User's Guide

MSPM0 NONMAIN 闪存配置指南

Sal Ye, Wei Gao

摘要

MSPM0 NONMAIN 是一个专用的闪存区域，用于配置芯片启动相关参数和扩展功能选择。本文档介绍了 NONMAIN 的详细配置，并为配置 NONMAIN 部分以供应用程序使用提供了指导。

内容

商标

1 简介

图 1-1 展示了所有 MSPM0 器件共享的通用平台存储器映射。该映射具有四个不同的存储器区域。

- MAIN 闪存
- NONMAIN 闪存
- FACTORY 区域
- ROM

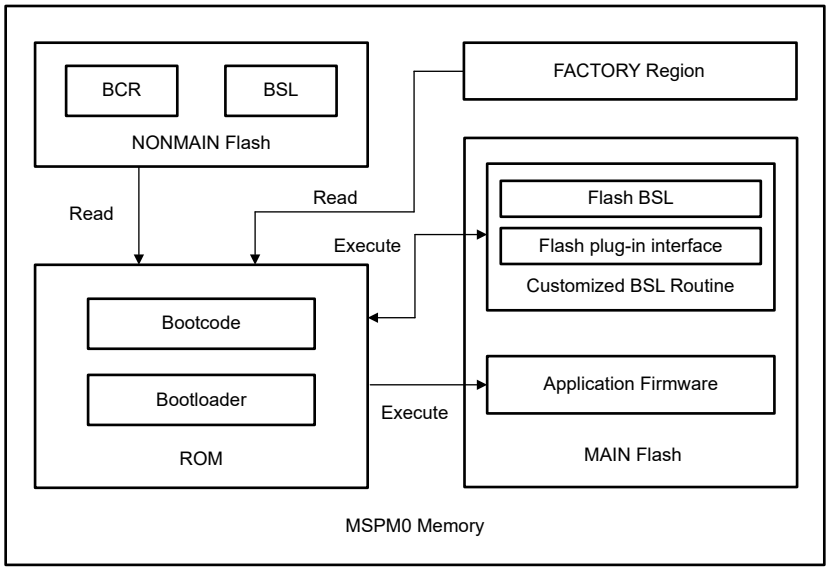


图 1-1. MSPM0 存储器映射

MAIN 闪存是存储可执行代码和数据的存储器。MSPM0 器件支持单存储体或双存储体 MAIN 闪存，请参阅器件特定数据表了解更多详细信息。

NONMAIN 是闪存中的一个专用区域，存储着引导配置例程 (BCR) 和引导加载程序 (BSL) 配置用于引导器件的数据。BCR 和 BSL 具有配置策略，这些策略可以保留为默认值（在开发和评估期间是典型值），也可以通过更改编程到 NONMAIN 闪存区域中的值来针对特定用途进行修改（在生产编程期间是典型值）。

备注

擦除 NONMAIN 闪存后，请验证是否已将正确的值加载到 NONMAIN 闪存中。如果负载验证失败，会导致在下次 BOOTRST 后永久锁定器件。

FACTORY 区域是一个存储器映射闪存区域，该区域提供描述器件功能的只读数据以及任何出厂提供的修整信息，供应用软件使用。

只读存储器 (ROM) 包含一个不可变的信任根引导配置例程 (BCR) 和引导加载程序 (BSL)。在主应用程序启动前，BCR 始终是紧跟器件的 BOOTRST 之后在 Cortex-M0+ 处理器上运行的第一个代码。BCR 还会通过硬件或软件调用引导加载程序 (BSL) 并授权 BSL 访问。图 1-2 展示了 MSPM0 器件的大体引导流程。有关更多详细信息，请参阅 *MSPM0 MCU 中的网络安全机制*应用手册。

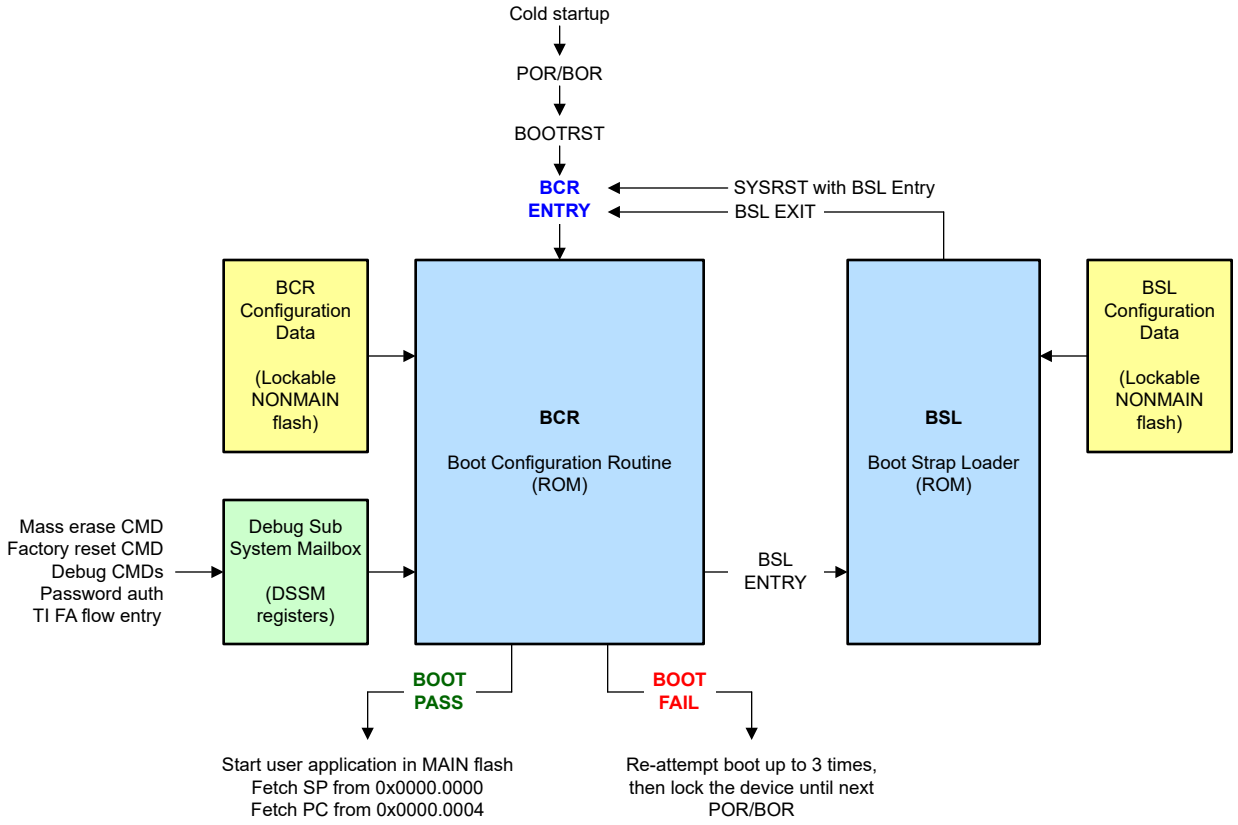


图 1-2. MSPM0 启动流程

BCR 会设置器件安全策略，配置器件使其运行，并视需要启动 BSL（如果 ROM 或闪存 BSL 存在）。如果 BCR 启动 BSL，则可使用 BSL 通过标准串行接口（UART 或 I2C）来对器件存储器（闪存和 SRAM）进行编程或验证。

1.1 术语

引导代码 (BCR)	在 BOOTRST 之后运行的启动例程，用于配置器件以执行应用程序。
BCR 配置	包含用于引导代码的所有用户可配置参数的配置结构，位于 NONMAIN 闪存中。
引导加载程序 (BSL)	用于将数据加载到器件存储器的引导例程。
BSL 配置	包含用于引导代码的所有用户可配置参数的配置结构，位于 NONMAIN 闪存中。
客户安全代码 (CSC)	该代码在 TI 引导代码之后运行，并从闪存执行来进一步配置安全元素。有关更多详细信息，请参阅安全引导用户指南

2 NONMAIN 架构

本节详细介绍 NONMAIN 结构、支持的功能以及各种 NONMAIN 配置的应用场景。

2.1 MSPM0 系列概述

MSPM0 系列支持不同类型的 NONMAIN 布局，以限制 NONMAIN 配置支持的可用功能。

表 2-1 展示了哪些 NONMAIN 布局类型可与哪些器件配合使用。有关更多详细信息，请参阅器件特定技术参考手册的*引导配置*部分（链接见节 10）。

表 2-1. NONMAIN 布局类型

NONMAIN 布局类型	支持的器件	概述
类型 A	MSPM0L110x MSPM0L130x MSPM0L134x MSPM0Gx10x MSPM0Gx50x	• 不支持 CSC • 密码以明文格式存储 • 仅支持 CRC32 应用程序完整性校验 • 支持 ROM 和闪存 BSL(1)
类型 B	MSPM0C1103 MSPM0C1104 MSPS003Fx	• 不支持 CSC • 无应用程序完整性校验 • 无基于密码的配置 • 不支持 BSL
类型 C	MSPM0Lx22x	• 支持 CSC • 密码以 SHA256 哈希格式存储 • 应用程序完整性校验提供了 CRC32 或 SHA256 哈希选项 • 支持 ROM 和闪存 BSL
类型 D	MSPM0C1105 MSPM0C1106 MSPM0H321x	• 支持 CSC • 密码以 SHA256 哈希格式存储 • 应用程序完整性校验支持 CRC32 或 SHA256 • 支持闪存 BSL
类型 E	MSPM0L111x MSPM0G511x MSPM0G5187	• 支持 CSC • 密码以 SHA256 哈希格式存储 • 应用程序完整性校验提供了 CRC32 或 SHA256 哈希选项 • UART 默认波特率配置 • 在 BSL 中禁用 NRST • 支持 ROM 和闪存 BSL
类型 F	MSPM0Gx51x MSPM0G352x	• 支持 CSC • 密码以 SHA256 哈希格式存储 • 应用程序完整性校验使用 CRC32 或 SHA256 哈希选项 • UART 默认波特率配置 • 支持 ROM 和闪存 BSL

(1) 闪存 BSL 是用户在 MAIN 闪存中定义的备用 BSL 接口。

2.2 NONMAIN 配置概述

MSPM0 NONMAIN 闪存具有不同类型的结构布局。在某些布局类型中，某些功能会被缩减或不受支持。

表 2-2. NONMAIN 配置参数

配置参数			寄存器字段	NONMAIN 布局类型					
				A	B	C	D	E	F
BCR	BCR 配置 ID		BCRCONFIGID	✓	✓	✓	✓	✓	✓
	串行线调试 (SWD) 策略配置	访问策略	BOOTCFG0	✓	✓	✓	✓	✓	✓
		调试策略							
		调试密码	PWDDEBUGLOCK	✓		✓	✓	✓	✓
		TI 故障分析	BOOTCFG1	✓		✓	✓	✓	✓
	SWD 恢复出厂设置配置	恢复出厂设置访问属性	BOOTCFG3	✓	✓	✓	✓	✓	✓
		恢复出厂设置密码	PWDFACTORYRESET	✓		✓	✓	✓	✓
	SWD 批量擦除配置	批量擦除访问属性	BOOTCFG3	✓		✓	✓	✓	✓
		批量擦除密码	PWDMASSERASE	✓		✓	✓	✓	✓
	闪存静态写保护 (SWP) 配置	MAIN 静态写保护	FLASHSWP0 FLASHSWP1 FLASHSWP2	✓	✓	✓	✓	✓	✓
		NONMAIN 静态写保护	BOOTCFG4						
	客户安全代码 (CSC)	CSC 策略	BOOTCFG5			✓	✓	✓	✓
		闪存存储体交换策略				✓		✓	✓
		调试保持	BOOTCFG4			✓	✓	✓	✓
	快速引导模式		BOOTCFG2	✓		✓	✓	✓	✓
	应用程序摘要校验配置	应用程序摘要校验策略	BOOTCFG6	✓		✓	✓	✓	✓
		应用程序起始地址	APPDIGESTSTART						
		应用程序长度	APPDIGESTLENGTH						
		应用程序校验和	APPDIGEST						
	BSL 策略	启用调用引脚检查	BOOTCFG1	✓		✓	✓	✓	✓
		启用 BSL 模式	BOOTCFG2						
	BCR 校验和	CRC16-CCITT	BCRCRC				✓		
		CRC32-ISO3309		✓		✓		✓	✓
BSL	BSL 配置 ID		BSLCONFIGID	✓		✓	✓	✓	✓
	BSL 引脚配置	调用引脚配置	BSLCONFIG0	✓		✓	✓	✓	✓
		NRST 引脚配置	BSLCONFIG3						✓
	UART 接口配置	引脚配置	BSLPINCFG0	✓		✓		✓	✓
		波特率配置	BSLCONFIG1					✓	✓
	I2C 接口配置	引脚配置	BSLPINCFG1	✓		✓		✓	✓
		目标地址配置	BSLCONFIG2						
	插件接口配置	启用插件接口	BSLPLUGINCFG	✓		✓		✓	✓
		函数指针地址分配	BSLPLUGINHOOK						
	备用 BSL 配置	启用备用 BSL 接口	BSLCONFIG1	✓		✓	✓	✓	✓
		备用 BSL 接口地址分配	SBLADDRESS						
	BSL 安全配置	BSL 访问密码	PWDBSL	✓		✓		✓	✓
		BSL 读出功能	BSLCONFIG0						
		BSL 警报配置	BSLCONFIG1						
		应用程序完整性校验	BSLAPPVER						
	BSL 校验和	CRC16-CCITT	BSLCRC				✓		
		CRC32-ISO3309		✓		✓		✓	✓

备注

对于不同类型的 NONMAIN 布局，寄存器地址偏移可能不同。有关详细信息，请参阅器件特定技术参考手册，链接见 节 10。

2.3 NONMAIN 存储器

表 2-3 展示了 NONMAIN 闪存的整个存储器部分。寄存器存储器映射因 NONMAIN 布局类型而异。有关更多详细信息，请参阅器件特定技术参考手册，链接见 节 10。

表 2-3. NONMAIN 存储器映射

NONMAIN 部分	起始地址	终止地址
BCR 配置	41C0.0000h	41C0.00FFh
BSL 配置	41C0.0100h	41C0.01FFh
保留的存储器	41C0.0200h	41C0.03FFh

一些器件提供了第二个 NONMAIN 扇区 (1KB)，访问时有以下限制：

- 始终不能被擦除（包括恢复出厂设置、批量擦除操作）
- 在 CSC 调用 INITDONE 之前只能被写入，之后只能被读取。

这种保护机制有助于实现基于硬件的单调计数器功能。由于该扇区只能被编程，因此它实际上可以用作非易失性递增计数器（或递减计数器，具体取决于软件对计数器值的解释）。CSC 可以在该扇区中维护修订版或回滚保护信息，确保低于计数器值的版本无法激活。

请参阅器件特定数据表以确定器件是否支持次级 NONMAIN 扇区。

3 NONMAIN 配置

本节详细介绍了 NONMAIN 配置，包括 BCR 配置和 BSL 配置。

3.1 BCR 配置

引导配置例程 (BCR) 是 BOOTRST 之后在器件上运行的第一个固件。用户可以在引导的启动阶段设置器件的不同属性。

TI 建议使用 Sysconfig 工具自定义 BCR 配置，有关更多详细信息，请参阅使用 SysConfig 配置 BCR。

3.1.1 BCR 配置 ID

TI 为不同类型的 NONMAIN 布局定义了默认的 BCR 配置 ID。用户可根据应用场景覆盖配置 ID。执行恢复出厂设置以恢复默认 BCR 配置 ID。

表 3-1 展示了每个器件的默认值。

表 3-1. 默认 BCR 配置 ID

NONMAIN 布局类型	器件系列	默认值
类型 A	MSPM0L110x	0x00000001
	MSPM0L130x	
	MSPM0L134x	
	MSPM0Gx10x	
	MSPM0Gx50x	
类型 B	MSPM0C1103	0x00000003
	MSPM0C1104	
	MSPS003Fx	
类型 C	MSPM0Lx22x	0x01000002
类型 D	MSPM0C1105	0x05000000
	MSPM0C1106	
	MSPM0H321x	
类型 E	MSPM0L111x	0x06000000
	MSPM0G511x	
	MSPM0G5187	
类型 F	MSPM0Gx51x	0x04000000
	MSPM0G352x	

3.1.2 串行线调试 (SWD) 策略

串行线调试 (SWD) 相关策略可配置器件物理调试接口中的可用功能。MSPM0 器件支持三个通用安全等级：无限制（级别 0）、自定义限制（级别 1）和完全限制（级别 2）。表 3-2 展示了三种通用安全级别（从限制性最低到限制性最高），包括每个级别之间的差异。

表 3-2. 通用安全级别

级别	场景	SW-DP 访问策略	应用程序调试策略	批量擦除策略	恢复出厂设置策略	TI FA 策略
0	无限制	EN	EN	EN、DIS(1)	EN	EN
1	自定义限制	EN	EN、EN（具有密码）、DIS	EN、EN（具有 PWDIS）	EN、EN（具有 PW）、DIS	EN、DIS
2	完全限制	DIS	无关紧要(2) （禁用 SW-DP 时，无法访问）			

(1) MSPM0C1103/4 不支持批量擦除操作。

(2) 当 SW-DP 策略是*禁用 SW-DP* 时，从 SWD 接口的角度来看，批量擦除和恢复出厂设置策略是无关的。但是，如果启用了引导加载程序 (BSL)，则批量擦除和恢复出厂设置策略会影响可用的 BSL 功能。有关保护 BSL 的详细信息，请参阅 BSL 安全性部分 (节 3.2.6)。

默认情况下，TI 提供的 MSPM0 处于无限制状态。无限制状态便于进行生产编程、评估和开发。但该状态不建议用于量产，因为无限制状态会留下较大的攻击面。

确定保护需求时需考虑 SWD 接口的 4 个主要用途：

- 应用程序调试访问：在 IDE 工具中进行调试
- 批量擦除访问：擦除 MAIN 存储器区域
- 恢复出厂设置访问：出厂复位访问：擦除 MAIN 存储器区域并将 NONMAIN 器件配置存储器复位至 TI 出厂默认值（级别 0）
- TI 故障分析访问：使 TI 能够启动故障分析

SWD 安全策略通过 NONMAIN 存储器中的 16 位模式匹配字段实现，具有以下特性：

- 需要精确的模式匹配，才能启用较低的安全状态
- 如果该 16 位字段中的任何值与确切定义的模式不匹配，都会导致相应参数进入最高安全状态

SWD 还支持自定义四组密码（总共 128 位），用于通过调试子系统邮箱 (DSSM) 命令解锁器件。有关详细说明，请参阅*适用于 MSPM0 的硬件编程和调试器指南*应用手册。

3.1.2.1 访问策略

BCR 支持通过 BOOTCFG0.SWDP_MODE 字段修改 SWD 访问策略，该配置可使器件进入最高限制状态（级别 2）。

当选择级别 2（禁用 SW-DP）时，物理调试端口 (SW-DP) 被完全禁用，所有 SWD 可访问的功能（应用程序调试、批量擦除、恢复出厂设置和 TI 故障分析）都不能通过 SWD 访问，无论各自的配置如何。

如果 BSL 被启用，那么 BSL 仍然使用批量擦除和恢复出厂设置配置字段来授权来自 BSL 接口的批量擦除或恢复出厂设置命令。有关详细信息，请参阅节 3.2。

3.1.2.2 调试策略

当 SWD 访问策略保持启用时，用户可通过 BOOTCFG0.DEBUGACCESS 字段进一步修改 SWD 调试策略。

MSPM0 支持 SWD 启用、禁用及 *128 位*密码加密。器件仅在引导阶段验证密码。用户执行用于密码认证的 DSSM 命令，其中包括用于解锁 SWD 接口的复位行为。

MSPM0 提供了两种类型的密码存储，即明文和 SHA2-256 摘要。密码存储在 PWDDEBUGLOCK 字段中。明文密码直接存储在 NONMAIN 中，存在密码泄露风险。SHA2-256 摘要存储模式不会将 *128 位*密码直接存储在 NONMAIN 存储器区域中。而是通过 *128 位*密码生成 *256 位*哈希值并存储至 NONMAIN，从而增强密码存储安全性。

用户可通过以下方式生成和设置 SHA2-256 摘要密码：

1. 确定解锁 SWD 接口的 *128 位*密码。将密码设置为 32 位对齐格式，以形成 4 个 *32 位*密码。
2. 反转 4 个 *32 位*密码字节序，因为 SHA 采用 *8 位*寻址格式。
3. 计算哈希值时，将 4 个反转后的 *32 位*密码合并为一个字符串。
4. 计算输入字符串的 SHA256 值。
5. 将输出 SHA256 值分解为 8 个 *32 位*字。
6. 反转输出的 8 个 *32 位*字的字节序，因为 SHA2-256 计算采用 *8 位*寻址格式。
7. 将反转后的密码按顺序存储到 PWDDEBUGLOCK.DIGEST 字段中。

备注

密码中的 *0* 值很重要。在计算过程中不要删除 *0* 值。

此流程适用于所有 *256 位* SHA2-256 摘要密码的生成（若器件支持），包括：

- SWD 访问权限
- 恢复出厂设置
- 批量擦除
- BSL 访问（*256 位*密码）

节 4.2 展示了如何使用 SysConfig 工具设置密码。节 6.3 展示了如何使用 CCS 工具解锁器件。

3.1.2.2.1 明文密码示例

图 3-1 展示了 SWD 密码以明文形式存储的情况，假设用户将 *128 位*密码设置为 *0123456789ABCDEF67452301EFCDAB89*。

	32-bit Hex	*8-bit Hex*
PWDDEBUGLOCK[0].DIGEST :	0x01234567	67 45 23 01
PWDDEBUGLOCK[1].DIGEST :	0x89ABCDEF	EF CD AB 89
PWDDEBUGLOCK[2].DIGEST :	0x67452301	01 23 45 67
PWDDEBUGLOCK[3].DIGEST :	0xEFCDAB89	89 AB CD EF

图 3-1. 明文密码示例

3.1.2.2.2 SHA2-256 密码示例

下面是 SHA2-256 摘要密码的计算示例，假设用户将 *128 位*密码设置为 *0123456789ABCDEF67452301EFCDAB89*。

1. 将密码设置为 32 位对齐：0x01234567、0x89ABCDEF、0x67452301、0xEFCDAB89
2. 反转密码字节序，因为 SHA2-256 按字节计算：0x67452301、0xEFCDAB89、0x01234567、0x89ABCDEF
3. 将 4 组反转后的 *32 位*密码合并为一个字符串：67452301EFCDAB890123456789ABCDEF
4. 计算 SHA2-256 值（选择 *HEX* 作为输入编码）：8420347FCB0F019E15564A0F65B8E197ECDF9F92D1ECA2BBAB1B8CB314C763DA（SHA256 在线工具）
5. 将 SHA2-256 值分解为 8 个 *32 位*字：0x8420347F、0xCB0F019E、0x15564A0F、0x65B8E197、0xECDF9F92、0xD1ECA2BB、0xAB1B8CB3、0x14C763DA
6. 反转输出字节序：0x7F342084、0x9E010FCB、0x0F4A5615、0x97E1B865、0x929FDFEC、0xBBA2ECD1、0xB38C1BAB、0xDA63C714
7. 将 8 个 *32 位*密码按顺序存储在 PWDDEBUGLOCK.DIGEST 字段中

3.1.2.3 批量擦除和恢复出厂设置策略

*SWD 批量擦除*是指仅擦除 MAIN 闪存区域，通常包括用户应用程序。存储在 NONMAIN 闪存区域中的 BCR 和 BSL 策略不受批量擦除的影响。批量擦除适用于在保持器件 NONMAIN 配置不变的情况下擦除所有应用程序代码和数据。

*SWD 恢复出厂设置*是指擦除 MAIN 闪存区域，然后将非 NONMAIN 闪存区域复位为默认值。这种擦除对于完全复位 BCR 和 BSL 器件启动策略非常有用，同时还擦除应用程序代码和数据。

BCR 配置借助于使用 DSSM 从调试探针通过 SWD 接口发送到器件的命令，提供批量擦除和恢复出厂设置功能。这些命令在 SWD 安全级别 2 中不可用，但在安全级别 0 和 1 中可供选用。当器件未配置为 SWD 安全级别 2 时，批量擦除和恢复出厂设置命令有三种可能的配置位置：启用、使用唯一的 128 位密码启用或禁用。

备注

当恢复出厂设置被单独配置为禁用时，BSL *无法*执行恢复出厂设置命令。

请参阅器件特定数据表，确定支持的 *128 位*密码存储模式（明文或 SHA2-256）。请参阅 节 3.1.2.2 了解有关计算 SHA2-256 摘要密码的说明，参阅 节 8.2 了解生成 SWD 批量擦除或恢复出厂设置命令的流程。

3.1.2.4 TI 故障分析

TI 故障分析访问功能允许 TI 通过 SWD 启动故障分析 (FA) 回流。TI FA 流程始终会在授予 TI 访问权限前强制恢复出厂设置。这可以确保 TI 在启动故障分析流程时，不具备读取器件闪存中客户专有信息的机制。

用户可通过 BOOTCFG1.TI_FA_MODE 字段禁用 TI 故障分析。此访问设置将阻止 TI 对器件恢复出厂设置及执行后续 FA 操作。

3.1.3 闪存静态写保护

闪存保护和完整性策略规定闪存的哪些扇区被锁定而无法修改，以及引导过程中在启动用户应用程序之前要校验哪些扇区的完整性。

支持两种类型的闪存静态保护：MAIN 闪存上的静态写保护和 NONMAIN 闪存上的静态写保护。

备注

在执行批量擦除或恢复出厂设置时，DSSM 命令会覆盖指定的静态写保护策略。而 BSL 命令*不*会覆盖，但相应的受保护闪存区域保持不变。

3.1.3.1 MAIN 闪存静态写保护

对于闪存容量最高 32KB 的器件，需要配备 FLASHSWP0.DATA 字段来保护整个 32KB 存储器。数值 *0* 表示对相应扇区应用写保护。对于 FLASHSWP0，1 位表示闪存中的 1 个扇区 (1KB)。

对于闪存容量最高 256KB 的器件，需要配备 FLASHSWP0 和 FLASHSWP1 字段来保护整个 256KB 存储器。数值 *0* 表示对相应扇区应用写保护。对于 FLASHSWP1，1 位表示闪存中的 8 个扇区，且低 4 位会被忽略。

对于闪存容量最高 512KB 的器件，需要配备 FLASHSWP0、FLASHSWP1 和 FLASHSWP2 字段来保护整个 512KB 存储器。数值 *0* 表示对相应扇区应用写保护。对于 FLASHSWP2，1 位表示闪存中从 256K 开始的 8 个扇区。

备注

当启用存储体交换功能并在多存储体器件中执行交换时，静态写保护地址也会交换。

3.1.3.2 NONMAIN 闪存静态写保护

当配置 BOOTCFG4 以保护 NONMAIN 时，整个 NONMAIN 区域将受到写保护。这使得当引导配置例程转换为执行 MAIN 闪存中的引导加载程序或用户应用程序代码时，该区域在功能上不可更改。如果应用程序代码或引导加载程序尝试对 NONMAIN 进行任何编程或擦除，都会导致硬件闪存操作错误，并且扇区不会被修改。

3.1.4 客户安全代码 (CSC)

客户安全代码 (CSC) 是在 TI 引导代码之后运行的代码。CSC 从闪存执行以进一步配置安全元素。CSC 是客户自有的安全软件，请验证 CSC 是否位于 0x0 地址。图 3-2 展示了简化的 CSC 执行流程。

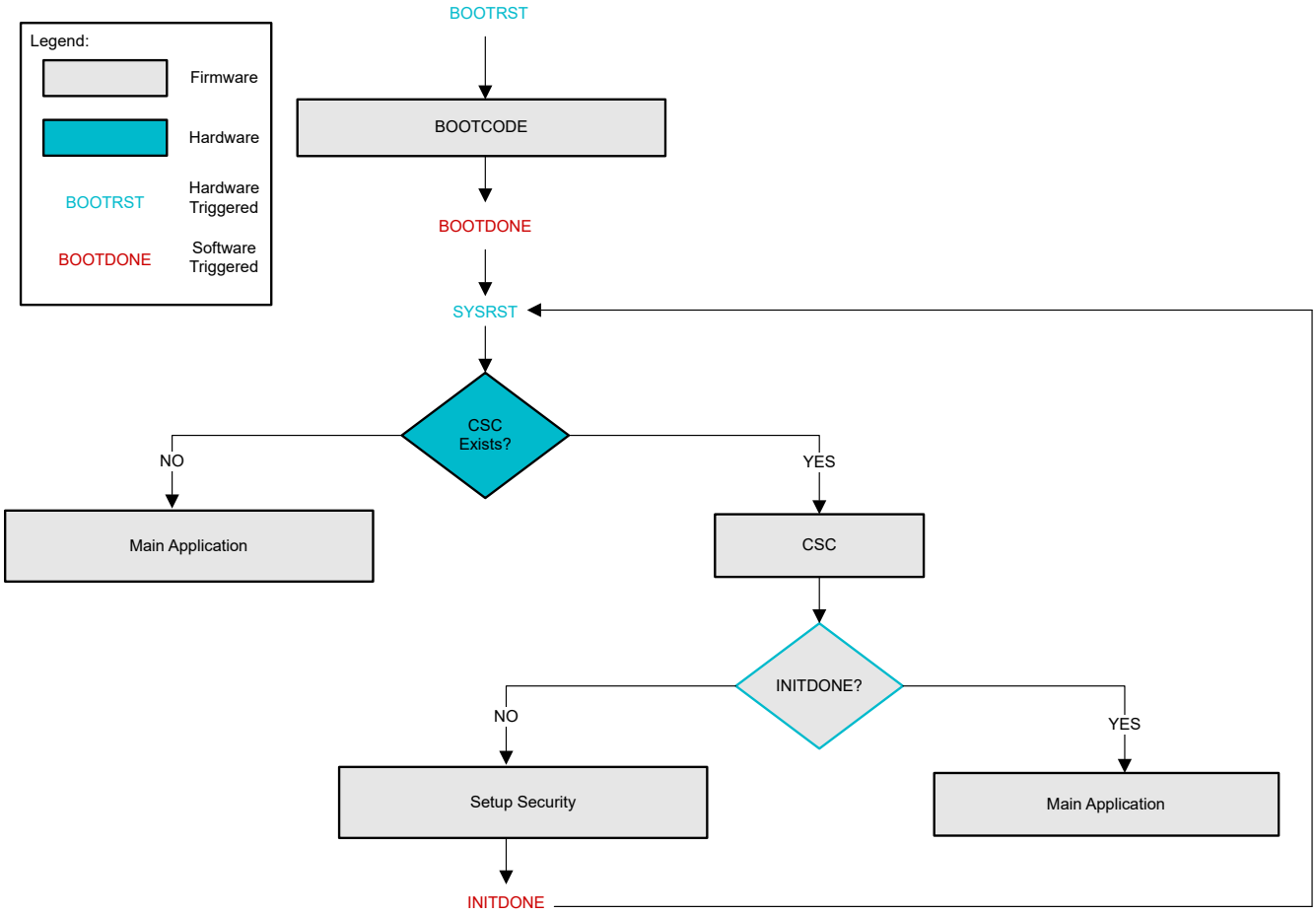


图 3-2. 安全引导和启动序列

TI 安全模型概念同时信任 TI 引导代码和 CSC。为了信任 CSC，TI 建议启用调试保持和闪存静态写保护。

3.1.4.1 CSC 策略

默认情况下，CSC 处于禁用状态。用户使用 BOOTCFG5.CSCEXISTS 字段启用 CSC。当 CSC 完成执行后，系统将经历第二次 SYSRST，然后就会启动主应用程序。

备注

当 CSC 启用时，引导代码会在引导阶段绕过 BSL 调用引脚检查。器件*无法*通过硬件调用进入 BSL 模式。在此期间，软件调用方法仍然可用。

3.1.4.2 闪存存储体交换策略

在多存储体器件中，存储体交换默认处于禁用状态。多个存储体具有相同的属性（写、读和执行），但位于不同的存储器地址。

用户可通过 BOOTCFG5.FLASHBANKSWAPPOLICY 字段启用存储体交换。引导代码读取此配置并将适当的 KEY (0xCA) 写入 SYSCTL.SECCFG.FLBANKSWPPOLICY。在硬件默认设置中，可交换配置为启用状态，且下存储体用作逻辑存储体 0。而 BOOTCFG5.FLASHBANKSWAPPOLICY 字段的默认值会禁用存储体交换策略，因此整个主闪存区域具有读、写、执行访问属性。

存储体交换行为发生在 CSC 阶段，并在器件发出 INITDONE 后生效。根据哪个（或哪对）存储体可执行，该（该对）存储体将获得读和执行权限，并失去写和擦除权限。另一个（或一对）存储体可读和写，但不可执行。此机制强制执行以下策略：发出 INITDOWN 后，固件更新的唯一保存位置是可写存储体，且绝不能被执行。

备注

即使用户启用了存储体交换，在 CSC 阶段整个主闪存区域仍具有读、写和执行访问属性。

有关存储体交换功能的更多详细信息，请参阅 *MSPM0 系列中的闪存多存储体功能*应用手册。

3.1.4.3 调试保持

对于支持 CSC 功能的器件，BCR 配置还提供了在 CSC 执行期间锁定 SWD 接口访问能力的属性。器件在 CSC 中发出 INITDOWN 后才会释放调试访问权限。

默认情况下，调试保持功能处于禁用状态，TI 建议用户在应用程序调试期间将其保持禁用。当产品进入量产阶段且启用 CSC 时，TI 建议通过 BOOTCFG4.DEBUGHOLD 字段启用调试保持，以将 CSC 设为不可访问状态来加以保护。

3.1.5 快速引导模式

通过启用快速引导模式，可以缩短 BCR 的执行时间。快速引导模式可使用以下方法加快引导过程：

- 限制进入 BSL：启用快速引导模式后，仅通过 SYSCTL 寄存器调用方法和 DSSM 调用方法进入引导加载程序。
- 绕过应用程序摘要校验（即使启用了应用程序摘要校验）。有关更多信息，请参阅节 3.1.6。

可将快速引导模式设置为通过在 NONMAIN 闪存中配置 BOOTCFG2.FASTBOOTMODE 字段来启用。

3.1.6 应用程序摘要校验

BCR 支持在启动用户应用程序之前的引导过程中对包含在 MAIN 闪存区域中的应用程序代码和数据执行完整的 CRC32 或 SHA2-256 完整性校验。应用程序摘要校验对于在执行部分或全部应用程序代码和数据之前验证这些代码和数据的完整性非常有用。

请按照以下步骤启用应用程序摘要校验：

1. 选择应用程序摘要校验策略，可设置为 CRC32 或 SHA2-256（BOOTCFG6.APPDIGESTMODE 字段）
2. 设置应用程序摘要校验的 32 位起始地址（APPDIGESTSTART.ADDRESS 字段）
3. 设置适用于 CRC 或 SHA 摘要校验的应用程序长度（按字节指定）（APPDIGESTLENGTH.LENGTH 字段）。
4. 使用给定数据（按字节指定）计算 CRC32/SHA2-256 值
5. 将预先计算出的 CRC32/SHA2-256 值分解为 32 位格式，并将其存储到 BCR 配置中（APPDIGEST.DIGEST 字段）。

如果在引导时应用程序摘要校验失败，则不会启动 MAIN 闪存中的应用程序。如果启用了 BSL，将进入 BSL。如果未启用 BSL，引导会失败。

3.1.6.1 CRC32 摘要校验示例

以下是 CRC32 完整性校验的计算示例，假设用户将起始地址设置为 0x0001.0000，长度为 8 字节。应用程序数据请参考 节 3.1.6。

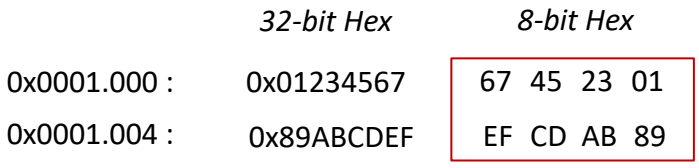


图 3-3. CRC32 摘要校验示例数据

选择 *JAMCRC* 作为计算模型以获取 CRC32 输出值。*JAMCRC* CRC 模型采用反转后的输入/输出位，使用多项式 0x04C11DB7，初始值为 0xFFFFFFFF，输出 XOR 值为 0x00000000。

按以下步骤计算示例 CRC32 值：

1. 选择的应用程序摘要校验策略为 CRC 校验 (BOOTCFG6.APPDIGESTMODE = AABBh)
2. 设置应用程序摘要校验的 32 位起始地址 (APPDIGESTSTART.ADDRESS = 0001.0000h)
3. 设置适用于 CRC 摘要校验的应用程序长度（按字节指定）(APPDIGESTLENGTH.LENGTH = 8h)
4. 根据给定输入计算 CRC32 输出
   a. 将输入数据字节序反转，因 CRC32 采用 8 位格式计算：67, 45, 23, 01, EF, CD, AB, 89
   b. 将 8 位输入数据合并为一个字符串：67452301EFCDAB89
   c. 计算 CRC32 值（选择 *HEX* 作为输入编码、*JAMCRC* 作为模型）：0x24648719（CRC32 在线工具）
5. 向 BCR 配置中注册 CRC32 输出值 (APPDIGEST.DIGEST = 2464.8719h)。若存在多个 APPDIGEST.DIGEST 寄存器，仅使用 DIGEST 的首个 32 位字进行 CRC32 校验

备注

验证应用程序 CRC32 摘要校验的长度是否为偶数。

3.1.6.2 SHA2-256 摘要校验示例

以下是 SHA2-256 完整性校验的计算示例，假设用户将起始地址设置为 0x0001.0000，长度为 8 字节。图 3-4 展示了示例应用程序数据。

	32-bit Hex	*8-bit Hex*
0x0001.000 :	0x01234567	67 45 23 01
0x0001.004 :	0x89ABCDEF	EF CD AB 89

图 3-4. SHA2-256 摘要校验示例数据

按以下步骤计算示例 SHA2-256 值：

1. 选择的应用程序摘要校验策略为 SHA2-256 (BOOTCFG6.APPDIGESTMODE = CCDDh)
2. 设置应用程序摘要校验的 32 位起始地址 (APPDIGESTSTART.ADDRESS = 0001.0000h)
3. 设置适用于 SHA2-256 摘要校验的应用程序长度（按字节指定）(APPDIGESTLENGTH.LENGTH = 8h)
4. 根据给定输入计算 SHA2-256 输出
   a. 反转输入 32 位数据字节序，因为 SHA2-256 按字节计算：67, 45, 23, 01, EF, CD, AB, 89
   b. 将 8 位输入数据合并为一个字符串：67452301EFCDAB89
   c. 计算 SHA2-256 值（选择 *HEX* 作为输入编码）：FDD232547CEEE35ADD35783CA7D518D2A49ABCCD0B60B028C8A9C629EBA6201F（SHA2-256 在线工具）
   d. 将 SHA2-256 值分解为 8 个 32 位字：0xFDD23254、0x7CEEE35A、0xDD35783C、0xA7D518D2、0xA49ABCCD、0x0B60B028、0xC8A9C629、0xEBA6201F
5. 反转 32 位输出字字节序：0x5432D2FD、0x5AE3EE7C、0x3C7835DD、0xD218D5A7、0xCDBC9AA4、0x28B0600B、0x29C6A9C8、0x1F20A6EB
6. 将 8 个 32 位 SHA2-256 校验和按顺序存储在 APPDIGEST.DIGEST 字段中

3.1.7 BSL 策略

对于支持 ROM 或闪存 BSL 的器件，TI 通过 BSL 接口提供了对器件存储器的访问能力。使用一个默认 BSL 调用引脚进行硬件调用 BSL，请参阅 节 3.2.2。

用户可通过 BOOTCFG2.BSLMODE 字段禁用 BSL，这将完全禁用 BSL 模式。BSL 软件调用（通过 SYSRST）也会被阻断。

或者，通过 BOOTCFG1.BSL_PIN_INVOKE 字段禁用 BSL 调用引脚检查，从而阻断 BSL 硬件入口。禁用调用引脚检查后，器件在引导过程中*不*再检查调用引脚状态，从而禁用 BSL 硬件入口。用户仍可通过应用程序代码中的软件调用方法进入 BSL。

3.1.8 BCR 校验和

MSPM0 器件在引导阶段会对 BCR 配置数据执行 CRC 校验。如果器件支持 CRC32-ISO3309，则会应用 32 位 CRC 摘要。否则，会应用 CRC16-CCITT 和 16 位 CRC 摘要。

以下是 CRC 计算的配置：

- 选择具有预选 CRC 标准的多项式（CRC32-ISO3309 或 CRC16-CCITT）
- 反映输入值和输出值
- 初始值设置为 0xFFFFFFFF（或 0xFFFF）
- 最终 XOR 值设置为 0x00000000（或 0x0000）

TI 建议使用 SysConfig 工具自动计算 BCR 校验和，更多详细信息请参阅 节 4.2。

3.1.8.1 CRC 校验失败处理

如果 BCR 配置数据在引导期间未通过 CRC 校验，则会导致灾难性的引导错误并施加以下限制：

- 错误原因作为引导诊断记录在 CFG-AP 中
- BSL 不会被调用，即使 BSL 配置为启用
- 用户应用程序不会启动
- 应用程序调试访问不会启用
- 如果启用了或使用密码启用了，则会执行待处理的 SWD 恢复出厂设置命令
- 如果已启用，则会执行待处理的 TI 失效分析流程条目
- 引导过程最多会重试 3 次
  - 如果第 2 次或第 3 次尝试通过，器件将正常引导
  - 如果第 3 次尝试仍未通过，则在下一次 BOR 或 POR 之前不会再进行启动尝试

3.2 BSL 配置

引导加载程序 (BSL) 例程是基于 ROM 的固件，用于将数据加载到器件存储器中。用户通过 NONMAIN 的 BSL 配置为进入和执行 BSL 设置不同的属性。

TI 建议使用 SysConfig 工具自定义 BSL 配置，更多详细信息请参阅节 4.3。

3.2.1 BSL 配置 ID

TI 为不同的 NONMAIN 布局定义了默认的 BSL 配置 ID。用户可根据应用场景覆盖 BSL ID。执行恢复出厂设置以恢复默认 BSL 配置 ID。

默认 BSL 配置 ID 展示了每个器件的默认值：

表 3-3. 默认 BSL 配置 ID

NONMAIN 布局类型	器件系列	默认值
类型 A	MSPM0L110x	0x00000001
	MSPM0L130x	
	MSPM0L134x	
	MSPM0Gx10x	
	MSPM0Gx50x	

表 3-3. 默认 BSL 配置 ID（续）

NONMAIN 布局类型	器件系列	默认值
类型 B	MSPM0C1103	不支持
	MSPM0C1104	
	MSPS003Fx	
类型 C	MSPM0Lx22x	0x00000001
类型 D	MSPM0C1105	0x05000000
	MSPM0C1106	
	MSPM0H321x	
类型 E	MSPM0L111x	0x06000000
	MSPM0G511x	
	MSPM0G5187	
类型 F	MSPM0Gx51x	0x04000000
	MSPM0G352x	

3.2.2 调用引脚配置

引导加载程序支持在 BOOTRST 之后使用 GPIO 进行硬件调用。TI 器件配置了特定的 GPIO 和极性。请参阅器件特定数据表确定默认调用引脚分配。

用户可使用 BSLCONFIG0 字段修改默认调用引脚。调用引脚配置包含：

- Pad
- 端口
- 引脚
- 极性

若原理图中的默认调用引脚初始状态与调用引脚配置匹配，器件加电后将直接进入 BSL 而不会运行应用程序代码。

3.2.3 基于 ROM 的通信接口

MSPM0 支持可配置的 I2C 或 UART 作为串行线 BSL 接口。基于 ROM 的 BSL 允许用户自定义 UART 或 I2C 引脚，但不支持切换通信接口实例（例如从 UART0 更改为 UART1）。用户可自定义 UART 或 I2C 引脚，默认引脚分配请参阅设备特定数据表。

对于配备 USB 接口的器件，基于 ROM 的接口还支持器件固件升级 (DFU) 选项。

3.2.3.1 UART 接口

对于支持 ROM BSL 的器件，可使用 UART 接口。BSL 配置支持通过 BSLPINCFG0 字段修改 UART 引脚配置（焊盘和 MUX）。由于 UART 实例不变，器件引脚的特性会限制可用引脚范围。请参阅器件特定数据表了解可用引脚。

UART 默认波特率为 9600。用户可通过 BSL 命令动态修改 UART 通信波特率（命令仍以 9600 波特率发送）。此外，NONMAIN 的 E 型布局支持通过 BSLCONFIG1.UART_DEFBAUDRATE 字段修改默认波特率。

3.2.3.2 I2C 接口

对于支持 ROM BSL 的器件，可使用 I2C 接口。BSL 配置支持通过 BSLPINCFG1 字段修改 I2C 引脚配置（焊盘和 MUX）。器件引脚的属性限制了特定 I2C 实例的可用引脚。请参阅器件特定数据表了解可用引脚。

I2C 的默认目标地址为 0x48。MSPM0 支持通过 BSLCONFIG2.I2CTARGETADDR 字段修改目标地址。最大支持 7 位目标地址。

有关 BSL UART 和 I2C 接口的更多详细信息，请参阅 *MSPM0 引导加载程序用户指南*。

3.2.3.3 USB 接口

对于配备 USB 外设的器件，USB 接口可用于基于 ROM 的 BSL。BSL 配置不支持 USB 引脚配置。请参阅器件特定技术参考手册，链接见节 10。

3.2.4 闪存插件接口

ROM 引导加载程序提供了一个选项，可将自定义接口实现作为闪存插件添加到 ROM BSL 内核中。闪存插件功能带来了自定义接口的优势，无需重新实现完整的 BSL 内核。

使用闪存插件接口：

- ROM BSL 中不可用的新接口可添加到用于自动检测的接口列表中
  - 示例：SPI、CAN 等

或

- 可以覆盖 ROM 接口实现 (UART/I2C)
  - 示例：实例选择、参数修改等。

要使用此选项，可将闪存插件映像加载到主闪存中，并在非主闪存中的 BSL 配置中注册映像地址。请确认已为加载插件映像的闪存区域启用闪存静态写保护，以防止在引导加载过程中将闪存插件擦除。有关闪存静态写保护的更多信息，请参阅节 3.1.3。

按照以下步骤启用闪存插件接口：

1. 设置闪存插件接口的存在状态（BSLPLUGINCFG.FLASHPLUGINEXISTS 字段）。
2. 在 BSLPLUGINCFG.PLUGINTYPE 字段中定义插件接口类型。类型选项包括：UART、I2C 或任何新接口。
3. 定义闪存插件接口使用的 SRAM 大小（BSLPLUGINCFG.SRAMUSED 字段）。最大支持 0xFF。
4. 为以下各项定义闪存插件接口函数指针：Init、Receive、Send 和 Deinit API。

闪存插件接口*不*会覆盖基于 ROM 的 BSL 协议。确认用户定义的 API 函数遵循 ROM BSL 标准。有关更多详细信息，请参阅 *MSPM0 引导加载程序*用户指南。

SDK 示例提供了一系列闪存插件接口实现示例。

3.2.5 备用 BSL 接口

ROM BSL 提供了使用备用（次级）BSL 的选项。实现方式：将备用 BSL 映像加载到主闪存中，并将其地址注册到 BSLCONFIG1.ALTBSLCONFIG 字段。

调用 BSL 时，ROM BCR 会检查备用 BSL 是否存在（BSLCONFIG1.ALTBSLCONFIG 字段）。如果启用了备用 BSL，则 ROM BCR 会跳转到备用 BSL。ROM BSL 预计此时不会执行。

BCR 配置中的 BSL 策略 (节 3.1.7) 同样适用于次级 BSL。当禁用 BSL 策略设置时，硬件引脚检查或软件复位不会调用次级 BSL。

为避免在引导加载过程中意外擦除，请在主闪存静态写保护中对加载次级 BSL 的闪存区域启用写保护 (节 3.1.3.1)。对于次级引导加载程序，非主闪存写保护为可选功能。NONMAIN 擦除后，应正确恢复次级 BSL API 指针。

次级 BSL 为 BSL 的自定义提供了更高灵活性。可根据用户期望修改通信接口、协议和安全功能的完整配置。如果器件支持，可按以下步骤启用次级 BSL：

1. 使用 BSLCONFIG1.ALTBSLCONFIG 字段设置主闪存区域中的替代 BSL 调用。
2. 使用 SBLADDRESS.ADDRESS 字段设置主闪存的次级 BSL 入口地址。

有关次级 BSL 实现的更多详情，请参阅 *MSPM0 引导加载程序 (BSL)* 用户指南和 *MSPM0 引导加载程序实现*应用手册。

SDK 示例提供了一系列次级 BSL 接口实现示例。

3.2.6 BSL 安全配置

BSL 内核提供了一个安全功能分支来防止潜在攻击：

- 访问密码
- 读出
- 警报
- 应用程序完整性校验

安全功能在 ROM BSL 和闪存插件接口中始终生效 (节 3.2.4)。用户可在备用 BSL 接口中完全自定义和覆盖 BSL 行为 (节 3.2.5)，因此安全功能对于备用 BSL 而言是可选项。

3.2.6.1 访问密码

使用用户指定的密码保护 ROM BSL 访问。密码位于 PWDBSL.PASSWORD 字段中。无法选择禁用 BSL 访问密码。

在向 BSL 内核发送正确的密码来解锁 BSL 之前，大多数 BSL 功能均无法访问。有关更多详细信息，请参阅 *MSPM0 引导加载程序 (BSL) 用户指南*。如果向 BSL 提供了错误的密码，BSL 会暂停 2 秒，随后可以再次尝试发送正确的密码。在密码尝试失败三次后，将激活安全警报功能 (节 3.2.6.3)。

BSL 访问密码支持两种类型：256 位明文和 256 位 SHA 加密。256 位明文密码直接存储在 BSL 配置中。另一种密码是 256 位 SHA 加密密码（原始密码的长度为 256 位），可参阅 节 3.1.2.2 了解 SHA2-256 计算过程。BSL 配置存储明文密码（256 位，所有字节均为 0xFF）或 SHA 加密密码（256 位，根据所有字节均为 0xFF 的 256 位数据计算得出）的默认值。

3.2.6.2 读出功能

支持 ROM BSL 的器件在 BSL 配置中具有一个可配置的属性，允许通过 BSL 接口读回器件存储器数据。所有闪存均可通过 BSL 主机完全访问，但部分 SRAM 的访问受限。有关更多详细信息，请参阅 *MSPM0 引导加载程序 (BSL) 用户指南*第 3.3.1 节（SRAM 存储器的使用）。

出于安全考虑，读出功能默认处于禁用状态。用户可通过 BSL 配置中的 BSLCONFIG0.READOUTEN 字段启用读出功能。

3.2.6.3 警报功能

如果发生超过 3 次 BSL 密码校验错误，ROM BSL 会启动警报功能。默认情况下，当发出安全警报时，器件不会执行额外操作。用户可通过 BSLCONFIG2.ALERTACTION 字段配置警报行为。

在 BSL 配置中，支持两种额外的警报行为：

- 触发 BSL 恢复出厂设置。如果 MAIN 或 NONMAIN 闪存中的扇区受到静态写保护，这些扇区不会受到 BSL 恢复出厂设置的影响。
- 重新配置 NONMAIN 区域以禁用 BSL。如果 NONMAIN 闪存受到静态写保护，则不支持重新配置。

设置闪存静态写保护 (节 3.1.3)，以保留特定 MAIN 闪存或整个 NONMAIN 闪存。

3.2.6.4 应用程序完整性校验

BSL 支持通过 BSL 接口返回应用程序版本号。这项支持使得 BSL 主机能够在不能读取固件的情况下查询固件版本。版本字段地址的长度为 32 位。在相应的 MAIN 闪存中对版本值进行编程。

使用 BSLAPPVER.ADDRESS 字段设置应用程序版本字的地址。只有当指定地址对应于有效的闪存地址时，才会返回版本数据。如果未对给定的闪存地址进行编程，则 ROM BSL 将返回 0h。

3.2.7 BSL 校验和

MSPM0 器件在引导阶段会对 BSL 配置数据执行 CRC 校验。BSL CRC 校验和与 BCR 校验和采用相同的计算流程。有关详细说明，请参阅节 3.1.8。

TI 建议使用 SysConfig 工具自动计算 BSL 校验和，更多详细信息请参阅节 4.3。

3.2.7.1 CRC 校验失败处理

如果 BSL 配置数据在 BSL 调用期间未通过 CRC 校验，则会导致灾难性的引导错误并施加以下限制：

- 错误原因作为引导诊断记录在 CFG-AP 中
- BSL 不会被调用，即使 BSL 配置为启用
- 用户应用程序不会启动
- 应用程序调试访问不会启用
- 启动过程最多会重试 3 次
  - 如果第 2 次或第 3 次尝试通过，器件将正常启动
  - 如果第 3 次尝试仍未通过，则在下一次 BOR 或 POR 之前不会再进行启动尝试

4 使用 SysConfig 进行 NONMAIN 配置

本节介绍如何使用 TI 图形配置工具 SysConfig 来配置 NONMAIN 内存的各种功能选项。

4.1 SysConfig 介绍

SysConfig 工具是一个直观而全面的图形实用程序集合，用于配置引脚、外设、子系统和其他元件，可帮助用户直观地管理、发现和解决冲突，以便将更多时间投入到创建应用程序上。SysConfig 的输出文件包括头文件和源文件，这些文件可与软件开发套件 (SDK) 示例配合使用或用于配置自定义项目。

SysConfig 工具以独立安装程序形式提供。可手动将安装程序链接到 Code Composer Studio (CCS)、IAR 或 Keil，或使用云工具门户。作为 TI 生态系统的一部分，CCS 默认集成了 SysConfig。有关更多详细信息，请参阅 SysConfig 用户指南。

图 4-1 展示了该工具中用于配置 NONMAIN 区域的部分，用户可以选择*问号*来了解该元件的更多详细信息。

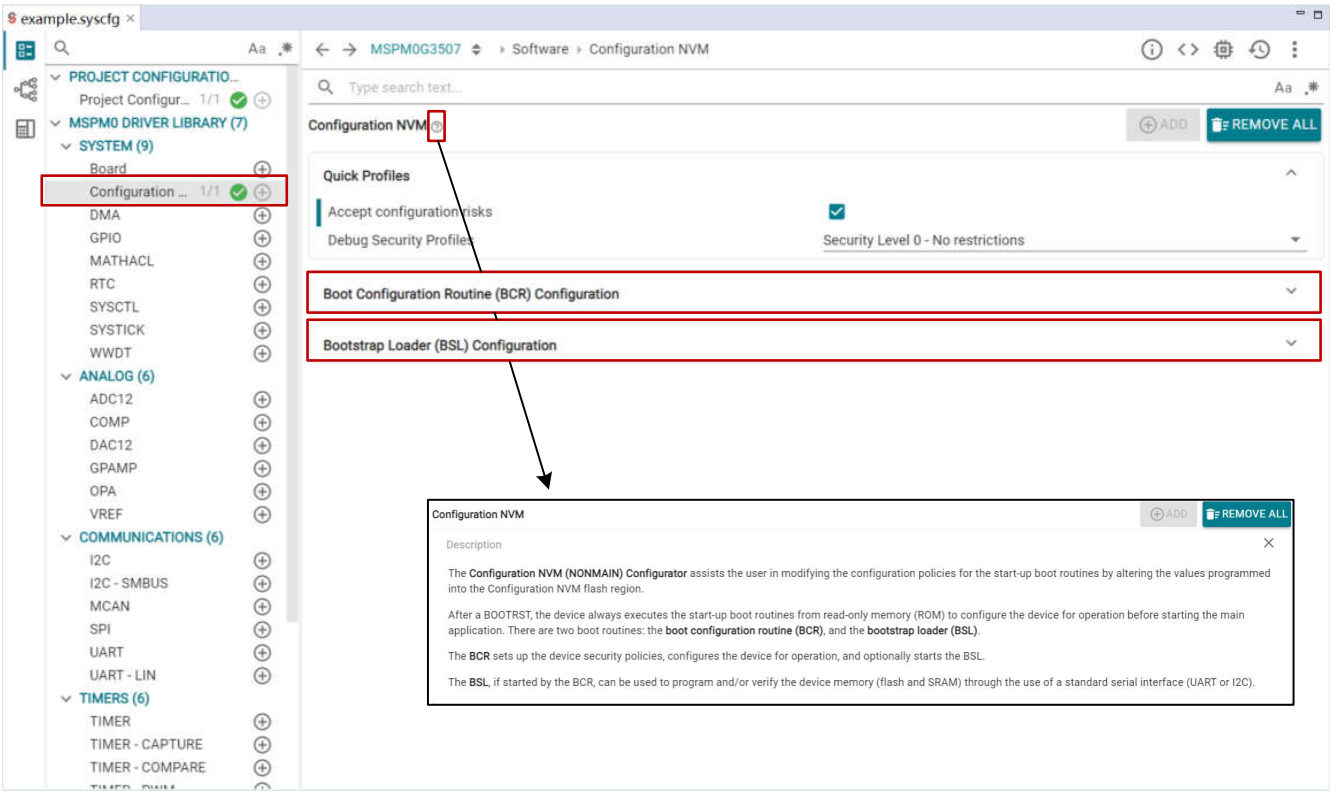


图 4-1. SysConfig 工具：配置 NVM

4.2 使用 SysConfig 进行 BCR 配置

图 4-2 展示了 MSPM0G3507 的 BCR 配置。根据 BCR 配置，调试安全配置文件可设置为 0 级、1 级和 2 级，详细说明请参阅 节 3.1.2。BCR 配置 ID 使用默认值，且不支持在工具中修改。CRC 校验和根据自定义 BCR 配置自动计算得出。

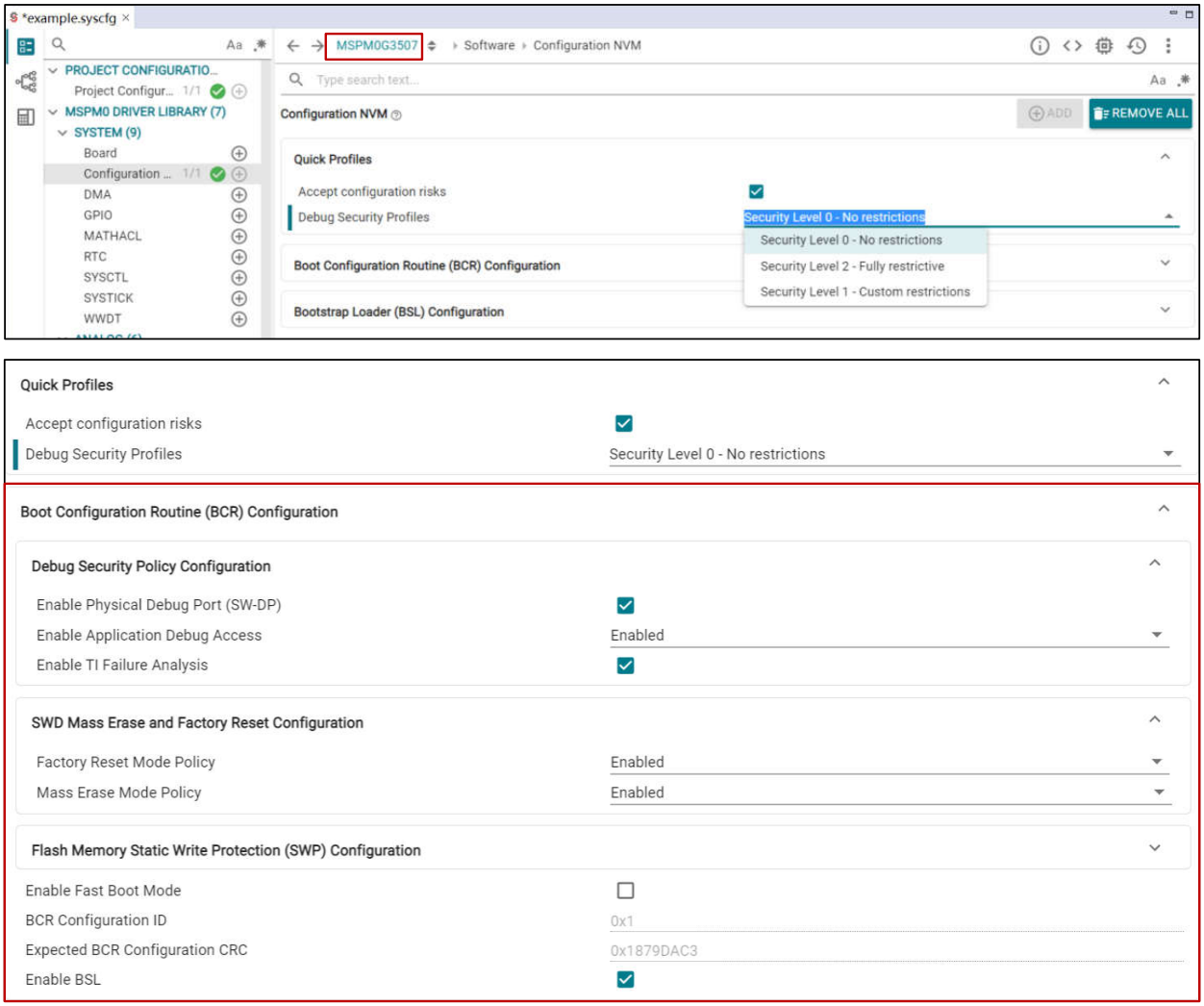


图 4-2. BCR 配置

4.2.1 密码配置

使用 128 位明文密码来加密 SWD 接口、恢复出厂设置和批量擦除。对于支持 SHA2-256 密码的器件，可在 NONMAIN BCR 配置中设置 256 位加密密码。图 4-3 展示了 MSPM0G3507（型号 A）的 SWD 密码配置，以及 MSPM0G3519（型号 F）的 SWD 密码配置。

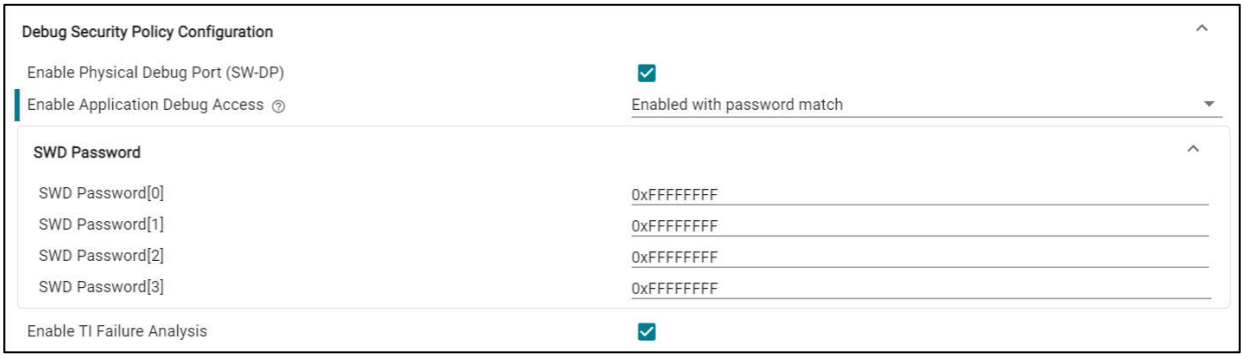


图 4-3. 明文 SWD 密码配置

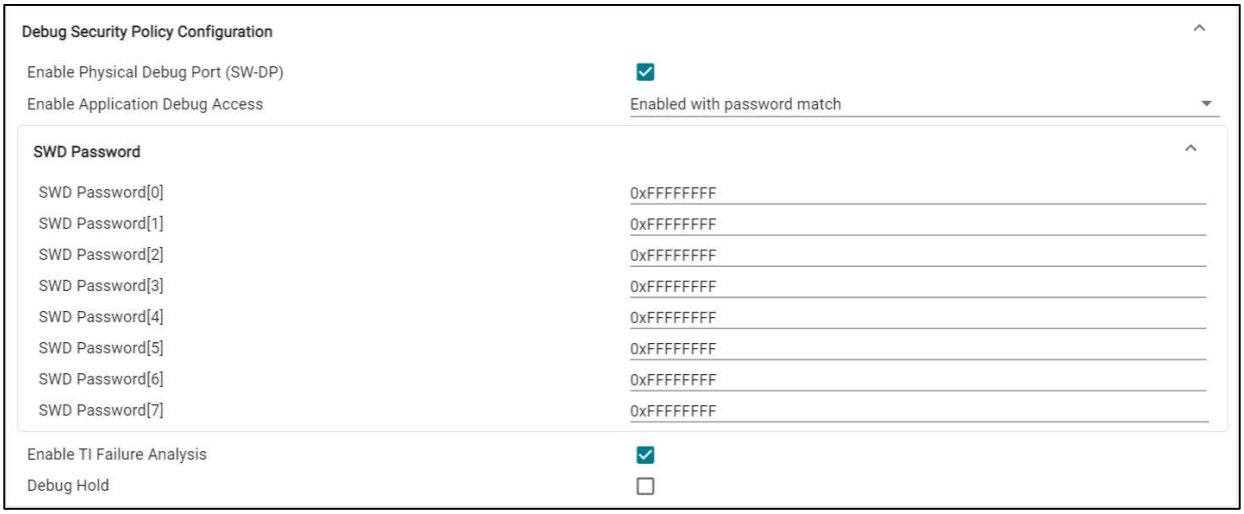


图 4-4. SHA2-256 SWD 密码配置

SysConfig 工具集成了 SHA2-256 密码生成流程（请参阅节 3.1.2.2），如图 4-5 所示。如果器件支持 SHA2-256 密码，可选择 *SWD Password* 旁边的问号来显示流程示例。示例中还演示了 SHA256 在线工具。

示例流程适用于所有 256 位 SHA2-256 密码的生成，包括：

- SWD 访问权限
- 恢复出厂设置
- 批量擦除
- BSL 访问权限

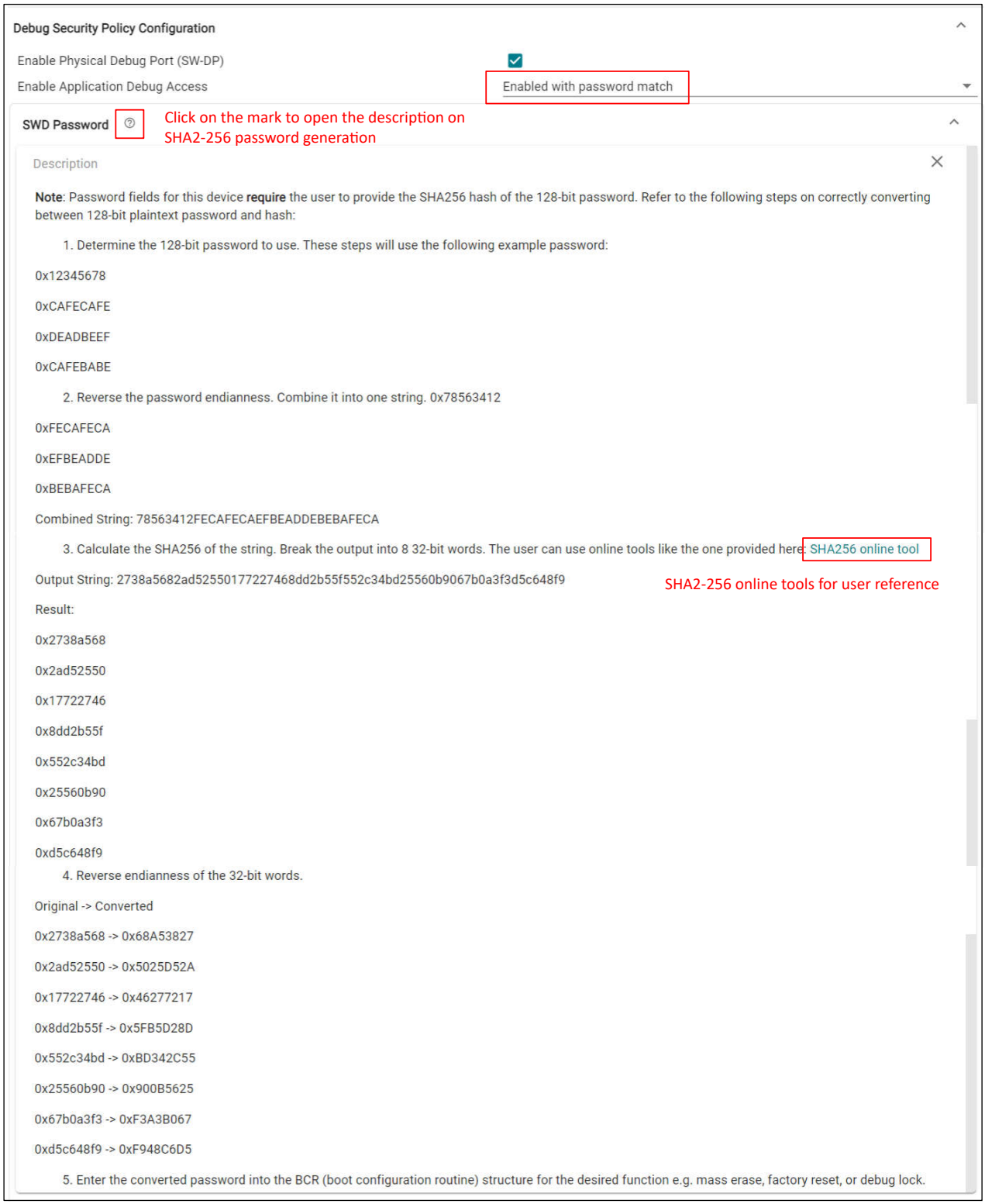


图 4-5. SHA2-256 密码生成流程

4.2.2 闪存静态写保护

要启用闪存静态写保护，可将 MAIN SWP 的位设置为 0。最小存储器范围为 1KB 或 8KB，具体取决于受保护的扇区顺序。如需了解更多详细信息，请参阅节 3.1.3。

图 4-6 展示了如何配置 MSPM0G3507 的静态写保护。对于 FLASHSWP0（低地址扇区），1 位表示闪存中的 1 个扇区 (1KB)，数值 0 表示对相应扇区应用写保护。

对于 FLASHSWP1（其余扇区），1 位表示闪存中的 8 个扇区。低 4 位及高 16 位会被忽略，即 FLASHSWP1 保护 32KB 至 128KB 的存储器地址。数值 0 表示对相应扇区应用写保护。

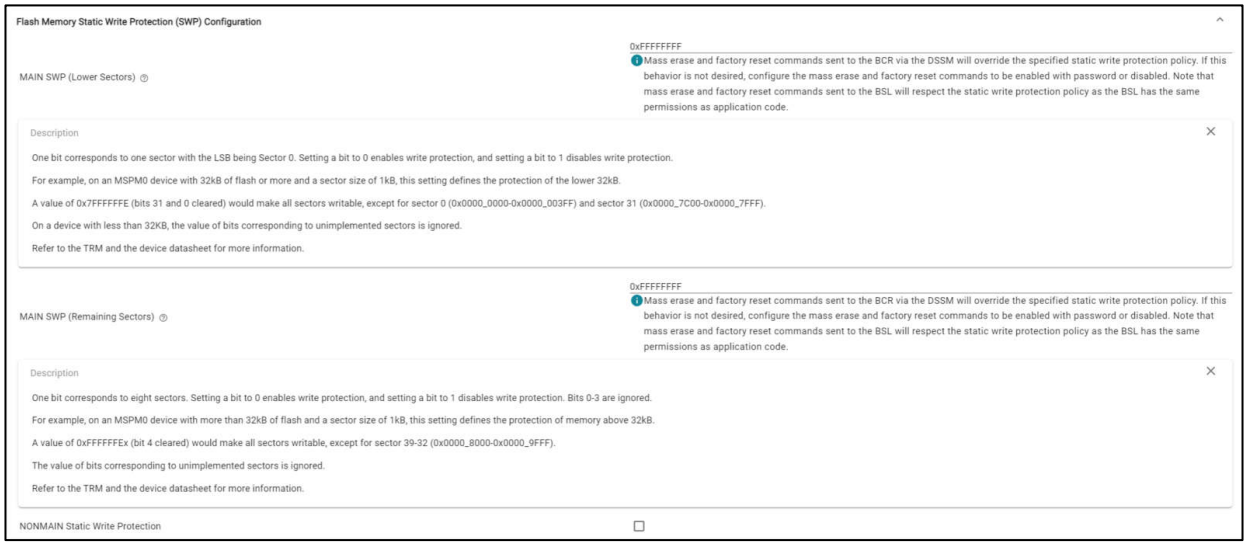


图 4-6. 闪存静态写保护

SysConfig 工具同样支持 NONMAIN 静态写保护。器件锁定后，将启用 NONMAIN 区域的 FLASHCTL 写/擦除操作。只有通过 DSSM 命令触发的恢复出厂设置可恢复 NONMAIN 区域。

4.2.3 其他 BCR 配置

SysConfig 中的一些 BCR 配置非常简单，可以启用或禁用，包括：

- 启用快速引导模式
- 启用 TI 故障分析
- 启用调试保持（如果支持）
- 启用 CSC 策略（如果支持）
- 启用闪存存储体交换策略（如果支持）
- 启用 BSL 模式（如果支持）
- 启用 BSL 调用引脚检查（如果支持）
  - 在 SysConfig 工具中，*BSL 配置*（而非 *BCR 配置*）包括 *Enable BSL Invoke Pin Check* 选项，以及其他 BSL GPIO 调用配置

备注

SysConfig 工具不支持应用程序摘要校验配置。有关更多详细信息，请参阅 节 3.1.6。

4.3 使用 SysConfig 进行 BSL 配置

图 4-7 展示了 MSPM0L1107 的 BSL 配置（类型 E 布局）。

图 4-7. BSL 配置

4.3.1 BSL 访问密码

为了使器件支持 ROM 或闪存 BSL，256 位 BSL 访问密码始终处于启用状态。MSPM0 支持两种密码类型。第一种为 256 位明文密码，默认值为 0xFF。如 节 4.3 所示，第二种为 SHA2-256 加密密码，原始密码全为 0xFF（256 位），因此 SysConfig 工具会将原始密码生成的 SHA2-256 值存储至 PWDBSL.PASSWORD 字段。

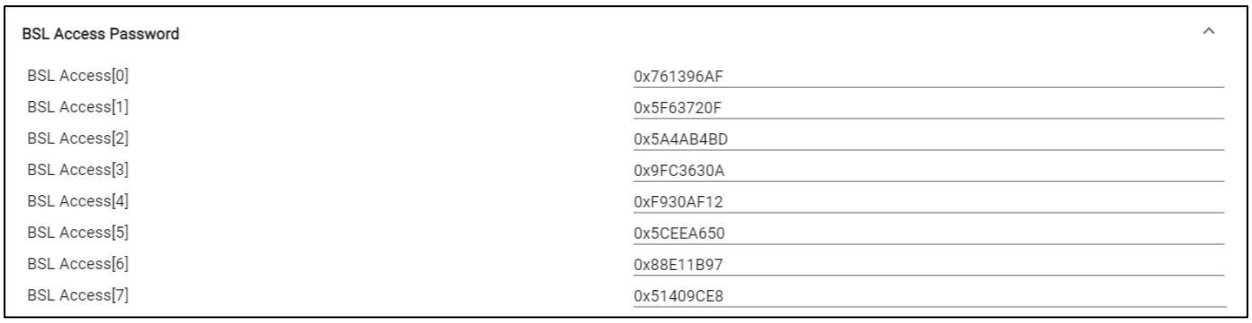


图 4-8. SHA2-256 BSL 默认访问密码

4.3.2 BSL 调用引脚配置

该器件使用自定义调用引脚属性来支持 ROM 或闪存 BSL。节 4.3 展示了该配置的详细信息。

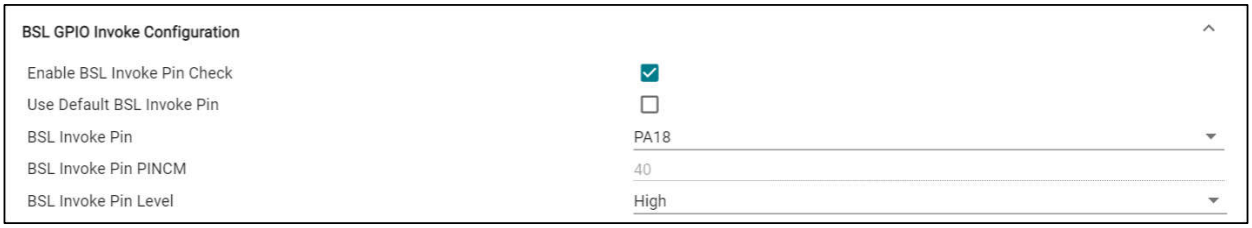


图 4-9. BSL 调用引脚

如果用户在 BSL 配置中禁用了 BSL 调用引脚检查，则 BCR 会跳过硬件引脚调用检查，同时仍可通过软件进入 BSL。禁用后，在 SysConfig 工具中，BCR 配置的 BOOTCFG1.BSL_PIN_INVOKE 字段设置为 0xFFFF，BSL PIN 配置保持不变。

4.3.3 BSL 通信接口

对于支持 ROM BSL 的器件，提供了 I2C 和 UART 接口作为通用 BSL 接口。对于支持 USB 外设的器件，可使用支持 DFU 的 USB 接口作为 BSL 接口。

BSL 配置使用户能够灵活地自定义 BSL 接口，如 节 4.3 所示。默认情况下，BSL 配置支持引脚选择和 I2C 目标地址更改。对于支持 UART 波特率修改的器件，用户可为 UART 通信选择九种不同的常用波特率。否则，UART 初始波特率设置为 9600。为实现更高数据吞吐量，可采用初始 9600 波特率发送 BSL 命令来修改 UART 波特率。

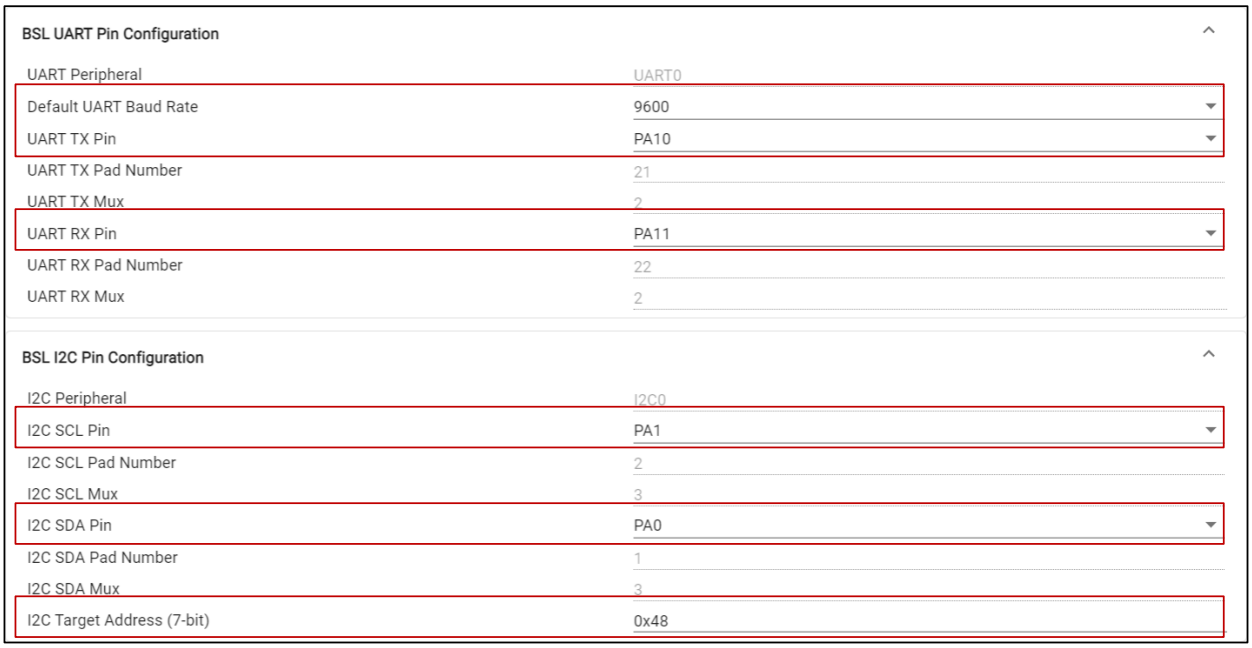


图 4-10. BSL 通信接口

4.3.4 闪存插件接口

对于支持 ROM BSL 的器件，BSL 配置为用户提供了灵活性，可自定义将加载至 MAIN 闪存的 BSL 接口（如 SPI、UART、IIC、CAN 等）。在 NONMAIN BSL 配置中注册闪存插件接口的存储器地址。

节 4.3 显示闪存插件接口有 4 个 API，包括：

- Init
- Receive
- 发送
- De-init

ROM BSL 通过 BSL 配置中注册的钩子函数调用 4 个 API。有关更多详细信息，请参阅 *MSPM0 引导加载程序 (BSL) 用户指南*的第 7 章（*接口插件*）。

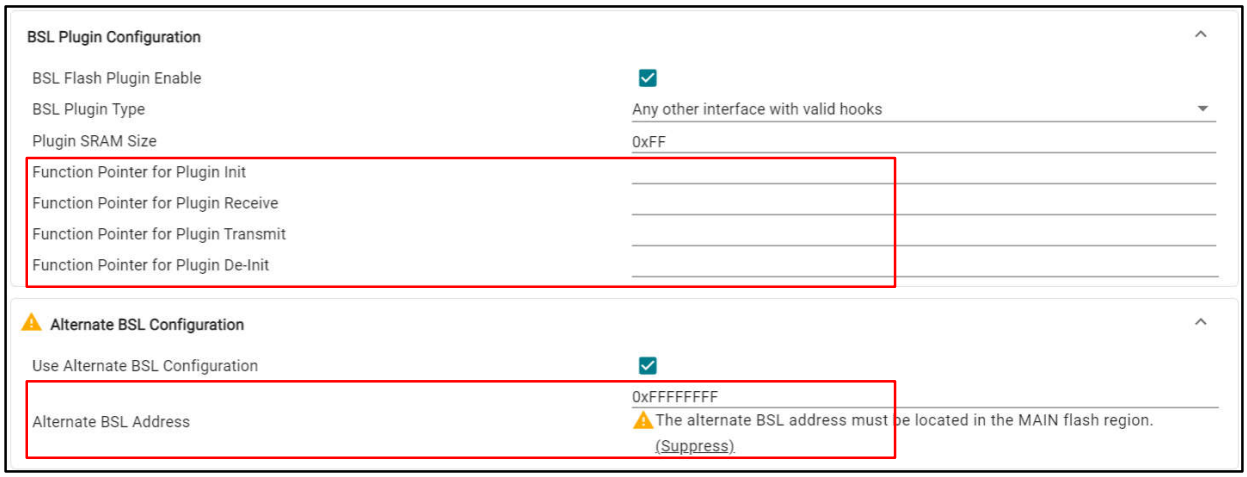


图 4-11. BSL 插件和备用 BSL 配置

4.3.5 备用 BSL 接口

在 SysConfig 工具中选择使用备用 BSL 配置后，用户可将备用 BSL（位于 MAIN 闪存中）的起始地址设置到 BSL 配置寄存器中，如节 4.3 所示。有关更多详细信息，请参阅 *MSPM0 引导加载程序 (BSL) 用户指南*第 6 节（*次级加载程序*）。

启用备用 BSL 后，当 BCR 将 BSL 调用条件置为有效时，会跳过 ROM BSL 并调用备用 BSL。

4.3.6 其他 BCR 配置

除 BSL 配置 ID 外，还有若干功能支持通过 SysConfig 工具进行自定义（并非所有 NONMAIN 布局类型均适用），如 图 4-7 所示，包括：

- BSL 应用程序版本
- 启用 BSL 读出
- BSL 安全警报配置
- 禁用 NRST

当 BSL 配置发生变更时，SysConfig 工具将自动计算 *CRC 校验和*值，用户只需专注于自定义 BSL 功能即可轻松生成 NONMAIN 固件文件。

5 在应用程序代码中进行 NONMAIN 配置

NONMAIN 区域也支持通过 FLASHCTL 进行动态修改，该功能广泛应用于应用程序代码和自定义引导加载程序中。图 5-1 和 SDK 示例提供了在应用程序代码中修改 NONMAIN 闪存字段的通用流程，可供用户参考。

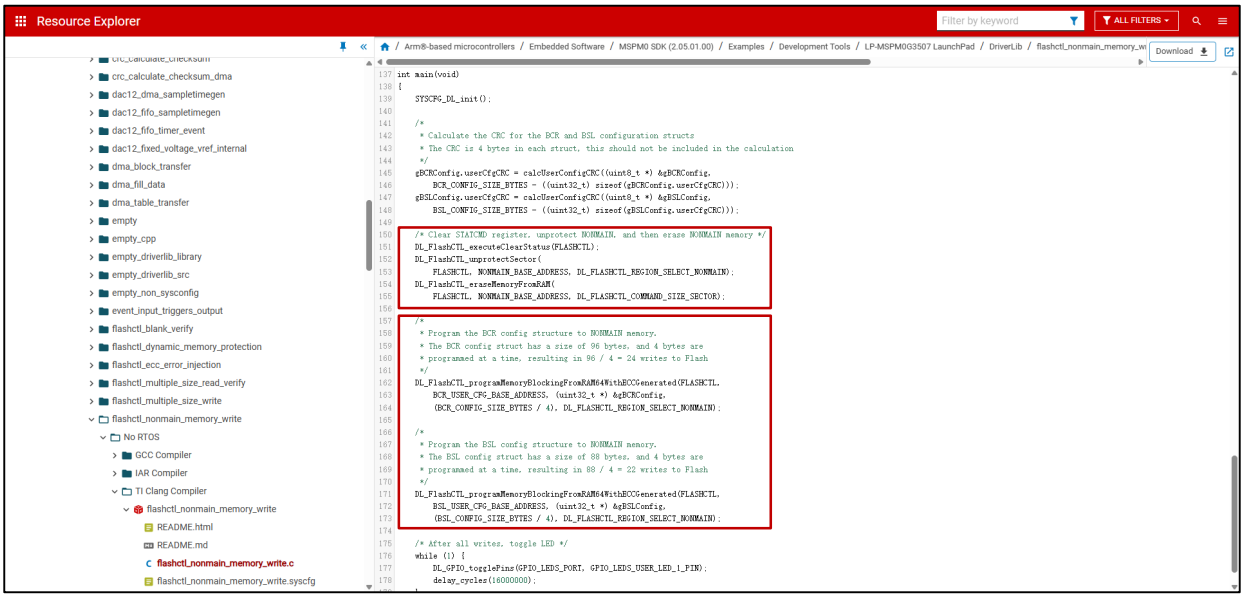


图 5-1. 包含 NONMAIN 修改的示例项目

编程 NONMAIN 存储器时需要 ECC 代码，因为引导代码会通过 ECC 校验读取 NONMAIN 存储器。每次修改 NONMAIN 寄存器都需要用户擦除整个 NONMAIN 扇区，并加载整个 NONMAIN 配置寄存器的新值。

TI 建议在安全状态下擦除 NONMAIN 字段并对其进行编程。在应用程序代码中频繁更新 NONMAIN 配置会引入该字段变空（不稳定的电源等）的风险。

6 使用 IDE 工具进行 NONMAIN 操作

TI 建议使用 SysConfig 工具管理 NONMAIN 配置。SysConfig 工具已集成至 Code Composer Studio (CCS)。TI 提供了独立版 SysConfig，以便与 Keil 或 IAR 等其他 IDE 工具配合使用。有关在不同 IDE 工具中使用 SysConfig 的更多详情，请参阅 节 10。

6.1 NONMAIN 配置文件

图 6-1 显示 SysConfig 工具生成两个文件（boot_config.c 和 boot_config.h）来配置 NONMAIN 闪存。

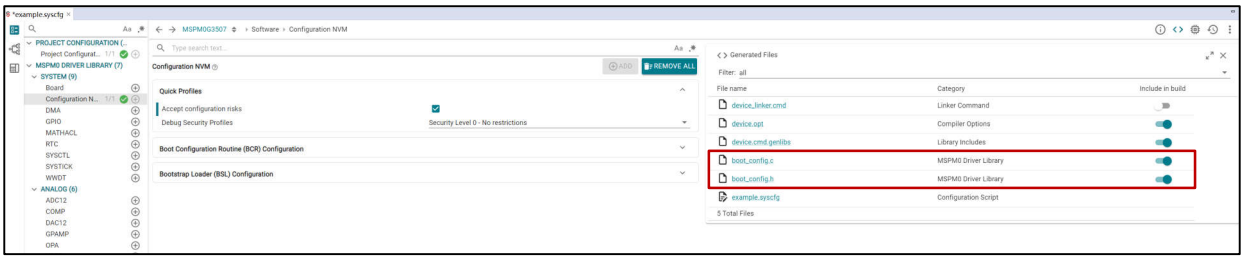


图 6-1. 配置文件

boot_config.c 文件包含 NONMAIN 结构的所有配置，这些配置会自动编译到输出文件中。boot_config.c 文件包含 SysConfig 工具中设置的默认值。用户*无法*手动修改这两个文件，因为 SysConfig 工具会管理和覆盖这些文件。当用户在 NVM 元件中修改 project-specific.syscfg 文件后，这两个文件会重新生成并更新至最新配置。

要使用 SysConfig 进行额外修改，可通过取消勾选 ***include in build*** 按钮来排除这两个文件的编译。这样，SysConfig 生成的文件将不会编译，用户可手动将这些文件添加到项目并进行额外修改。

6.2 项目擦除属性

NONMAIN 闪存区域属于非易失性存储器，写入新值前需执行擦除操作。要成功将 NONMAIN 区域固件加载到器件中，需设置擦除属性来擦除 NONMAIN。

对于 Code Composer Studio：右键单击项目打开项目属性。图 6-2 展示了详细信息。

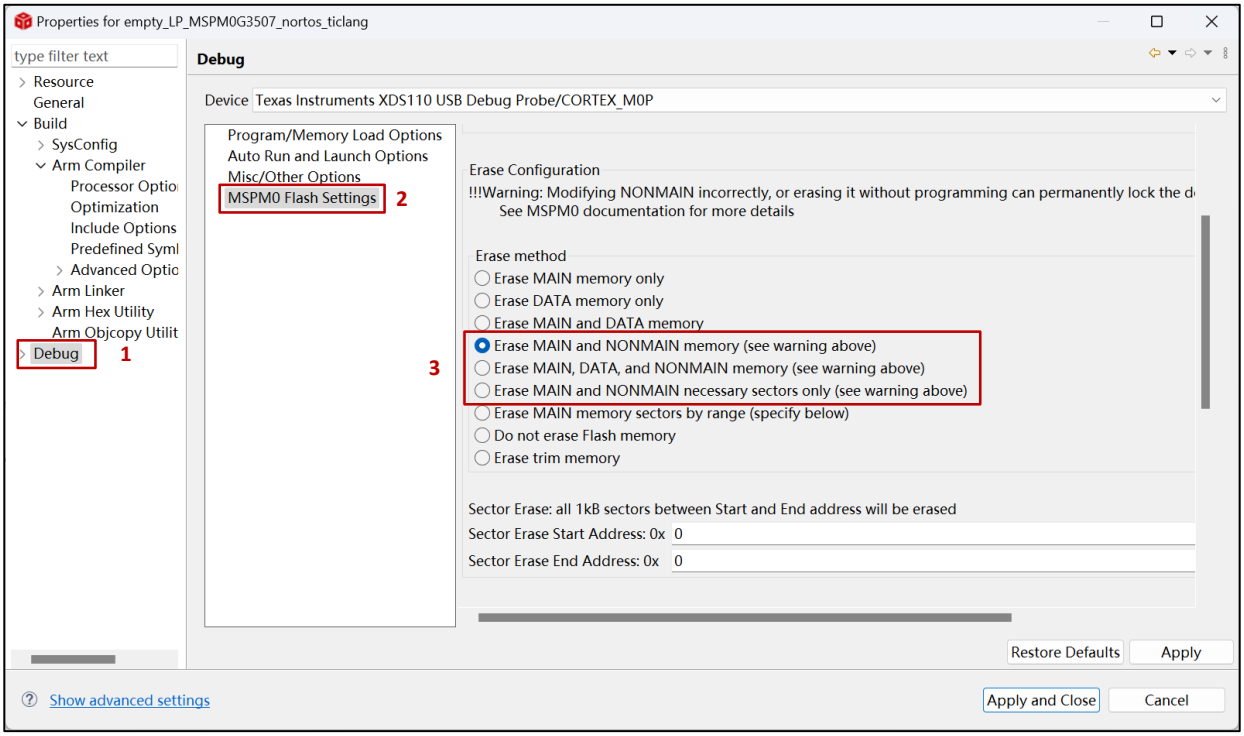


图 6-2. CCS 擦除属性设置

对于 Keil：选择并打开目标选项，在调试设置中添加 NONMAIN。图 6-3 展示了 Keli 擦除属性设置。

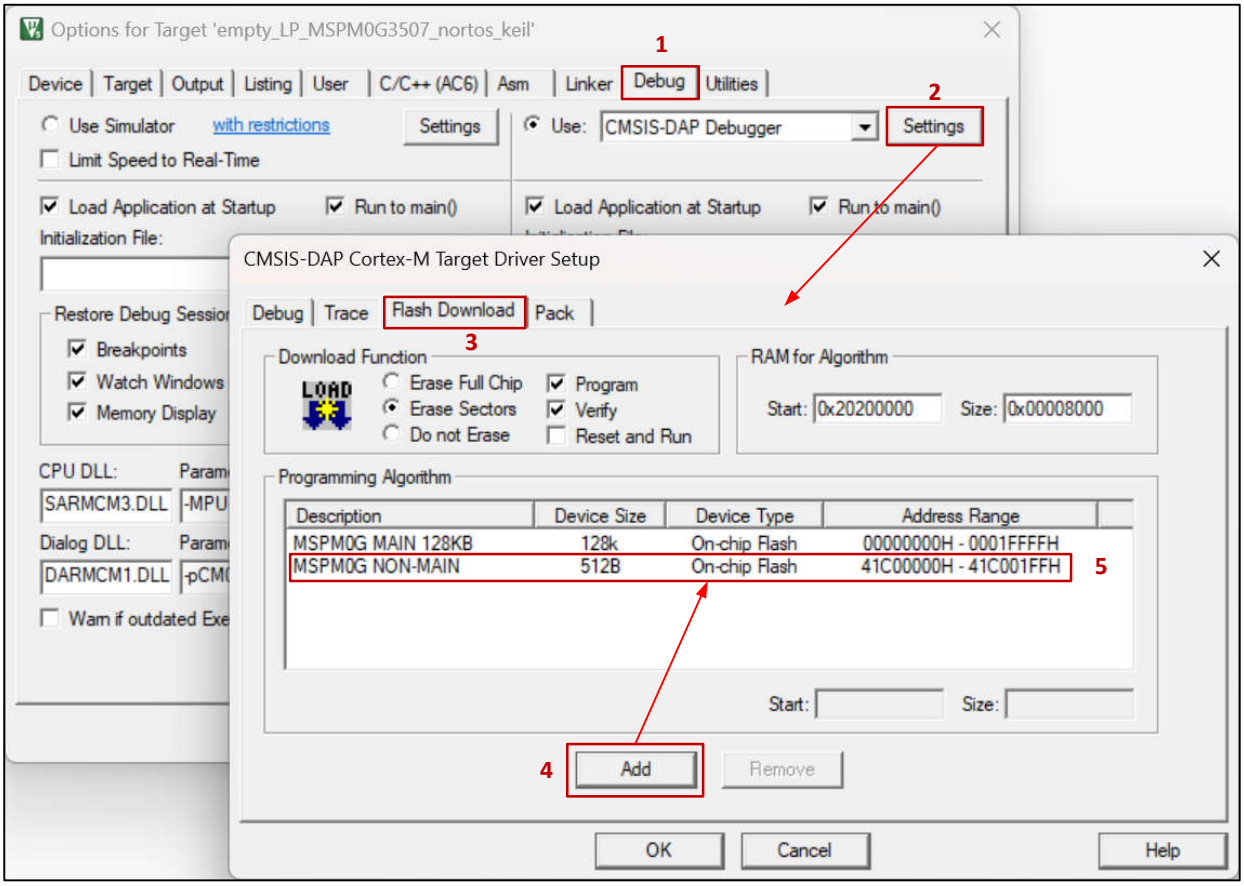


图 6-3. Keil 擦除属性设置

对于 IAR：右键单击项目打开选项。确认已通过额外参数添加非主区域。图 6-4 展示了 IAR 设置。

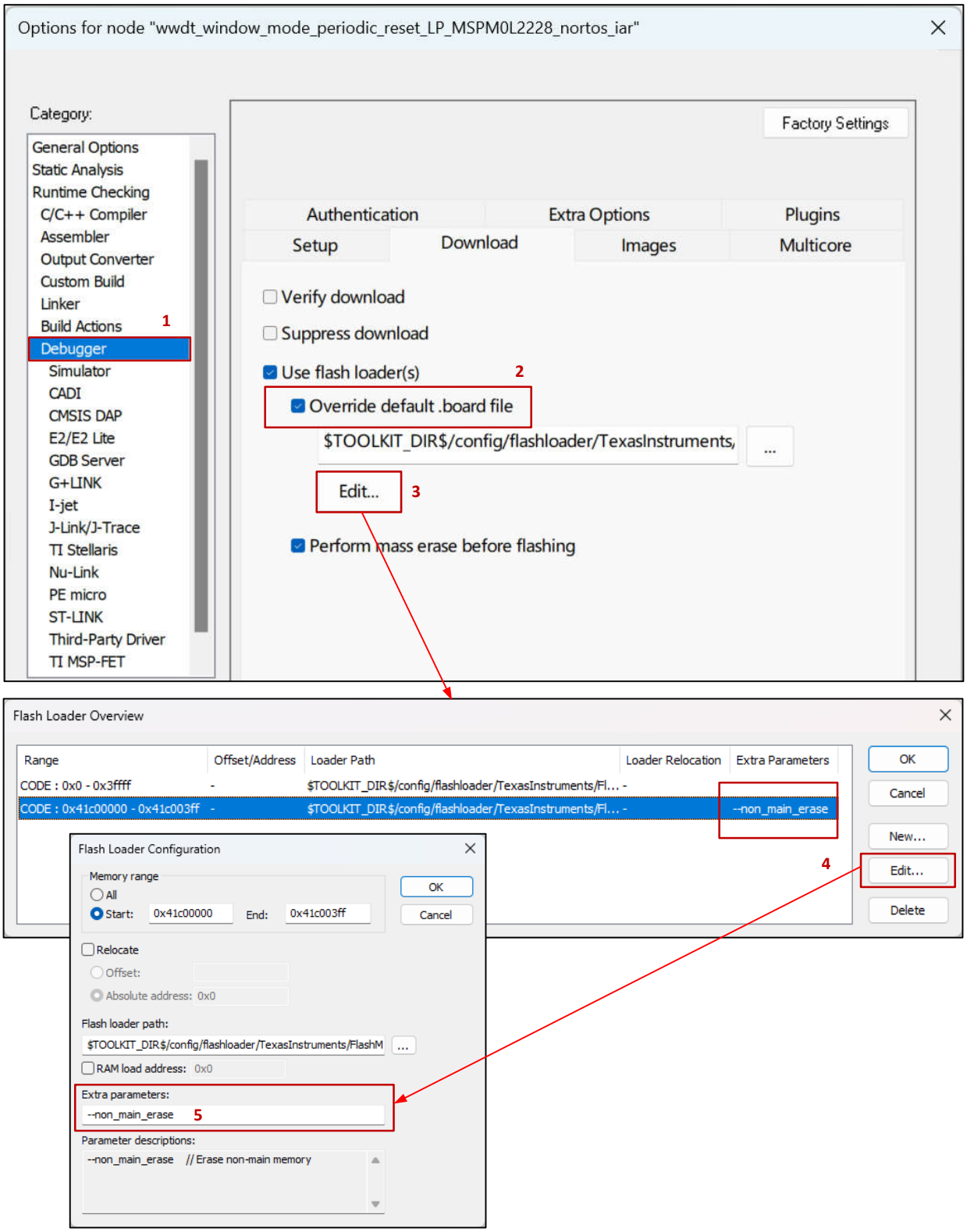


图 6-4. IAR 擦除属性设置

完成 NONMAIN 配置并编程后，TI 建议移除 NONMAIN 字段，避免每次对 NONMAIN 重复编程相同值。

6.3 密码保护调试

当 SWD 调试策略设置为密码加密时，需在调试或下载新固件前执行用于 SWD 密码认证的 DSSM 命令。

TI 通过 XDS110 工具为 MSPM0 调试配置提供了.ccxml 文件。图 6-5 展示了为 DSSM 命令设置 128 位密码的步骤：

1. 在 *targetConfigs* 文件夹中，打开 *.ccxml* 文件。
2. 选择 *Advanced* 以打开 XDS110 *Target Configuration*。
3. 选择 *MSPM0 器件*以打开 *Device Properties* 页。
4. 在 SWD Password 框中输入 128 位密码。

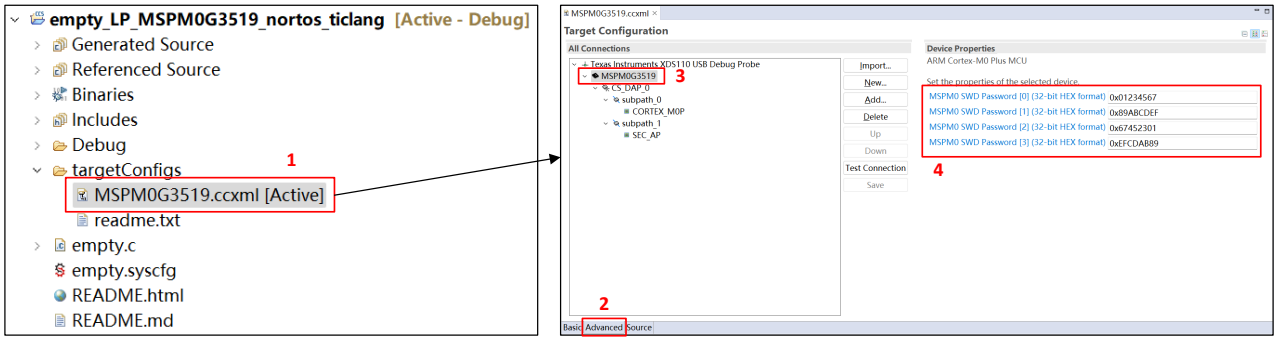


图 6-5. 使用 .ccxml 文件设置密码

备注

.ccxml 文件中设置的密码适用于所有 DSSM 操作密码，包括 SWD 访问、恢复出厂设置和批量擦除。

设置正确的密码后，运行 DSSM 脚本以解锁 SWD 接口。图 6-6 展示了使用 CCS v12 运行 DSSM 脚本的步骤：

1. 在顶部菜单中的 *View* 下，选择 *Target Configurations*。
2. 选择目标项目的 *.ccxml* 文件。.ccxml 文件设置了密码。
3. 选择 *Launch the selected configuration*。
4. 在调试窗口中，选择 *XDS110 USB Debug Probe*。
5. 在顶部菜单中，选择 *Scripts*。
6. 选择目标 *MSPM0 器件*以打开命令列表。
7. 选择 *DebugAccessPasswordAuthentication_Auto* 以生成 DSSM 命令。CCS 会自动调用 DSSM 命令并发送 .ccxml 文件中的密码。
8. 命令执行完成且器件解锁后，通过 *Load Program or Load Symbol* 执行进一步调试。

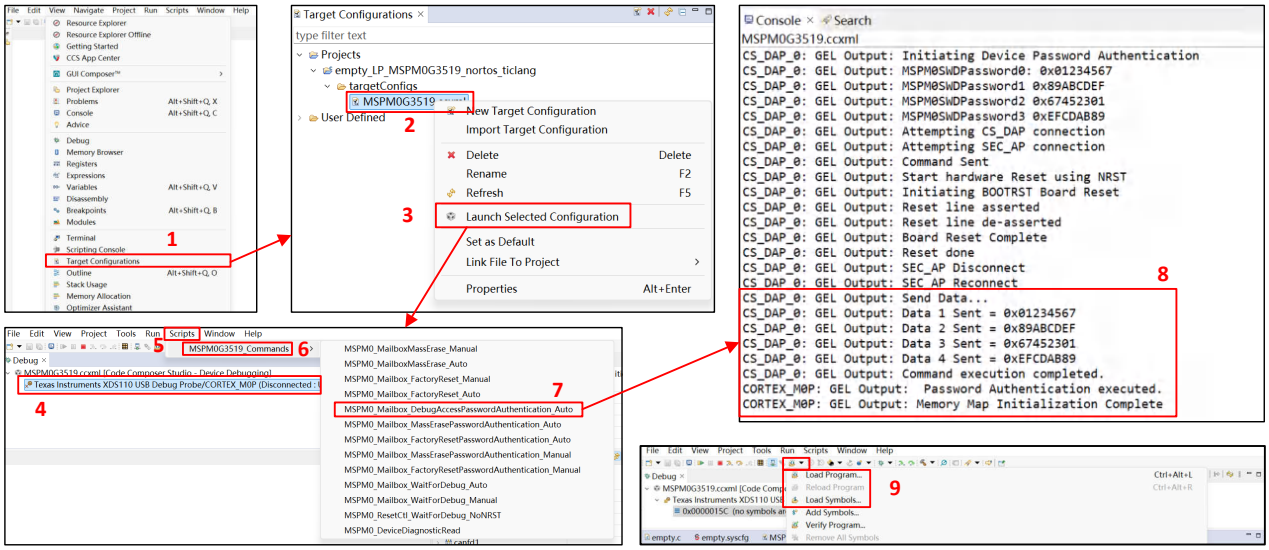


图 6-6. 带 SWD 密码认证的 DSSM 命令

如果用户在 CCS v20 中调试，则执行 DSSM 命令的过程略有不同。图 6-7 展示了该过程。

1. 在项目中，找到 *targetConfigs* 并选择 *.ccxml* 文件。
2. 右键单击 .ccxml 文件并选择 *Start Project-less Debug*。
3. 在顶部菜单中，选择 *Scripts*，然后选择 *MSPM0 Device Commands*。
4. 选择并执行 DSSM 命令。选择 *DebugAccessPasswordAuthentication_Auto* 以解锁 SWD 接口。
5. 成功执行完命令后，找到顶部菜单。选择 *Run*，然后选择 *Debug Project* 或 *Flash Project*。

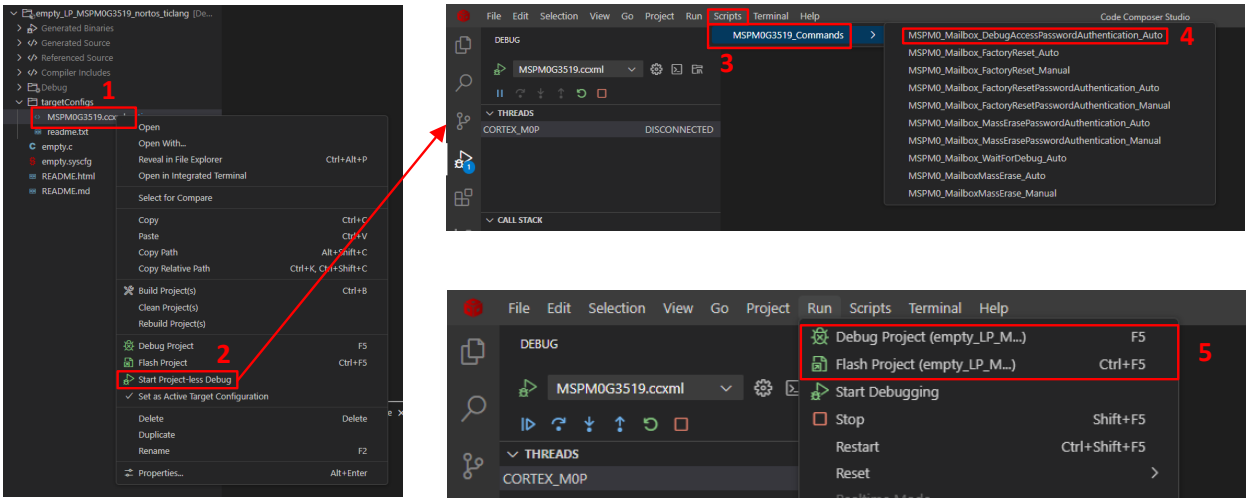


图 6-7. 使用 CCS v20 的 DSSM 命令

备注

成功执行调试访问的密码认证序列后，器件将保持解锁状态直至下次 BOOTRST。

7 使用编程工具进行 NONMAIN 操作

NONMAIN 是闪存的专用区域，用于存储 BCR 和 BSL 用于引导器件的配置数据。该区域不用于任何其他目的。要更改引导配置中的任何参数，有必要擦除整个 NONMAIN 扇区，并使用所需的设置对 BCR 和 BSL 配置结构进行重新编程。

针对 NONMAIN 字段为空可能导致器件永久锁定的风险，TI 建议用户缩短 NONMAIN 为空并等待新数据加载的持续时间。以下是两种首选方法：

- 单独擦除 NONMAIN 后立即加载 NONMAIN。
- 依次擦除 MAIN 和 NONMAIN，然后依次写入 NONMAIN 和 MAIN。
  - 若编程器无法验证擦除与编程顺序，则分别加载 MAIN 和 NONMAIN 固件。

7.1 使用 UniFlash 进行 NONMAIN 操作

UniFlash 是一款软件工具，用于对 TI 微控制器和无线连接器件上的片上闪存以及 TI 处理器的板载闪存进行编程。UniFlash 提供图形界面和命令行界面。

要对 NONMAIN 闪存进行编程，可选择包括 NONMAIN 存储器的擦除属性。图 7-1 展示了该过程。

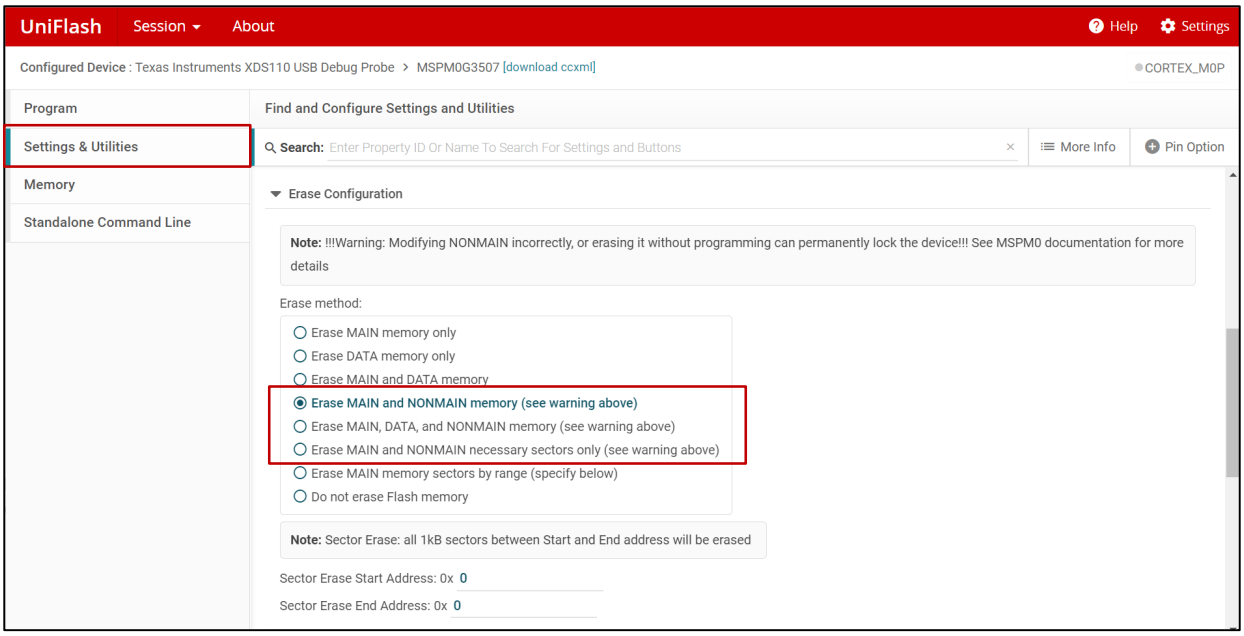


图 7-1. 使用 UniFlash 进行 NONMAIN 操作

要仅对 NONMAIN 进行编程，可按以下步骤操作：

1. 准备固件以仅包含 NONMAIN 区域固件。TI-Hex 或 Intel-Hex 格式可轻松生成单独的 NONMAIN 区域固件。
2. 选择擦除属性 *Erase MAIN and NONMAIN necessary sectors (see warning above)*。
3. 对器件进行编程。仅 NONMAIN 会被擦除并重新编程。

7.2 使用 J-Flash 进行 NONMAIN 操作

J-Flash 是 SEGGER 开发的闪存编程软件，让用户能够对嵌入式微控制器 (MCU) 的内部和外部闪存进行编程。可使用 GUI (J-Flash) 或命令行界面控制 J-Flash。

要对 NONMAIN 字段进行编程，需启用 NONMAIN 存储体并选择包括 NONMAIN 字段的擦除属性。图 7-2 展示了使用 J-Flash 对 NONMAIN 进行操作。

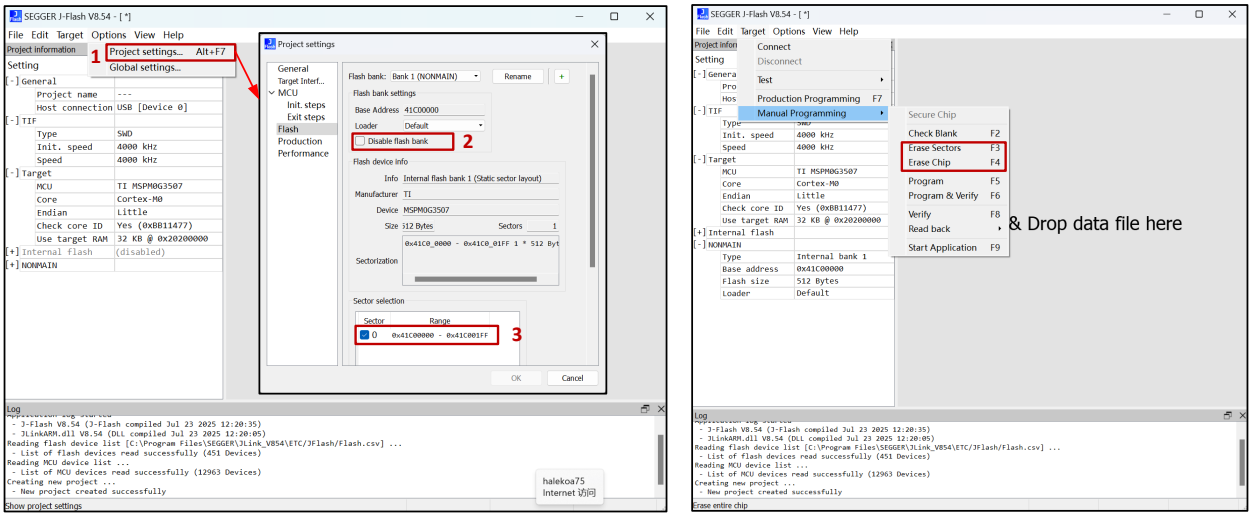


图 7-2. 使用 J-Flash 进行 NONMAIN 操作

如果选择手动编程方法，请确保在加载新 NONMAIN 固件之前器件未复位。如果器件复位，空 NONMAIN 将导致器件永久锁定。

备注

TI 建议为新 MSPM0 器件使用最新版 J-Flash。

要仅对 NONMAIN 进行编程，可按以下说明操作：

1. 准备固件，仅包含 NONMAIN 区域固件。
2. 选择排除了 MAIN 闪存的擦除属性：禁用 MAIN 闪存或不选择任何 MAIN 闪存扇区。
3. 使用 *erase sectors* 对器件进行编程。仅 NONMAIN 字段会擦除并重新编程。

7.3 使用 C-GANG 进行 NONMAIN 操作

C-GANG 是一款低成本的 gang 编程器，可同时对多达六个相同的目标进行编程。C-GANG 与多种 TI 微控制器兼容，包括 MSPM0 和 MSP430 器件。

C-GANG 支持对 NONMAIN 字段进行编程，可按以下步骤操作：

1. 选择 *Setup*。选择 *Memory* 以打开 NONMAIN 配置窗口
2. 直接在 C-GANG GUI 中修改 NONMAIN 配置（左侧），或者从代码文件导入配置（右侧）
3. 在 Memory Protection 中，选择 *Enable*。
4. 选择 *Secure / Protect* 可单独对 NONMAIN 字段进行擦除或编程。
5. 选择 *AUTO PROG* 可依次擦除 MAIN 并对 MAIN 字段进行编程。擦除 NONMAIN 字段并对其进行编程。

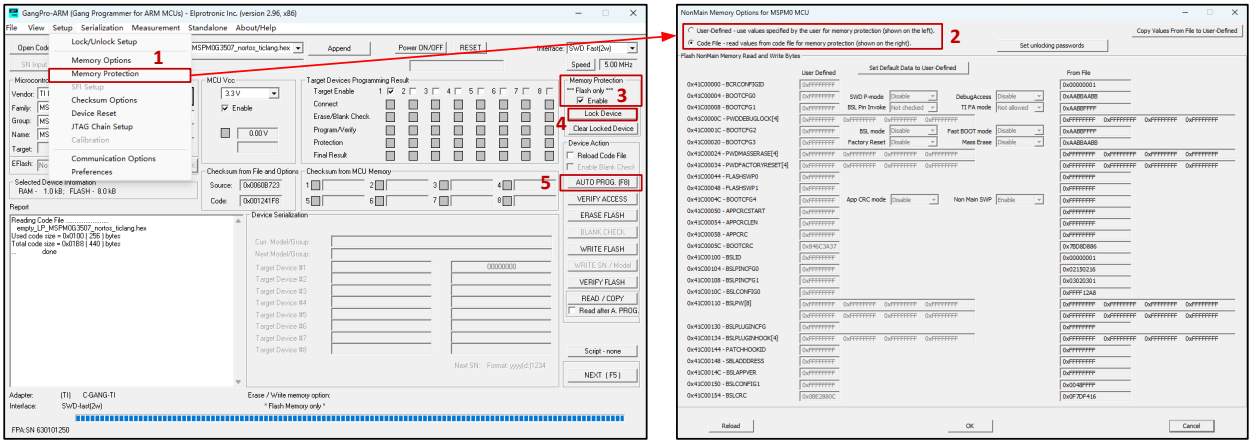


图 7-3. 使用 C-GANG 进行 NONMAIN 操作

7.4 使用 MSP-GANG 进行 NONMAIN 操作

MSP-Gang 编程器 (MSP-GANG) 是一款 MSPM0/MSP430™/MSP432™ 集成电路器件编程器，可同时对多达八个完全相同的 MSPM0/MSP430/MSP432 闪存或 FRAM 器件进行编程。MSP-Gang 编程器使用标准的 RS-232 或 USB 连接与主机 PC 相连，并提供灵活的编程选项，让用户可以完全自定义流程。

备注

MSP-GANG 的数量有限，而且不支持更新版本的 MSPM0 器件。请参阅 C-GANG 了解我们新的快速 Gang 编程解决方案。

MSP-GANG 支持在 NONMAIN 字段中进行编程，可按以下步骤操作：

1. 找到 Memory 选项，选择 *All Memory*。
2. 选择 *Secure / Protection Option* 以打开 NONMAIN 配置窗口。
3. 选择 *NONMAIN Memory Write Enable*。
4. 直接在 MSP-GANG GUI 中修改 NONMAIN 配置（左侧），或者从代码文件导入配置（右侧）。
5. 要擦除并编程 NONMAIN 字段，可选择 *Secure / Protect*。

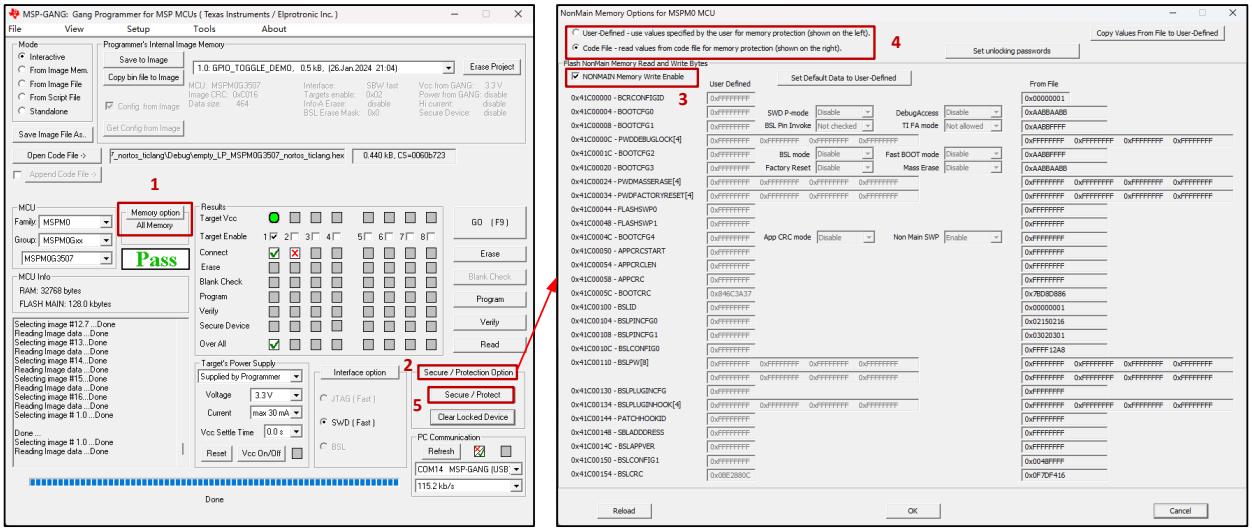


图 7-4. 使用 MSP-GANG 进行 NONMAIN 操作

8 常见问题解答 (FAQ)

8.1 MCU 锁定状态分析

如果 MSPM0 进入锁定状态，MSPM0 可能*无法*通过 SWD 接口建立连接。本节介绍了 MCU 被锁定的常见原因。

8.1.1 硬件问题分析

以下硬件因素可能会导致器件锁定：

- 硬件电路设计缺陷
- 连接调试器时出现问题
- 周期性外部复位信号（至 NRST）

8.1.1.1 硬件电路设计

确认 MSPM0 在少数电源引脚中配备了合适的外部电容器，以确保稳定运行。检查连接至 VDD、Vcore（如果支持）和 NRST 引脚的电路和引脚电压。

可在器件特定数据表的*基本应用原理图*部分找到参考电路。Vcore 电压（如果支持）应为 1.35V（典型值），表示用于数字内核功能的内部 PMU 电路工作正常。

8.1.1.2 调试器连接

使用独立调试器调试 MSPM0 器件时，请检查硬件引脚分配。为确保成功连接器件，需验证 SWDIO 和 SWCLK 是否按正确顺序可靠连接。有时调试器默认*不*会向器件供电；请确认器件已通过板载或外部电源轨正常供电。

若在噪声环境中调试 MSPM0，可尝试降低调试速率，或在 SWCLK 和 SWDIO 引脚上并联一个小电容器。

8.1.1.3 外部复位信号

如果 NRST 线路上连接外部复位信号，调试 MSPM0 器件时需禁用该复位信号或保持该信号为高电平。向 NRST 引脚发出复位信号（低电平有效）会中断 SWD 连接。如果存在持续复位信号，器件可能进入锁定状态。

8.1.2 软件问题分析

以下软件因素可能会导致器件锁定：

- CPU 进入故障状态
- BCR 配置
- 低功耗模式（STOP 或 STANDBY）
- SHUTDOWN IO 状态
- SWD IO（SWDLCK 或 SWDIO）功能被禁用
- WDT 或 IWDT 被复位（如果在应用程序代码中启用）
- 软件 POR 或 BOOTRST

8.1.2.1 CPU 进入故障状态

当 CPU 进入故障状态时，ARM 器件会自动执行连续自复位，直到 CPU 退出故障状态。CPU 在发生嵌套异常时进入故障状态（例如双硬故障或 NMI），该异常由非法 CPU 活动引起（例如无效地址修改或外设错误配置）。

当 CPU 进入故障状态时，持续复位行为会中断 SWD 连接，导致器件进入锁定状态。

8.1.2.2 BCR 配置

当 BCR 配置设置为禁用 SW-DP 或禁用 SWD 访问时，器件将在调试模式下进入锁定状态。当 BCR 配置设置为加密 SWD 访问时，若加载了错误的密码，器件将在调试模式下进入锁定状态。

此外，加载错误的 NONMAIN 配置可能导致器件进入永久锁定状态。

8.1.2.3 低功耗模式（STOP 或 STANDBY）

当器件进入 STOP 或 STANDBY 模式时，无法发现 AHB-AP（ARM 调试）内核，从而使器件进入锁定状态。若要主动连接到 AHB-AP，请验证调试器是否首先连接了 PWR-AP。有关更多详细信息，请参阅*适用于 MSPM0 的硬件编程和调试器指南*应用手册。

XDS110 支持在 STOP 或 STANDBY 模式下连接器件。对于其他第三方调试器（例如 J-Link），可尝试使用最新软件版本来解决此低功耗模式调试问题。

8.1.2.4 SHUTDOWN IO 状态

数字 IO 引脚在进入 SHUTDOWN 模式时会被锁存并保持。数字 IO 引脚状态包括：

- 输出低电平或高电平
- 上拉或下拉
- 启用输入或输出
- 驱动配置

退出 SHUTDOWN 模式后，IO 保持上一状态，直到应用软件通过设置 SHDNIOREL 寄存器中的 RELEASE 位以及匹配的 KEY 值来释放该状态。

退出 SHUTDOWN 模式时，串行线调试 (SWD) 引脚也会保持锁定状态，直到应用软件设置 RELEASE 位。因此，从 SHUTDOWN 模式唤醒时无法建立调试连接，直到应用软件释放 IO。

TI 调试工具 XDS110 现在支持直接调试从 SHUTDOWN 模式唤醒的器件。

8.1.2.5 SWD IO 功能

POR 之后，当引导过程完成且 CPU 启动应用程序时，串行线调试 (SWD) IO 将配置为 SWD 模式（SWDIO 拉至高电平、SWCLK 拉至低电平）。

当不再需要调试支持时，可以在软件中将 SWD 引脚重新配置为通用 IO (GPIO)，以便在应用中使用这些引脚。要禁用 SWD 功能，请设置 SYSCTL 的 SWDCFG 寄存器中的 DISABLE 位以及 KEY (62h)。

一旦 SWD 引脚功能被禁用，器件将保持锁定状态，只有触发 POR 才能重新启用 SWD 引脚功能。

8.1.2.6 WDT 或 IWDT 复位

如果未在正确的时间窗口内馈送 WDT 或 IWDT，会产生复位并可能导致器件进入锁定状态。

通常，调试器通过 SYSCTL 生成 SYSRST 来禁用 WDT。如果未产生复位，或设置为 CPU 级复位，WDT 可能导致器件进入锁定状态，并在 SWD 连接阶段将复位置为有效。

对于包含 IWDT 外设的 MSPM0 器件，调试器会产生 LFSS 复位或 IWDT 复位来禁用 IWDT。否则器件可能进入锁定状态，因为 SYSRST 不会复位 LFSS 元件（包括 IWDT）。

8.1.2.7 软件 POR 或 BOOTRST

软件 POR 或 BOOTRST 会中断 SWD 连接，如果在建立 SWD 连接时发生中断，可能导致器件进入锁定状态。

请禁用软件 POR 和 BOOTRST 以稳定地调试 MSPM0 器件。

8.2 解锁 MSPM0 器件

本节介绍在调试访问因软件问题而失败时如何解锁器件。

8.2.1 强制 MCU 进入 BSL 模式

对于支持 BSL 且 BSL 已启用（默认启用）的器件，可使用硬件调用方法强制 MCU 进入 BSL 模式。此时 MCU 将执行 ROM 中的 BSL 代码，不会运行应用程序代码。器件在 BSL 模式下仍保持 SWD 连接。

以下是强制 MCU 进入 BSL 模式的方法：

- 将 BSL 调用引脚（默认为 PA18）连接到 VCC（使用或不使用上拉电阻器）。重新为器件供电。
- 将 BSL 调用引脚（默认为 PA18）连接到 VCC（使用或不使用上拉电阻器）。强制拉低 NRST 超过 1 秒以触发 POR。

备注

如果 MCU 进入 BSL 模式后 10 秒内未向 UART 或 I2C 接口发送 BSL 连接命令，MCU 将进入 STANDBY0 模式。

如果软件问题导致 MCU 进入锁定状态，此方法可帮助用户解锁器件。这种方法在以下情况下不可行：

- NONMAIN 配置 CRC 校验失败。
- NONMAIN 配置禁用了 SWD 接口或锁定了调试访问。
- 器件支持 VBAT 功能且启用了 IWDT，需要用户手动关闭 VBAT 以禁用 IWDT。
- BSL 硬件调用方法被禁用。

8.2.2 发送 BSL 命令

如果 NONMAIN 配置禁用了 SWD 接口或锁定了调试访问，可在进入 BSL 模式后发送 BSL 命令来解锁器件。

应用 BSL 通信接口（请参阅 节 3.2.3）来发送 BSL 命令。ROM BSL 内核支持恢复出厂设置命令，以恢复默认的 NONMAIN 配置。有关更多详细信息，请参阅 *MSPM0 引导加载程序用户指南*。

这种方法在以下情况下不可行：

- BCR 配置禁用了恢复出厂设置。
- BCR 配置启用了 NONMAIN 静态写保护。
- BCR 配置禁用了 BSL。

8.2.3 执行 DSSM 命令

为了实现复位功能，MSPM0 支持 DSSM 命令，包括批量擦除和恢复出厂设置。请参阅节 6.3，其中以 DebugAccessPasswordAuthentication_Auto 命令为例，介绍了如何使用 CCS 工具演示 DSSM 命令。其他 DSSM 命令（恢复出厂设置或批量擦除）采用相同的方法，包括在 .ccxml 文件的相同位置设置 4 个 32 位密码。

以下是 CCS 工具支持的命令列表：

- DebugAccessPasswordAuthentication_Auto
- FactoryReset_Auto
- FactoryReset_Manual
- FactoryResetPasswordAuthentication_Auto
- FactoryResetPasswordAuthentication_Manual
- MassErase_Auto
- MassErase_Manual
- MassErasePasswordAuthentication_Auto
- MassErasePasswordAuthentication_Manual

TI 建议使用 FactoryReset_Auto 解锁器件。FactoryReset_Auto 命令可在 NRST 引脚上生成复位信号并恢复默认 NONMAIN 配置。

如果软件问题中断了恢复出厂设置过程（通常由软件复位引起），可按照以下步骤选择 FactoryReset_Manual 来解锁器件：

1. 按照 节 6.3 中所示的步骤启动配置文件 (CCS v12) 或启动无项目调试 (CCS v20)。
2. 使 NRST 线路保持低电平状态（连接到 GND）。
3. 使用 FactoryReset_Manual 执行 DSSM 命令（NRST 线路保持低电平）。
4. 当 CCS 控制台窗口出现 *Press the reset button* 时，释放 NRST 线路（断开与 GND 的连接）。
5. 恢复出厂设置命令随即执行。

若 FactoryReset_Manual 解锁器件失败，可通过将 BSL 调用引脚连接至 VCC 进一步强制 MCU 进入 BSL 模式（前提是 BSL 模式存在且已启用），然后执行 FactoryReset_Auto 命令。在命令执行期间，需保持 BSL 调用引脚（默认为 PA18）连接到 VCC，以避免器件运行应用程序代码。

在以下情况下，这种方法不可行：

- NONMAIN 配置禁用了 SWD 恢复出厂设置命令。
- 器件支持 VBAT 功能且启用了 IWDT，需要用户手动关闭 VBAT 以禁用 IWDT。

请参阅 *MSPM0 MCU 开发指南*用户指南第 7.1 节（*解锁 MCU*），使用不同的工具生成恢复出厂设置命令。

8.3 调试错误概述

本节介绍使用 CCS 工具时常见的调试错误，并详细说明可能的恢复方法。

8.3.1 无错误代码：DAP 连接错误

可能的根本原因：

- SWD 连接问题
- MSPM0 器件损坏
- CPU 进入故障状态
- 在连接期间发生软件（POR 或 BOOTRST）或硬件复位

可能的恢复方法：

- 检查 SWD 硬件连接
- 检查 VCC、NRST 和 Vcore（如果支持）是否在数据表规格范围内
- 强制拉低 NRST，然后连接器件。如果没有出现 DAP 连接错误，尝试 节 8.2 中详述的方法

8.3.2 无错误代码：连接到 MSPM0 内核失败

可能的根本原因：

- NONMAIN 配置会禁用或加密 SWD 接口
- CPU 进入故障状态
- 在连接期间发生软件（POR 或 BOOTRST）或硬件复位

可能的恢复方法：

- 确认 NRST 线路上没有发生低电平脉冲
- 如果对 SWD 接口进行加密，则通过 DSSM 生成 SWD 访问密码认证
- 请按照 节 8.2 中列出的步骤操作

8.3.3 错误 - 6305：PRSC 模块无法写入例程寄存器

请参阅 节 8.3.2。

8.3.4 错误 - 260：尝试连接到 XDS110 失败

可能的根本原因：

- XDS110 未连接或被占用。
- 无效的 XDS110 固件、无效的 XDS110 序列号等。

可能的恢复方法：

- 将 XDS110 断电重启并检查 USB 连接。
- 复位 XDS110 固件。

8.3.5 错误 - 261：来自 XDS110 的无效响应

可能的根本原因：

- XDS110 调试器进入故障状态。
- CPU 进入故障状态。

可能的恢复方法：

- 对 XDS110 调试器断电重启。
- 请按照 节 8.2 中的说明操作。

8.3.6 错误 - 615：目标无法识别格式正确的 SWD 标头

请参阅 节 8.3.5。

8.3.7 错误 - 1001：此器件不支持请求的操作

可能的根本原因：

- 发生软件或硬件复位。
- CPU 进入故障状态。

可能的恢复方法：

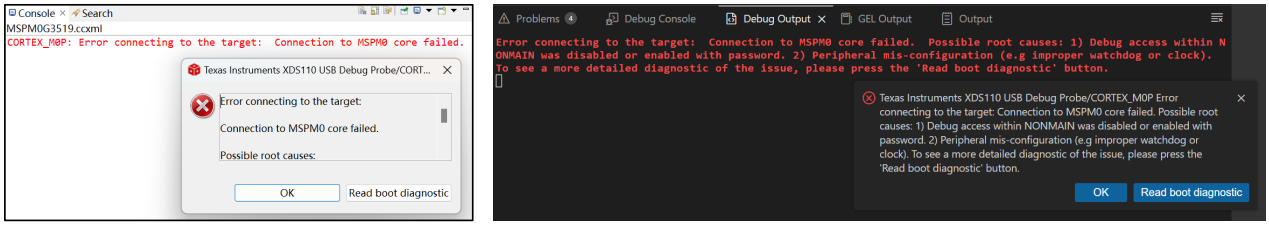
- 检查确认 NRST 线路上没有发生低电平脉冲
- 如果设备进入锁定状态，请按照 节 8.2 中的说明操作。

8.3.8 错误 - 2131：无法访问器件寄存器

请参阅 节 8.3.7。

8.4 MSPM0 引导诊断

当器件进入锁定状态且 DAP 保持连接时，CCS 工具提供了读取 MSPM0 器件引导诊断信息的方法，如 图 8-1 所示。



CCS Version 12.x　　CCS Version 20.x

图 8-1. 使用 CCS 工具读取引导诊断信息

所列为常见的引导诊断读取值。

表 8-1. 常见引导诊断读取值

器件诊断读取值	详细说明
0x0000.0000	引导失败。可能的原因是 NRST 线路保持低电平或时钟源无效。
0x0000.0007	引导成功。应用程序代码已执行并使器件进入锁定状态。
0x0001.0136	引导失败。可能的原因是 NONMAIN CRC 校验和验证失败。
0x0004.110A	引导成功。BSL 被调用并执行。
0x0007.0000	引导失败。在引导代码中触发 NMI 错误。

9 总结

本用户指南概述了 MSPM0 系列中的 NONMAIN 配置，并提供了配置 NONMAIN 寄存器的分步说明。除了基本知识之外，本文档还为用户列出了参考资料和进一步阅读材料。

10 参考资料

- 德州仪器 (TI)，*MSPM0 C 系列 24MHz 微控制器*技术参考手册
- 德州仪器 (TI)，*MSPM0 G 系列 80MHz 微控制器*技术参考手册
- 德州仪器 (TI)，*MSPM0 H 系列 32MHz 微控制器* 技术参考手册
- 德州仪器 (TI)，*MSPM0 L 系列 32MHz 微控制器*技术参考手册
- 德州仪器 (TI)，*MSPM0 MCU 开发指南*用户指南
- 德州仪器 (TI)，*MSPM0 引导加载程序*用户指南
- 德州仪器 (TI)，*MSPM0 引导加载程序实现*应用手册
- 德州仪器 (TI)，*MSPM0 系列中的闪存多存储体功能*应用手册
- 德州仪器 (TI)，*MSPM0 MCU 中的网络安全机制*应用手册
- 德州仪器 (TI)，*适用于 MSPM0 的硬件编程和调试器指南*应用手册
- 德州仪器 (TI)，*安全引导用户指南*
- 德州仪器 (TI)，*适用于 Code Composer Studio v12 (Eclipse) 的 MSPM0 SDK QuickStart 指南*
- 德州仪器 (TI)，*适用于 Code Composer Studio v20 QuickStart 指南*
- 德州仪器 (TI)，*适用于 IAR 的 MSPM0 SDK QuickStart 指南*
- 德州仪器 (TI)，*适用于 Keil 的 MSPM0 SDK QuickStart 指南*
- *德州仪器 (TI)，UniFlash 闪存编程工具*
- *德州仪器 (TI)，C-GANG 设计和开发工具*
- *德州仪器 (TI)，MSP-GANG 设计和开发工具*
- *SEGGER Microcontroller GmbH，J-Flash Program internal & external microcontroller flash tool*
- *Emn178.Github.io，CRC32 and SHA256 online tool*

重要通知和免责声明

TI“按原样”提供技术和可靠性数据（包括数据表）、设计资源（包括参考设计）、应用或其他设计建议、网络工具、安全信息和其他资源，不保证没有瑕疵且不做出任何明示或暗示的担保，包括但不限于对适销性、与某特定用途的适用性或不侵犯任何第三方知识产权的暗示担保。

这些资源可供使用 TI 产品进行设计的熟练开发人员使用。您将自行承担以下全部责任：(1) 针对您的应用选择合适的 TI 产品，(2) 设计、验证并测试您的应用，(3) 确保您的应用满足相应标准以及任何其他安全、安保法规或其他要求。

这些资源如有变更，恕不另行通知。TI 授权您仅可将这些资源用于研发本资源所述的 TI 产品的相关应用。严禁以其他方式对这些资源进行复制或展示。您无权使用任何其他 TI 知识产权或任何第三方知识产权。对于因您对这些资源的使用而对 TI 及其代表造成的任何索赔、损害、成本、损失和债务，您将全额赔偿，TI 对此概不负责。

TI 提供的产品受 TI 销售条款)、TI 通用质量指南 或 ti.com 上其他适用条款或 TI 产品随附的其他适用条款的约束。TI 提供这些资源并不会扩展或以其他方式更改 TI 针对 TI 产品发布的适用的担保或担保免责声明。 除非德州仪器 (TI) 明确将某产品指定为定制产品或客户特定产品，否则其产品均为按确定价格收入目录的标准通用器件。

TI 反对并拒绝您可能提出的任何其他或不同的条款。

最后更新日期：2025 年 10 月