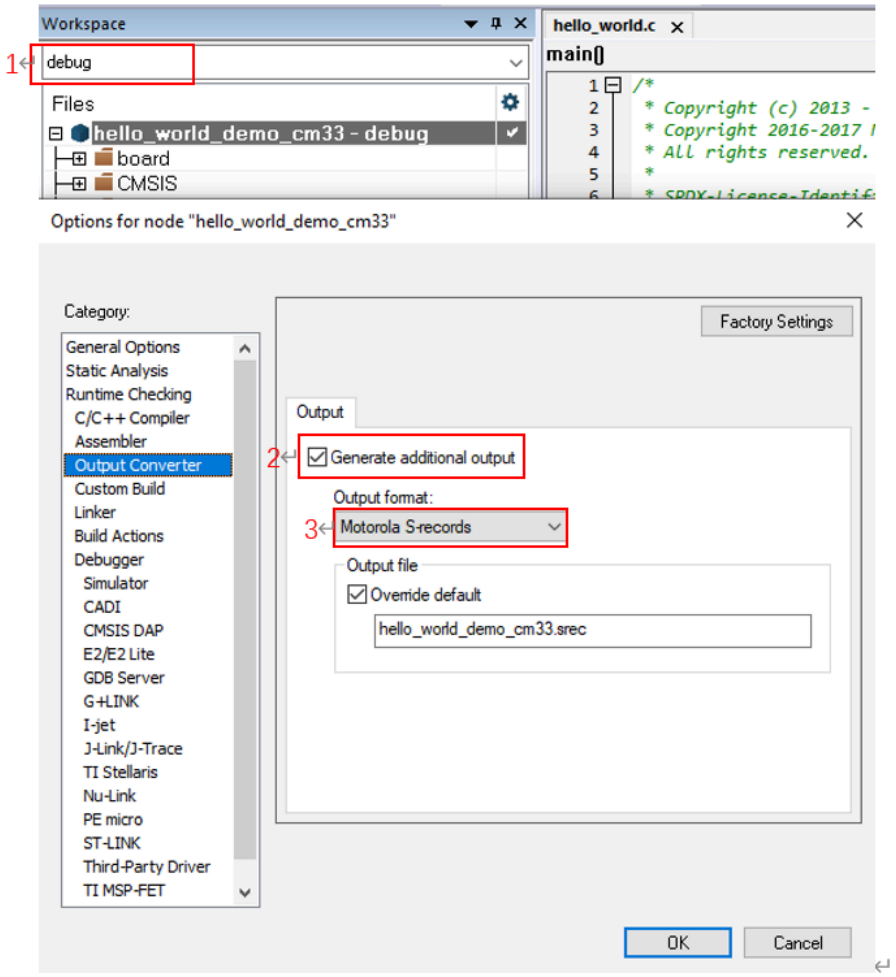

Figure 10. Build the non-XIP image

2. To test the FlexSPI NAND flash, perform the following steps:
   a. Switch to the serial Download mode.
   b. Click boot memory configuration and select FlexSPI NAND.
   c. Click the **Test** button. If the connection is correct, the test is successful.

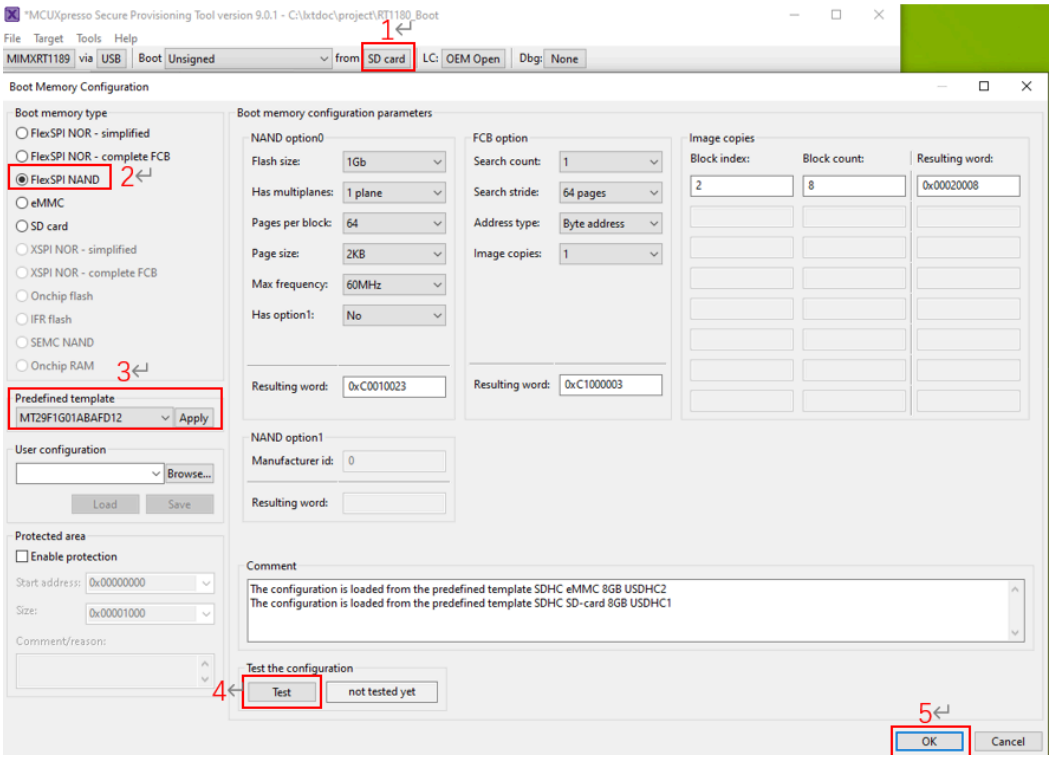

Figure 11. FlexSPI NAND flash configuration in MCUXpresso secure provisioning tool

3. To debug the NAND flash debug message on the debug console, perform the following steps:
   a. Switch the Boot mode to 101 (serial NAND flash with 2K page via FlexSPI1, primary group, PortA).
   b. Press the POR button on the MIMXRT1180-EVK, “hello world” can be seen on the serial debug console.

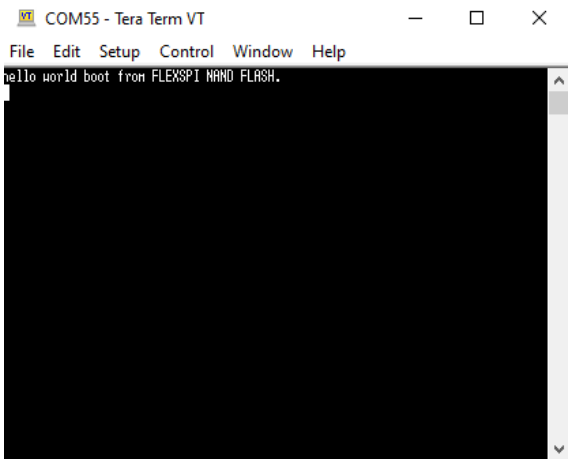

Figure 12. NAND flash debug message on debug console

3.3 Boot from eMMC

1. The eMMC boot image is a non-XIP image. Build the image similar to FlexSPI NAND flash. For more details, see Section 3.2.
2. To configure the eMMC, perform the following steps:
   a. Set the Boot mode to serial download.

b. Open the MCUXpresso secure provisioning tool and click “Boot Memory Configuration”.
c. Select the **SDHC eMMC 8GB USDHC2** and click the **Apply** button.
d. Click the **Test** button. If the connection is correct, the test is successful.

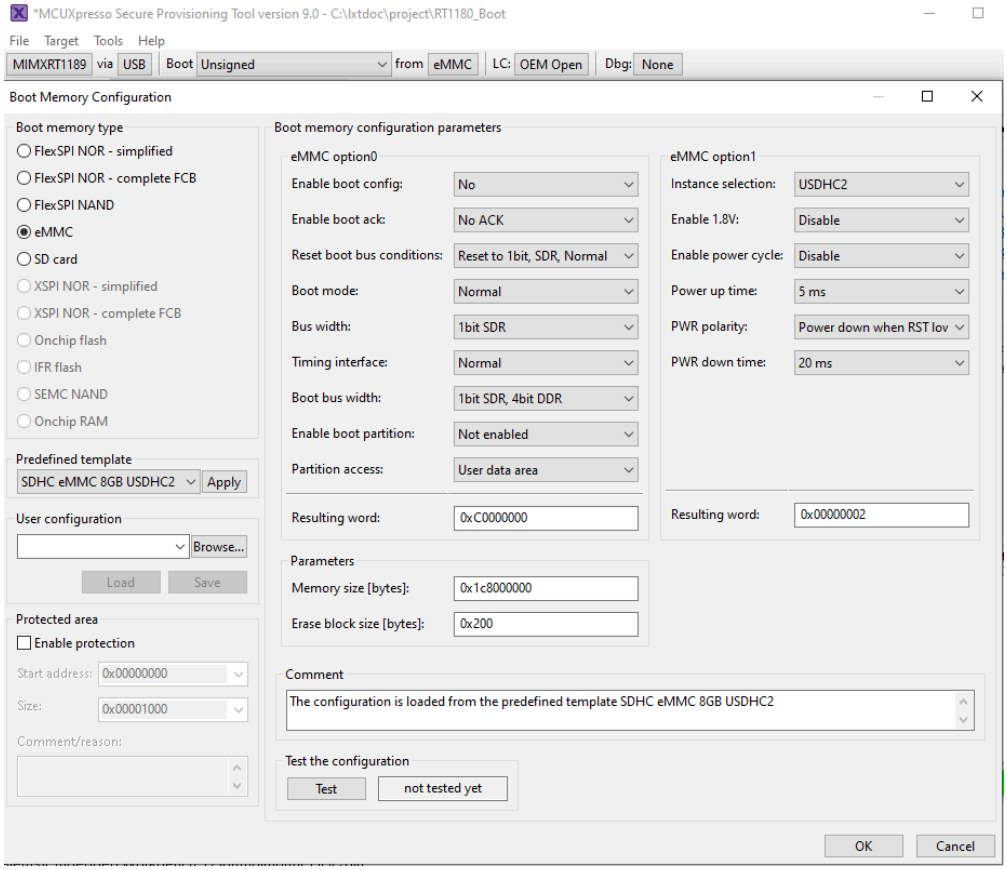

Figure 13. eMMC configuration in MCUXpresso secure provisioning tool

3. To debug the eMMC debug message on the debug console, perform the following steps:
   a. Build and write the image to the eMMC.
   b. Switch the Boot mode from serial download to 010 (eMMC 8 bits via uSDHC2).
   c. Press the POR button on the MIMXRT1180-EVK, “hello world” can be seen on the serial debug console.

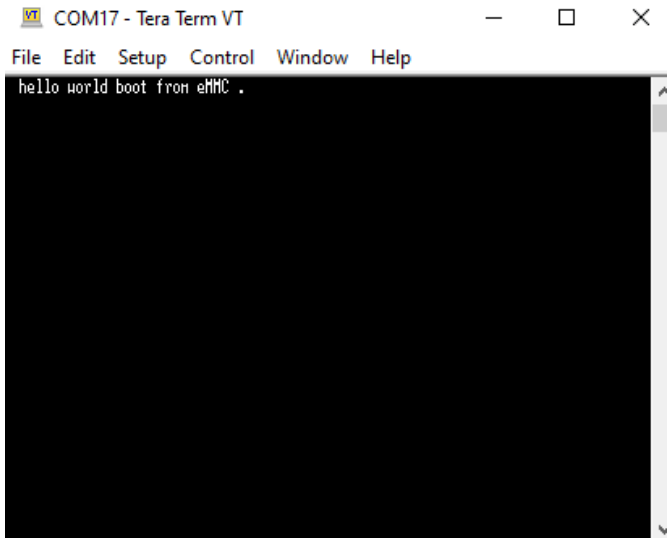

Figure 14. eMMC debug message on debug console

3.3.1 eMMC boot configurations

The i.MX RT1180 platform has three Boot mode pins. To boot from eMMC, set the Boot mode pins[0:2] to 010. The eMMC with bus width 8 bits boots the i.MX RT1180 via uSDHC2 boot device, which has fixed instance and bus width. If you need other configurations like eMMC bus speed mode, burn the fuse.

The following are the options to boot from eMMC via uSDHC1:

- 4-bits mode
- 4-bits DDR mode
- 8-bits DDR mode

The following is the example of booting from eMMC with bus width 4 bits via uSDHC2:

1. Download the image to the eMMC via the MCUXpresso secure provisioning tool.
2. Burn the fuse BOOT_CFG0[2:0] to 2, which selects the eMMC.
3. Burn the fuse BOOT_CFG0[6] to 1, the selected fuse configuration is valid for ROM booting.
4. Burn fuse BOOT_CFG4[0] to 1, the selected uSDHC2 is used for the eMMC boot. If you boot from the fuse, the default eMMC bus width is 4 bits. Table 4 shows the fuse configuration when booting from eMMC with bus width 4 bits via uSDHC2.

Table 4. Fuse configuration when booting from eMMC

FUSE	Definitions	Settings
BOOT_CFG0[2:0]	BOOT_MODE_FROM_FUSE	eMMC
BOOT_CFG0[6]	BT_FUSE_SEL	Fuse configuration is valid for ROM booting
BOOT_CFG4[0]	USDHC_PORT	uSDHC2 is used for the eMMC boot

5. Switch the Boot mode to 000 (boot from internal fuses).
6. Press the POR button on the MIMXRT1180-EVK and the boot is successful.

3.4 Boot from the SD card

1. The SD card cannot XIP. Build a non-XIP image first.
2. To configure the SD card, perform the following steps:
   a. Set the Boot mode to serial download.
   b. Open the MCUXpresso secure provisioning tool and click “Boot Memory Configuration”.
   c. Select the **SD card**.
   d. Click the **Test** button. If the connection is correct, the test is successful.

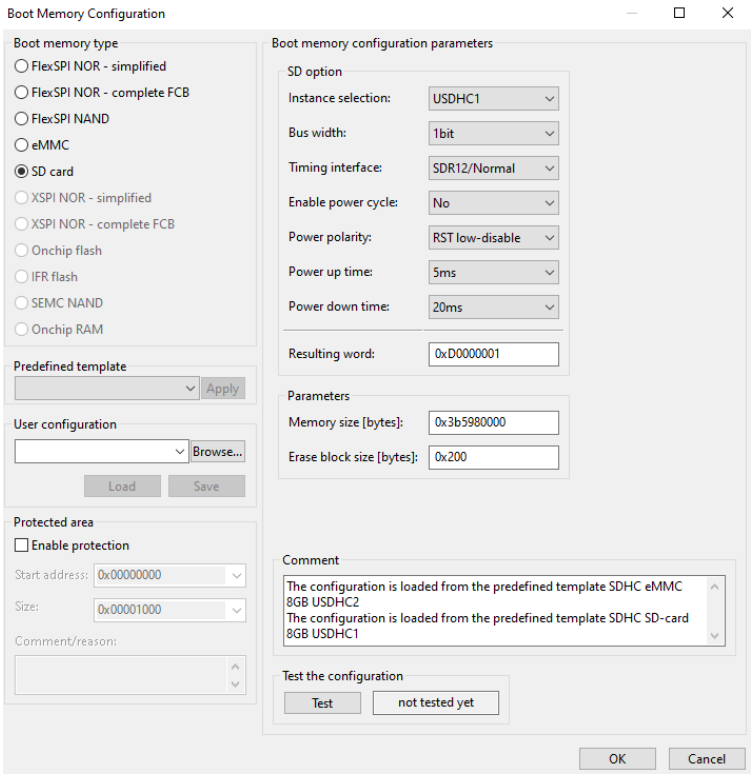

Figure 15. SD card configuration in MCUXpresso secure provisioning tool

3. To debug the SD card debug message on the debug console, perform the following steps:
   a. Build and write the image to the SD card.
   b. Switch the Boot mode to 011 (SD 4 bits via uSDHC1).
   c. Press the POR button on the MIMXRT1180-EVK, “hello world” can be seen on the serial debug console.

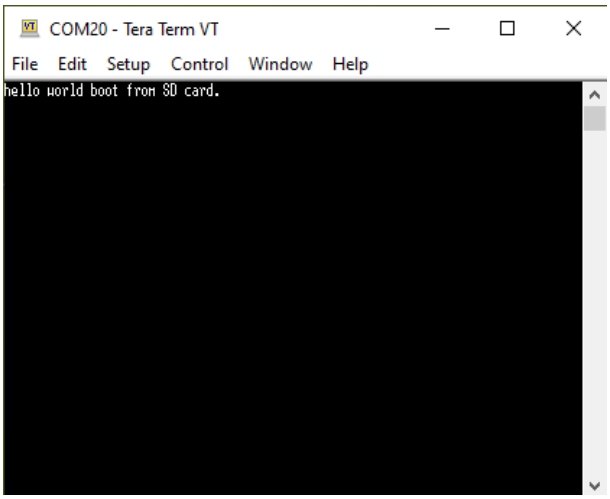

Figure 16. SD card debug message on debug console

The i.MX RT1180 platform has three Boot mode pins. To boot from the SD card, set the Boot mode pins[0:2] to 011. The SD card with bus width 4 bits boots the i.MX RT1180 via uSDHC1 boot device, which has fixed instance and bus width. If you need other configurations, burn the fuse.

3.5 Recovery boot

1. The recovery boot starts when the primary boot fails. Burn the RECOVERY_BOOT_EN fuse.

BOOT_CFG0[4]	RECOVERY_BOOT_EN	0 – Recovery boot is disabled 1 – Recovery boot is enabled	0

Figure 17. RECOVERY_BOOT_EN fuse

2. Build the non-XIP image via IAR. The LPSPI flash cannot XIP. Download the image via blhost as the MCUXpresso secure provisioning tool does not support LPSPI flash boot.

Figure 18. IAR build non-XIP image

3. Add the container to the non-XIP image as follows:
   a. In the MCUXpresso secure provisioning tool, select the image to build.
   b. Click the **Build image** button.

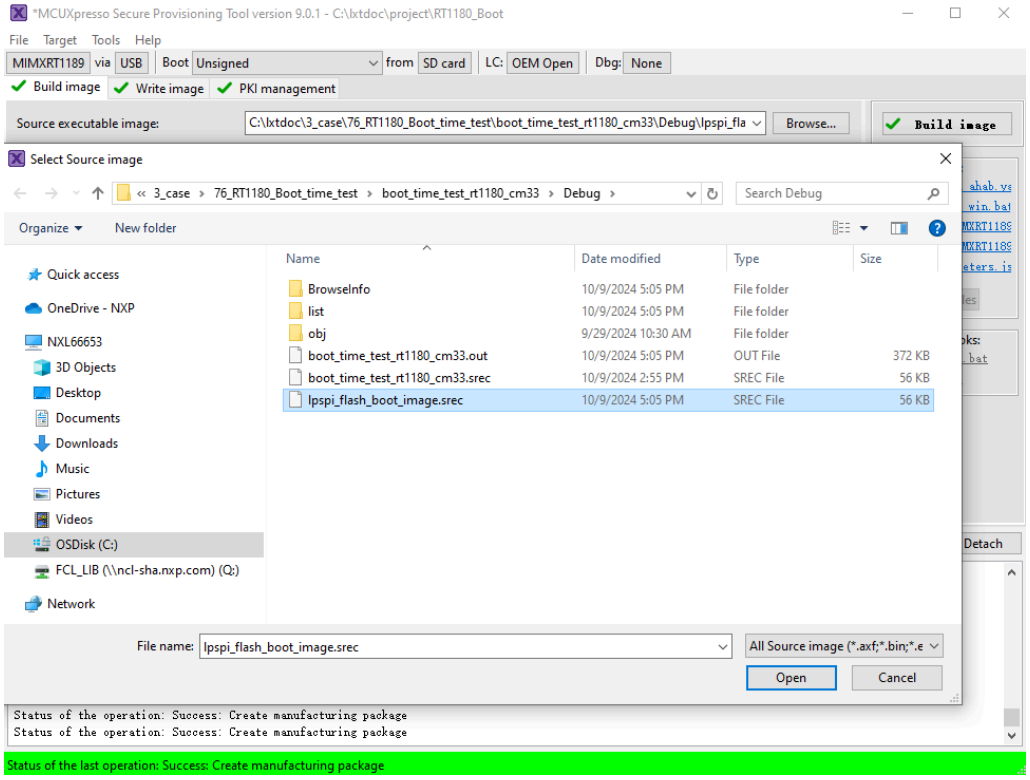

Figure 19. Build image via MCUXpresso secure provisioning tool

4. Switch to the **Write image** page. Click the **Create manufacturing package ...** button.

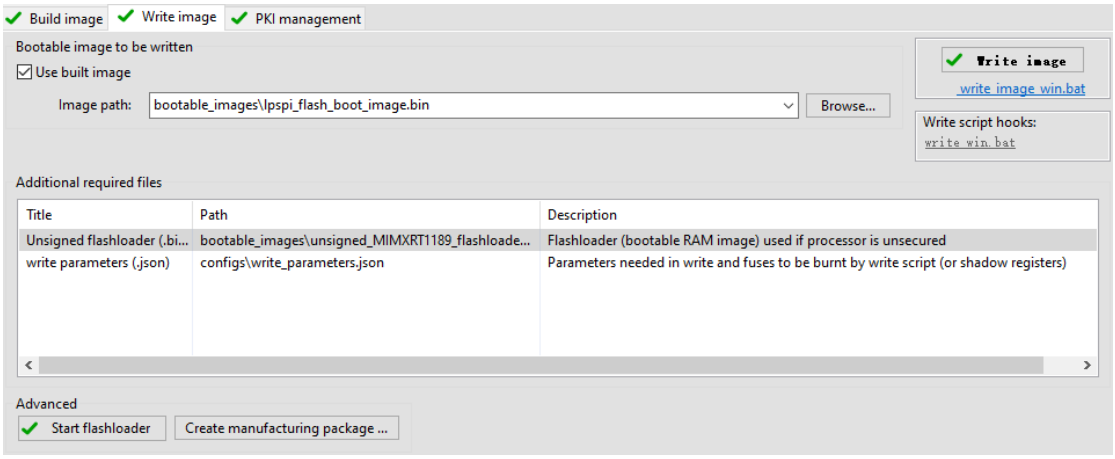

Figure 20. Create the manufacturing package

5. Find the image after adding the container in `manufacturing package`.

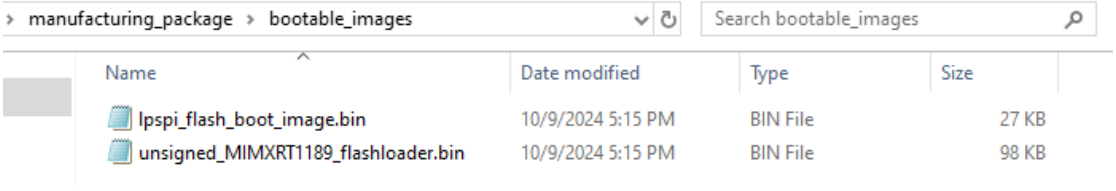

Figure 21. Image after adding container

6. For more details, see “Layout for Serial NOR Flash/E2PROM” in *i.MX RT1180 Reference Manual* (document IMXRT1180RM). Add MCB to the image. In this case, add 1024 zeros before the image.

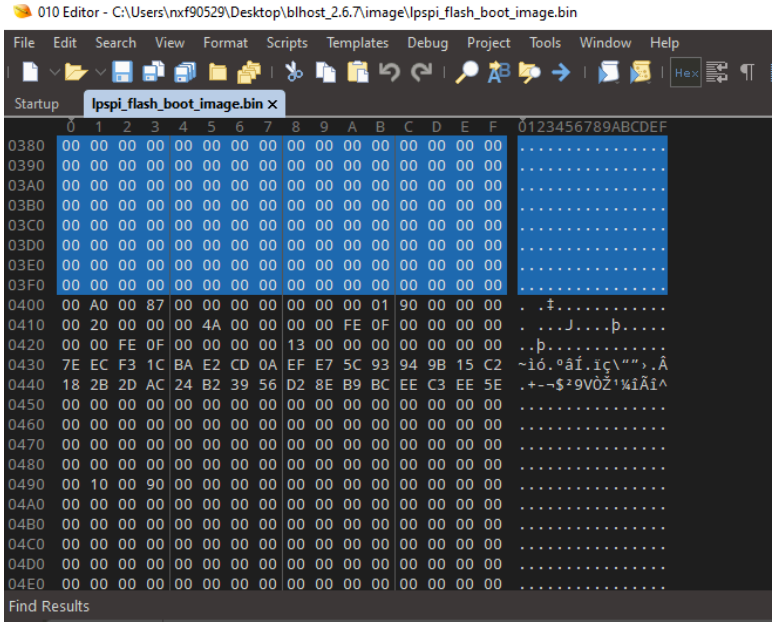

Figure 22. Add 1024 zeros before the image

7. Download the image to the LPSPI flash via blhost. Figure 23 shows the blhost command.

```
Blhost cmd burn image to spi flash
cd C:\Users\nxf90529\Desktop\blhost_2.6.7\bin\win
blhost.exe -u 0x1fc9,0x014c -- get-property 1
blhost.exe -u 0x1fc9,0x014c -- load-image ../../flashloader/flashloader_utility.bin
blhost.exe -u 0x15A2,0x0073 -- get-property 1
blhost.exe -u 0x15A2,0x0073 -- fill-memory 0x20000000 4 0xc1100000
blhost.exe -u 0x15A2,0x0073 -- fill-memory 0x20000004 4 0x00000000
blhost.exe -u 0x15A2,0x0073 -- configure-memory 0x110 0x20000000
blhost.exe -u 0x15A2,0x0073 -- flash-erase-region 0x0 0x20000 0x110
blhost.exe -u 0x15A2,0x0073 -- write-memory 0x0 ../../image/lpspi_flash_boot_image.bin 0x110
blhost.exe -u 0x15A2,0x0073 -- read-memory 0x0 0x400 0x110
```

Figure 23. Blhost command to download the image

8. To debug the eMMC debug message on the debug console, perform the following steps:
   a. Ensure that there is no image in XIP flash.
   b. Switch the Boot mode to 0100 (serial NOR flash).
   c. Press the POR button on the MIMXRT1180-EVK and the boot is successful.

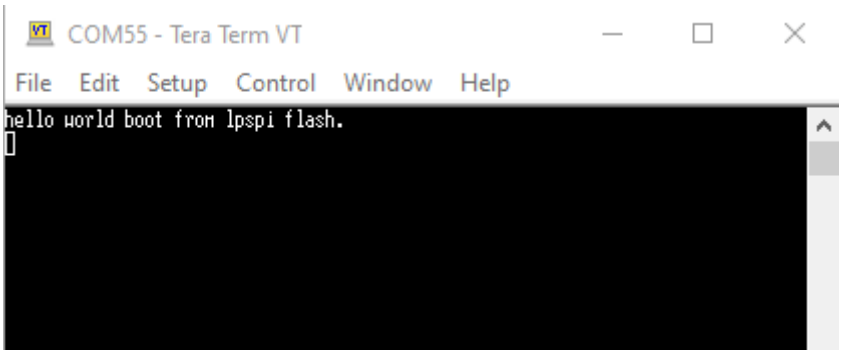

Figure 24. eMMC debug message on debug console

4 Test the boot time from different boot devices

This section tests the boot time from different boot devices.

4.1 Method for testing the boot time

The boot time is measured from the point of 65 % of POR_B to the point of running to “SystemInit” of apps code.

The POR_B test point on MIMXRT1180-EVK is J92-2. Figure 25 shows the POR_B test point.

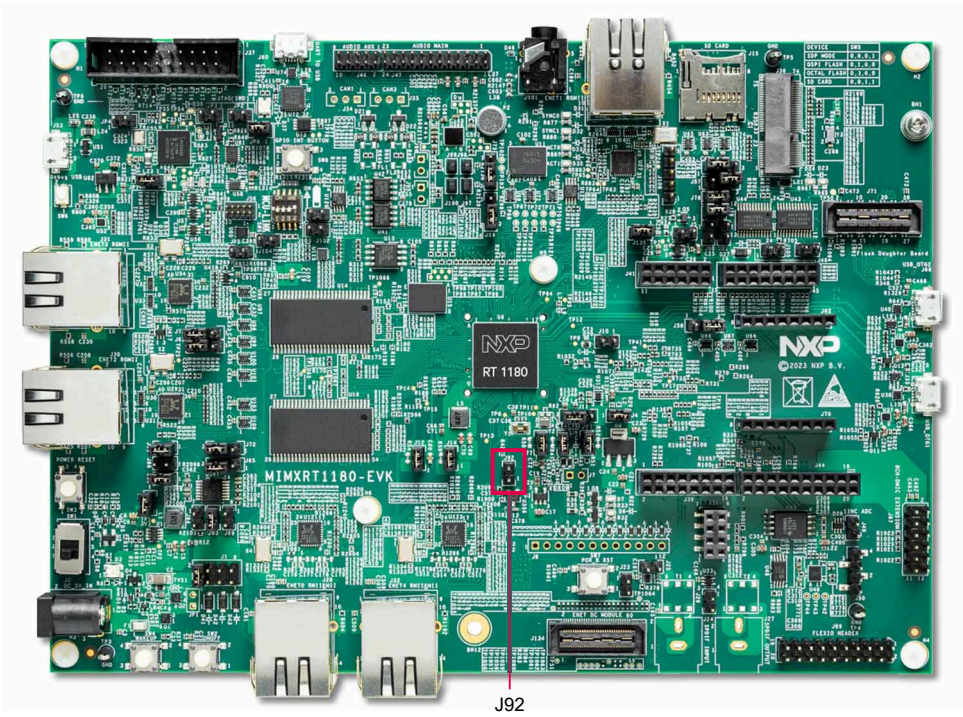

Figure 25. POR_B test point on MIMXRT1180-EVK

The general-purpose input/output (GPIO) in “systemInit” of apps code toggles so that the oscilloscope can capture it.

Figure 26 shows the end time test point.

```
void SystemInit(void)
{

/* EVK GPIO_AD_00 */
#if 1
  RGPIO4->PDDR |= (1UL << 0);
  RGPIO4->PTOR = 1;
#endif

#if ((__FPU_PRESENT == 1) && (__FPU_USED == 1))
    SCB->CPACR |= ((3UL << 10 * 2) | (3UL << 11 * 2)); /* set CP10, CP11 Full Access */
#endif                                                 /* ((__FPU_PRESENT == 1) && (__FPU_USED == 1)) */

#if defined(__MCUXPRESSO)
    extern uint32_t g_pfnVectors[]; // Vector table defined in startup code
    SCB->VTOR = (uint32_t)g_pfnVectors;
#endif
```

Figure 26. End time test point

4.2 The boot time for different boot devices

Test the Quad serial peripheral interface (SPI) flash boot time with the images size of 8626 bytes and 412122 bytes. The start time and the end time test points are the same as given in Section 4.1.

1. The boot time for the 8626 bytes image is about 18.9 ms.

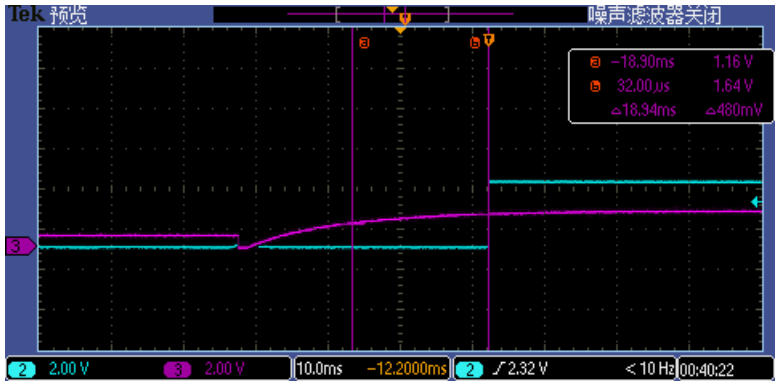

Figure 27. Test the Quad SPI flash boot time for 8626 bytes

2. The boot time for the 412122 bytes image is about 26.3 ms.

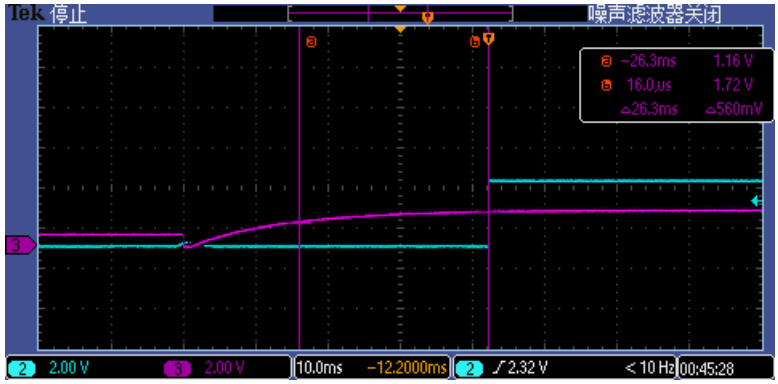

Figure 28. Test the Quad SPI flash boot time for 412122 bytes

3. Table 5 shows the boot time for other boot devices.

Table 5. The boot time for different boot devices with different sizes

Boot devices	Image size	Boot time	Image size	Boot time
Quad SPI flash	8626 bytes	18.9 ms	412122 bytes	26.3 ms

Table 5. The boot time for different boot devices with different sizes*...continued*

Boot devices	Image size	Boot time	Image size	Boot time
FlexSPI NAND flash	8799 bytes	22.89 ms	368862 bytes	41.8 ms
eMMC 4 bits	8806 bytes	247 ms	368870 bytes	263 ms
eMMC 8 bits	8806 bytes	243 ms	368870 bytes	251 ms
SD card	8806 bytes	129 ms	368866 bytes	140.5 ms
LPSPI flash	12118 bytes	22.9 ms	132142 bytes	60.5 ms

5 Conclusion

This document outlines the following details:

- i.MX RT1180 system boot
- i.MX RT1180 booting from the different boot devices
- Use of MCUXpresso secure provisioning tool
- Method to test the boot time from different boot devices

For more details, see *i.MX RT1180 Reference Manual* (document IMXRT1180RM).

6 Acronyms

Table 6 lists the acronyms used in this document.

Table 6. Acronyms

Term	Description
AES	Advanced encryption standard
AHAB	Advanced high assurance boot
CBC	Cipher block chaining
DCD	Device configuration data
EEPROM	Electrically erasable programmable read-only memory
eMMC	embedded multi-media card
ESC	EtherCAT slave controller
GPIO	General-purpose input/output
IEE	Inline encryption engine
LPSPI	Low-power serial peripheral interface
LPUART	Low-power universal asynchronous receiver/transmitter
OTFAD	On-the-Fly AES decryption
POR	Power reset
PSRAM	Pseudostatic random-access memory
RAM	Random-access memory
SDK	Software development kit
SDRAM	Synchronous dynamic random-access memory
SEMC	Smart external memory controller

Table 6. Acronyms...*continued*

Term	Description
SPI	Serial peripheral interface
TSN	Time sensitive networking
UART	Universal asynchronous receiver/transmitter
XIP	Execute-in-place
XMCD	External memory configuration data

7 References

Table 7 lists the references used to supplement this document.

Table 7. Related documentation/resources

Document	Link/how to access
i.MX RT1180 Reference Manual (document IMXRT1180RM)	IMXRT1180RM

8 Note about the source code in the document

Sample code shown in this document has the following copyright and BSD-3-Clause license:

Copyright 2025 NXP Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials must be provided with the distribution.
3. Neither the name of the copyright holder nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

9 Revision history

Table 8 summarizes the revisions to this document.

Table 8. Revision history

Document ID	Release date	Description
AN14589 v.1.0	04 March 2025	Initial public release

Legal information

Definitions

Draft — A draft status on a document indicates that the content is still under internal review and subject to formal approval, which may result in modifications or additions. NXP Semiconductors does not give any representations or warranties as to the accuracy or completeness of information included in a draft version of a document and shall have no liability for the consequences of use of such information.

Disclaimers

Limited warranty and liability — Information in this document is believed to be accurate and reliable. However, NXP Semiconductors does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information. NXP Semiconductors takes no responsibility for the content in this document if provided by an information source outside of NXP Semiconductors.

In no event shall NXP Semiconductors be liable for any indirect, incidental, punitive, special or consequential damages (including - without limitation - lost profits, lost savings, business interruption, costs related to the removal or replacement of any products or rework charges) whether or not such damages are based on tort (including negligence), warranty, breach of contract or any other legal theory.

Notwithstanding any damages that customer might incur for any reason whatsoever, NXP Semiconductors' aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the Terms and conditions of commercial sale of NXP Semiconductors.

Right to make changes — NXP Semiconductors reserves the right to make changes to information published in this document, including without limitation specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

Suitability for use — NXP Semiconductors products are not designed, authorized or warranted to be suitable for use in life support, life-critical or safety-critical systems or equipment, nor in applications where failure or malfunction of an NXP Semiconductors product can reasonably be expected to result in personal injury, death or severe property or environmental damage. NXP Semiconductors and its suppliers accept no liability for inclusion and/or use of NXP Semiconductors products in such equipment or applications and therefore such inclusion and/or use is at the customer's own risk.

Applications — Applications that are described herein for any of these products are for illustrative purposes only. NXP Semiconductors makes no representation or warranty that such applications will be suitable for the specified use without further testing or modification.

Customers are responsible for the design and operation of their applications and products using NXP Semiconductors products, and NXP Semiconductors accepts no liability for any assistance with applications or customer product design. It is customer's sole responsibility to determine whether the NXP Semiconductors product is suitable and fit for the customer's applications and products planned, as well as for the planned application and use of customer's third party customer(s). Customers should provide appropriate design and operating safeguards to minimize the risks associated with their applications and products.

NXP Semiconductors does not accept any liability related to any default, damage, costs or problem which is based on any weakness or default in the customer's applications or products, or the application or use by customer's third party customer(s). Customer is responsible for doing all necessary testing for the customer's applications and products using NXP Semiconductors products in order to avoid a default of the applications and the products or of the application or use by customer's third party customer(s). NXP does not accept any liability in this respect.

Terms and conditions of commercial sale — NXP Semiconductors products are sold subject to the general terms and conditions of commercial sale, as published at https://www.nxp.com/profile/terms, unless otherwise agreed in a valid written individual agreement. In case an individual agreement is concluded only the terms and conditions of the respective agreement shall apply. NXP Semiconductors hereby expressly objects to applying the customer's general terms and conditions with regard to the purchase of NXP Semiconductors products by customer.

Export control — This document as well as the item(s) described herein may be subject to export control regulations. Export might require a prior authorization from competent authorities.

Suitability for use in non-automotive qualified products — Unless this document expressly states that this specific NXP Semiconductors product is automotive qualified, the product is not suitable for automotive use. It is neither qualified nor tested in accordance with automotive testing or application requirements. NXP Semiconductors accepts no liability for inclusion and/or use of non-automotive qualified products in automotive equipment or applications.

In the event that customer uses the product for design-in and use in automotive applications to automotive specifications and standards, customer (a) shall use the product without NXP Semiconductors' warranty of the product for such automotive applications, use and specifications, and (b) whenever customer uses the product for automotive applications beyond NXP Semiconductors' specifications such use shall be solely at customer's own risk, and (c) customer fully indemnifies NXP Semiconductors for any liability, damages or failed product claims resulting from customer design and use of the product for automotive applications beyond NXP Semiconductors' standard warranty and NXP Semiconductors' product specifications.

HTML publications — An HTML version, if available, of this document is provided as a courtesy. Definitive information is contained in the applicable document in PDF format. If there is a discrepancy between the HTML document and the PDF document, the PDF document has priority.

Translations — A non-English (translated) version of a document, including the legal information in that document, is for reference only. The English version shall prevail in case of any discrepancy between the translated and English versions.

Security — Customer understands that all NXP products may be subject to unidentified vulnerabilities or may support established security standards or specifications with known limitations. Customer is responsible for the design and operation of its applications and products throughout their lifecycles to reduce the effect of these vulnerabilities on customer's applications and products. Customer's responsibility also extends to other open and/or proprietary technologies supported by NXP products for use in customer's applications. NXP accepts no liability for any vulnerability. Customer should regularly check security updates from NXP and follow up appropriately.

Customer shall select products with security features that best meet rules, regulations, and standards of the intended application and make the ultimate design decisions regarding its products and is solely responsible for compliance with all legal, regulatory, and security related requirements concerning its products, regardless of any information or support that may be provided by NXP.

NXP has a Product Security Incident Response Team (PSIRT) (reachable at PSIRT@nxp.com) that manages the investigation, reporting, and solution release to security vulnerabilities of NXP products.

NXP B.V. — NXP B.V. is not an operating company and it does not distribute or sell products.

Trademarks

Notice: All referenced brands, product names, service names, and trademarks are the property of their respective owners.

NXP — wordmark and logo are trademarks of NXP B.V.

Contents