

User's Guide

# MSPM0 引导加载程序



摘要

MSPM0 引导加载程序 ( 也称为 BSL ) 提供了一种通过标准串行接口对器件存储器 ( 包括应用程序存储器 ( 闪存 ) 和数据存储器 (RAM) ) 进行修改的方法。

若要调用引导加载程序，需要遵循特定的进入序列，它不会在每次上电时执行。此外，可通过复位或特定命令来启动应用程序，从而退出引导加载会话。

内容

1 简介.....	2
1.1 BSL 特性概览.....	2
1.2 术语.....	2
1.3 其他资源.....	2
2 BSL 选项.....	3
3 BSL 架构.....	5
3.1 设计.....	5
3.2 调用 BSL.....	6
3.3 存储器.....	7
3.4 BSL 配置.....	8
3.5 BSL 状态.....	8
4 引导加载程序协议.....	10
4.1 数据包格式.....	10
4.2 UART 和 I2C BSL 协议.....	10
4.3 引导加载程序内核命令.....	11
4.4 引导加载程序内核响应.....	19
4.5 引导加载程序安全性.....	21
5 使用引导加载程序的示例程序流程.....	23
6 基于闪存的 BSL.....	25
6.1 次级 BSL/ROM 调用的闪存 BSL.....	25
6.2 独立闪存 BSL.....	26
7 接口插件.....	29
7.1 实现.....	29
7.2 闪存插件类型.....	31
7.3 覆盖现有接口.....	32
8 修订历史记录.....	33

商标

所有商标均为其各自所有者的财产。

# 1 简介

## 1.1 BSL 特性概览

引导加载程序 (BSL) 提供了一种通过 UART 或 I2C 等标准串行接口对器件存储器进行编程或验证的方法。

可通过串行接口访问的 ROM BSL 主要特性包括：

- 闪存的编程和擦除
- 能够返回代码或数据区域的 32 位 CRC (最小区域大小为 1KB) 以验证编程
- 能够启用代码或数据读取 (默认禁用)
- 能够通过指向主闪存的指针返回固件版本号
- 指定硬件调用 GPIO 的能力
- 访问始终受到 256 位密码的保护
- 可配置的安全警报处理，用于抵抗蛮力攻击
- 能够添加新接口作为闪存插件
- 能够使用自定义引导加载程序

## 1.2 术语

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

引导代码 (BCR) - 在启动复位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，位于非主闪存存储器中

BSL 配置 - 包含用于引导加载程序的所有用户可配置参数的配置结构，位于非主闪存存储器中

## 1.3 其他资源

### 1. 技术参考手册

- a. [MSPM0 G 系列微控制器](#)
- b. [MSPM0 L 系列微控制器](#)
- c. [MSPM0 H 系列微控制器](#)
- d. [MSPM0 C 系列微控制器](#)

### 2. 工具和示例

- a. [MSPM0 软件开发套件](#)

### 3. 应用手册

- a. [MSPM0 引导加载程序实现](#)

## 2 BSL 选项

根据默认情况下支持的 BSL 特性，MSPM0 器件可分为三类。

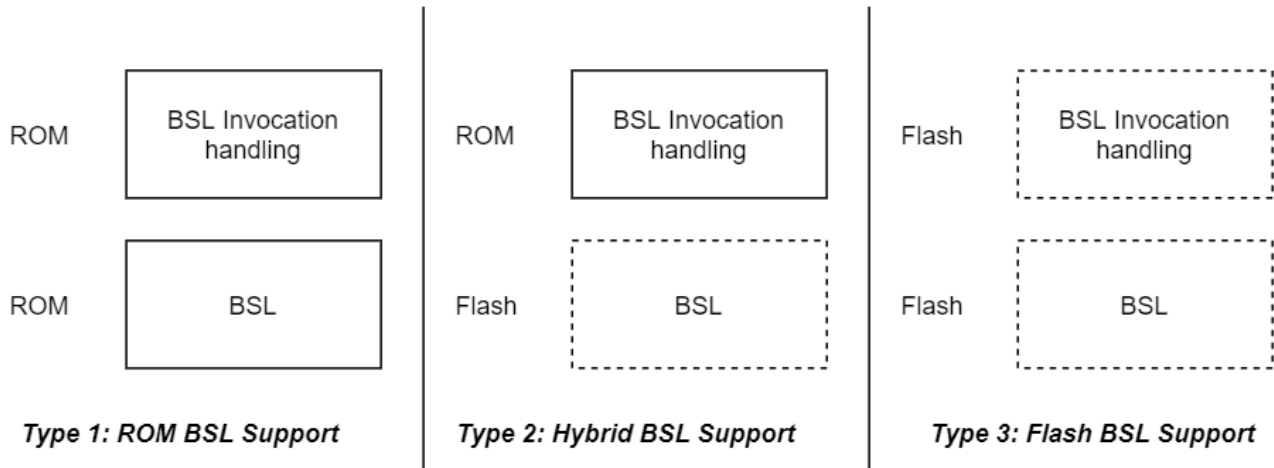


图 2-1. MSPM0 器件中的 BSL 选项

### 第 1 类：ROM BSL 支持

具备 ROM BSL 实现的器件将引导加载程序嵌入在 ROM 中。它将负责处理更新存储器时的调用和内核操作。本用户指南的后续几节将详细介绍 ROM BSL 的[特性](#)、[架构](#)和[协议](#)。

- 支持 UART 和 I2C 接口
- 出厂新器件可立即投入使用
- 这些器件还支持闪存存储器中的自定义实现，例如次级 BSL 或接口插件形式。

有关[次级 BSL](#)或[接口插件](#)的更多详细信息，请参阅后续部分。

### 第 2 类：混合 BSL 支持

在这些器件中，ROM 和闪存共同承担功能职责。在 ROM 引导代码 (BCR) 配置中启用了引导加载程序时，BCR 会在每次启动时检查[调用条件](#)。这部分的处理方式与 ROM BSL 类似。满足条件时，控制权会转移到位于闪存存储器中的 BSL。内核 BSL 操作在该存储器中实现。

- 可以是支持器件中任何可用接口的自定义 BSL 实现
- 在新器件中，首次需要通过调试器接口加载闪存 BSL (以及可选的应用程序映像)。

此类别中闪存 BSL 的实现和配置过程遵循与次级 BSL 相同的程序。有关更多详细信息，请参阅[节 6](#)。

### 第 3 类：闪存 BSL 支持

这些器件不具备针对 BSL 活动的 ROM 支持。闪存实现应完全处理 BSL 调用标准和功能。

- 可以是支持器件中任何可用接口的自定义 BSL 实现
- 在新器件中，首次需要通过调试器接口加载闪存 BSL (以及可选的应用程序映像)。

表 2-1. 器件中的 BSL 支持

器件系列	器件	ROM BSL 支持	混合 BSL 支持	闪存 BSL 支持
MSPM0Gx	MSPM0Gx	✓		
MSPM0Lx	MSPM0Lx	✓		
MSPM0Cx	MSPM0C1105/6		✓	
	MSPM0C1104			✓

表 2-1. 器件中的 BSL 支持 (续)

器件系列	器件	ROM BSL 支持	混合 BSL 支持	闪存 BSL 支持
MSPM0Hx	MSPM0H321x		✓	

## 3 BSL 架构

### 3.1 设计

当检测到有效的引导加载程序调用条件时，引导代码将调用引导加载程序。仅当在 BCR 配置的 BSL 模式字段中启用了引导加载程序时，才会调用它。

引导加载程序启动后，它首先执行“Init”阶段，在该阶段完成 BSL 配置的初始检查，并将器件配置为运行引导加载程序。

接下来，引导加载程序进入“接口自动检测”阶段。在此阶段，BSL 会配置所有可用的 BSL ROM 接口和闪存插件接口（如果已注册）。然后，BSL 逐个轮询所有接口的数据。当在其中一个接口中接收到有效的[连接数据包](#)时，该接口将被视为用于进一步通信的有源接口，所有其他接口都将被禁用。接口搜索将持续 10 秒，如果未检测到接口，则器件将进入待机模式。

接下来，引导加载程序进入“命令接收”阶段。在此阶段，BSL 将无限循环等待来自主机的命令。接收到有效命令后，将处理该命令，并将来自 BSL 内核的响应发送回主机。然后，它返回到循环并等待下一条命令，依此类推。如果接收到“Start Application”命令，引导加载程序将触发系统复位，然后执行引导代码并调用应用程序。该阶段的超时也为 10 秒。如果未接收到有效命令，引导加载程序将被锁定并进入睡眠模式。

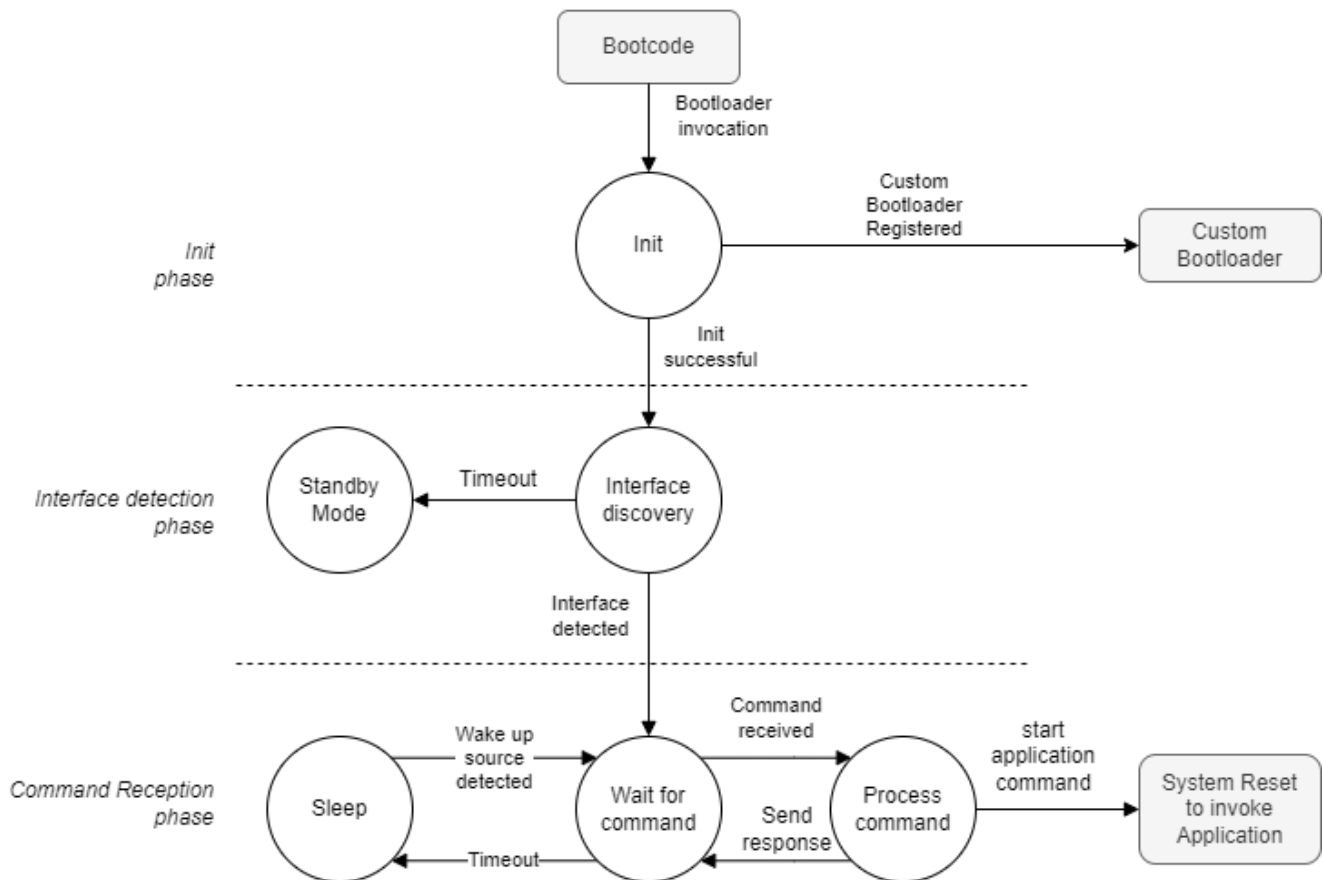


图 3-1. BSL 架构

#### 3.1.1 超时特性

引导加载程序将在未检测到任何活动时超时，并进入低功耗模式以节省功耗。

这已在以下两个阶段实施。

##### 1. 接口自动检测

## 2. 命令接收

### 3.1.1.1 接口自动检测

在接口检测阶段，如果在任何接口上持续 10 秒未收到有效的连接命令，引导加载程序将进入待机模式。

需要 POR 才能退出此状态，并通过再次创建 BSL 调用条件来使用引导加载程序。

### 3.1.1.2 命令接收

在命令接收阶段，如果 10 秒内未收到有效命令，引导加载程序将进入睡眠模式。要将器件从睡眠模式唤醒，应在有源接口上进行数据传输。

引导加载程序将在进入睡眠模式之前锁定，以减少攻击面。因此，从低功耗模式唤醒后，需要通过发送 256 位 BSL 密码来再次解锁引导加载程序（请参阅[解锁引导加载程序命令](#)）。

---

#### 备注

引导加载程序使用 LFCLK 进行超时检查。如果应用程序将外部时钟配置为 LFCLK 的时钟源，BSL 将在通过应用程序 BSL 请求调用时使用相同的时钟

---

## 3.2 调用 BSL

只有在满足任何 BSL 调用条件并且在 BCR 配置中启用了引导加载程序时，引导加载程序才应由引导代码调用。

在 BCR 配置中启用快速引导模式后，引导加载程序只能由调试邮箱命令和应用程序请求调用。跳过其他检查以节省执行时间。

### 3.2.1 空白器件

引导代码通过检查栈指针 (0x00000000) 和复位矢量 (0x00000004) 地址的擦除状态来检测空白器件。当两个闪存存储器地址都为空时，系统将调用引导加载程序。

### 3.2.2 应用程序请求

要从应用程序调用引导加载程序，请将 RESETLEVEL 设置为 BOOTLOADERENTRY 并通过 RESETCMD 寄存器触发复位。该序列会导致系统复位，并执行引导代码和调用引导加载程序。

由于系统复位已发出，因此在退出应用程序时所有外设配置均会复位。

### 3.2.3 基于 GPIO 的调用

用于 BSL 调用的 GPIO 可在非主存储器的 BSL 配置中进行配置。

全新器件将在 BSL 配置中具有 TI 编程的默认引脚详细信息。

可以在 BCR 配置中禁用基于 GPIO 引脚的调用。默认处于启用状态。

GPIO 应在 POR 前置为有效，并且此状态在 POR 后应至少保持 T\_start ms。然后，可以取消置位 GPIO 引脚状态。

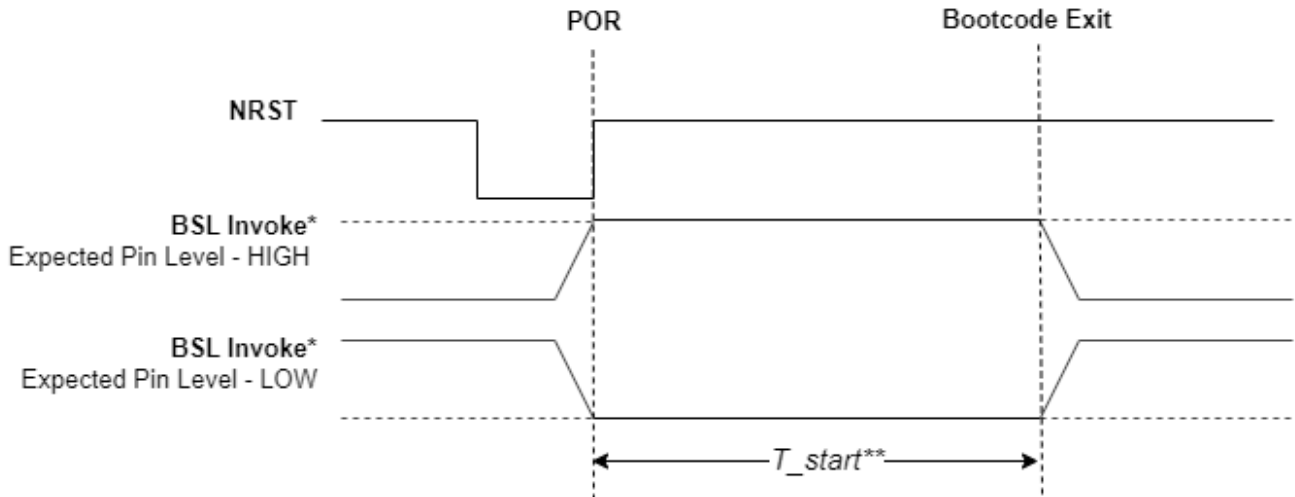


图 3-2. 从 GPIO 调用

\* - 可在 BSL 配置中配置用作 “BSL Invoke” 和 “Expected Pin Level” 的 GPIO 引脚

\*\* -  $T_{start}$  是指器件特定数据表中指定的冷启动时间

#### 备注

如果启用了基于引脚的 BSL 调用，则应将所配置的 GPIO 引脚拉高或拉低。它不应保持悬空，否则可能会导致意外的 BSL 执行。

### 3.2.4 调试邮箱命令

当调试接口可用时，引导加载程序调用命令可通过调试子系统邮箱 (DSSM) 发送。

有关 DSSM 命令用法的更多详细信息，请参阅 MSPM0x\_Technical\_Reference\_Manual DSSM 命令部分。

### 3.2.5 启动前应用程序验证

在 BCR 配置中配置应用程序 CRC/哈希值校验时，会在每次启动时验证应用程序存储器完整性。如果验证失败且启用了引导加载程序，则会调用引导加载程序。如果引导加载程序禁用，则会导致启动失败。有关更多详细信息，请参阅器件技术参考手册中的“启动配置”部分。

## 3.3 存储器

### 3.3.1 SRAM 存储器使用情况

SRAM 存储器布局说明了用于引导加载程序运行的存储器。

- 数据段和栈段 - 供 BSL 运行使用。在退出引导加载程序时，SRAM 的这些段会被清除。
- 可变缓冲区空间 - 用于存储 BSL 通信期间所接收/发送数据包的缓冲区空间

主机允许进行读写访问的 SRAM 存储器是 BSL 缓冲区起始地址到 [SRAM 结束地址 - 0x120]，其中 SRAM 结束地址由每个器件中可用的 SRAM 存储器决定。由于与可变缓冲空间共享同一 SRAM 空间，因此其内容在 SRAM 写入/读取操作期间有可能被覆盖。

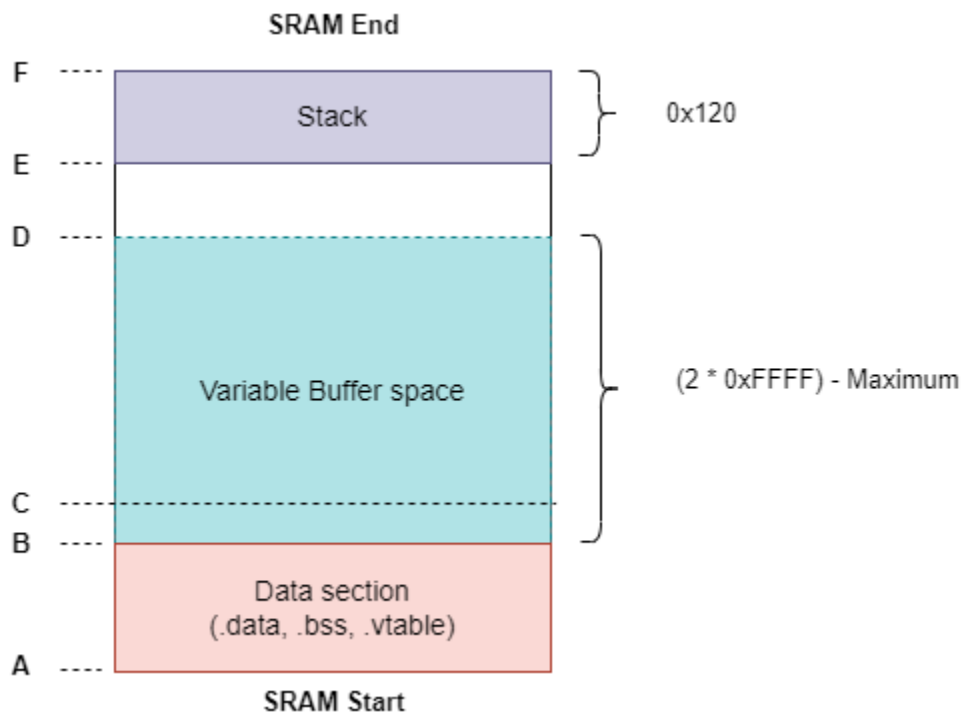


图 3-3. SRAM 使用情况

A - SRAM 起始地址 (0x20000000)

B - 未注册任何闪存插件接口时，从“Get Device Info”命令响应中获知的“BSL Buffer Start Address”

C - 从“Get Device Info”命令响应中获知的“BSL Buffer Start Address”。当没有注册闪存插件接口时，它将与“B”相同

D - BSL 缓冲区结束地址 = “BSL 缓冲区起始地址” + ( 2 \* “BSL 最大缓冲区大小” )，其中 BSL 缓冲区起始地址和 BSL 最大缓冲区大小可从“Get Device Info”命令响应中得知

E - 栈分配的起始地址 (E - 0x120)。当“BSL 最大缓冲区大小”小于 0xFFFF 时，它将与“D”相同

F - 器件中可用 SRAM 存储器的结束地址。要获悉此地址，请参阅器件特定数据表。

B-C 段：

- 在 BSL 配置中注册时，将为闪存插件操作分配的数据段

C-D 段：

- 用于存储数据包的缓冲区空间
- 最大大小为 (2 \* 0xFFFF)

C-E 段：

- 可通过 BSL 命令进行 SRAM 读取和写入操作的存储器

### 3.4 BSL 配置

非主闪存存储器中的 BSL 配置允许用户自定义 BSL 使用的某些参数。

要了解有关可用配置的更多信息，请参阅 MSPM0 技术参考手册的“配置存储器”部分。

### 3.5 BSL 状态

BSL 状态提供了有关执行期间发生的任何错误的详细信息，这些错误导致了 BSL 无响应。此 32 位状态信息存储在特定的 SRAM 地址“0x200000C0”中，如果在 BCR 配置中启用了调试访问，则可以通过调试器读取此信息。



表 3-1. BSL 状态的解释

字节 4	字节 3	字节 2	字节 1
保留	HW 误差	硬件错误详细信息	SW 误差

错误	说明
0x01	BSL 配置 CRC 错误

#### HW 误差

错误	说明	硬件错误详细信息
0x07	NMI 异常	NMIIDX - NMI 中断索引寄存器数据

## 4 引导加载程序协议

### 4.1 数据包格式

BSL 数据包具有分层结构。BSL 内核命令包含由 BSL 待处理的实际命令数据。除了标准 BSL 命令，每个内核命令的前后有被称为外设接口代码（PI 代码）的包装器数据。该包装数据是所使用的外设和协议的特定信息，并且它包含有允许 BSL 内核命令正确传输的信息。包装器和内核命令将构成一个 BSL 数据包

PI 代码	BSL 内核数据	PI 代码
-------	----------	-------

### 4.2 UART 和 I2C BSL 协议

UART 和 I2C BSL 协议的数据包具有以下结构。

- **标头**字节表示使用的协议和数据包类型（命令或响应数据包）。
- **长度**字段包含以字节为单位的 BSL 内核数据的大小。
- **BSL 内核数据**包含命令/响应 ID 和地址，即命令所需的数据
- **CRC** 字段包含针对“BSL 内核数据”中数据计算得出的 CRC，用于检查完整性。CRC 可以是 32 位 CRC，也可以是 16 位 CRC，具体取决于器件。在任一情况下，字段长度都是恒定的。如果是 16 位 CRC，则剩余字节将补零。

PI 代码		BSL 内核数据	PI 代码
标头（1 字节）	长度（2 字节）	BSL 内核命令/响应	CRC（4 字节）

根据内核数据字段，数据包被归类为命令数据包或响应数据包。

命令数据包是发送到 BSL 的第一个数据包。第二个数据包是从 BSL 接收的响应数据包。响应数据包包含两个组件 BSL 确认和 BSL 内核响应。在这两个命令中，每个发送的命令数据包都会从 BSL 接收确认。但并非每个命令都会收到 BSL 内核响应。

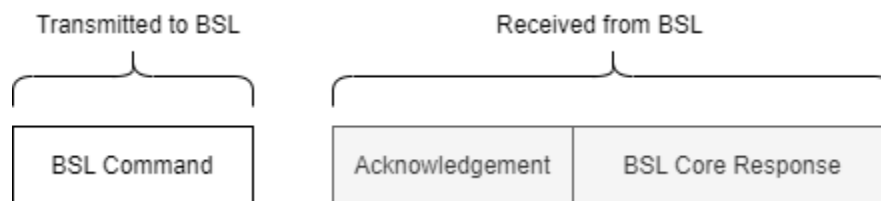


图 4-1. BSL 协议

#### 4.2.1 BSL 确认

BSL 软件的外设接口部分对 BSL 数据包的包装器部分进行解析。如果数据传输中有错误，一个错误消息被立即发出。在成功接收到所有数据后一个 ACK 被发出，这并不意味着命令已经成功执行（或者甚至命令是有效的）而是意味着数据包被正确格式化并被传递到 BSL 内核软件进行分析。

BSL 协议规定，每个发送的 BSL 数据包除了已发送的 BSL 数据包外，以单字节确认进行响应。表列出了来自 BSL 的确认响应。如果发送了除 ACK 以外的确认字节，则 BSL 不发送任何 BSL 数据包。主机编程器必须检查确认错误并重新尝试发送。

数据	含义
0x00	BSL_ACK ( 成功接收到数据包 )
0x51	BSL_ERROR_HEADER_INCORRECT
0x52	BSL_ERROR_CHECKSUM_INCORRECT
0x53	BSL_ERROR_PACKET_SIZE_ZERO
0x54	BSL_ERROR_PACKET_SIZE_TOO_BIG
0x55	BSL_ERROR_UNKNOWN_ERROR

数据	含义
0x56	BSL_ERROR_UNKNOWN_BAUD_RATE

## 4.2.2 外设配置

### 4.2.2.1 UART

通过以下配置启用 UART：

- 使用 UART0
- 波特率默认为 9600bps。它可以通过 Change Baud Rate 命令进行更新
- 数据宽度：8 位
- 停止位数：1
- 无奇偶校验位
- 用于 RXD 和 TXD 的引脚取自 BSL 配置

### 4.2.2.2 I2C

BSL 中的 I2C 接口可用作 I2C 目标。主机充当控制器并驱动通信。

- 使用 I2C0
- 默认情况下，I2C 目标地址为 0x48。它可以在 BSL 配置中进行配置
- SCL 和 SDA 线路需要外部上拉
- 用于 SDA 和 SCL 的引脚取自 BSL 配置

#### 备注

- BSL 不会验证在 UART 和 I2C 接口的非主中配置的引脚的详细信息。BSL 期望引脚配置正确。
- 不要对 UART 和 I2C 使用相同的引脚。

### 4.2.2.3 CRC

使用硬件 CRC 模块计算 CRC。在支持 CRC32 的器件中，使用 32 位 CRC 值。在仅支持 CRC16 的器件中，使用 16 位 CRC，剩余字节则补零，以构成 4 个字节。请参阅特定于器件的数据表，以确定受支持的 CRC 多项式。

数据的 CRC 根据以下各项计算：

- CRC32-ISO3309 ( 或 ) CRC16-CCITT 多项式
- 位反转配置
- 初始种子 - 0xFFFFFFFF

## 4.3 引导加载程序内核命令

BSL 命令	受保护	CMD	起始地址	数据 ( 字节 )	BSL 内核响应
CMD 连接	否	0x12	-	-	否
CMD Unlock Bootloader	否	0x21	-	D1...D32 ( 密码 )	是
CMD Flash Range Erase	是	0x23	A1...A4	A1...A4 ( 结束地址 )	是
CMD 批量擦除	是	0x15	-	-	是
CMD Program Data	是	0x20	A1...A4	D1...Dn ,	是
CMD Program Data Fast	是	0x24	A1...A4	D1...Dn ,	否
CMD Memory Read back	是	0x29	A1...A4	L1...L4	是
CMD 恢复出厂设置	是	0x30	-	D1...D16 ( 密码 )	是
CMD Get Device 信息	否	0x19	-	-	是

BSL 命令	受保护	CMD	起始地址	数据 ( 字节 )	BSL 内核响应
CMD Standalone Verification	是	0x26	A1...A4	L1...L4	是
CMD Start application	否	0x40	-	-	否
CMD Change Baud rate	否	0x52	-	D1 ( 波特率 ID )	否

#### 缩写:

A1...A4

地址字节，其中 A1 是最低有效字节

D1...Dn

数据字节，其中 “n” 受 BSL 最大缓冲区大小限制

L1...L4

数据长度字节，其中 C1 是最低有效字节

#### 4.3.1 连接

##### 结构

标头	长度		CMD	CRC32			
0x80	0x01	0x00	0x12	C1	C2	C3	C4

##### 描述

连接命令是第一个用于通过特定接口 ( UART 或 I2C ) 在主机和目标之间建立连接的命令。

##### 保护

否

##### 命令返回

仅 [BSL 确认](#)

##### 示例

主机 : 80 01 00 12 3A 61 44 DE

BSL : 00

#### 4.3.2 获取器件信息

##### 结构

标头	长度		CMD	CRC32			
0x80	0x01	0x00	0x19	C1	C2	C3	C4

##### 描述

该命令用于获取可用于数据事务的版本信息和缓冲区大小。

##### 保护

否

##### 命令返回

BSL 确认和带有器件信息的 BSL 内核响应。请参阅[器件信息](#)了解更多详细信息

##### 示例

BSL:00 08 19 00 31 00 01 00 01 00 00 00 00 01 00 C0 06 60 01 00 20 01 00 00 00 01 00 00 00 49 61 57 8C

是

## 地址

要编程的存储器区域的起始地址。A1...A4，其中 A1 是 32 位地址的最低有效字节。

## 数据

要写入指定地址的数据字节。可发送的数据的最大大小受器件缓冲区大小的限制。通过 **Get Device Info 命令** 可以知道缓冲区大小。

## 命令返回

BSL 确认和带有有关操作状态的消息的 BSL 内核响应。有关更多详细信息，请参阅节 4.4.1 部分。

## 示例

主机：80 0D 00 20 00 00 00 00 00 00 00 04 00 00 00 08 7A DC AE B8

BSL：00 08 02 00 3B 00 38 02 94 82

### 4.3.5 快速编程数据

## 结构

标头	长度		CMD	地址	数据	CRC32			
0x80	L1	L2	0x24	A1...A4	D1...Dn	C1	C2	C3	C4

## 描述

Program Data Fast 命令与 Program Data 命令相同，只是该命令执行非阻塞写入，以加快编程过程。对于此命令，BSL 不会发送 BSL 内核消息响应来指示编程是否成功。

## 保护

是

## 地址

要编程的存储器区域的起始地址。A1...A4，其中 A1 是 32 位地址的最低有效字节。

## 数据

要写入指定地址的数据字节。可发送的数据的最大大小受器件缓冲区大小的限制。通过 **Get Device Info 命令** 可以知道缓冲区大小。

## 命令返回

BSL 确认。

## 示例

主机：80 0D 00 24 00 01 00 00 01 02 03 04 05 06 07 08 72 10 2A 18

BSL：00

### 4.3.6 回读数据

## 结构

标头	长度		CMD	地址	数据	CRC32			
0x80	0x09	0x00	0x29	A1...A4	L1...L4	C1	C2	C3	C4

## 描述

此命令用于从地址 A1...A4 开始读取数据。

应在 BSL 配置中启用读取，从而使用此命令读取数据。在 BSL 配置中，默认情况下会禁用它。

允许主闪存 ( 应用程序存储器 ) 、非主闪存 ( 配置存储器 ) 和 SRAM 存储器读取数据。

#### 备注

主机无法完全访问 SRAM 存储器。有关更多详细信息，请参阅 [SRAM 存储器使用情况](#)。

#### 保护

是

#### 地址

要回读的存储器区域的起始地址。A1...A4，其中 A1 是 32 位地址的最低有效字节。

#### 数据

要读取的数据大小 ( 以字节为单位 )，L1...L4，其中 L1 是最低有效字节。可读取的数据的最大大小受器件缓冲区大小的限制。通过 [Get Device Info 命令](#) 可以知道缓冲区大小。

#### 命令返回

如果回读命令有效，则使用 BSL 确认和带有所请求数据的 BSL 内核响应。有关更多详细信息，请参阅 [节 4.4.3](#)。

如果回读命令具有无效的地址/长度，或者如果读数被禁用，相应的错误将在 BSL 确认后作为消息响应发送。

#### 示例

主机：80 09 00 29 00 0C 00 00 08 00 00 00 32 9D B0 35

BSL：00 08 09 00 30 FF FF FF FF FF FF FF F6 2B A1 73

#### 4.3.7 闪存范围擦除

#### 结构

标头	长度		CMD	地址	数据	CRC32			
0x80	0x09	0x00	0x23	A1...A4 ( 起始地址 )	A1...A4 ( 结束地址 )	C1	C2	C3	C4

#### 描述

Flash range erase 命令用于擦除指定的闪存存储器区域。闪存按扇区擦除 (1KB)，不能进行更小范围的擦除。

当起始地址和结束地址驻留在不同的闪存扇区中时，BSL 会擦除起始地址和结束地址之间的所有闪存扇区，包括包含这些地址的扇区。

此命令只能用于擦除主闪存存储器。不能擦除非主闪存存储器。

结束地址不应小于起始地址。

#### 保护

是

#### 地址

要擦除的存储器区域的起始地址。A1...A4，其中 A1 是 32 位地址的最低有效字节。

#### 数据

要擦除的存储器区域的结束地址。A1...A4，其中 A1 是 32 位地址的最低有效字节。

#### 命令返回

BSL 确认和带有有关操作状态的消息的 BSL 内核响应。有关更多详细信息，请参阅 [节 4.4.1](#) 部分。

#### 示例

主机 : 80 09 00 23 00 01 00 00 FF 03 00 00 2B E6 BE D8

BSL : 00 08 02 00 3B 00 38 02 94 82

#### 4.3.8 批量擦除

##### 结构

接头	长度		CMD	CRC32			
0x80	0x01	0x00	0x15	C1	C2	C3	C4

##### 说明

批量擦除命令会擦除器件中可用的完整主闪存存储器 ( 应用程序存储器 ) 。

BCR 配置存储器中的批量擦除配置不会影响此 BSL 命令。

当一个闪存区域在 BCR 配置存储器中受到静态写保护时，此区域不能被擦除。

##### 受保护

是

##### 命令返回

BSL 确认和带有有关操作状态的消息的 BSL 内核响应。更多详细信息，请参阅节 4.4.1。

##### 示例

主机 : 80 01 00 15 99 F4 20 40

BSL : 00 08 02 00 3B 00 38 02 94 82

#### 4.3.9 恢复出厂设置

##### 结构

接头	长度		CMD	数据	CRC32			
0x80	L1	L2	0x30	D1...D16	C1	C2	C3	C4

##### 说明

factory reset 命令会擦除完整的主闪存 ( 应用程序 ) 存储器和非主闪存 ( 配置 ) 存储器。

处理此命令会受到 BCR 配置存储器中的出厂复位配置的影响。

##### 恢复出厂设置

- 在无密码时允许 ( 如果 “Enabled” )
- 在有密码时允许 ( 如果 “Enabled with Password” )
- 不允许 ( 如果 “Disabled” )

当一个闪存区域在 BCR 配置存储器中受到静态写保护时，此区域不能被擦除。

##### 受保护

是

##### 数据

存储在 BCR 配置存储器中的 16 字节出厂复位密码。默认密码全为 0xFF。仅当 BCR 配置中的恢复出厂设置为 “Enabled with Password” 时，才需要密码。

##### 命令返回

BSL 确认和带有有关操作状态的消息的 BSL 内核响应。更多详细信息，请参阅节 4.4.1。



### 小心

恢复出厂设置后，除非已恢复非主配置，否则系统极易受到潜在锁定情况的影响，在这种情况下，无法再次访问器件。

### 示例

场景 1：无密码即可恢复出厂设置

主机：80 01 00 30 DE 20 24 0B

BSL：00 08 02 00 3B 00 38 02 94 82

场景 2：通过密码恢复出厂设置

主机：80 11 00 30 FF FF FF FF FF FF FF FF FF FF FF FF 8A 28 EA DC

BSL：00 08 02 00 3b 00 38 02 94 82

### 4.3.10 独立验证

#### 结构

接头	长度		CMD	地址	数据	CRC32			
0x80	0x09	0x00	0x26	A1...A4	D1...D4	C1	C2	C3	C4

#### 说明

此命令用于验证存储在给定存储器范围内的数据的 CRC。这样可以更快地验证编程的数据。它要求数据大小至少为 1KB。

允许主闪存（应用程序存储器）、非主闪存（配置存储器）和 SRAM 存储器进行 CRC 验证。

#### 备注

主机无法完全访问 SRAM 存储器。有关更多详细信息，请参阅节 3.3.1。

### 受保护

是

#### 地址

要验证的存储器区域的起始地址。A1...A4，其中 A1 是 32 位地址的最低有效字节。

#### 数据

要验证的数据大小（以字节为单位），L1...L4，其中 L1 是最低有效字节。1kB ≤ 大小 ≤ 64KB。

#### 命令返回

BSL 确认和带有所请求的存储器区域计算出的 CRC 值的 BSL 内核响应。有关更多详细信息，请参阅节 4.4.5。

如果验证命令的地址/长度无效，则相应的错误将作为 BSL 确认后的消息响应发送。请参阅节 4.4.1。

### 示例

主机：80 09 00 26 00 00 00 00 0C 00 00 1C E9 07 E1

BSL：00 08 05 00 32 A0 45 71 82 91 1F 94 EC

### 4.3.11 启动应用程序

#### 结构

标头	长度		CMD	CRC32			
0x80	0x01	0x00	0x40	C1	C2	C3	C4

## 描述

Start application 命令会发出系统复位，这会导致引导加载程序退出、重新运行引导代码，进而启动应用程序。

## 保护

否

## 命令返回

BSL 确认

## 示例

Host:80 01 00 40 E2 51 21 5B

BSL:00

### 4.3.12 更改波特率

## 结构

接头	长度		CMD	数据	CRC32			
0x80	0x02	0x00	0x52	D1	C1	C2	C3	C4

## 说明

该命令可用于更改 UART 接口的波特率。新波特率将在为此数据包发送 BSL 确认后生效。

BSL UART 的默认波特率为 9600bps。

## 备注

更新波特率后，如果使用解锁引导加载程序命令发送了错误的 BSL 密码，则波特率设置将恢复为默认值。应以默认波特率进行进一步通信。

## 受保护

否

## 数据

表中指定的 D1 波特率。

ID	波特率 (bps)
1	4800
2	9600
3	19200
4	38400
5	57600
6	115200
7	1000000
8	2000000
9	3000000

## 命令返回

BSL 确认

## 示例

主机：80 02 00 52 03 6C 83 A2 AF

BSL：00

### 4.4 引导加载程序内核响应

BSL 响应	RSP	数据
Memory read back	0x30	D1...Dn
Get Device Info	0x31	D1...D24
Standalone verification	0x32	D1...D4
消息	0x3B	MSG
详细错误	0x3A	D1..D3

## 缩写:

### MSG

字节，包含 BSL 内核响应，并描述请求操作结果。其可以是一个错误代码，也可以是对成功操作的确认。当要求 BSL 响应数据时（例如，存储器、版本、CRC 或缓冲器大小），操作应答未成功发生，BSL 内核立即发送数据。

### D1..Dn

数据字节，其中“n”受 BSL 最大缓冲区大小限制

#### 4.4.1 引导加载程序内核消息

## 结构

接头	长度		RSP	数据	CRC32			
0x08	0x02	0x00	0x3B	MSG	C1	C2	C3	C4

## 说明

对于某些命令，BSL 会将消息响应发送到主机，以指示已处理命令的状态。该表列出了来自 BSL 的所有可能消息。

MSG	含义	可能原因 <sup>(1)</sup>
0x00	操作成功	
0x01	BSL 锁定错误	尚未使用引导加载程序解锁密码命令解锁 BSL，或者在 BSL 解锁之后，命令接收阶段会发生超时
0x02	BSL 密码错误	发送了错误的密码来解锁引导加载程序。
0x03	多个 BSL 密码错误。已采取安全警报措施。	为解锁引导加载程序已发送错误的密码 3 次。
0x04	未知命令	提供给 BSL 的命令未被识别为有效命令
0x05	存储器范围无效	给定的存储器范围无效。
0x06	命令无效	指定给 BSL 的命令是已知命令，但在该时刻无效，无法处理。
0x07	已禁用恢复出厂设置	BCR 配置中禁用了恢复出厂设置
0x08	恢复出厂设置密码错误	当 BCR 配置的 factory reset 为“Enabled with password”时，使用 factory reset 命令发送的密码不正确或没有密码
0x09	读出错误	在 BCR 配置中禁用存储器读出
0x0A	地址或长度对齐无效	闪存编程的起始地址或数据长度不是 8 字节对齐的
0x0B	独立验证的长度无效	发送用于独立验证的数据大小小于 1KB

(1) 此处列出的可能原因不是出现状态或错误的唯一原因。它仅列出了导致出现错误（可由主机纠正）的可能软件原因。

## 4.4.2 详细错误

结构

接头	长度		RSP	数据	CRC32			
0x08	0x02	0x00	0x3A	D1..D3	C1	C2	C3	C4

说明

D1 - 错误类型

D3、D2 - 错误详细信息

可能的值

错误类型		错误详细信息	
值	说明	值	说明
0xF0	闪存错误	0xXX	包含 FLASHCTL.STATCMD 寄存器值

## 4.4.3 存储器回读

结构

标头	长度		RSP	数据	CRC32			
0x08	L1	L2	0x30	D1...Dn	C1	C2	C3	C4

描述

该命令会返回所请求的数据以响应回读命令

数据

数据 D1..Dn，其中“n”受 BSL 最大缓冲区大小限制。

## 4.4.4 器件信息

结构

标头	长度		RSP	数据	CRC32			
0x08	0x19	0x00	0x31	D1...D24	C1	C2	C3	C4

描述

该命令会返回版本信息和 BSL 缓冲区大小，以响应 Get Identity 命令

数据

识别字节	数据字节
命令解释器版本	[D02-D01]
构建 ID	[D04- D03]
应用程序版本	[D08-D05]
有源插件接口版本	[D10-D09]
BSL 最大缓冲区大小	[D12-D11]
BSL 缓冲区起始地址	[D16-D13]
BCR 配置 ID	[D20-D17]
BSL 配置 ID	[D24- D21]

## 应用程序版本：

32 位应用程序版本取自 BSL 配置中指定的地址

## BSL 存储器大小：

可用于存储发送/接收到的 BSL 数据包的 RAM 数据缓冲区大小。

## 示例

主机：80 01 00 19 B2 B8 96 49

BSL：00 08 19 00 31 00 01 00 01 00 00 00 00 01 00 C0 06 60 01 00 20 01 00 00 00 01 00 00 00 49 61 57 8C

在上述给定响应中，

命令解释器版本 - 0x0100

构建 ID - 0x0100

应用程序版本 - 0x00000000

有源插件接口版本 - 0x0001

BSL 最大缓冲区大小 - 0x06C0

BSL 缓冲区起始地址 - 0x20000160

BCR 配置 ID - 0x00000001

BSL 配置 ID - 0x00000001

## 4.4.5 独立验证

### 结构

标头	长度		RSP	数据	CRC32			
0x08	0x05	0x00	0x32	D1...D4	C1	C2	C3	C4

### 描述

该命令会返回 CRC 值以响应 standalone verification 命令

### 数据

为所请求的存储器区域计算的 32 位 CRC 值。D1...D4，其中 D1 是 CRC32 的最低有效字节。

## 4.5 引导加载程序安全性

### 4.5.1 受密码保护的命令

可以直接或间接访问存储器中数据的所有命令都受密码保护。密码可在非主存储器的 BSL 配置中配置。

发送错误的密码后，器件将在接下来的 2 秒内睡眠，并且在此期间不接受任何命令，以使蛮力破解攻击更加困难。当错误密码发送 3 次时，BSL 会执行安全警报操作。

#### 4.5.1.1 安全警报

在 BSL 配置中，安全警报可配置为以下三种模式中的任何一种。

- 恢复出厂设置** - 恢复出厂设置将擦除整个主闪存和非主闪存存储器。无论 BCR 配置中的出厂复位模式如何，都将执行擦除。如果闪存的某些部分受到静态写保护，BSL 不能擦除整个闪存存储器。在这种情况下，擦除非主闪存存储器后，引导加载程序密码将恢复为默认值。然后可以使用新密码解锁 BSL 并对其进行编程。
- 禁用引导加载程序** - 在 BCR 配置中禁用引导加载程序并退出 BSL。除非更新了 BCR 非主配置以启用引导加载程序，否则不能再进入 BSL。如果为非主器件启用了静态写保护，则不会禁用 BSL。

具有 2KB 以上 SRAM 存储器的器件也支持此特性。

### 3. 不执行任何操作 - 不采取任何行动。

---

#### 备注

在“恢复出厂设置”安全警报配置中，如果非主闪存存储器保持被擦除状态，器件会被锁定，无法再次访问器件。

---

#### 4.5.2 进入 BSL

只允许通过引导代码进入引导加载程序。

可在 BCR 配置中禁用引导加载程序。

## 5 使用引导加载程序的示例程序流程

本节介绍了 BSL 主机通过引导加载程序加载映像的典型序列。此采样序列会擦除闪存存储器并对其中的新固件进行编程。

- 应通过空白器件检测、基于引脚的调用或应用程序请求启动引导加载程序。
- 调用后，发送连接命令以通过所需的接口与 BSL 建立连接。
- 如果使用 UART 接口，则可以将波特率更改为更高的值，以加快进一步的通信，并且波特率是可选的。
- 要完全擦除闪存存储器，请使用 **Mass erase** 命令。仅当需要更新非主闪存时，才使用出厂复位命令。这是因为，如果非主闪存被擦除并保持未编程状态，器件就会被锁定。
- 对固件映像进行编程
- 对已编程的存储器区域进行 CRC 验证，以检查已编程数据的正确性。这是一个可选步骤。
- 应用程序可通过 “Start application” 命令启动。

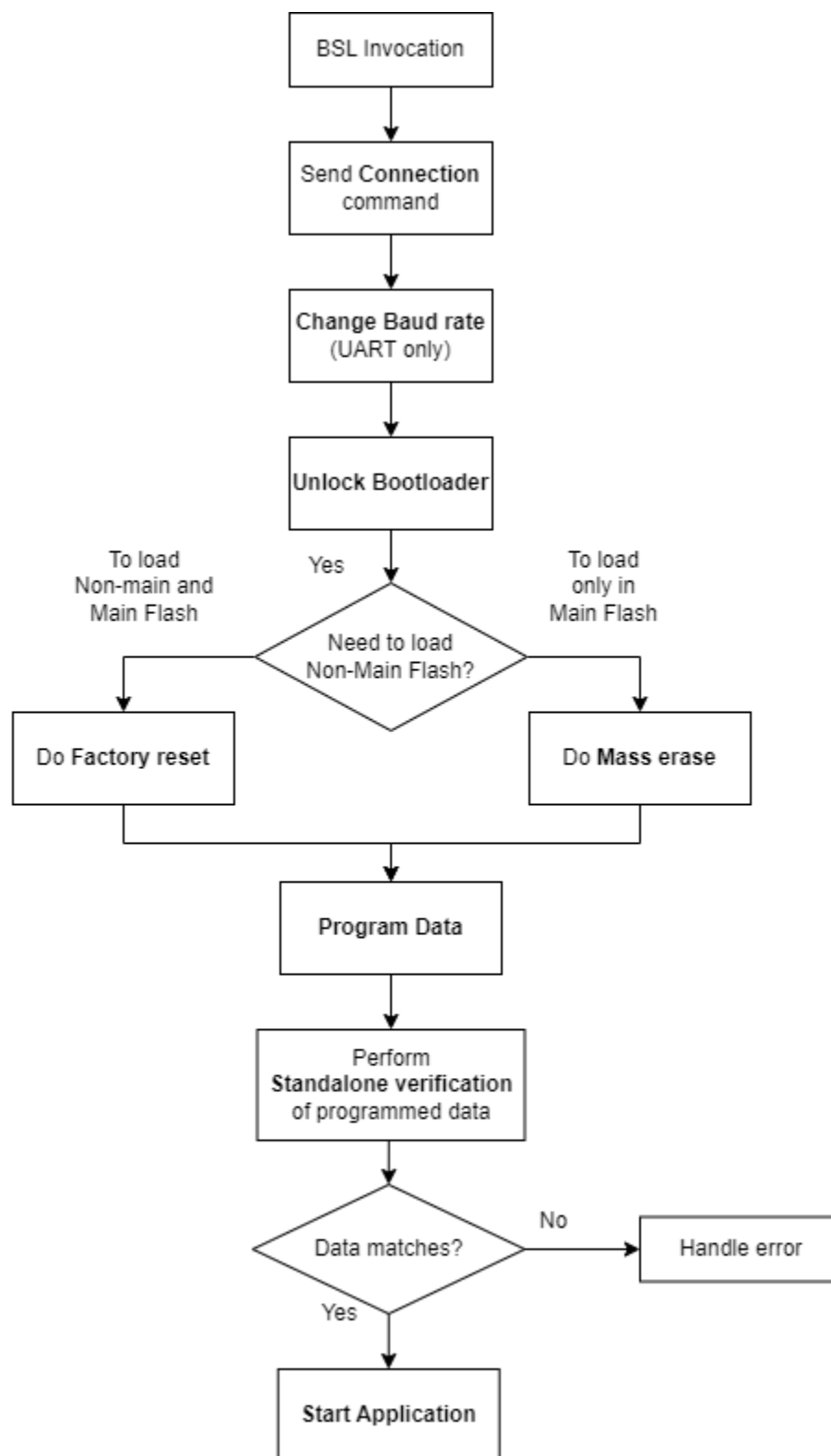


图 5-1. BSL 主机序列



## 6 基于闪存的 BSL

闪存引导加载程序是用户实现的引导加载程序例程，放置在闪存存储器中，从而允许进行自定义和更新。闪存 BSL 的实现因器件的 **BSL 类型** 而异。

- 具备 ROM BSL 支持的第 1 类器件提供了一个在闪存存储器中实现自定义引导加载程序的选项，称为次级/备用引导加载程序。在此类器件中，ROM BCR 将负责调用，而闪存实现则应通过器件中任何可用的串行接口处理 BSL 操作（例如程序和存储器回读）。
- 第 2 类器件缺少 ROM BSL 支持，采用与次级 BSL 相同的实现方式。

第 1 类和第 2 类闪存 BSL 实现都遵循相同的调用流程和配置程序。有关第 1 类和第 2 类器件的信息，请参阅 [节 6.1](#)。

- 除了 BSL 操作外，第 3 类器件还需要处理 BSL 调用。

有关第 3 类器件的信息，请参阅 [节 6.2](#)。

---

### 备注

如果闪存 BSL 存储器未受到写保护，引导加载过程可能会擦除闪存 BSL 代码，从而可能导致器件锁定。能否恢复取决于调试配置。

---

### 6.1 次级 BSL/ROM 调用的闪存 BSL

当 ROM BCR 检测到 BSL 调用条件时，它将调用这个基于闪存的 BSL，后者应负责通过串行接口处理 BSL 操作。

为此，应将主闪存存储器中的 BSL 加载到 0x0000 以外的任意地址。应用程序应放置在闪存存储器的起始位置。随后将其注册到非主闪存的 BSL 配置中。下次通过某种 [调用方法](#) 调用 BSL 时，器件将检查闪存 BSL 配置字段并分支到该字段，且控制不会返回到 ROM。

BCR 配置中的 BSL 模式配置也适用于闪存 BSL。禁用此设置后，将不会调用闪存 BSL。

在 BCR 配置中，加载 BSL 的闪存存储器区域应受到写保护，以避免在引导加载过程中发生意外擦除。非主存储器写保护为可选配置。但应注意确保闪存 BSL 指针保留在非主存储器的 BSL 配置中。

#### 6.1.1 次级 BSL 示例

第 1 类器件的次级引导加载程序示例作为 SDK 示例的一部分提供，以供参考。本节对此进行更加详细的介绍。

#### 说明

此次级引导加载程序示例支持对存储器中的数据进行编程/验证，其 BSL 协议格式与器件中的主 BSL (ROM BSL) 相同，并且可以采用与 ROM BSL 相同的方式调用。

它支持以下主要功能

- 编程数据
- 闪存存储器擦除
- 回读数据
- CRC 验证
- 启动应用程序

它使用 UART 接口与主机通信。

此示例负责次级引导加载程序的实施及注册。因此，一旦将该映像加载到器件中，就无法使用器件中的主引导加载程序。只有次级引导加载程序将处于运行状态。要将器件恢复为使用主引导加载程序，必须使用 SWD\_Factory\_Reset 命令（通过调试子系统邮箱恢复出厂设置）。

#### 示例用法

- 将 UART\_RX 和 UART\_TX 与 BSL 主机（任何带有 UART 的微控制器）连接。
- 编译并加载示例。

- 使用 BSL 调用引脚或任何其他调用方法创建 BSL 调用条件。
- 从主机发送 **GetDeviceInfo** 命令。
- 器件应使用可用的版本信息和 SRAM 缓冲区空间进行响应。
- 类似地，发送擦除、编程、验证命令来对存储器中的数据进行编程。

## 软件文件详细信息

文件名称	详细信息
secondary_bsl.c	初始化 BSL 运行所需的外设。接收来自通信接口的命令数据包并将其传递到命令处理层。还负责在 BSL 配置存储器中注册次级引导加载程序。
bsl_ci.c	解释命令数据包，处理命令并将响应发送回主机
bsl_ci.h	包含 BSL 命令和响应的定义。此外还有 bsl_ci.c 的函数声明
bsl_uart.c	处理主机和 BSL 内核之间的通信
bsl_uart.h	包含 BSL 确认的定义和 bsl_uart.c 的函数声明
ti_msp_dl_config.h	包含特定于器件的配置，例如 UART 引脚、所用外设的基址等
boot_config.h	包含 BCR 和 BSL 配置结构
factory_config.c	实现了用于获取出厂配置的器件特定数据（如 SRAM 存储器大小）的函数。
factory_config.h	包含 factory_config.c 的出厂配置结构和函数声明
startup_mspm0x_ticlang	包含矢量表、复位处理程序和其他处理程序的启动文件
mspm0x.cmd	指定内存区域（次级引导加载程序映像应驻留的存储器位置）和 SRAM 区域（次级引导加载程序应运行的位置）的链接器命令文件。

## 定制

此示例提供了次级引导加载程序的参考设计。可根据需要对其自定义。BSL 内核层（secondary\_bsl.c、bsl\_ci.c）或接口层（bsl\_uart.c）是进行自定义的主要位置。

### 要执行的步骤

- 根据需要修改代码
- 更改完成后，编译代码
- 根据需要修改 BCR 配置中的闪存写入保护设置
- 计算 BCR 配置的 CRC 并存储新的 CRC 值
- 再次编译代码
- 加载自定义 BSL 映像

## 6.2 独立闪存 BSL

此闪存 BSL 实现应负责处理调用机制以及 BSL 操作。当器件结束复位时，它将始终执行放置在闪存存储器起始位置的代码。

若要添加闪存 BSL 支持，应将调用和 BSL 实现加载到闪存中。此外，在 BCR 配置中，加载 BSL 的闪存存储器区域应受到写保护，以避免在引导加载过程中发生意外擦除。

### 6.2.1 独立闪存 BSL 示例

MSPM0 SDK 包含适用于第 3 类器件的闪存 BSL 实现示例。本部分将对此进行详细说明。

#### 说明

闪存 BSL 示例支持对存储器中的数据进行编程/验证。其中一个 UART/I2C 接口可用于与主机通信。

可通过以下三种方式中的任何一种调用：

- 应用程序启动时无应用程序
- BSL 引脚调用
- 基于应用程序的调用

它支持以下功能

- 编程数据
- 获取器件信息
- 闪存存储器擦除
- 回读数据
- CRC 验证
- 启动应用程序

使用引导代码用户配置对闪存 BSL 区域实施静态写保护。将此映像加载到器件后，闪存引导加载程序将处于活动状态，并将在每次复位时查找调用条件。未找到 BSL 请求时，它将分支到应用程序。稍后，如果闪存 BSL 需要更新或需要删除，则必须使用 **SWD\_Factory\_Reset** 命令从闪存存储器中擦除闪存 BSL。

用户可以配置接口引脚和其他设置。还可以选择性地仅启用所需的功能，以优化存储器空间占用。

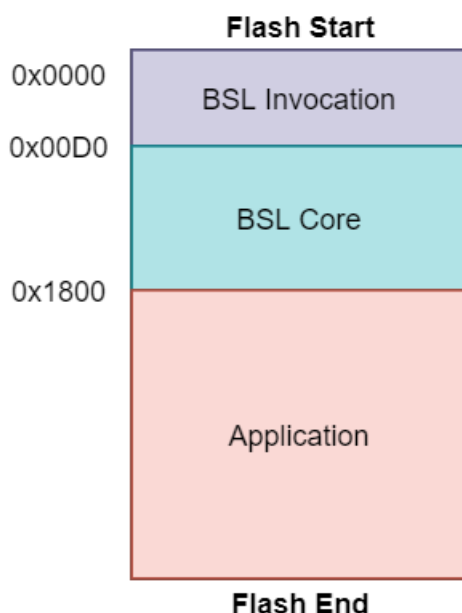


图 6-1. 独立闪存 BSL 存储器布局

在闪存存储器中，调用逻辑放置在起始地址 0x0000 处。然后放置 BSL。默认情况下，这两部分将共占用闪存的前 6KB 空间。应用程序预计从 0x1800 开始。这些地址可能因 BSL 中添加或删除的特性而异。

#### 示例用法

- 将 UART\_RX 和 UART\_TX 或 I2C\_SDA 和 I2C\_SCL 与 BSL 主机相连
- 在 **flash\_bsl\_modules.h** 中仅启用所需的 BSL 功能
- 编译并加载示例。
- 使用 BSL 调用引脚或任何其他调用方法创建 BSL 调用条件。
- 从主机发送 **GetDeviceInfo** 命令。器件应使用可用的版本信息和 SRAM 缓冲区空间进行响应。
- 类似地，发送擦除、编程、验证命令来对存储器中的数据进行编程。

#### 软件文件详细信息

文件名称	详细信息
flashBSL.c	初始化 BSL 运行所需的外设。接收来自通信接口的命令数据包并将其传递到命令处理层。 请求调用 BSL 后执行。
flashBSL_defines.h	包含 flashBSL.c 的配置和其他定义

文件名称	详细信息
flashBSL_modules.h	包含闪存 BSL 的不同特性，可根据需要配置
flashBSL_invocation .c	检查不同的调用条件，并根据结果分支到应用程序或 BSL
flashBSL_invocation .h	包含与调用相关的配置和宏定义
flashBSL_uart .c	处理主机和 BSL 内核之间的通信
flashBSL_uart .h	包含 BSL 确认的定义
flashBSL_i2c .c	处理主机和 BSL 内核之间的通信
flashBSL_i2c .h	包含 BSL 确认的定义
flashBSL_ci.c	解释命令数据包，处理命令并将响应发送回主机
flashBSL_ci.h	包含 BSL 命令和响应的定义。此外还有 flashBSL_ci.c 的函数声明
boot_config .c	包含启动配置详细信息。它还包含可根据闪存 BSL 实现自定义的 BSL 配置部分。 注意：它并非源自 SYSCFG 工具。
boot_config .h	包含 boot_config .c 的定义
startup_msp0c1104_ticlang.c	BSL 的启动文件。它将 BSL 和调用作为两个独立的例程进行处理。
mspm0c1104.cmd	处理上述存储器布局。

## 7 接口插件

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。这提供了自定义接口的优势，无需重新实现完整的 BSL 内核。

借助此功能：

- 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。例如：SPI、CAN 等（或）
- 可以覆盖 ROM 接口实现 (UART/I2C)

要使用此选项，闪存插件映像应加载到主闪存中并在非主闪存存储器中的 BSL 配置中注册。请参阅 MSPM0xx 技术参考手册的“配置存储器（非主）”部分。

此外，在 BCR 配置中，加载插件和非主配置存储器的闪存存储器区域应受到写保护，以防止闪存插件及其在非主配置中的注册在引导加载过程中被擦除。

### 备注

擦除主闪存中的闪存插件区域可能会导致器件锁定。因此需要写保护。

### 7.1 实现

插件必须负责数据处理和与 BSL 主机的通信。该接口闪存插件通过以下 4 个 API 与 ROM BSL 内核耦合。

- Init
- 接收
- 发送
- Deinit

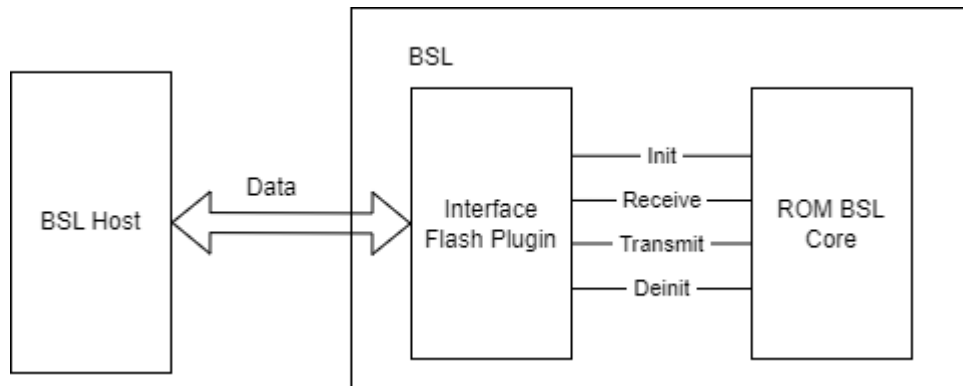


图 7-1. 插件实现

闪存插件映像的构建方式应与任何其他应用程序类似，并将加载到主闪存中。但与应用程序不同，启动代码或主函数将不会执行。ROM BSL 将通过在非主存储器的 BSL 配置中注册的挂钩调用上述 4 个 API。

#### 7.1.1 Init

##### Prototype

```
uint16_t init(uint8_t* buffer, uint16_t bufferSize);
```

**Buffer** - 指向从 BSL 内核发送的 SRAM 数据缓冲区的指针。

**bufferSize** - 可供用作数据缓冲区的 SRAM 存储器大小的 1/2。如果使用两个缓冲区，1 个用于发送，1 个用于接收，一个缓冲区的地址将是“**buffer**”，另一个缓冲区的地址将是“**buffer + bufferSize**”

**return** - 返回 16 位插件版本信息

##### 描述

Init 函数应负责配置所需的接口并初始化用于数据处理的参数。它还负责初始化所使用的任何其他全局变量，因为不会运行启动代码来初始化它们。

### 7.1.2 接收

#### Prototype

**uint32\_t receive(void);**

**return** - 返回接收到的数据包的 32 位起始地址。数据包的格式应与 ROM BSL 协议 [节 4](#) 中描述的格式相同。

#### 描述

接收功能应负责从 BSL 主机读取数据包。只有在接收到完整数据包并检查数据正确性（数据的 CRC 验证）时，它才应与 BSL 内核共享数据包。此外，每个数据包只应共享一次地址。如果 BSL 内核调用此函数，当没有接收到数据包或正在接收数据包时，它应该返回“0”。

如果成功接收到数据包而没有出现任何问题，则应像 ROM BSL 插件那样向主机确认（请参阅 [ROM BSL 确认](#)）。如果出现任何问题，应通过 NACK 向主机报告数据包，并且数据包不应与 ROM BSL 内核共享。

### 7.1.3 发送

#### Prototype

**uint8\_t send(uint8\_t\* data, uint16\_t length);**

**data** - 指向要发送到主机的 BSL 内核响应数据包的指针。其格式与 [节 4](#) 部分中所述的格式相同

**length** - BSL 内核响应数据包的长度，不包括 CRC 4 字节的长度。

**return** - 如果传输成功，则返回“1”。如果传输失败，则为“0”。

#### 描述

Transmit 必须向主机发送 BSL 内核响应数据包。它还负责计算和添加数据包的 CRC。Transmission API 应仅在完全传输数据后返回。

### 7.1.4 Deinit

#### Prototype

**bool deinit(void);**

#### 返回

如果 deinit 完成，则返回“True”。

#### 描述

此函数会复位接口配置并取消寄存中断处理程序（如果已注册）。

### 7.1.5 重要注意事项

在开发闪存插件时需要记住的要点。

1. 加载闪存插件的主闪存存储器区域应受静态写保护
2. 所有全局变量都应由“Init”函数初始化
3. 4 个插件 API 的函数原型应符合 BSL 用户指南中的规定。
4. SRAM 存储器使用情况
  - a. VTOR - SRAM 起始地址 (0x20000000)。如果使用中断，VTOR 应放置在 SRAM 的开头，因为 ROM BSL 使用该地址空间
  - b. 栈起始地址 - 器件中可用 SRAM 存储器的结束地址
  - c. 栈大小 - 不应超过 ROM BSL 栈大小
  - d. 数据段（.data、.bss）- 当器件中未注册闪存插件时，Get Device Info 命令返回的“BSL Buffer Start address”应为数据段的起始地址。

- e. 数据段大小 - 数据段 ( .data、.bss ) 消耗的大小应在 BSL 非主配置存储器中配置。

## 7.2 闪存插件类型

闪存插件中使用的通信接口类型在 BSL 配置存储器的 BSLPLUGINCFG.PLUGINTYPE 字段中进行配置 ( 请参阅技术参考手册 )。

如果插件类型具有任何 ROM 接口类型 ( UART 或 I2C )，闪存插件将覆盖 ROM 实现。如果该类型具有 ROM 中不可用的新类型 ( 例如 SPI )，插件将作为新接口添加。

表 7-1. 插件类型值

编号	接口	插件类型
1	UART	0x1000
2	I2C	0x2000
3	任何其他接口	0xFFFF

如图 7-2 所示，检查闪存插件类型后，更新后的接口列表将用于自动检测。即使在闪存插件注册后，如果 ROM 接口未被覆盖，也可以使用这些接口。

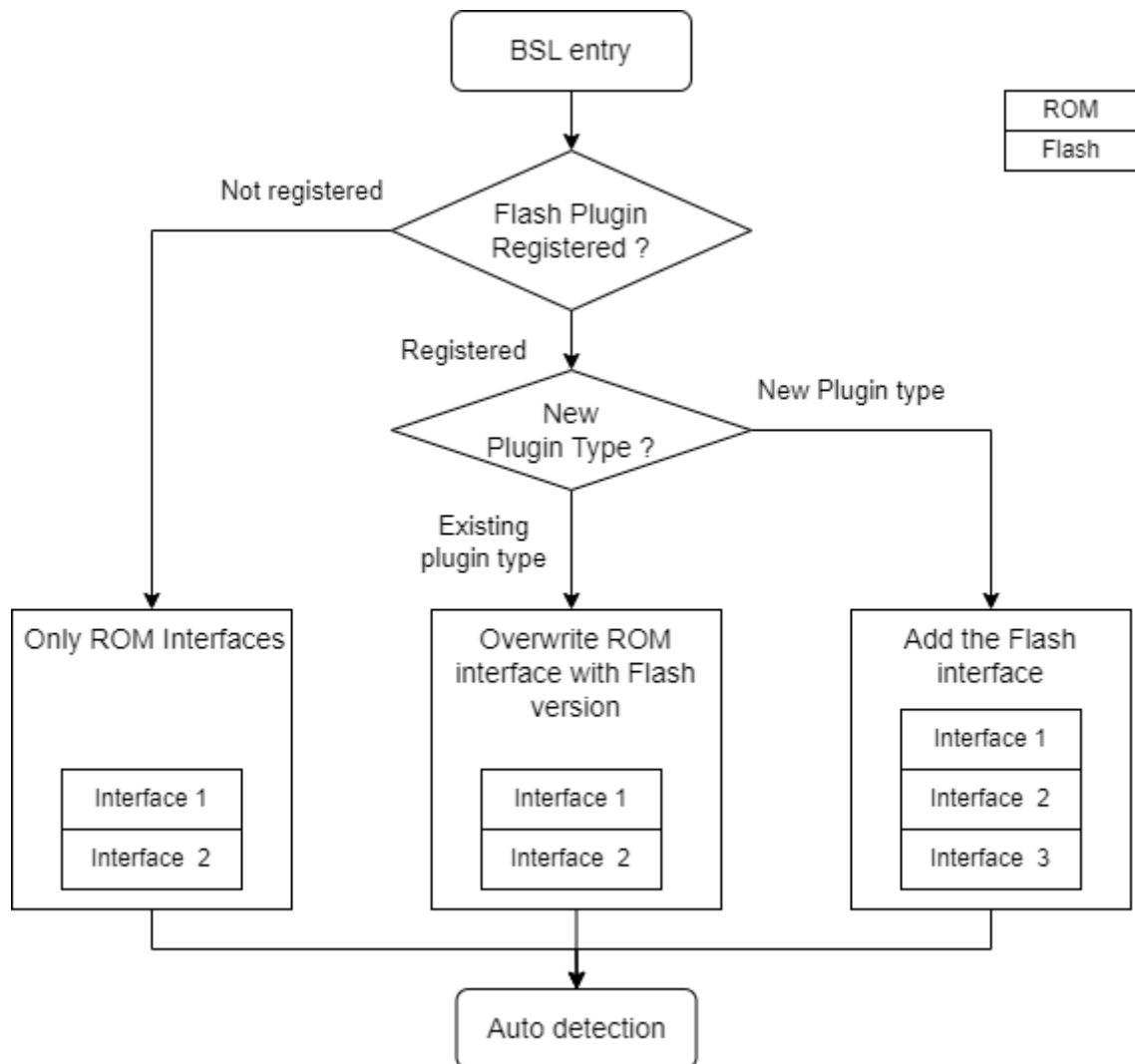


图 7-2. 闪存插件



## 7.3 覆盖现有接口

此功能允许用户使用闪存版本覆盖现有 ROM BSL 接口、UART 和 I2C。在这种情况下，一旦加载了闪存插件，ROM 版本即变为无效，不能使用。要将器件恢复为使用 ROM 接口，请使用 SWD\_Factory\_Reset (通过调试子系统邮箱恢复出厂设置)。

### 7.3.1 UART 接口闪存插件示例

使用 UART 通信的闪存插件示例作为 SDK 示例的一部分提供，以供参考。本节对此进行更加详细的介绍。

#### 说明

UART 接口闪存插件通过以下 4 个 API 挂钩来处理 BSL 主机和 ROM BSL 之间的数据事务。

- BSL\_PI\_UART\_init
- BSL\_PI\_UART\_receive
- BSL\_PI\_UART\_send
- BSL\_PI\_UART\_deinit

UART 闪存插件主要用于在需要时通过自定义实现覆盖 ROM BSL UART 接口。

#### 示例用法

- 将 UART\_RX 和 UART\_TX 与 BSL 主机 (任何带有 UART 接口的微控制器) 连接。
- 编译并加载示例。
- 使用 BSL 调用引脚或任何其他调用方法创建 BSL 调用条件。
- 从主机发送 **Connection** 命令。成功时会收到 BSL 确认。
- 从主机发送 **GetDeviceInfo** 命令。
- BSL 使用 UART 接口闪存插件版本信息进行响应。
- 类似地，发送擦除、编程、验证命令来对存储器中的数据进行编程。

#### 软件文件详细信息

文件名称	详细信息
bsl_uart.c	处理主机和 BSL 内核之间的通信。定义四个接口 API Init、Receive、Send 和 Deinit。
bsl_uart.h	包含 BSL 确认的定义和 bsl_uart.c 的函数声明
ti_msp_dl_config.h	包含特定于器件的配置，如 UART 引脚、时钟配置等。
boot_config.h	包含 BCR 和 BSL 配置结构
startup_mspm0x_ticlang	仅包含默认处理程序函数定义的启动文件。与典型的启动文件 (没有中断矢量表或复位处理程序) 不同，因为这些功能从未在闪存插件中使用，并且会被移除以降低存储器消耗。
mspm0x.cmd	指定存储器区域 (闪存插件映像的驻留位置) 和 SRAM 区域 (闪存插件的运行位置) 的链接器命令文件。

#### 定制

此示例提供了闪存插件的参考设计。可根据需要对其自定义。接口闪存插件 API 是主要的更改位置。

要执行的步骤：

- 根据需要修改闪存插件 API
- 更改完成后，编译代码
- 如果 API 指针已更新，则计算 BSL 配置的 CRC
- 根据需要修改 BCR 配置中的闪存写入保护设置
- 计算 BCR 配置的 CRC
- 将新 CRC 存储在 BCR 和 BSL 配置中。
- 再次编译代码。
- 加载闪存插件映像



## 8 修订历史记录

注：以前版本的页码可能与当前版本的页码不同

### Changes from FEBRUARY 28, 2023 to JANUARY 31, 2026 (from Revision \* (February 2023) to Revision A (January 2026))

	Page
• 在 MSPM0 器件中添加了不同的 BSL 类型。 .....	25

## 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。

版权所有 © 2026，德州仪器 (TI) 公司

最后更新日期：2025 年 10 月