应用手册

专用固件区读保护使用指南

适用范围：

- SYM32 系列所有支持专用固件区读保护功能的芯片

文档说明

本应用手册将详细介绍 FLASH 控制器的专用固件区读保护功能。

1 专用固件区

适用于本手册的芯片其 FLASH 存储器具有专用固件区（下文统一简称为 Slib 区）读保护功能，使能该功能可以禁止对 Slib 区以任何形式的读取操作（CPU、SWD、ISP 均无法读取 Slib 区内容），但不影响调用和运行处于 Slib 区的程序。

用户可将核心算法或数据安全地存储于 Slib 区以供其二次开发客户调用，而不用担心算法或数据泄露问题，能大量节省防盗、防逆向成本。

2 如何使能专用固件区读保护功能

Step1. 在 FLASH 特定地址写入控制字[注 1]，以 SYM32L010F8P6 为例，在 FLASH 地址 0xFFF0 ~ 0xFFFF 区域写入控制字，如下方所示。

0xFFF0	0xFFF1	0xFFF2	0xFFF3	0xFFF4	0xFFF5	0xFFF6	0xFFF7
SlibKey0	SlibKey1	SlibKey2	SlibKey3	SlibKey4	SlibKey5	SlibKey6	SlibKey7
0xFFF8	0xFFF9	0xFFFA	0xFFFB	0xFFFC	0xFFFD	0xFFFE	0xFFFF
StartIdx	EndIdx	0x5A	0x5A	0xA5	0xA5	CRCL	CRCH

各地址内控制字的含义：

- SlibKey0 ~ SlibKey7 为专用固件区密码，用于禁止专用固件区读保护功能；用户应根据实际需求进行设置并妥善保管，切勿将密码设置为全 FF、全 00
- StartIdx 为专用固件区的起始页面序号，用户应按实际情况进行设置
- EndIdx 为专用固件区的结束页面序号，固定为 0x7F
- CRCL 为采用 CRC16_X25 算法对 0xFFF0~0xFFFD 数据计算所得结果的低字节
- CRCH 为采用 CRC16_X25 算法对 0xFFF0~0xFFFD 数据计算所得结果的高字节

控制字的设置示例及参考请见本手册的样例演示章节——*Slib_LIB* 样例。

Step2. 写入控制字后，通过 NRST 复位或上电复位即可使能 Slib 区读保护功能。

读保护功能使能后，可查看 FLASH_SLIBCFG 寄存器[注 2]确认其 Slib 区起始页面序号和 Slib 区结束页面序号是否和控制字所配置的一致。如果成功使能则应该与控制字配置值一致，若为默认值则表示使能未成功[注 3]。

注 1：控制字格式以及写入的 FLASH 地址请见芯片的用户参考手册 FLASH 章节相关描述

注 2：该寄存器描述包括默认值请见芯片的用户参考手册的 FLASH 章节寄存器描述

注 3：若使能未成功，请确认控制字是否配置正确

3 如何禁止专用固件区读保护功能

不同芯片禁止专用固件区读保护功能的方法略有不同，总共有两种方法：

- 通过 ISP 协议执行片擦操作：芯片接收到片擦指令及正确的专用固件区密码时，将自动擦除本芯片的所有数据并禁止专用固件区读保护功能；芯片接收到片擦指令及错误的专用固件区密码时，不执行任何操作。

  ISP 协议详细介绍请参考《SYM32 系列 MCU 闪存串行编程协议》。
- 在 RAM 中对芯片执行片擦操作：配置编译器参数，从 RAM 中执行指令，对专用固件区域外地址执行写操作触发片擦；片擦操作完成后，本芯片的所有数据均被擦除并禁止专用固件区读保护功能。

  此方法仅适用于 SYM32L010 和 SYM32L011。

4 样例演示(MDK)

适用于本手册的芯片，其驱动库提供使用 Slib 读保护功能的演示样例。
下文均以 SYM32L010 驱动库以及 SYM32L010F8P6 芯片为例。
SYM32L010 驱动库目录下"…\Examples\EXT_Slibrary\"有演示样例如下图所示：

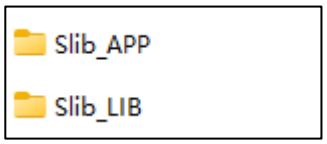

图 1 样例目录

分为两个部分：

- Slib_APP 样例以二次开发者的视角，演示调用 Slib 区固件，以及演示通过 CPU 和 SWD 读取 Slib 区内容。
- Slib_LIB 样例以一次开发者视角，演示编写 Slib 固件，以及演示如何启用 Slib 的读保护功能。

4.1 Slib_LIB 样例

该样例的目的是创建 Slib 区域并启用读保护功能，一次开发用户可采用此样例的方式，将核心算法编译下载在 Slib 区，并启用 Slib 区读保护功能。该样例对于 Slib 控制字的写入方式采用下载写入，即通过烧录就可直接将控制字写入 FLASH，下载后再进行 NRST 复位或上下电复位即可生效 Slib 区读保护功能。

该样例目录结构如图所示：

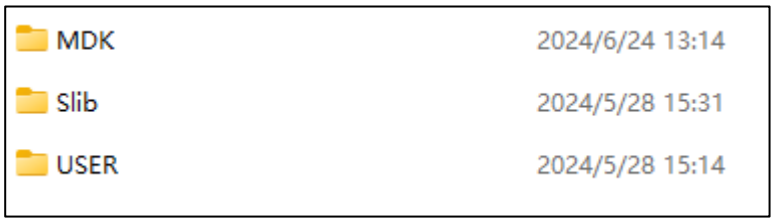

图 2 Slib_LIB 样例

Slib 文件夹中有 Slib.c 和 Slib.h 文件，Slib.c 文件在本样例中作为要被保护的固件库，一次开发者的核心算法将写在其中；Slib.h 文件内定义了被保护固件库的若干个函数指针宏，一次开发者需要将它提供给二次开发者，二次开发者就可以通过宏来调用 Slib 库。

以下给出具体操作步骤：

Step1. 为 Slib 区域分配合适的 FLASH 空间：用户需要根据要被保护的固件的体积大小来设计 Slib 的空间大小，并按设计的大小对 MDK 进行配置。按下图所示打开 MDK 选项卡，以该样例为例，on-chip ROM 被划分为两个区域 IROM1 和 IROM2，其中 IROM2 的配置就是为了将其作为 Slib 区域，本样例 Slib 被分配 0x400 Byte 空间。

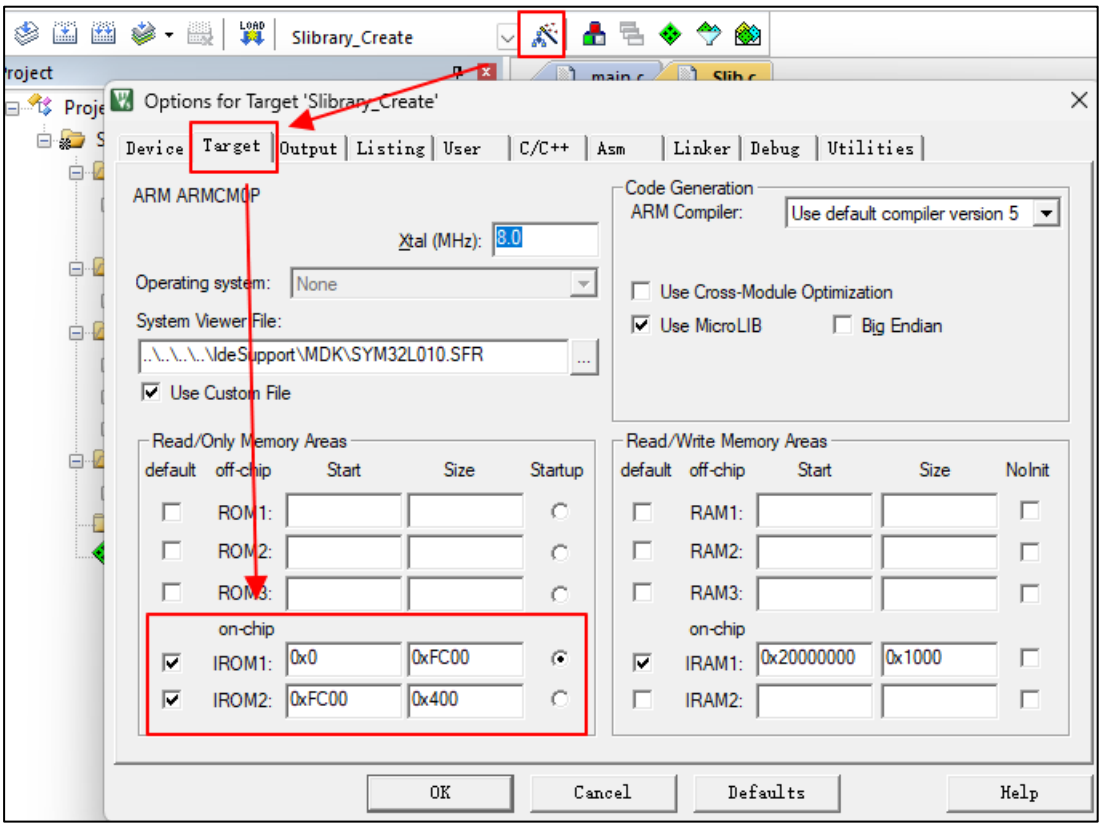

图 3 MDK 配置 FLASH 区域

同时，Linker 选项卡中需要勾选 Use Memory Layout from Target Dialog 选择框。

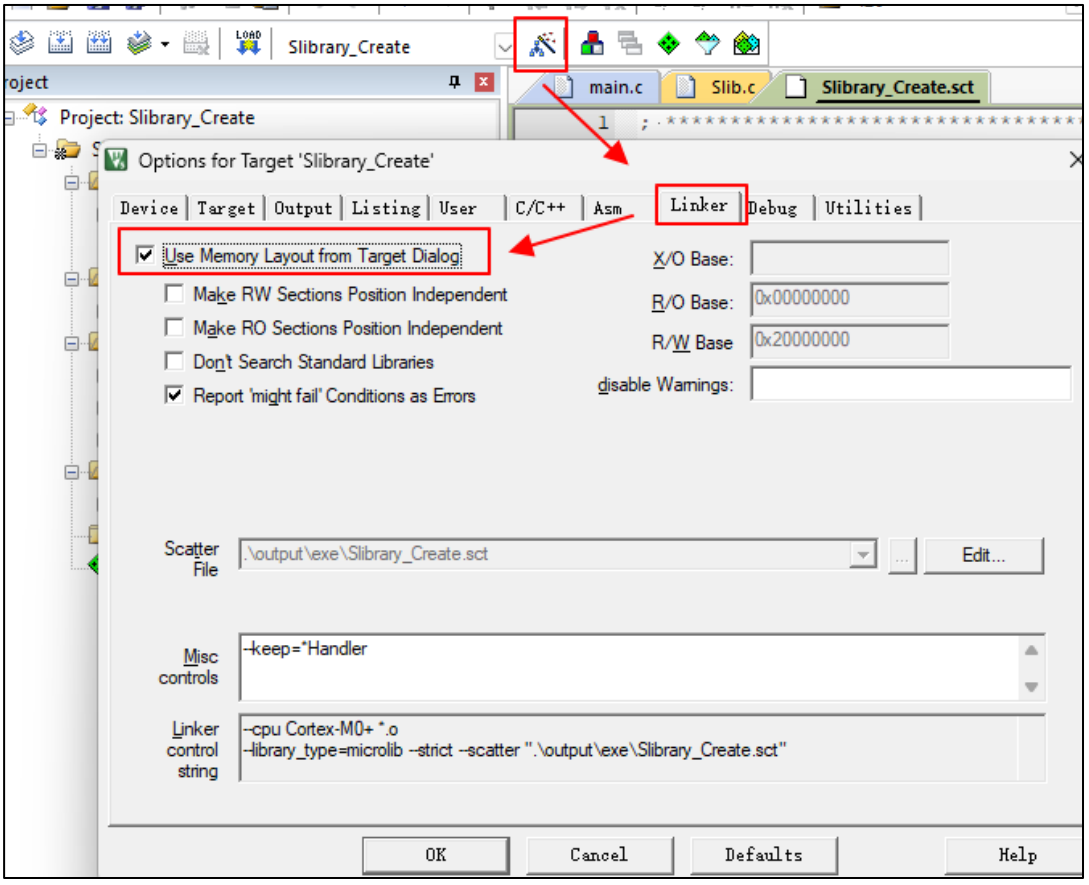

图 4 Linker 选项卡配置

Step2. 编写 Slib 区固件。其中包含一些**注意事项**如下列所示：

- 在 Slib.c 文件中使用"**__attribute__**"关键字将函数链接在指定地址，这样就可以通过函数指针的形式调用函数，方便二次开发。关于如何使用该关键字指定函数链接地址，参考下图所示：

```
void Slib_Led_Blink(void)           __attribute__((section(".ARM.__at_0x0000FC00")));
void Slib_Delay(uint32_t DlyCnt)    __attribute__((section(".ARM.__at_0x0000FE00")));
uint32_t Slib_GetValue(void)        __attribute__((section(".ARM.__at_0x0000FE40")));
```

上述的函数声明，需要在函数定义前声明。

- 通过宏的形式来确保 Slib 函数的可见性，在 Slib.h 文件中定义函数指针宏，如下图所示。将 Slib.h 文件提供给二次开发客户，二次开发时就能调用 Slib 区内的函数。

```
/*****************************************************************
 * @brief  Slib 区函数，Led 闪烁 PA03
 * @param  None
 * @retval None
 */
#define SlibApi_Led_Blink       ((void (*)(void))      (0x0000FC00UL + 1))
```

注意：指针值必须是：函数地址 +1，否则会导致偶发性错误。

Step3. 在 Slib.c 中定义 Slib 控制字常量数组，并链接地址到 0xFFF0，如下所示，其中 SlibKeys、StartIdx、EndIdx、CTRL_KEY 和 CRCL、CRCH 都是宏。定义了这样的数组，下载固件时便可直接把 Slib 控制字烧录到指定区域。

```
const  uint8_t  Slib_Record[]  __attribute__((section(".ARM.__AT_0x0000FFF0")))  =
{
    // SlibKey0~7
    SlibKeys,
    // StartIdx and EndIdx
    StartIdx & 0x7F, EndIdx & 0x7F,
    // Mcu Key
    CTRL_KEY,
    // CRC
    CRCL, CRCH
};
```

编辑 Slib 控制字，根据需要，编辑这些宏从而编辑要写入的控制字。

```
#define SlibKeys    0x55, 0x55, 0x55, 0x55, 0xAA, 0xAA, 0xAA, 0xAA    // SLIB 区域的 KEY 值（由用户配置）
#define StartIdx   (0x7E)                                             // SLIB 区域起始 Page 索引（由用户配置）
#define EndIdx     (0x7F)                                             // SLIB 区域结束 Page 索引（用户不可更改）
#define CTRL_KEY    0x49, 0x53, 0xA5, 0xA5                            // SLIB 区域特定字符序列（用户不可更改）
#define CRCL       (0x0)                                              // SLIB 区域的 CRC16_X25（由用户计算得到并填入）
#define CRCH       (0x0)                                              // 请计算正确的 CRC16_X25，并填写 CRCL 和 CRCH
```

其中 SlibKeys 有 8 字节，它是用户自己设置的密码，用于在 ISP 协议下擦除 Slib 并关闭 Slib 读保护；其中 CRCL 和 CRCH 分别为 CRC16 的低字节和高字节，CRC16 需要通过计算得出，可以使用 PC 计算或者形如该样例 main 程序使用 CRC 外设计算。

由于本样例分配 0x400 Byte 给 Slib，查阅芯片手册可知 SYM32L010F8P6 的 FLASH 存储器构成：其主存储区包含 128 个 Page，每 Page 包含 512 Byte。即 0x400 Byte 为 2 个 Page 的空间，所以 StartIdx 选为 0x7E，即 Slib 占用 0x7E 和 0x7F 这 2 个 Page。

Step4. 写好固件库并且配置好控制字后，编译、下载。

Step5. 下载固件后，再按下 NRST 按键，Slib 区域读保护生效。进入调试模式在 Memory 窗口访问 Slib 区 FLASH 如下图所示，生效后的 Slib 区 FLASH 读出全 0，同时 Slib 区内函数能正常调用，LED 闪烁。

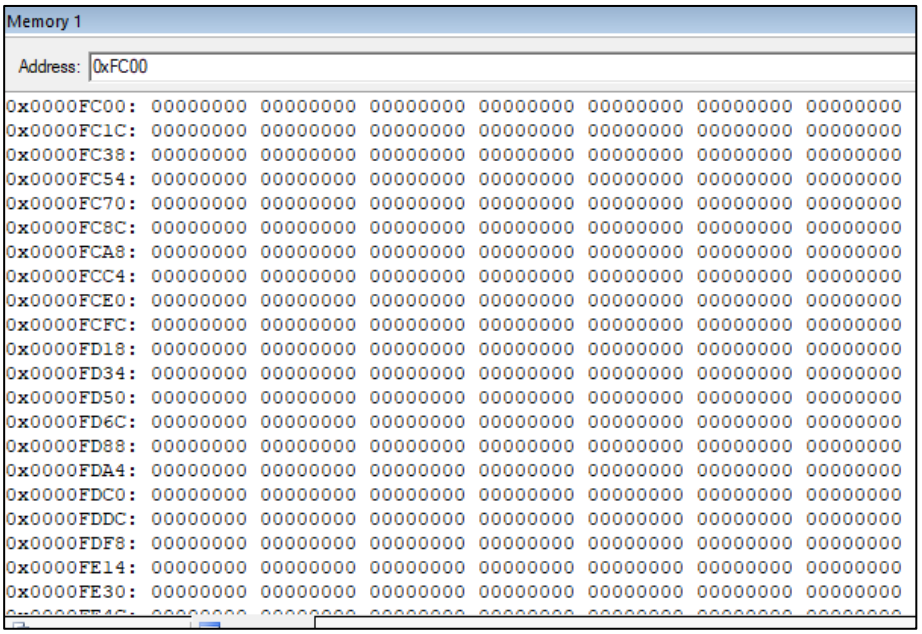

图 5 Slib 读保护生效后查看 Slib 中的 Flash 内容

4.2 Slib_APP 样例

该样例模拟二次开发者的环境：芯片已经下载 Slib 固件，同时 Slib 读保护已生效。除此之外，二次开发者需要拿到一次开发者提供的函数声明，以 Slib_LIB 样例为例，即为 Slib.h 文件，通过 Slib.h 中的函数指针宏，就能调用处于 Slib 中的程序。

以下给出该样例的具体操作步骤：

Step1. 将 Slib.h 添加至工程，不需添加 Slib.c 文件，工程结构如下图所示：

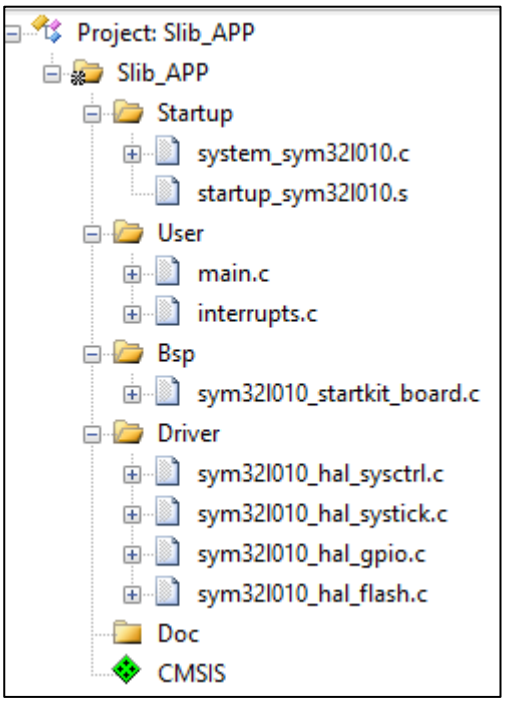

图 6 Slib_APP 工程目录

Step2. 由于预先下载的 Slib 区占用一定的 FLASH 空间，因此需要对 MDK 的 on-chip ROM 配置进行更改，如下图所示：

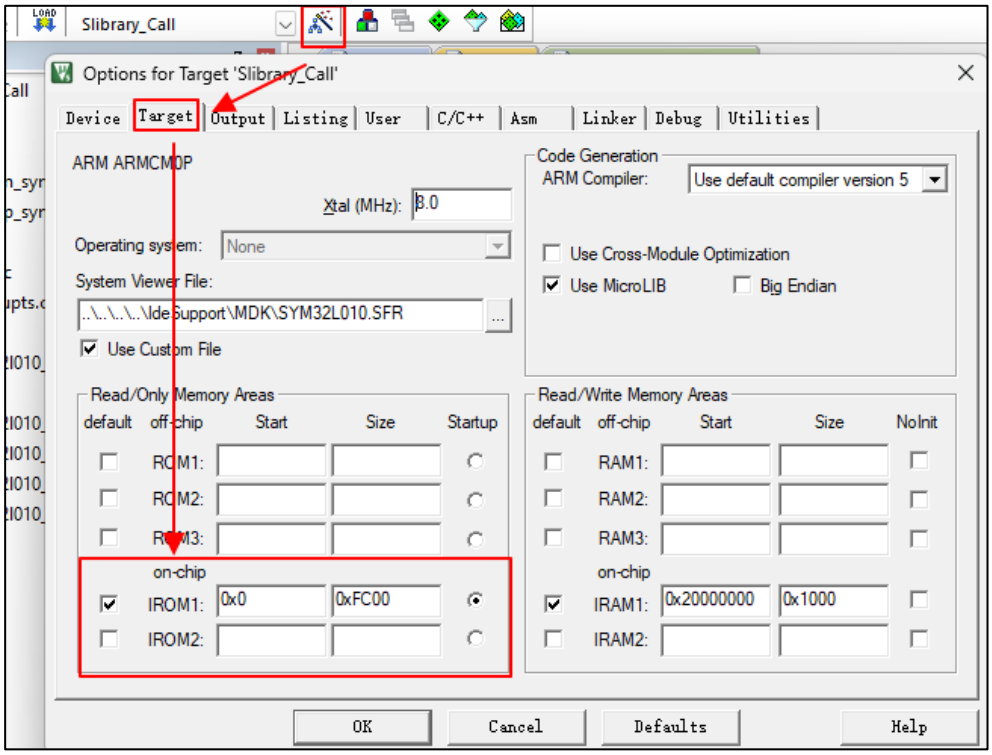

图 7 MDK 配置 FLASH 区域

Slib 区占用 0x400 Byte 空间，则需要调整 IROM1 的 Size 为 0xFC00。

同 Slib_LIB 样例，需要勾选 Use Memory Layout from Target Dialog 选择框。

Step3. 在 main 中直接调用宏，并验证 API 功能的正确性。尝试用 CPU 读取 Slib 区域的起始 16

个字节，验证是否为全 0。

若全部验证成功则 LED 快闪，约每秒闪烁 2 次；否则慢闪，约每 2 秒闪烁一次。

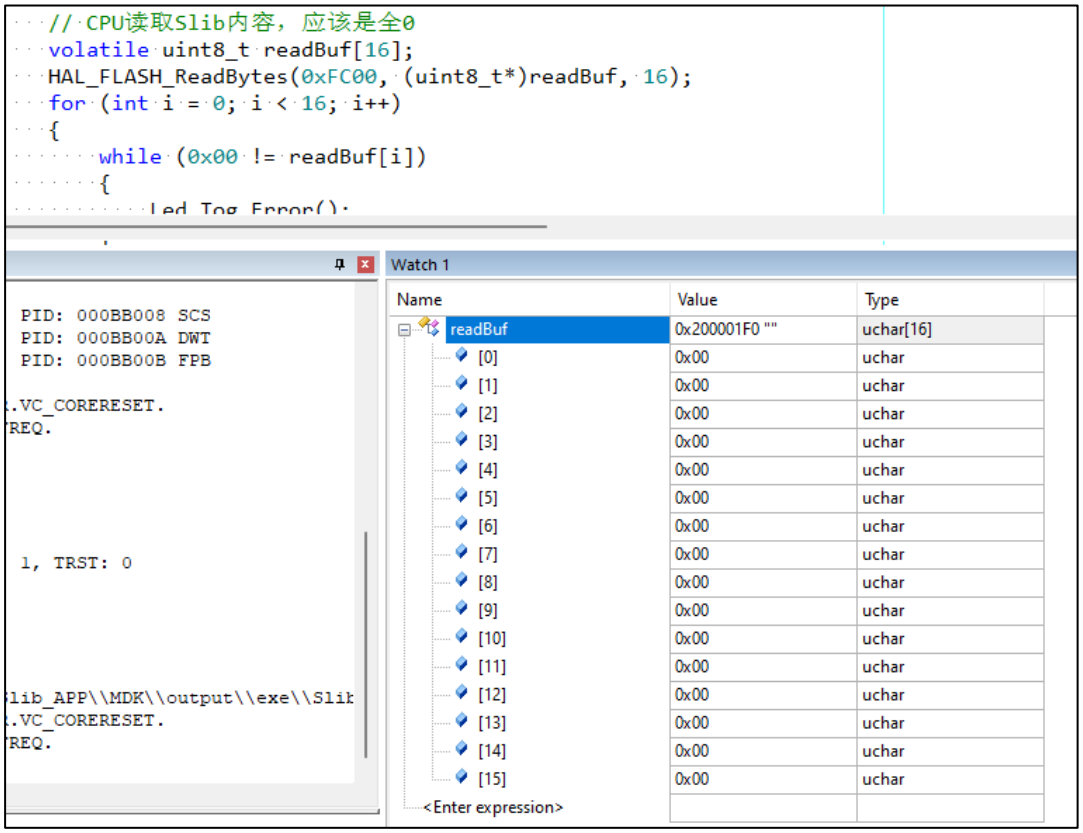

图 8 通过 CPU 读取 Slib

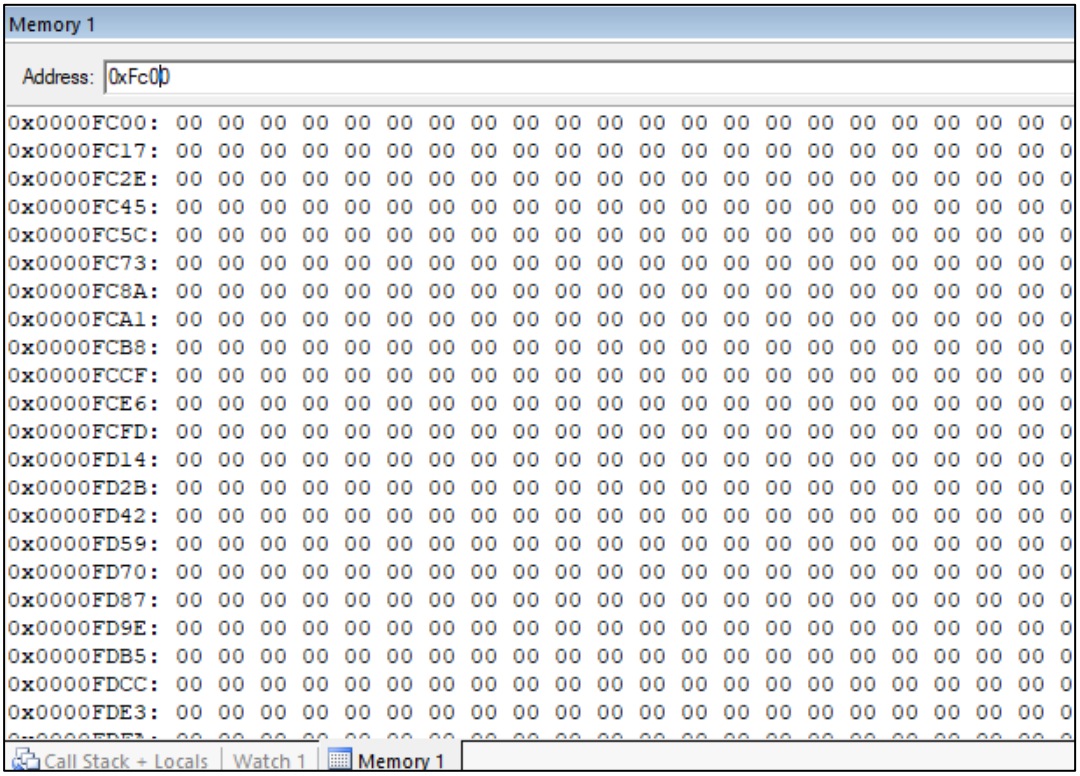

图 9 Memory 窗口下的 Slib 区域内容

版本记录

版本	修订日期	修订说明
Rev1.0	2024-06-20	初始版本