AN14178

MCXNx4x的Flash命令示例

第1版 — 2024年1月24日 **应用笔记**

文档信息

信息	内容
关键词	AN14178、MCXNx4x、MCX N、MCX N系列、MCXNx4x 的Flash命令控制器（Flash Command Controller）、Flash IAP、Flash编程、Arm Cortex-M33、通用MCU
摘要	本文档介绍了如何使用Flash命令控制器来执行Flash的读写操作；这比调用ROM API更高效。

1 介绍

本文档介绍了如何使用Flash命令控制器执行Flash的读写操作；这比调用ROM API更高效。在一些复杂的应用中，需要进行无阻塞的Flash操作。然而，命令写入序列可能更难以使用。本文档旨在对如何使用命令写入序列在MCXNx4x上烧录内部Flash进行介绍。

2 概述

该过程遵循一种通用的Flash命令写入序列，如图1所示。

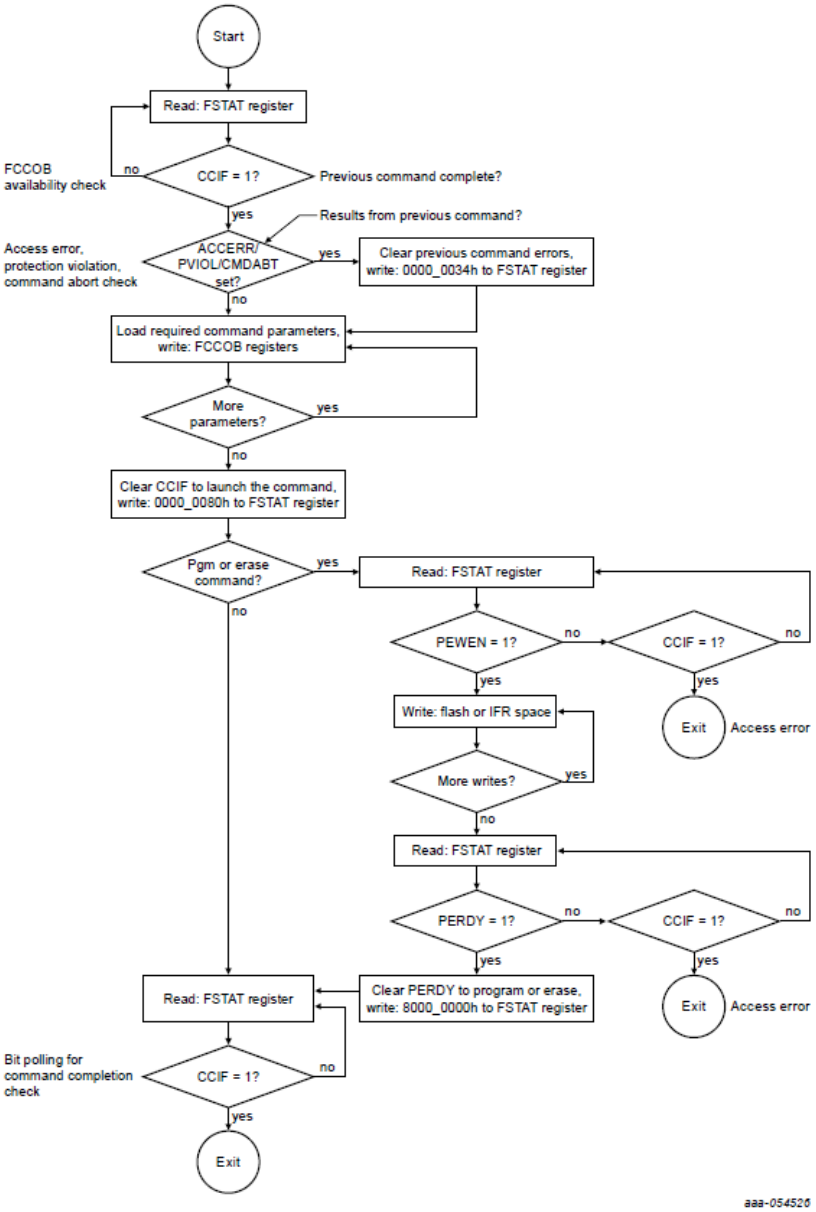

图1. 命令写入序列

2.1 综述

以下列出了所使用的步骤：

1. 初始化必要的时钟和寄存器。
2. 使用擦除扇区（erase sector）命令，一次性擦除内部Flash的一个扇区即8192字节，范围从0x10_0000到0x1F_FFFF。
3. 使用编程页面（program page）命令，一次编程一个页面即128字节，范围从0x10_0000到0x1F_FFFF。
4. 验证存储的值是否与预期值相匹配。
5. 此外，在每个命令之间，检查FSTAT寄存器进行错误的处理，并等待CCIF置位，然后再继续下一个命令。

有关更多详细信息，请参见图2。

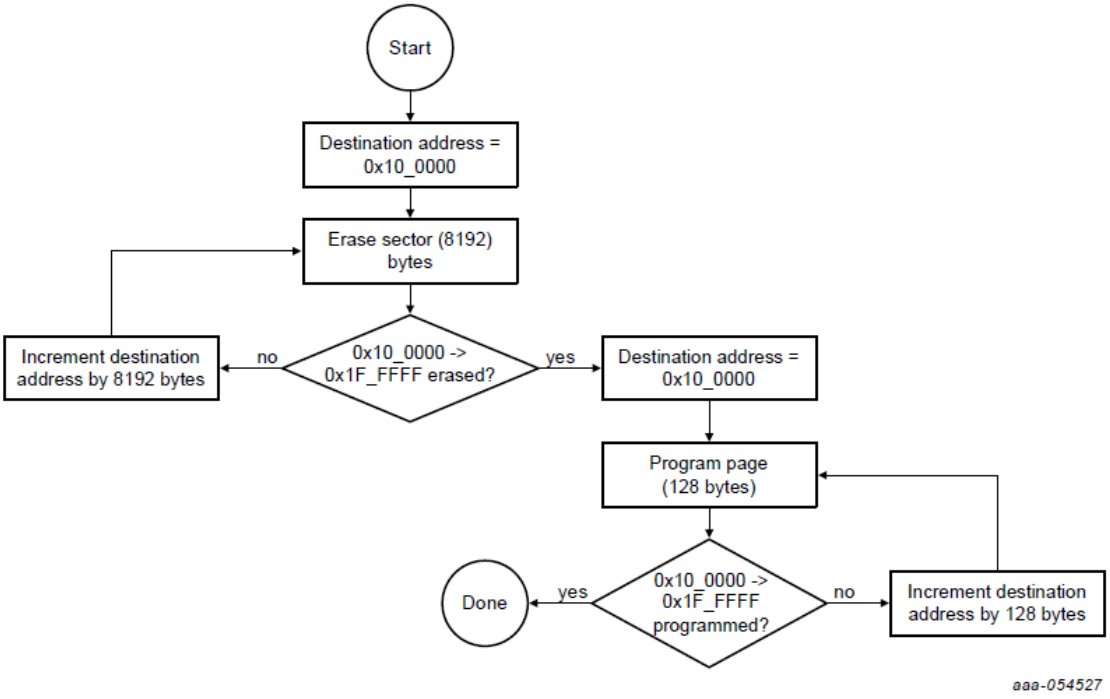

图2. Flash的擦除——写入流程

3 使用示例

这里提供了一个用例，其中包括一个MCUXpresso工程。该工程可擦除并烧录Flash的后半部分，大小为1MB。该示例可在本应用笔记的相关软件包中获得。

如第1节所述，此过程遵循通用的命令写入序列。以下小节重点介绍了本示例中使用的命令。

3.1 擦除扇区命令

这些步骤展示了擦除一个扇区的8192字节的过程。对于本示例工程，该过程会重复进行，直到Flash的整个后半部分被擦除。且它以一个目标地址destAdrss=0x10_0000开始，这是Flash后半部分的第一个索引。

1. 检查FMU FSTAT寄存器，确保CCIF已置位。即上一条命令已完成。

```
if (((FMU0->FSTAT & FMU_FSTAT_CCIF(1)) >> FMU_FSTAT_CCIF_SHIFT) == 1)
```

```
{
  //continue with programming
}
```

如果CCIF寄存器没有置位，那么就不能继续操作，必须等待上一个操作完成，然后才能开始执行下一个Flash控制器命令。在示例代码中，使用了一个while循环来等待CCIF寄存器完成置位。然而，开发人员应考虑应用程序是否需要并行运行其他任务。

2. 处理并清除FMU FSTAT寄存器中存在的所有错误标志。

```
//clear previous errors
FMU0->FSTAT = 0x34;
```

FSTAT_CLEARERR的值为0x34。

3. 通过将FMU FCCOB[0]设置为0x42 (ERSSCR)，指定命令为“擦除扇区”（erase sector）。

```
//42h is erase sector command ERSSCR
//specify command
FMU0->FCCOB[0] = 0x42;
```

4. 清除CCIF寄存器以启动命令。

```
//clear ccif to launch
FMU0->FSTAT = 0x80;
```

FSTAT_CLEARCCIF的值为0x80。这会向FSTAT[CCIF]位写入1，从而清除寄存器。

5. 检查FMU FSTAT PEWEN == 1，允许写入一个词组。

```
if (((FMU0->FSTAT & FMU_FSTAT_PEWEN(value)) >> FMU_FSTAT_PEWEN_SHIFT) == 1)
{
//continue
}
```

在FSTAT PEWEN等于1之前，无法继续执行擦除扇区命令的操作。在示例代码中，使用了一个while循环来等待PEWEN寄存器完成设置。然而，开发人员应考虑应用程序是否需要并行运行其他任务。

6. 将四个连续字节写入Flash，第一次写入必须与词组或扇区对齐。

***注：**这些写入的内容并不重要，因为该扇区将被擦除，但必须执行四次连续写入的操作，以便根据擦除扇区命令的实现来执行该命令。*

示例开头的目标地址是0x100000。这是Flash后半部分的第一个索引。

```
*(volatile uint32_t *)(destAdrss) = 0x0;
*(volatile uint32_t *)(destAdrss + 4) = 0x0;
*(volatile uint32_t *)(destAdrss + 8) = 0x0;
*(volatile uint32_t *)(destAdrss + 12) = 0x0;
```

7. 检查PERDY == 1，即已准备好执行操作。

```
if (((FMU0->FSTAT & FMU_FSTAT_PERDY(1)) >> FMU_FSTAT_PERDY_SHIFT) == 1)
{
//continue
}
```

除非PERDY已置为1（即已准备好执行操作），否则不能继续此操作。

在步骤6描述的序列操作中，PERDY必须在第四次连续的*(volatile uint32_t *) (destAdrss + 12) = 0x0写入完成之后立即被置为1。在示例代码中，使用了一个while循环来等待PERDY寄存器完成设置。然而，开发人员应考虑应用程序是否需要并行运行其他任务。

8. 通过向PERDY写入1来将其清零。操作将暂停，直至将其清零。

```
//controller should erase AND verify after we clear PERDY
FMU0->FSTAT = 0x80000000;
```

9. 检查FSTAT寄存器中是否存在任何错误。

```
if (((FMU0->FSTAT & FMU_FSTAT_ACCERR(1)) >> FMU_FSTAT_ACCERR_SHIFT) == 1)
{
   PRINTF("\r\n Access Error \r\n");
}
else if (((FMU0->FSTAT & FMU_FSTAT_PVIOL(1)) >> FMU_FSTAT_PVIOL_SHIFT) == 1)
{
   PRINTF("\r\n Protection Violation \r\n");
}
else if (((FMU0->FSTAT & FMU_FSTAT_CMDABT(1)) >> FMU_FSTAT_CMDABT_SHIFT) ==
 1)
{
   PRINTF("\r\n Operation Is Aborted \r\n");
}
else if(((FMU0->FSTAT & FMU_FSTAT_FAIL(1)) >> FMU_FSTAT_FAIL_SHIFT) == 1)
{
   PRINTF("\r\n Command Failed \r\n");
}
```

10. 在继续下一个命令控制器操作之前，确保FSTAT CCIF已置位。即该命令已完成。

```
if (((FMU0->FSTAT & FMU_FSTAT_CCIF(1)) >> FMU_FSTAT_CCIF_SHIFT) == 1)
{
//continue with programming
}
```

在示例代码中，使用了一个while循环来等待CCIF寄存器完成设置。然而，开发人员应考虑应用程序是否需要并行运行其他任务。

3.2 编程页面命令

以下步骤演示了执行一个编程页面命令的过程。示例工程将连续执行编程页面命令，直到0x10_0000 -> 0x1F_FFFF编程成功。

1. 检查FMU FSTAT寄存器确保CCIF已置位。这表示上一个命令已完成。

```
if (((FMU0->FSTAT & FMU_FSTAT_CCIF(1)) >> FMU_FSTAT_CCIF_SHIFT) == 1)
{
//continue with programming
}
```

CCIF寄存器必须置位为1，才能继续新的操作。在示例代码中，使用了一个while循环来等待CCIF寄存器完成设置。然而，开发人员应考虑应用程序是否需要并行运行其他任务。

2. 处理并清除FMU FSTAT寄存器中存在的所有错误标志。

```
//clear previous errors
FMU0->FSTAT = 0x34;
```

FSTAT_CLEARERR的值为0x34。

3. 通过将FMU FCCOB[0]设置为0x23 (PGMPG)，指定命令为“编程页面”（program page）。

```
//only need to specify command at call time
```

```
FMU0->FCCOB[0] = PGMPG;
```

4. 清除CCIF寄存器以启动命令。

```
//clear ccif to launch
FMU0->FSTAT = 0x80;
```

通过写入1来清零CCIF寄存器并启动命令。

5. 检查FMU FSTAT PEWEN == 2，允许进行页面编程的写入 - 一个页面。

```
if (((FMU0->FSTAT & FMU_FSTAT_PEWEN(value)) >> FMU_FSTAT_PEWEN_SHIFT) == 2)
{
//continue
}
```

FSTAT PEWEN必须置为2才能继续进行操作。在示例代码中，使用了一个while循环来等待PEWEN寄存器完成设置。然而，开发人员应考虑应用程序是否需要并行运行其他任务。

6. 向Flash中连续写入32个字。

```
//write 32 consecutive words to flash space
//one word = 4 bytes
for (int i = 0; i < 32; i++)
{
*(volatile uint32_t *)(destAdrss + index + (i*4)) = 0x12345678;
}
```

7. 检查FMU FSTAT PERDY == 1，即已准备好执行编程命令操作。

```
if (((FMU0->FSTAT & FMU_FSTAT_PERDY(1)) >> FMU_FSTAT_PERDY_SHIFT) == 1)
{
//continue
}
```

在示例代码中，使用了一个while循环来等待PERDY寄存器完成设置。然而，开发人员应考虑应用程序是否需要并行运行其他任务。

注： *在执行该命令之前，必须将FSTAT PERDY置为1。*

8. 通过向FMU FSTAT PERDY写入1来将其清零，否则，操作将保持停滞状态。

```
//clear PERDY
FMU0->FSTAT = 0x80000000;
```

9. 检查FSTAT寄存器中是否存在错误。

```
if (((FMU0->FSTAT & FMU_FSTAT_ACCERR(1)) >> FMU_FSTAT_ACCERR_SHIFT) == 1)
{
  PRINTF("\r\n Access Error \r\n");
}
else if (((FMU0->FSTAT & FMU_FSTAT_PVIOL(1)) >> FMU_FSTAT_PVIOL_SHIFT) == 1)
{
  PRINTF("\r\n Protection Violation \r\n");
}
else if (((FMU0->FSTAT & FMU_FSTAT_CMDABT(1)) >> FMU_FSTAT_CMDABT_SHIFT) ==
 1)
{
  PRINTF("\r\n Operation Is Aborted \r\n");
}
else if(((FMU0->FSTAT & FMU_FSTAT_FAIL(1)) >> FMU_FSTAT_FAIL_SHIFT) == 1)
{
  PRINTF("\r\n Command Failed \r\n");
}
```

10. 在继续下一个命令控制器操作之前，确保FSTATCCIF已置位。即该命令已完成。

```
if (((FMU0->FSTAT & FMU_FSTAT_CCIF(1)) >> FMU_FSTAT_CCIF_SHIFT) == 1)
{
//continue with programming
}
```

4 运行演示

要求：

1. MCUXpresso 11.7.1或更新版本
2. MCXNx4x EVK或FRDM
3. USB线
4. SDK版本2.13.0

步骤：

1. 下载相关的软件包。
2. 将工程导入到MCUXpresso IDE - 快速启动面板。点击“**从文件系统中导入工程...**”，见图3。

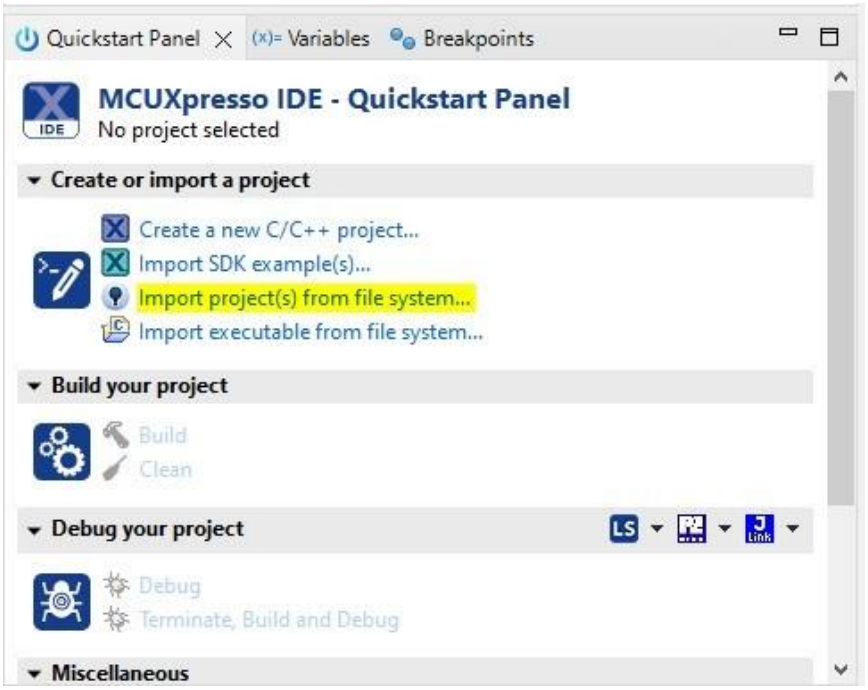

图3. 快速启动面板 - 导入工程

3. 点击“**浏览**...”，见图4。

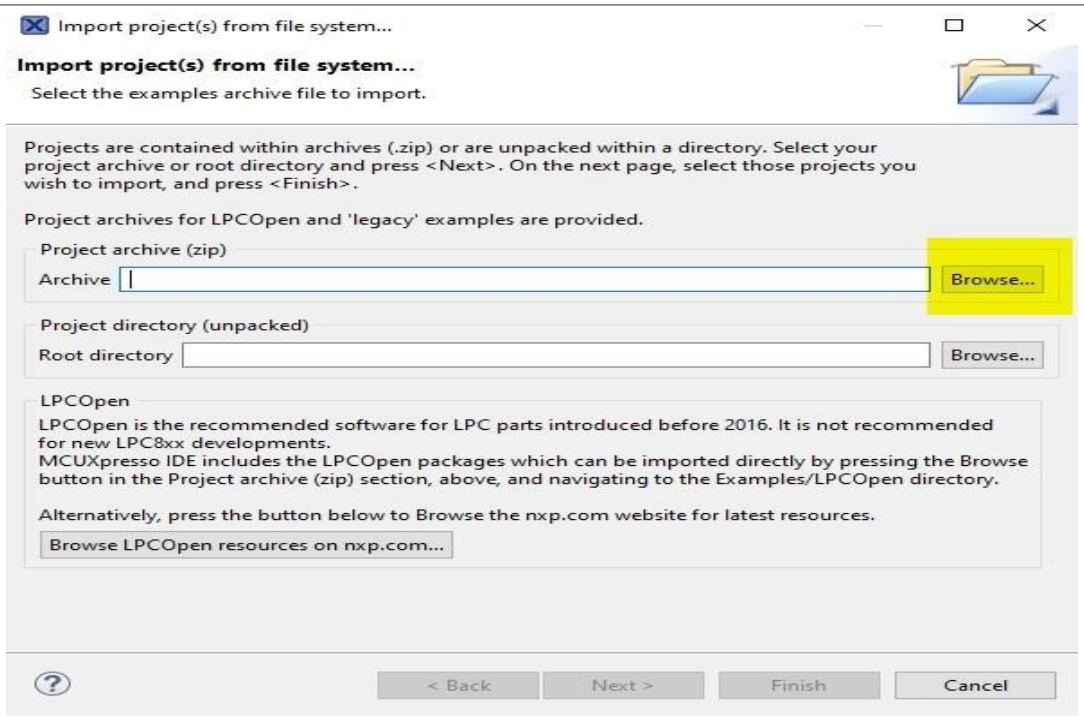

图4. 导入存档的工程

4. 在文件浏览器中找到并选择下载的`IAP_Flash_Commands.zip`。

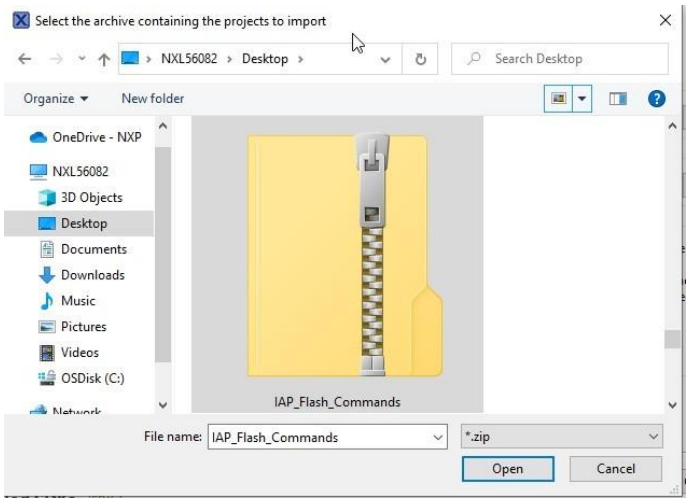

图5. 从文件浏览器中选择文件

5. 点击“**打开**”，见图5。
6. 点击“**下一步**”，见图6。

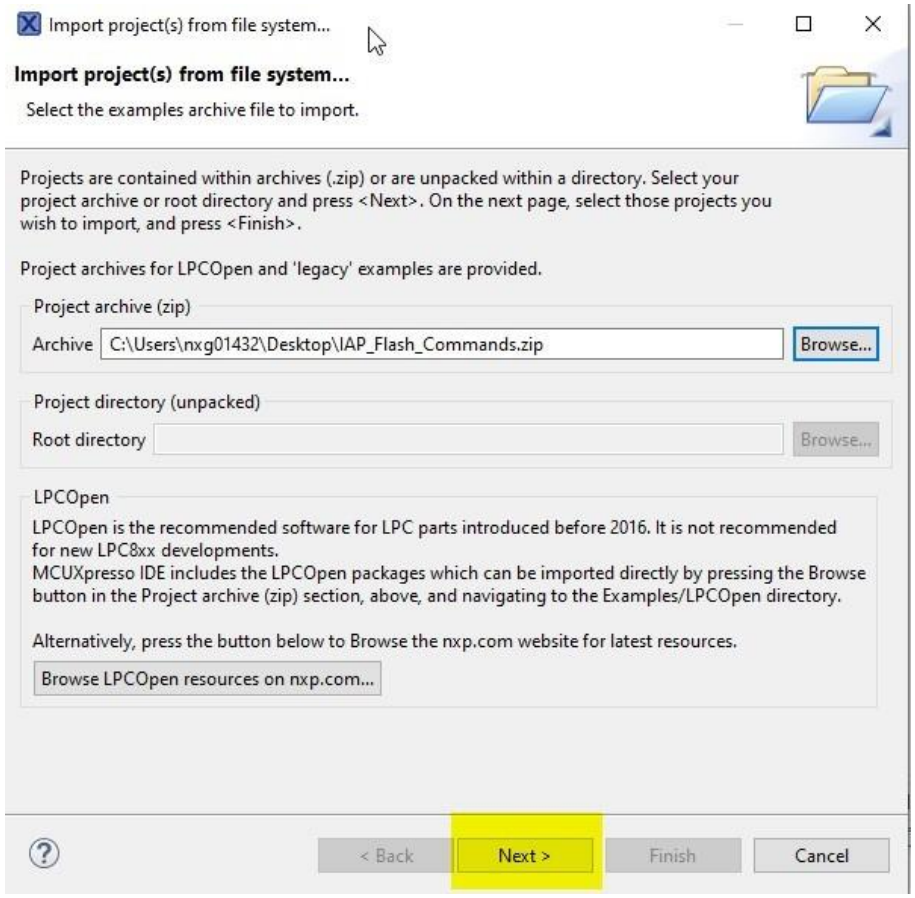

图6. 导入`IAP_Flash_Commands.zip`

7. 点击“**完成**”，见图7。

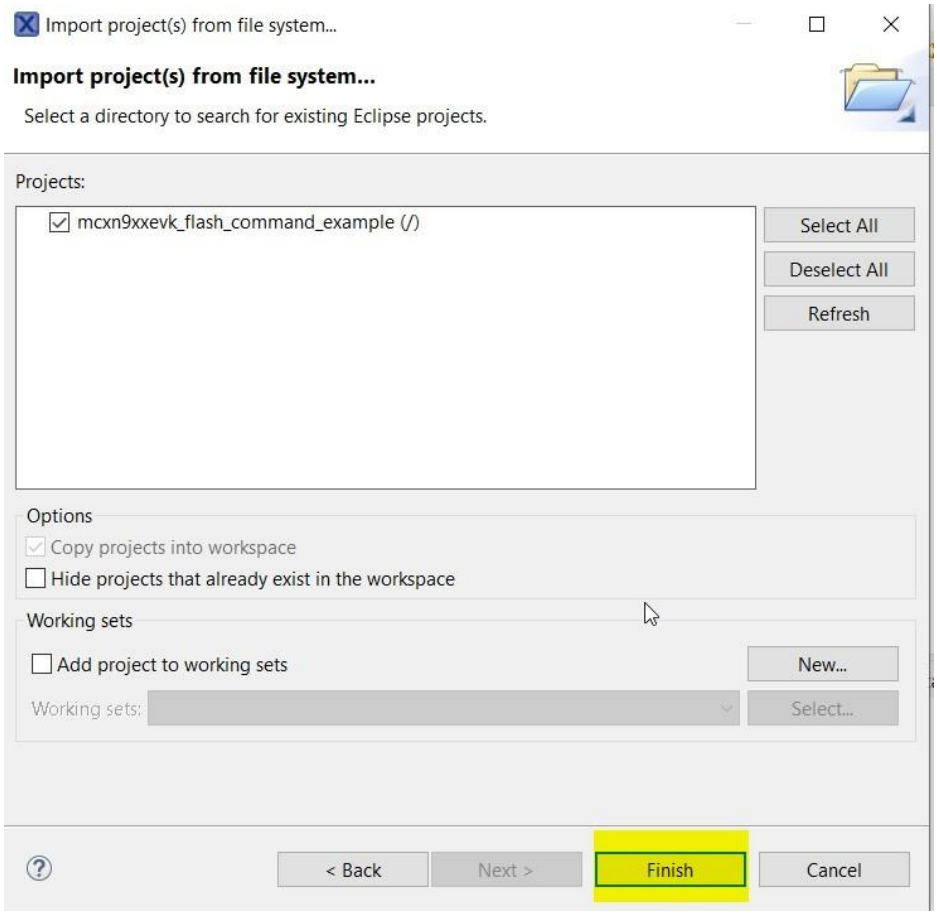

图7. 导入完成

工程下载并导入到MCUXpresso中后，使用一根micro-USB线连接PC主机和板上的MCU-Link USB端口J5（当使用MCX-N9XX-EVK时）或J17（当使用FRDM-MCXN947时）。

打开一个串口终端，并使用以下设置：

- 波特率：115200
- 数据位：8
- 奇偶校验：无
- 停止位：1
- 流控制：无

1. 点击工具栏的“**启动串口终端**”选项，见图8。

图8. 启动串口终端

2. “**启动终端**”窗口弹出。
3. 从下拉列表中选择终端 -> 选择**串口终端**，见图9。

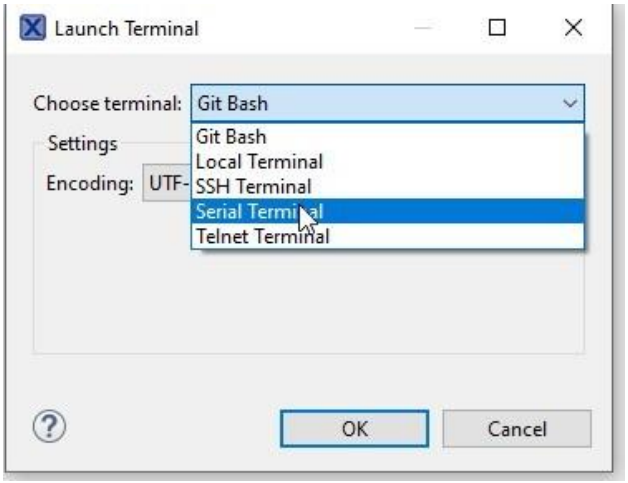

图9. 选择并启动串口终端

4. 选择与连接设备关联的**串行端口**，见图10。

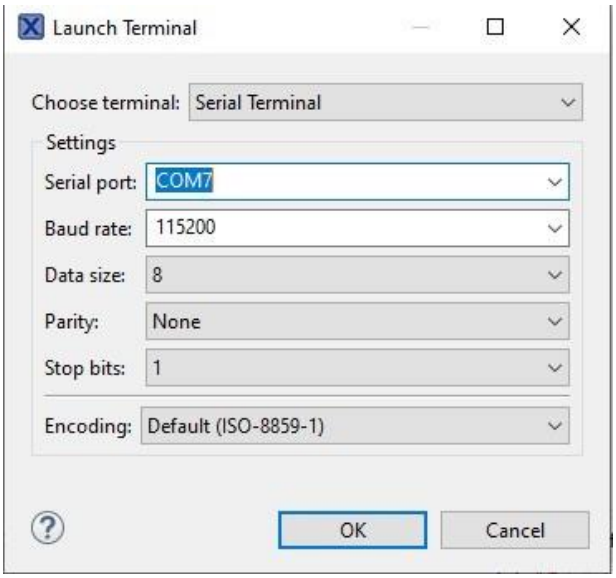

图10. 启动终端

注：*每个用户设备的串行端口均不同。*

5. 选择以下设置，如图10所示：
   - 波特率 -> `115200`。
   - 数据大小 -> `8`。
   - 奇偶校验 -> 无。
   - 停止位 -> `1`。
6. 点击“**OK**”。
7. 在快速启动面板中点击“**构建**”，见图11。

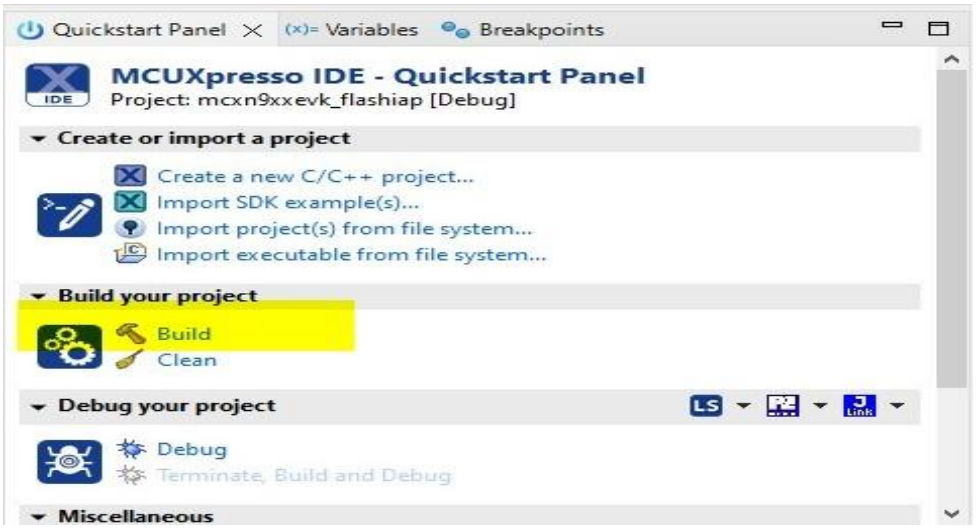

图11. 快速启动面板 - 构建工程

8. 点击“**调试**”，见图12。

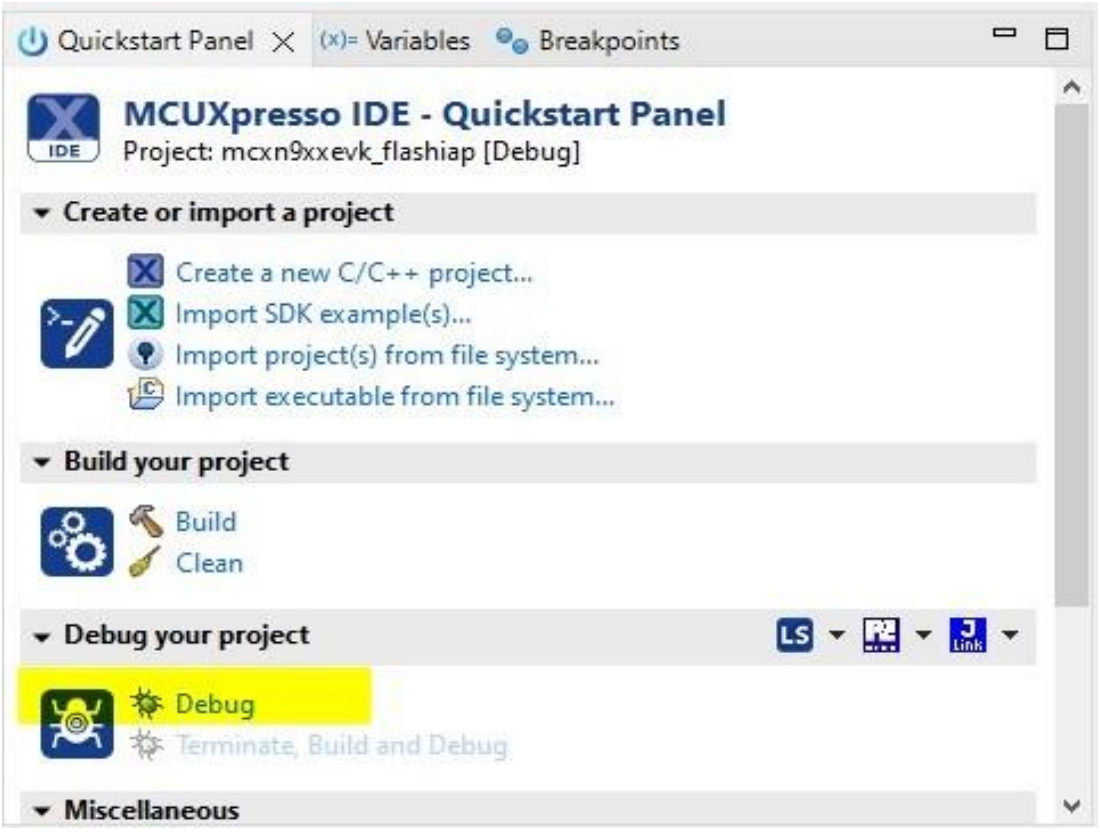

图12. 快速启动面板 – 调试工程

9. 点击“**OK**”，见图13。

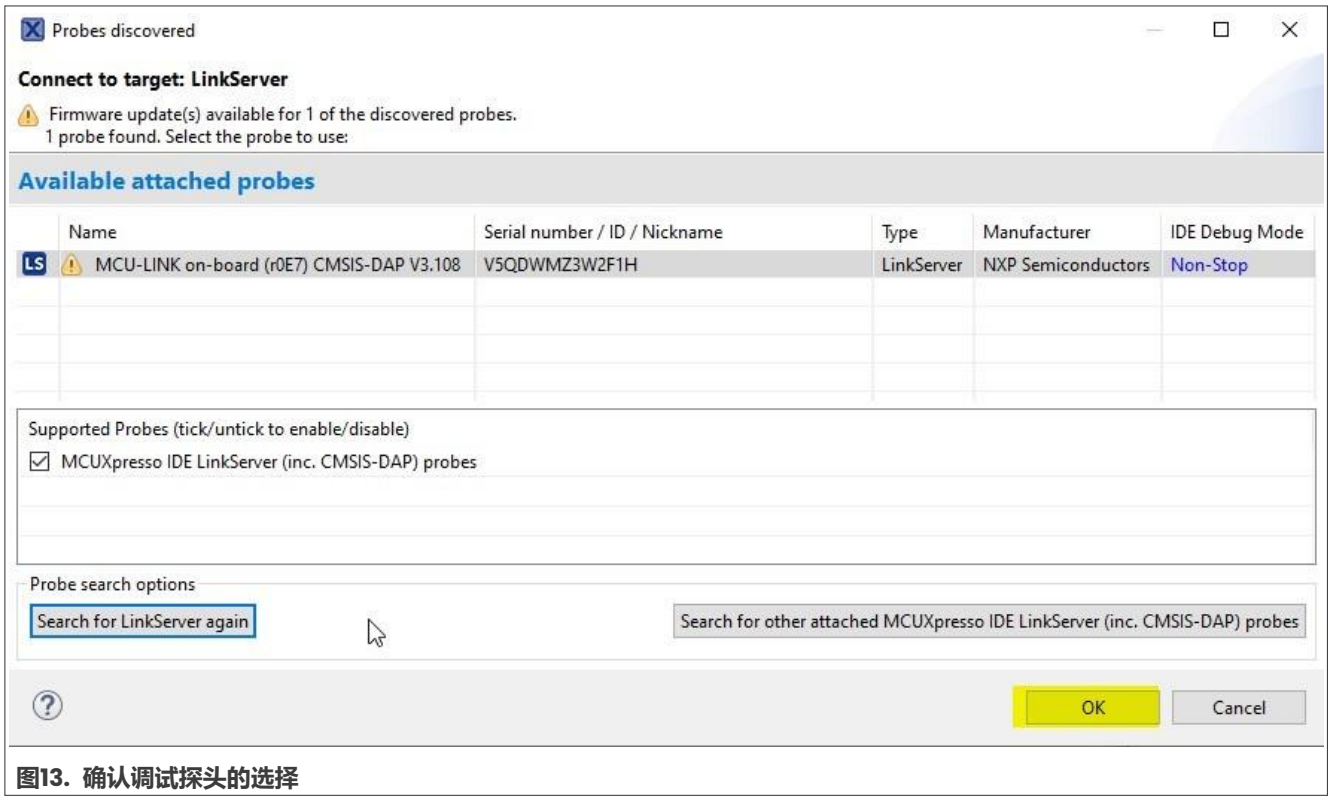

图13. 确认调试探头的选择

10. 现在，应该能够逐步执行代码了。点击工具栏中的“**逐步执行**” （**Step Over**）选项，见图14。

图14. “逐步执行”图标

11. 逐步执行至第**215**行，见图15。

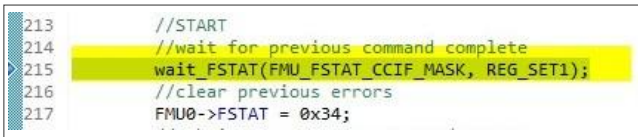

图15. 在第215行执行

12. 此时就完成了擦除扇区命令的第一步，打开外设查看器。点击“**外设+**”选项卡，见图16。

图16. 外设查看器选项卡

13. 展开“**FMU0**”，见图17。

FMU0			0x40043000	Flash
FSTAT	0x00000080	RW	0x40043000	Flash Status Register
FCNFG	0xff000000	RW	0x40043004	Flash Configuration Register
FCTRL	0x00000003	RW	0x40043008	Flash Control Register
FCCOB0	0x00000000	RW	0x40043010	Flash Common Command Object Registers
FCCOB1	0x00000000	RW	0x40043014	Flash Common Command Object Registers
FCCOB2	0x00000000	RW	0x40043018	Flash Common Command Object Registers
FCCOB3	0x00000000	RW	0x4004301c	Flash Common Command Object Registers
FCCOB4	0x00000000	RW	0x40043020	Flash Common Command Object Registers
FCCOB5	0x00000000	RW	0x40043024	Flash Common Command Object Registers
FCCOB6	0x00000000	RW	0x40043028	Flash Common Command Object Registers
FCCOB7	0x00000000	RW	0x4004302c	Flash Common Command Object Registers

图17. 外设查看器FMU0

14. 可以看到，FSTATCCIF寄存器已置为1，这意味着目前没有正在执行的命令，我们可以使用命令控制器执行命令，见图18。

FMU0			0x40043000	Flash
FSTAT	0x00000080	RW	0x40043000	Flash Status Register
FAIL	fail0	R	[0]	Command Fail Flag
CMDABT	cmdabt0	RW	[2]	Command Abort Flag
PVIOL	pviol0	RW	[4]	Command Protection Violation Flag
ACCERR	accerr0	RW	[5]	Command Access Error Flag
CWSABT	cwsabt0	RW	[6]	Command Write Sequence Abort Flag
CCIF	ccif1	RW	[7]	Command Complete Interrupt Flag

图18. FMU CCIF寄存器

15. 继续逐步执行代码并停在第**222**行，见图19。

```
221         //clear ccif to launch
222         FMU0->FSTAT = 0x80;
```

图19. 在第222行停止执行

16. 外设查看器显示，已将FMU -> FSTAT -> FCCOB[0]设置为0x42，这是擦除扇区的命令，见图20。

FCCOB0	0x00000042	RW	0x40043010	Flash Common Command Object Registers
CCOBn	0x42	RW	[31:0]	CCOBn

图20. FCCOB0寄存器

17. 再执行一步代码，可以看到已清零了CCIF，从而发起命令执行，见图21。

FMU0			0x40043000	Flash
FSTAT	0x01000900	RW	0x40043000	Flash Status Register
FAIL	fail0	R	[0]	Command Fail Flag
CMDABT	cmdabt0	RW	[2]	Command Abort Flag
PVIOL	pviol0	RW	[4]	Command Protection Violation Flag
ACCERR	accerr0	RW	[5]	Command Access Error Flag
CWSABT	cwsabt0	RW	[6]	Command Write Sequence Abort Flag
CCIF	ccif0	RW	[7]	Command Complete Interrupt Flag
CMDPRT	cmdprt01	R	[9:8]	Command protection level
CMDP	cmdp1	R	[11]	Command protection status flag
CMDDID	0x0	R	[15:12]	Command domain ID
DFDIF	dfdif0	RW	[16]	Double Bit Fault Detect Interrupt Flag
SALV_USED	salv_used0	R	[17]	Salvage Used for Erase operation
PEWEN	pewen01	R	[25:24]	Program-Erase Write Enable Control

图21. CCIF和PEWEN寄存器

18. 继续逐步执行并停在第**229**行。回顾前述操作，我们需要执行四次写入，第一次写入必须与扇区对齐。当前已停在序列操作中的第四次写入，见图22。

```
229         *(volatile uint32_t *)(destAdrss + 12) = 0x0;
```

图22. 在第229行停止执行

19. 单步执行此步代码，须看到PEWEN被清零且PERDY已置位，见图23。

PEWEN	pewen00	R	[25:24]	Program-Erase Write Enable Control
PERDY	perdy1	RW	[31]	Program-Erase Ready Control/Status Flag

图23. PEWEN和PERDY寄存器

20. 单步执行第**233**行代码，清除PERDY，见图24。

```
233         FMU0->FSTAT = 0x80000000;
234         //wait for previous command complete
235     wait_FSTAT(FMU_FSTAT_CCIF_MASK, REG_SET1);
```

图24. 单步执行第233行代码

21. 在外设查看器中，CCIF已设置为1，即命令已完成，见图25。

CCIF	ccif1	RW	[7]	Command Complete Interrupt Flag

图25. CCIF设置为1 – 命令完成

注：*擦除操作从0x10_0000开始，并擦除了一个扇区。*

22. 在“**外设+**”选项卡下，点击三个垂直点的图标并选择“**添加内存监视器**” -> **PROGRAM_FLASH1**，见图26。

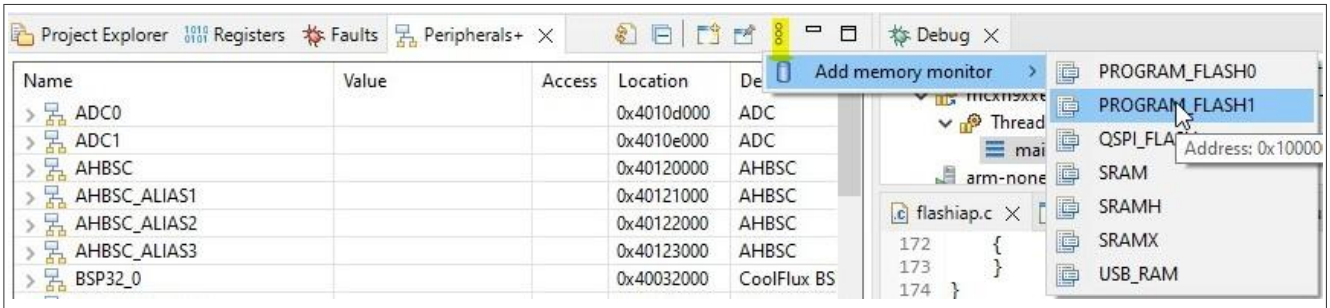

图26. 打开内存监视器 – PROGRAM_FLASH1

23. 完成擦除扇区的命令后，在“**内存->0x100000: 0x100000 <Hex>**”选项卡下，我们必须找到FFFFFFFF并继续擦除到0x102000，这意味着Flash的一个扇区已被擦除，见图27和图28。

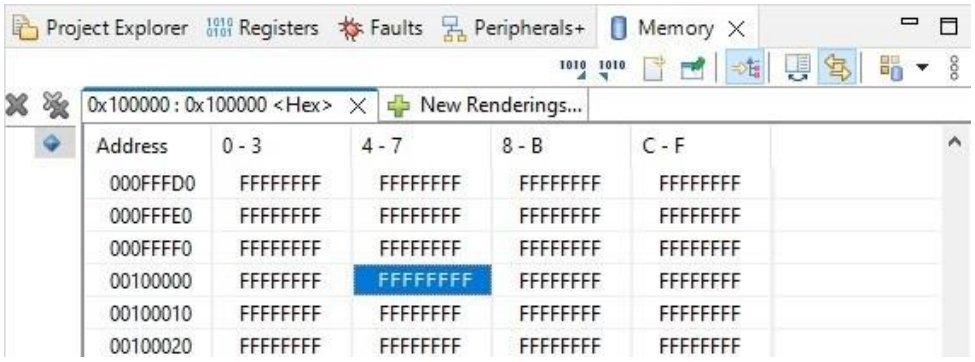

图27. 内存查看器中已擦除的扇区

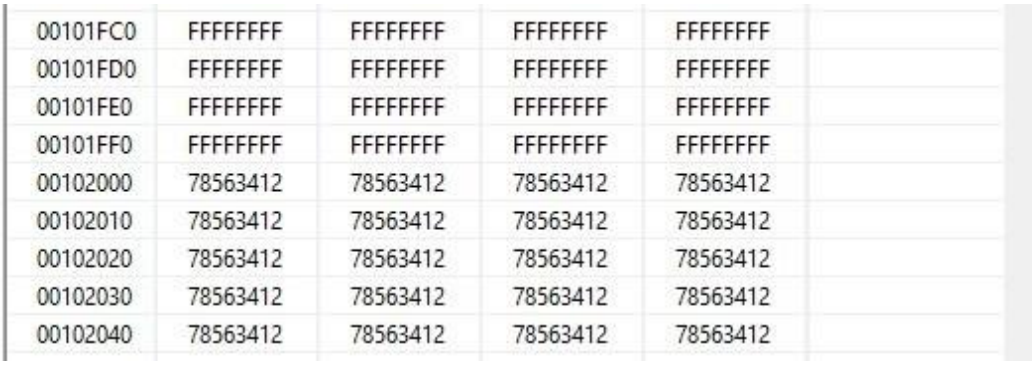

图28. 内存查看器中已擦除扇区的末端

24. 现在，可以选择继续逐步执行代码，或者终止调试会话，因为Flash程序命令也遵循类似的一种序列。
25. 执行完所有的Flash命令后，在内存监视器中，每个4字节的区域都应填充了十六进制值数`0x1234_5678`，见图29。

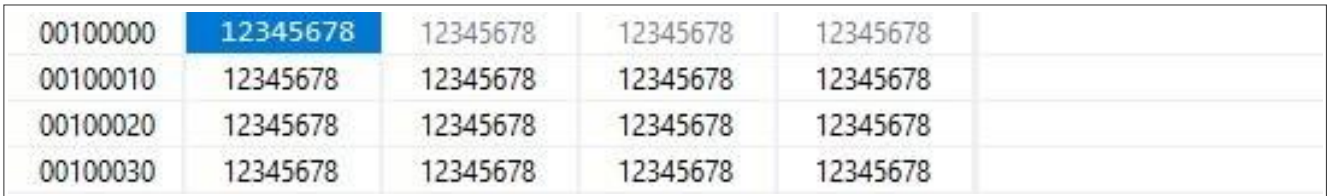

图29. Flash程序的后半部分

26. 终端窗口将显示以下消息，确认示例代码已成功运行。

```
Flash Command Erase / Programming example:
This application erases the flash area from 0x0010_0000 -> 0x001F_FFFF and
 then programs with 0x1234_5678.
Begin erase:   Success!
Begin Program: Success!

End of Flash Programming Example!
```

5 关于本文中源代码的说明

本文中所示的示例代码具有以下版权和BSD-3-Clause许可：

6 修订历史

表1总结了本文档的修订记录。

表1. 修订历史

文档编号	发布日期	说明
AN14178 v.1.0	2024年1月24日	初版发布

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.cn/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.

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.

MCX — is a trademark of NXP B.V.

目录

For more information, please visit: https://www.nxp.com.cn

Date of release: 24 January 2024
Document identifier: AN14178