

1 介绍

i.MX RT系列是恩智浦推出的业界首款跨界处理器。本文介绍如何将可启动映像编程到恢复用闪存芯片中，并使i.MX RT能够从这个恢复闪存芯片启动。

ROM中的引导加载程序支持恢复启动功能，即在主启动映像因异常行为而损坏时，例如，在映像升级期间出现问题从而破坏了启动映像，将芯片恢复到某种特定状态的选项。

该版本包括PC主机的blhost命令行应用程序。此应用程序可在开发阶段将应用程序下载到闪存芯片中。此版本还包括elftosb命令行应用程序。它可针对i.MXRT 600的ROM生成可启动映像文件。

本文中使用的示例软件基于i.MXRT 685 SDK 2.6.0。开发环境为IAR Embedded Workbench 8.40.2。硬件开发环境为X-IMXRT 685-EVK (Rev.E)。

2 i.MXRT600启动概述

2.1 启动功能

内部ROM存储器用于存储启动代码。重启后，Arm®处理器从此存储器开始执行代码。每次芯片上电或复位时，都会执行引导加载程序代码。

由于i.MX RT600没有内部闪存用于代码和数据存储，映像必须存储在其他地方，以便在重启时加载，或者CPU可以直接从外部存储器执行（XIP）。映像可以从外部闪存加载到内置SRAM中，也可以通过串行端口（UART、SPI、I²C、USB）下载。然后会验证代码，启动ROM再跳转到内置SRAM。

根据OTP位和ISP引脚的值以及映像头部的类型定义，引导加载程序决定是否将代码下载到内置SRAM，或是从外部存储器运行。引导加载程序首先检查OTP位的设置，然后检查ISP引脚。如果OTP字BOOT_CFG[0]中的位[3:0]未编程（4b'0000），则启动源由ISP启动引脚（PIO1_15、PIO1_16和PIO1_17）的状态来确定。

2.2 启动设置

如果OTP中的PRIMARY_BOOT_SRC位未设置，则i.MX RT600将读取ISP引脚的状态来确定启动源。

表1. 基于ISP引脚的启动模式和ISP下载器模式

启动模式	ISP2引脚 PIO1_17	ISP1引脚 PIO1_16	ISP0引脚 PIO1_15	说明
—	低	低	低	保留

下一页继续.....

目录

1 介绍.....	1
2 i.MXRT600启动概述.....	1
3 恢复启动模式.....	5
4 MIMXRT685 EVK评估板设置.....	8
5 编程工具.....	9



表1. 基于ISP引脚的启动模式和ISP下载器模式 (续)

启动模式	ISP2引脚 PIO1_17	ISP1引脚 PIO1_16	ISP0引脚 PIO1_15	说明
SDIO0 (SD卡)	低	低	高	从连接到SDIO 0接口的SD卡芯片启动。i.MXRT600将在SD卡芯片中查找有效映像。如果没有找到有效的映像，i.MXRT600将根据OTP DEFAULT_ISP_MODE位 (6 : 4 , BOOT_CFG[0]) 进入ISP启动模式。
端口B的FlexSPI 启动	低	高	低	从连接到FlexSPI接口0端口B的四线或八线SPI闪存芯片启动。i.MXRT600将在外接四线/八线SPI闪存芯片中查找有效映像。如果没有找到有效的映像，i.MXRT600将进入ISP启动模式。
端口A的FlexSPI 启动	低	高	高	从连接到FlexSPI接口0端口A的四线/八线SPI闪存芯片启动。i.MXRT600将在外接四线/八线SPI闪存芯片中查找有效映像。如果没有找到有效的映像，i.MXRT600将进入ISP启动模式。
SDIO 0 (eMMC)	高	低	低	从连接到SDIO 0接口的SD卡设备启动。i.MXRT600将在SD卡芯片中查找有效映像。如果没有找到有效的映像，i.MXRT600将根据OTP DEFAULT_ISP_MODE位 (6 : 4 , BOOT_CFG[0]) 进入ISP启动模式。
USB DFU (主器件启动)	高	低	高	采用USB DFU类设备通过USB高速端口将启动镜像下载到内置SRAM。
串行ISP (UART、SPI、 I ² C、USB-HID)	高	高	低	采用串行接口 (UART、SPI和I ² C、USB-HID) 对OTP、外接闪存、SD或eMMC设备进行编程。
串行主器件启动 (UART、SPI、 I ² C、USB-HID)	高	高	高	串行主器件启动 (SPI Slave、I ² C Slave或UART、USB-HID) 用于通过串口 (SPI Slave、I ² C Slave或UART、USB-HID) 下载启动镜像。

2.3 启动映像偏移量

引导加载程序从启动介质上的指定偏移量查找启动映像。详见表2。

表2. 不同启动介质上的映像偏移量

启动介质	映像偏移量
FlexSPI启动（串行NOR或闪存芯片）	0x1000
SD启动（SD卡）	0x1000
eMMC启动（eMMC存储器）	0x1000
恢复启动（SPI或闪存芯片）	0x1000

2.4 启动映像的文件头

一旦确定了启动模式，并且所选外部存储器（SD、eMMC或串行NOR闪存）上的启动映像时可用的，则ROM引导加载程序会开始将外接存储器中映像文件头部的64个字节复制到内置SRAM中。映像的开头遵循表3中提到的格式。

表3. 映像文件头格式

偏移量	字段	说明
0x00 - 0x1F	保留	—
0x20	imageLength	映像长度
0x24	imageType	映像类型 0x0000 - 明文映像 0x0001 - 明文签名映像 0x0002 - 明文CRC映像 0x0003 - 加密签名映像 0x0004 - 明文签名XIP映像 0x0005 - 明文CRC XIP映像 0x8001 - 包含KeyStore的明文签名映像 0x8003 - 包含KeyStore的加密签名映像
0x28	authBlockOffset/ crcChecksum	验证块偏移量或CRC32校验和
0x2C - 0x33	保留	—
0x34	imageLoadAddress	映像加载地址
0x38-0x3F	保留	—

引导加载程序通过检查偏移量0x24（imageType）处的映像类型，开始扫描用户映像。如果检测到有效的映像类型，则开始验证映像文件头。通过读取映像文件头中偏移量0x34处的映像加载地址（imageLoadAddress），并将其用作指向有效映像文件头结构的指针，继续审核映像文件头。如果imageType和imageLoadAddress均为非零，则imageLoadAddress指向的地址必须包含正在检查的映像文件头。

完成映像文件头的验证之后，通过检查映像类型字段来继续审核此文件头。如果可启动(非XIP)映像驻留在外部闪存中，则整个映像将被加载到内置SRAM中，然后映像文件头中的imageLength字段将被用作映像长度来执行CRC校验，用以确定是否启用了CRC校验功能。

2.5 串行ISP启动

i.MXRT600包含系统内编程 (ISP) 功能。引导加载程序提供了闪存编程工具，可通过MCU上的串行连接来操作。它使MCU在整个产品生命周期 (包括应用程序开发、最终产品制造等) 中的编程都非常简单快捷。主机端命令行和GUI工具可用于与引导加载程序通信。用户可以利用主机工具读取和烧写应用程序代码，并通过引导加载程序进行生产制造。

以下是ISP功能简介：

- 支持UART、SPI、I2C和USB外设接口。
- 自动检测活跃外设。
- 烧写OTP。
- 烧写串行NOR闪存。
- 烧写SD卡。
- 烧写eMMC设备。

每种启动芯片都有唯一的配置选项块。它指示闪存芯片属性。配置选项块应传递给主机工具。表4描述了恢复启动芯片的配置选项块。

表4. 恢复启动配置选项块

偏移量	字段	说明	
0x00	选项	[31:28] 标签	必须为0xC
		[27:24] 保留	此字段为保留字段。
		[23:20] SPI索引	Flexcomm SPI索引选择 0000 - SPI0 0001 - SPI1 0010 - SPI2 0011 - SPI3 0100 - SPI4 0101 - SPI5 0110 - SPI6 0111 - SPI7 其他保留
		[19:16] 保留	此字段为保留字段。
		[15:12] 存储器类型	存储器类型选择 0000 - 手动NOR闪存 0010 - SFDP NOR闪存 其他保留

下一页继续.....

表4. 恢复启动配置选项块 (续)

偏移量	字段	说明
	[11:8] 内存大小	0 ≤ n ≤ 11, 大小 = 512 KBytes * 2 ⁿ 12 ≤ n ≤ 15, 大小 = 32 KBytes * 2 ⁽ⁿ⁻¹²⁾
	[7:4] 扇区大小	0 ≤ n ≤ 1, 大小 = 4 KBytes * 2 ⁿ 2 ≤ n ≤ 5, 大小 = 32 KBytes * 2 ⁽ⁿ⁻²⁾ 其他保留
	[3:0] 页面大小	0 ≤ n ≤ 2, size = 256 Bytes * 2 ⁿ 3 ≤ n ≤ 5, size = 32 Bytes * 2 ⁽ⁿ⁻³⁾ 其他保留

3 恢复启动模式

在 i.MXRT600 中, 支持 SPI NOR 芯片作为恢复介质。

3.1 设计引脚分配

可以将 SPI0-SPI7 中的一个指定为连接 SPI NOR 芯片的接口。

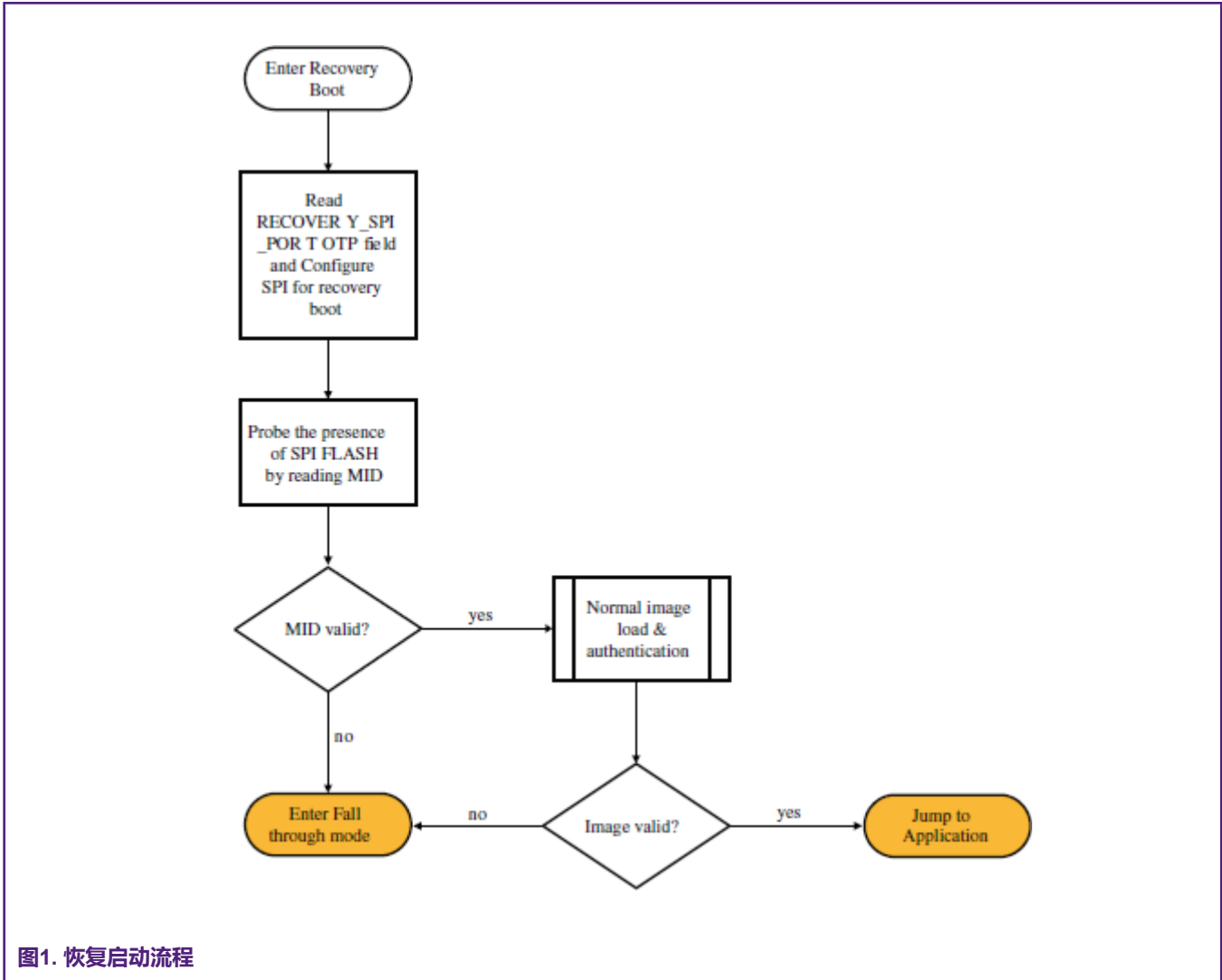
表 5 列出了 SPI 外设的引脚列表。

表5. 恢复启动SPI闪存引脚分配

启动接口	引脚	功能
SPI闪存	FC0	使用Flexcomm0引脚PIO0_0 (SCK)、PIO0_1 (MISO)、PIO0_2 (MOSI)和PIO0_3 (SSEL)
	FC1	使用Flexcomm1引脚PIO0_7 (SCK)、PIO0_8 (MISO)、PIO0_9 (MOSI)和PIO0_10 (SSEL)
	FC2	使用Flexcomm2引脚PIO0_14 (SCK)、PIO0_15 (MISO)、PIO0_16 (MOSI)和PIO0_17 (SSEL)
	FC3	使用Flexcomm3引脚PIO0_21 (SCK)、PIO0_22 (MISO)、PIO0_23 (MOSI)和PIO0_24 (SSEL)
	FC4	使用Flexcomm4引脚PIO0_28 (SCK)、PIO0_29 (MISO)、PIO0_30 (MOSI)和PIO0_31 (SSEL)
	FC5	使用Flexcomm5引脚PIO1_3 (SCK)、PIO1_4 (MISO)、PIO1_5 (MOSI)和PIO1_6 (SSEL)
	FC6	使用Flexcomm6引脚PIO3_25 (SCK)、PIO3_26 (MISO)、PIO3_27 (MOSI)和PIO3_28 (SSEL)
	FC7	使用Flexcomm7引脚PIO4_0 (SCK)、PIO4_1 (MISO)、PIO4_2 (MOSI)和PIO4_3 (SSEL)

3.2 恢复启动流程

如果主器件启动失败并且启用了恢复启动, 则引导加载程序进入恢复启动模式。



当恢复启动过程开始时，引导加载程序使用IRC48M中的24 MHz时钟，通过检查制造商ID来探测SPI NOR芯片的存在。一旦检测到，引导加载程序将继续将恢复映像从SPI NOR芯片加载到内置SRAM，使用24 MHz时钟，并对映像执行完整性检查/映像身份验证。

如果完整性检查/身份验证通过，引导加载程序将跳转到恢复启动映像。否则，它将转入ISP模式。

3.3 映像链接区

恢复映像只能是非XIP映像。它应该链接到内置的4.5 MB SRAM上。由于前512KB的SRAM已经被ROM和DSP占用，所以最好从0x80000开始链接恢复映像。

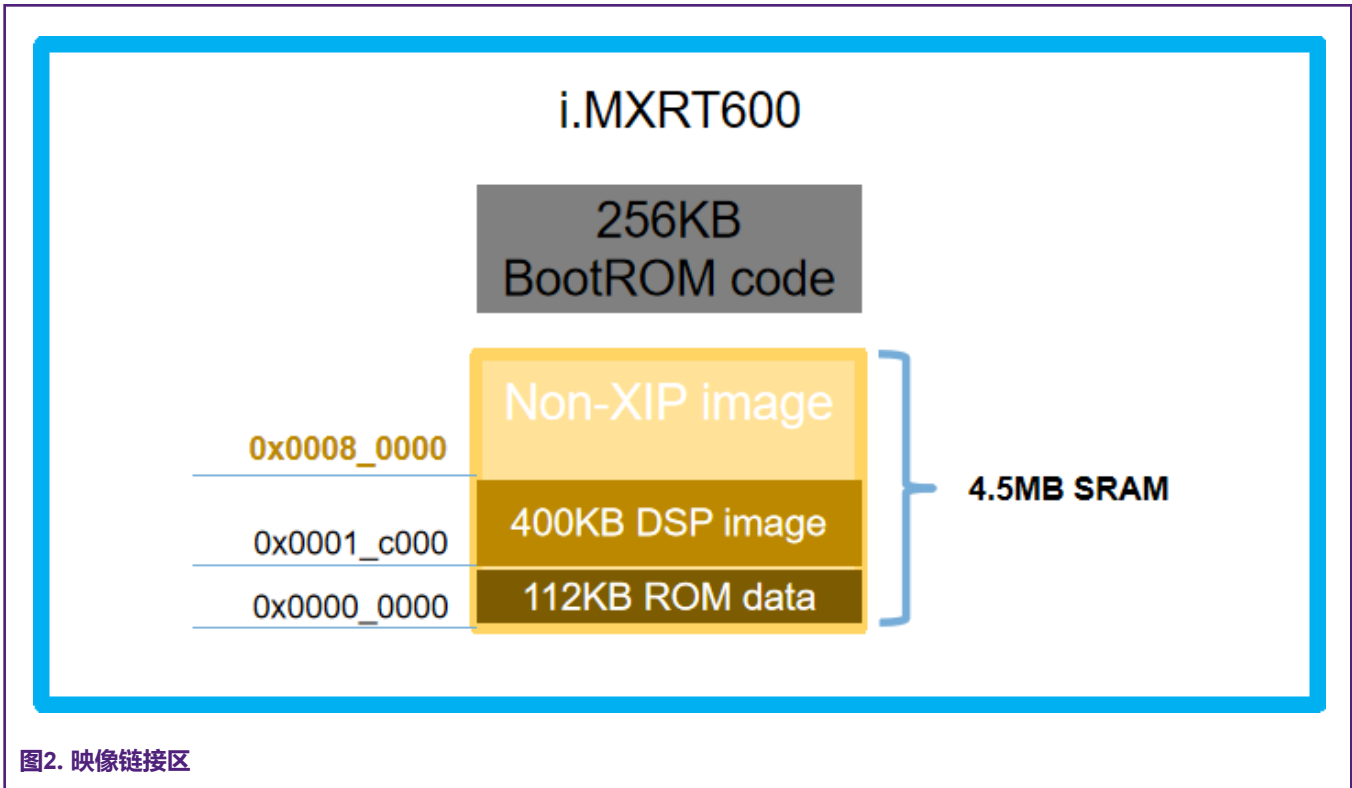


图2. 映像链接区

3.4 恢复启动的OTP设置

有两个与恢复启动相关的字段。这两个字段都分配在BOOT_CFG0 OTP字处。

- PRIMARY_BOOT_SRC从偏移量0开始，4位宽度。详见表6。

表6. 恢复启动的OTP字段

PRIMARY_BOOT_SRC	字段	主启动源（又名主器件启动源）
SPI_SLV_BOOT	b'0111	通过SPI接口从1位NOR闪存启动，所使用的SPI实例由熔丝字0x60第17位至第19位选择，详情请参阅熔丝映射。
FLEX_SPI_REC_BOOT_PORTB	b'1 011	从FlexSPI0端口B上的八线/四线SPI闪存芯片启动。如果未找到映像，则采用通过FlexComm连接的SPI闪存芯片来检查恢复启动。
FLEX_SPI_REC_BOOT_PORTA	b'1100	从FlexSPI0端口A上的八线/四线SPI闪存芯片启动。如果未找到映像，则采用通过FlexComm连接的SPI闪存芯片来检查恢复启动。
SDHC0_REC_BOOT	b'1101	从SDHC0端口芯片启动。如果未找到映像，则采用通过FlexComm连接的SPI闪存芯片来检查恢复启动。
SDHC1_REC_BOOT	b'1110	从SDHC1端口芯片启动。如果未找到映像，则采用通过FlexComm连接的SPI闪存芯片来检查恢复启动。

- REDUNDANT_SPI_PORT，从偏移量17开始，3位宽度。详见表7。

表7. 恢复启动OTP字段

REDUNDANT_SPI_PORT	用于冗余SPI闪存启动的FlexComm端口	值
FC0	使用Flexcomm0引脚PIO0_0 (SCK)、PIO0_1 (MISO)、PIO0_2 (MOSI)和PIO0_3 (SSEL)	3'b000
FC1	使用Flexcomm1引脚PIO0_7 (SCK)、PIO0_8 (MISO)、PIO0_9 (MOSI)和PIO0_10 (SSEL)	3'b001
FC2	使用Flexcomm2引脚PIO0_14 (SCK)、PIO0_15 (MISO)、PIO0_16 (MOSI)和PIO0_17 (SSEL)	3'b010
FC3	使用Flexcomm3引脚PIO0_21 (SCK)、PIO0_22 (MISO)、PIO0_23 (MOSI)和PIO0_24 (SSEL)	3'b011
FC4	使用Flexcomm4引脚PIO0_28 (SCK)、PIO0_29 (MISO)、PIO0_30 (MOSI)和PIO0_31 (SSEL)	3'b100
FC5	使用Flexcomm5引脚PIO1_3 (SCK)、PIO1_4 (MISO)、PIO1_5 (MOSI)和PIO1_6 (SSEL)	3'b101
FC6	使用Flexcomm6引脚PIO3_25 (SCK)、PIO3_26 (MISO)、PIO3_27 (MOSI)和PIO3_28 (SSEL)	3'b110
FC7	使用Flexcomm7引脚P4_0 (SCK)、P4_1 (MISO)、P4_2 (MOSI)和P4_3 (SSEL)	3'b111

4 MIMXRT685 EVK评估板设置

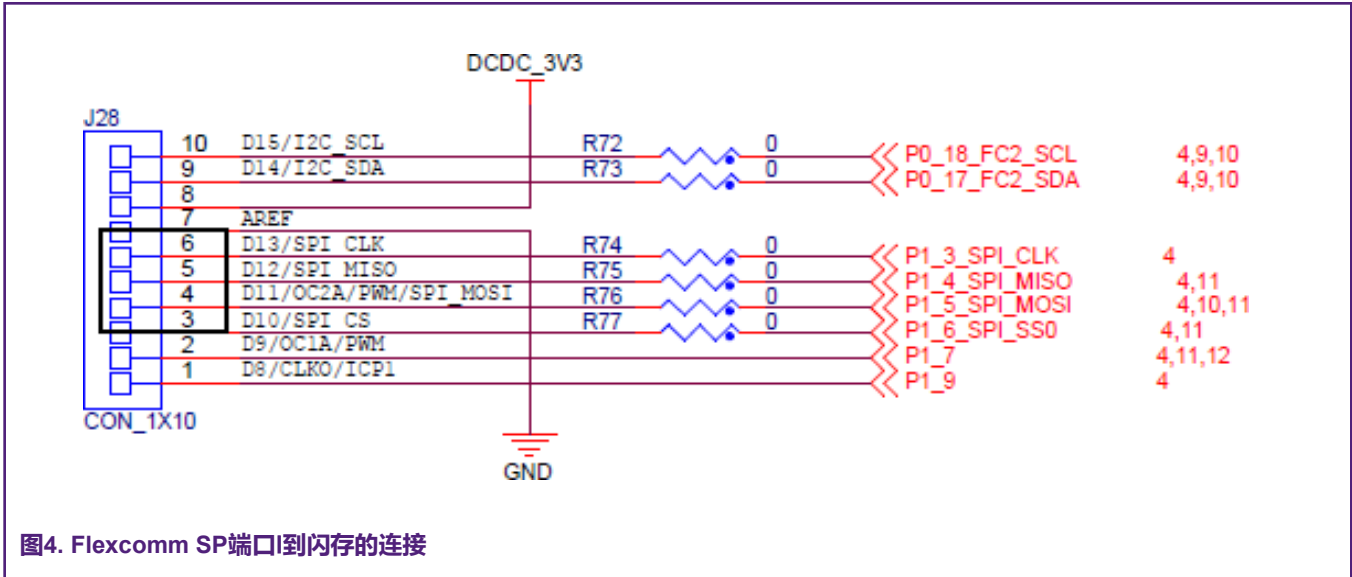
没有闪存连接到EVK评估板上的Flexcomm SPI端口。要启用QSPI NOR闪存的恢复启动功能，需要一个简单的闪存卡。



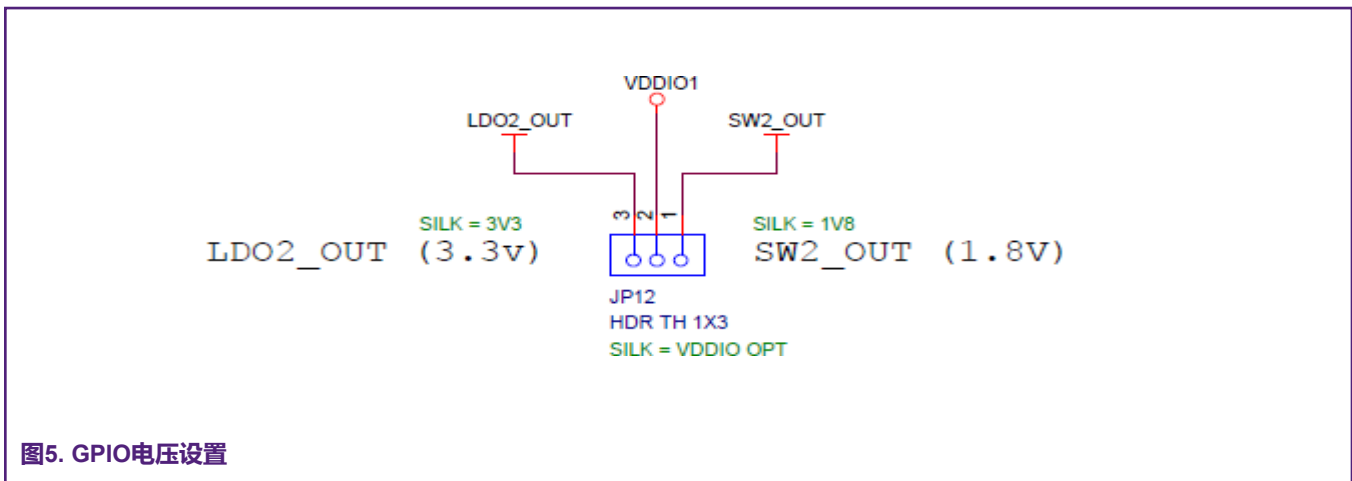
图3. QSPI NOR闪存卡

此存储卡中的是四线SPI NOR闪存。但是，如果将其用作恢复启动芯片，则只需将IO[1:0]、CLK和CS引脚连接到Flexcomm SPI端口。其他IO[3:2]引脚应被驱动至高电平。

选择Flexcomm的SPI5（J28的引脚3-6）连接闪存卡。



由于存储卡中的QSPI NOR闪存芯片的供电电压为3.3V，为了确保i.MXRT600 GPIO能够驱动这个闪存，应该连接JP12的引脚2-3。



5 编程工具

5.1 Blhost工具

Blhost是一个命令行主机程序，可与运行ROM引导加载程序的芯片进行交互。Blhost的版本应为v2.3或更高。

5.2 NXP-MCUBootUtility工具

NXP-MCUBootUtility是一个图形用户界面工具，可与运行ROM引导加载程序的芯片接口。这是一个真正的一站式工具。NXP-MCUBootUtility的版本应为v2.2或更高。

5.3 使用Blhost启用恢复启动

本章介绍了使用blhost工具将映像编程到QSPI NOR闪存以及从QSPI NOR恢复闪存进行启动的步骤。

1. 打开\SDK_2.6.0_EVK-MIMXRT685\boards\evkmimxrt685\driver_examples\gpio\led_output示例，选择项目配置为调试，如图6所示。

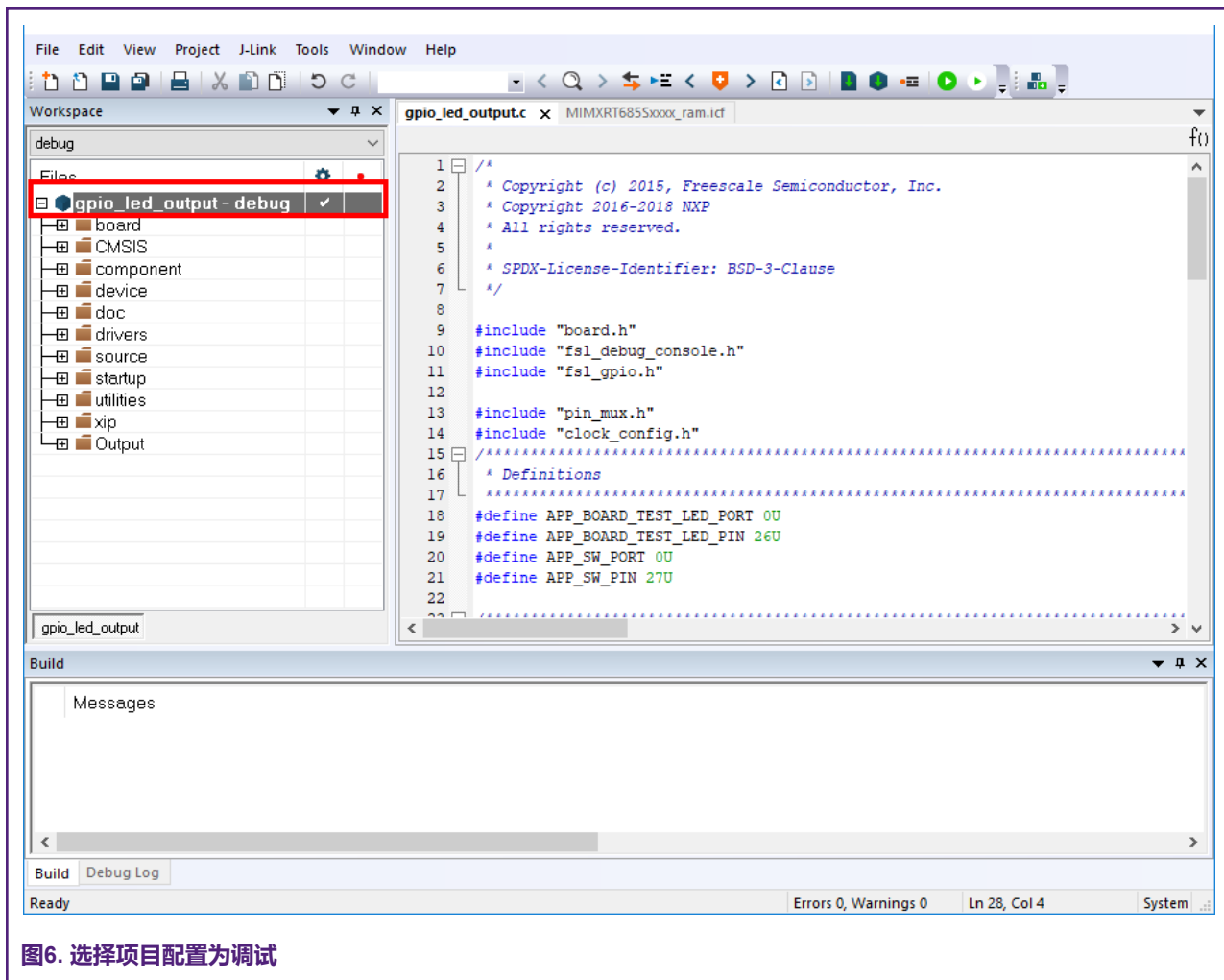


图6. 选择项目配置为调试

2. 构建项目并生成.bin格式的映像。您可以找到如图7所示的gpio_led_output.bin文件。

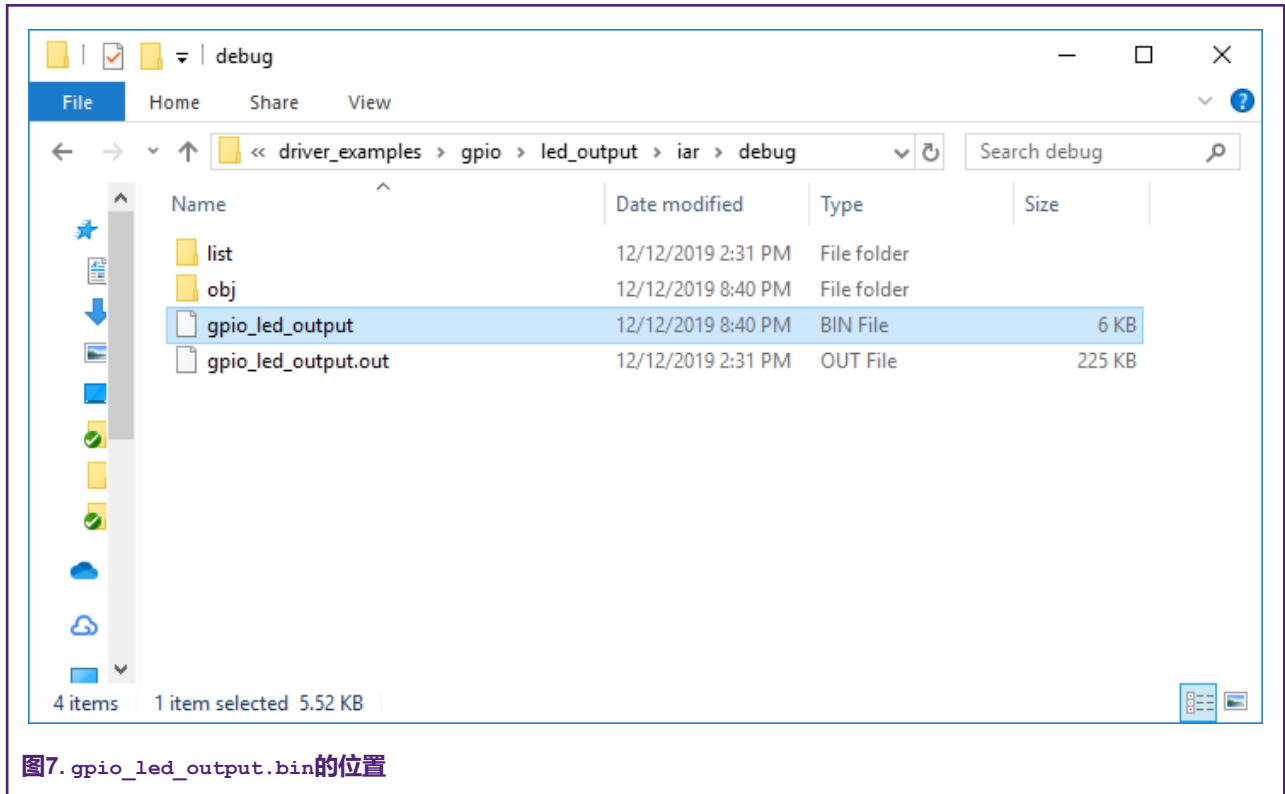


图7. `gpio_led_output.bin`的位置

3. 在`startup_MIMXRT685S_cm33.s`中，根据生成的`gpio_led_output.bin`的大小填写实际映像长度。重新构建项目以获得新的`gpio_led_output.bin`。

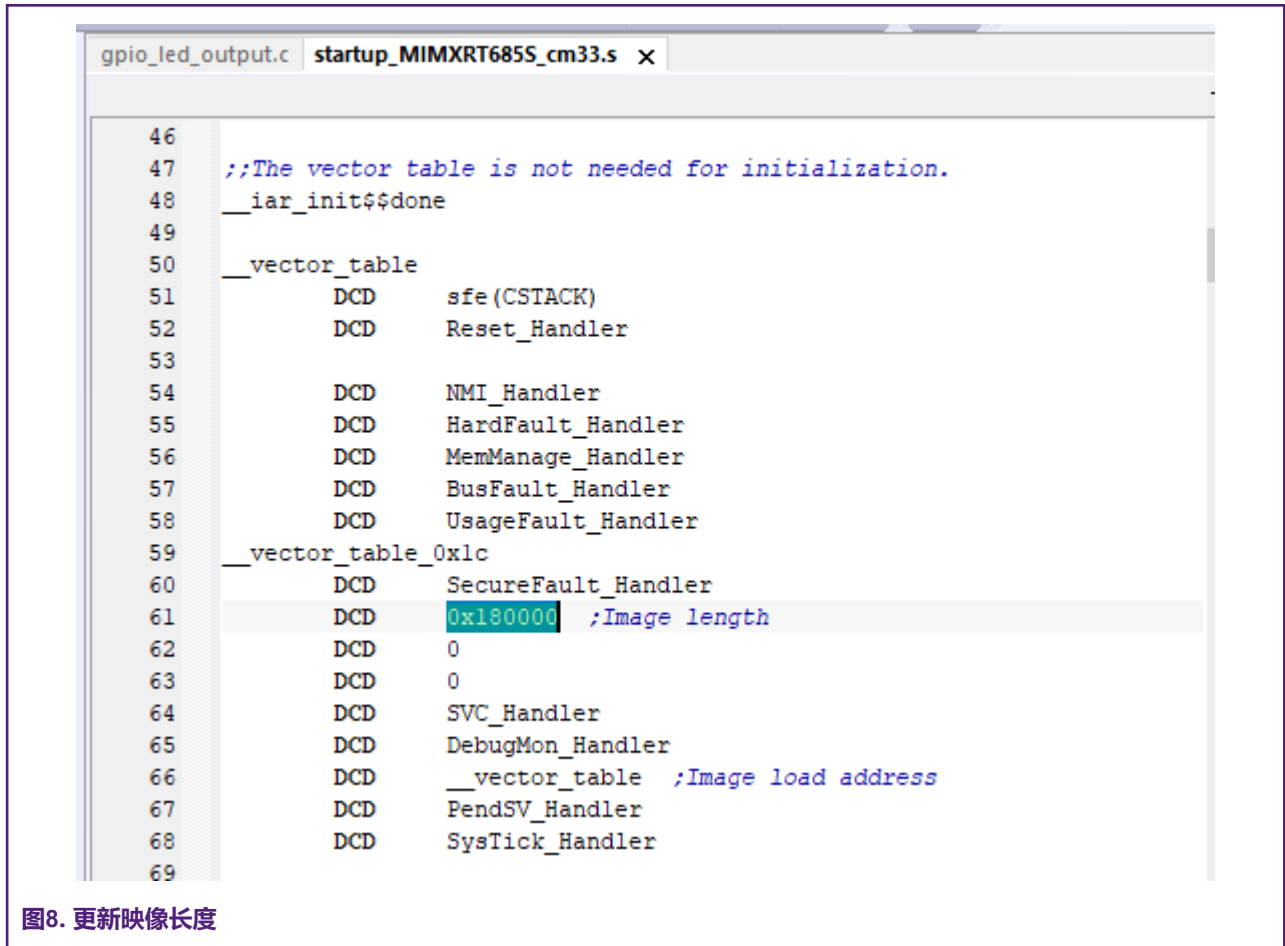


图8. 更新映像长度

4. 将新的gpio_led_output.bin复制到blhost文件夹中，如图9所示。

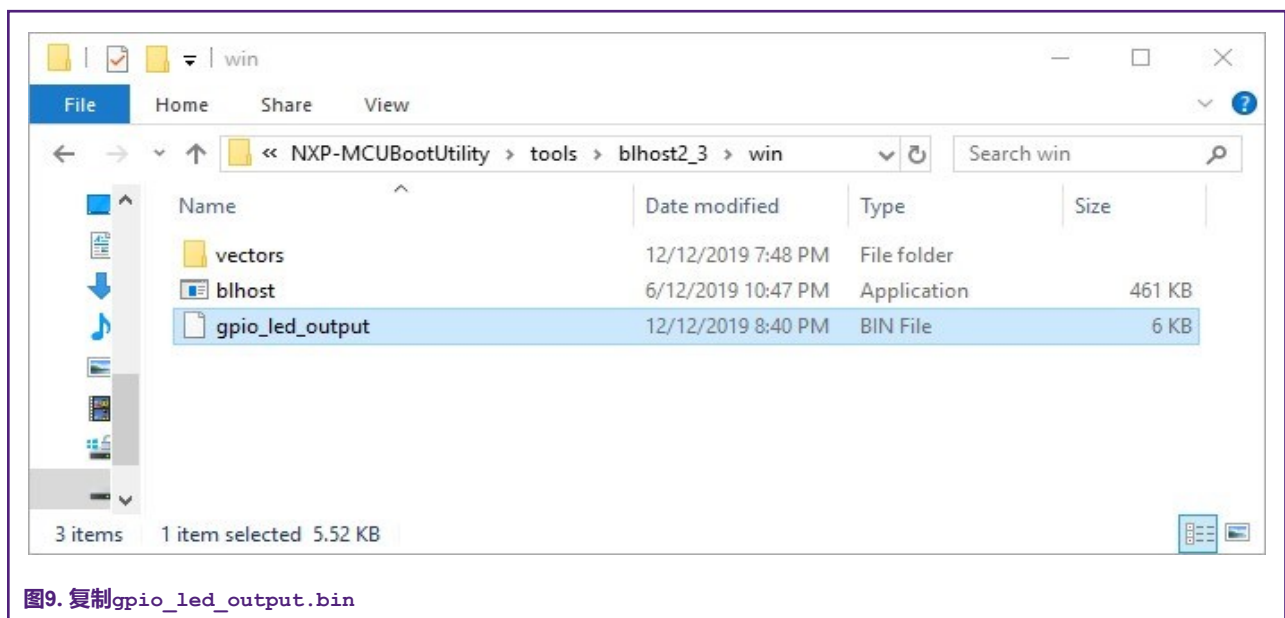


图9. 复制gpio_led_output.bin

5. 将SW5设置为1-ON、2-OFF和3-OFF，可将RT685-EVK评估板切换到串行ISP模式。将USB电缆连接到J7 USB端口并发出blhost命令，如图10所示。

```

Windows PowerShell
PS D:\NXP-MCUBootUtility\tools\blhost2_3\win> .\blhost.exe -u 0x1fc9,0x0020 -- fill-memory 0x1c000 0x4 0xc0500000
Inject command 'fill-memory'
Successful generic response to command 'fill-memory'
Response status = 0 (0x0) Success.
PS D:\NXP-MCUBootUtility\tools\blhost2_3\win> .\blhost.exe -u 0x1fc9,0x0020 -- configure-memory 0x110 0x1c000
Inject command 'configure-memory'
Successful generic response to command 'configure-memory'
Response status = 0 (0x0) Success.
PS D:\NXP-MCUBootUtility\tools\blhost2_3\win> .\blhost.exe -u 0x1fc9,0x0020 -- flash-erase-region 0x1000 0x2000 0x110
Inject command 'flash-erase-region'
Successful generic response to command 'flash-erase-region'
Response status = 0 (0x0) Success.
PS D:\NXP-MCUBootUtility\tools\blhost2_3\win> .\blhost.exe -u 0x1fc9,0x0020 -- write-memory 0x1000 .\gpio_led_output.bin 0x110
Inject command 'write-memory'
Preparing to send 5658 (0x161a) bytes to the target.
Successful generic response to command 'write-memory'
(1/1) 100% Completed!
Successful generic response to command 'write-memory'
Response status = 0 (0x0) Success.
Wrote 5658 of 5658 bytes.
PS D:\NXP-MCUBootUtility\tools\blhost2_3\win> .\blhost.exe -u 0x1fc9,0x0020 -- efuse-program-once 0x60 000a0000
Inject command 'efuse-program-once'
Successful generic response to command 'efuse-program-once'
Response status = 0 (0x0) Success.
PS D:\NXP-MCUBootUtility\tools\blhost2_3\win>

```

图10. blhost命令序列

Fill-Memory命令中的参数值0xc0500000是恢复启动配置选项块。详情请参见表4。

- 发出efuse-program-once 0x60 recoveryBootValue命令，设置OTP BOOT_CFG [0]中的PRIMARY_BOOT_SRC位。recoveryBootValue可以是4b'0111/4b'1011/4b'1100/4b'1101/4b'1110。确保任何主器件启动芯片中都没有有效的映像。复位此评估板，gpio led演示程序将正常运行。

5.4 使用NXP-MCUBootUtility启用恢复启动

本章介绍使用NXP-MCUBootUtility工具将映像编程到QSPI NOR闪存并从QSPI NOR恢复闪存启动的步骤。

- 重建\SDK_2.6.0_EVK-MIMXRT685\boards\evkmimxrt685\driver_examples\gpio\led_output项目并生成.srec格式的映像。
- 将RT685-EVK评估板切换到串行ISP模式，将USB线缆连接到J7 USB端口，然后打开NXP-MCUBootUtility。将MCU芯片设置为i.MXRT6xx，并将Boot Device（启动芯片）设置为FLEXCOMM SPI NOR。单击“Boot Device Configuration（启动芯片配置）”将Spi索引设置为5。单击Connect to ROM（连接到ROM）。

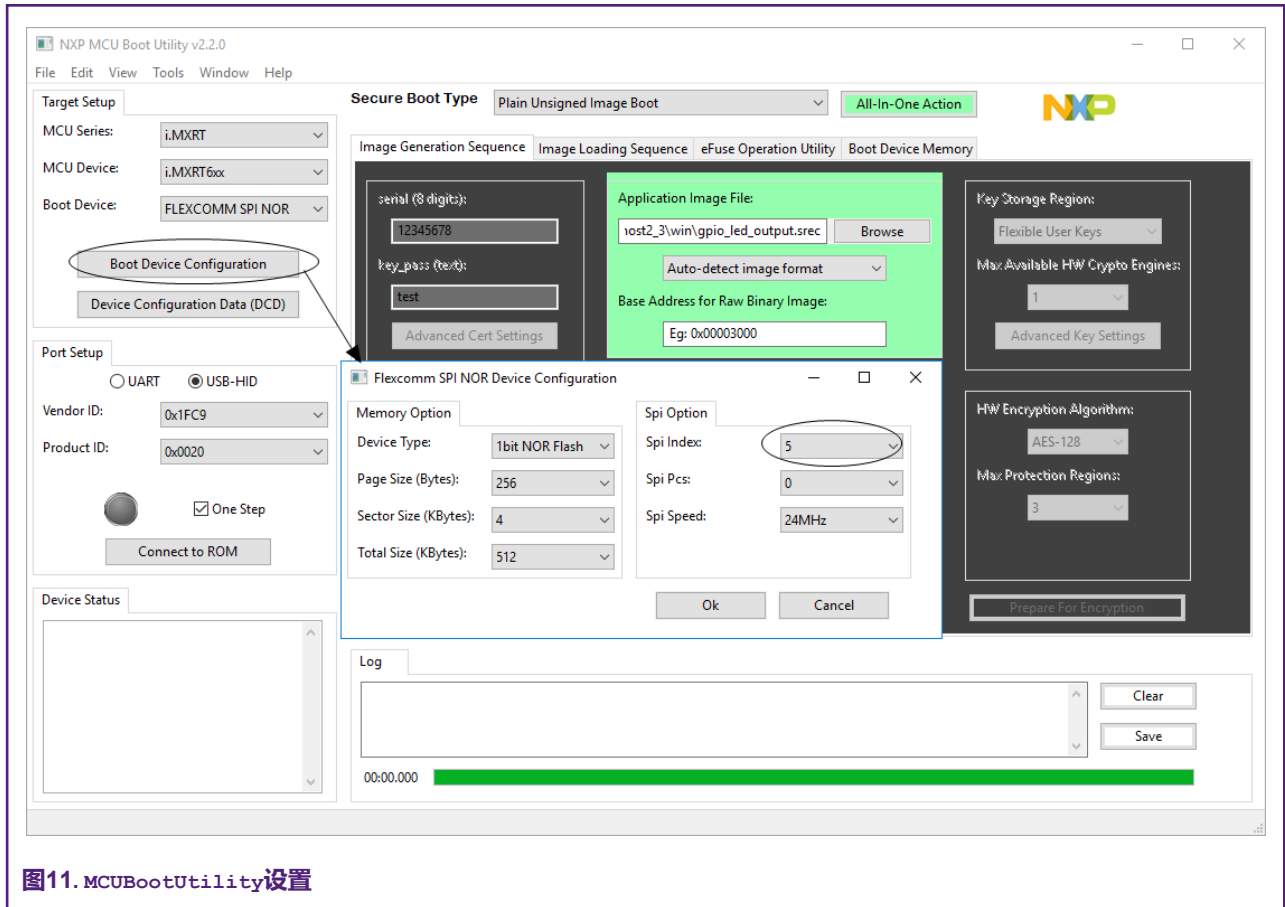


图11. MCUBootUtility设置

3. 如果工具能成功连接RT600 BootROM，则芯片状态面板上会显示芯片信息。浏览gpio_led_output.srec文件，然后单击All-in-One Action。

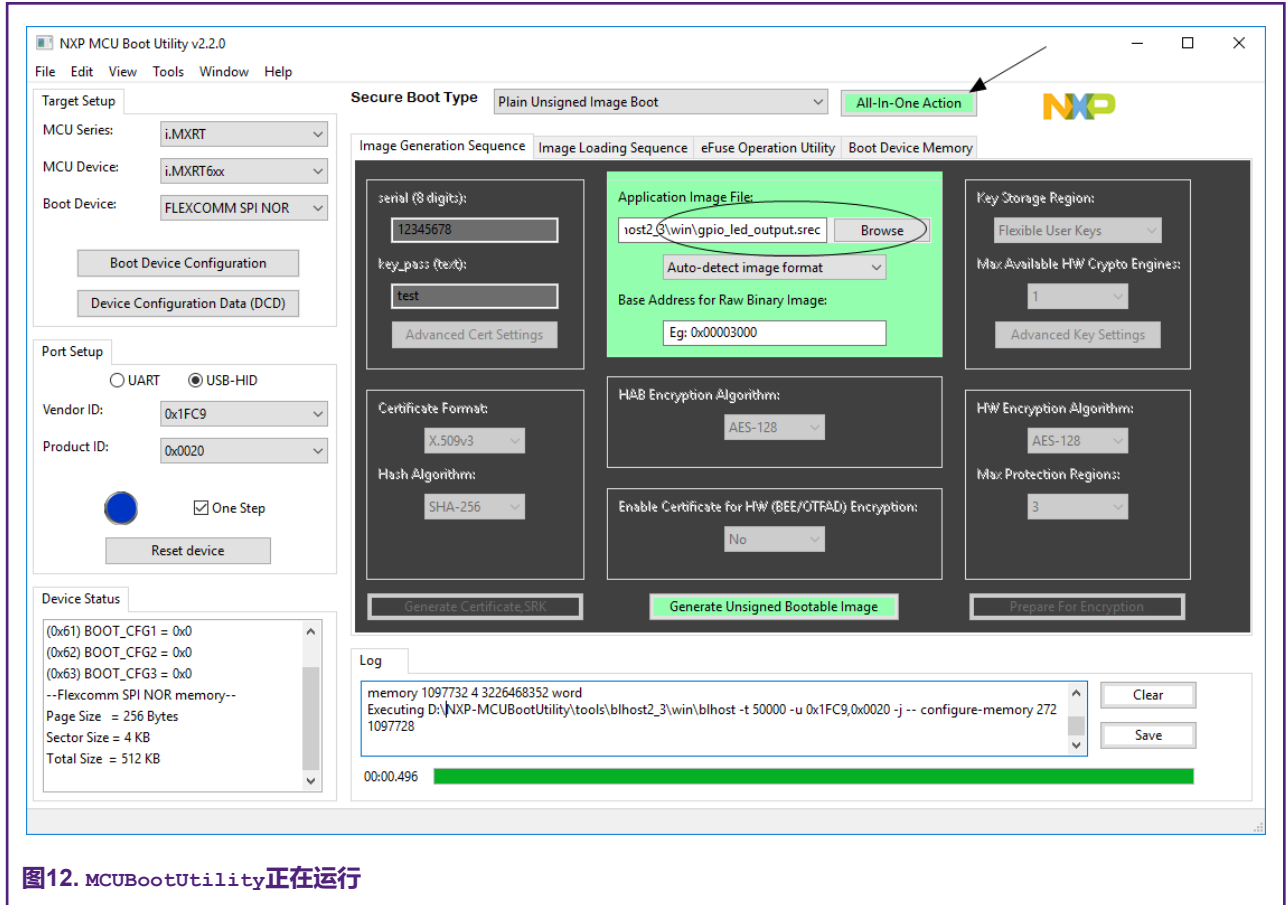


图12. MCUBootUtility正在运行

4. 烧写 `recoveryBootValue` 来设置 OTP `BOOT_CFG [0]` 中的 `PRIMARY_BOOT_SRC` 位。 `recoveryBootValue` 可以是 `4b'0111/4b'1011/4b'1100/4b'1101/4b'1110`。

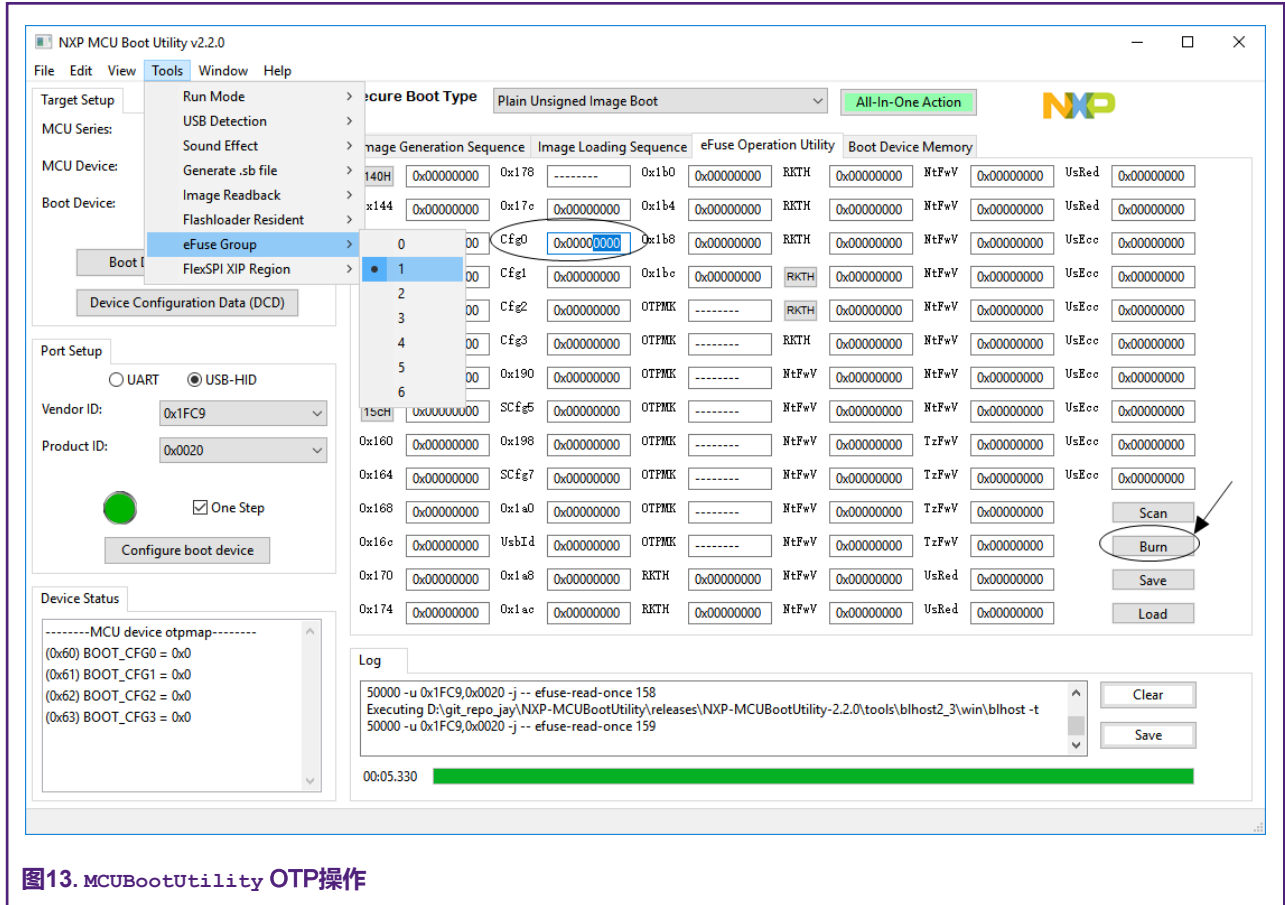


图13. MCUBootUtility OTP操作

5. 确保任何主启动芯片中都没有有效的映像。重启评估板，gpio-led演示程序将正常运行。

How To Reach Us

Home Page:

nxp.com

Web Support:

nxp.com/support

Information in this document is provided solely to enable system and software implementers to use NXP products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits based on the information in this document. NXP reserves the right to make changes without further notice to any products herein.

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

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, µVision, 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.

© NXP B.V. 2020.

All rights reserved.

For more information, please visit: <http://www.nxp.com>

For sales office addresses, please send an email to: salesaddresses@nxp.com

Date of release: 25/02/2020

Document identifier: AN12751

