文档编号：AN-MEM-002
第 1.0 版，2022 年 11 月

Z20K14xM Flash 保护指南

1 简介

本应用笔记主要介绍如何使用 Z20K14xM 系列 MCU 的 Flash 保护功能以及在应用开发过程中需要考虑的因素。

目录

2 Flash 保护介绍

2.1 Flash 保护的作用

Z20K14xM 系列 MCU 实现的 Flash 保护，可用于阻止无权限用户访问 Flash 和 RAM。需要注意的是，该保护功能无法阻止自身代码读取内存并向外发送。

设置 Flash 保护后，芯片会：

- 保护 Flash 和 RAM 中的内容
- 禁止在扩展模式下访问内部 Flash/RAM
- 禁止通过 J-Link 对调试口进行调试

2.2 Flash 保护功能

Flash 的保护是通过在地址 0x02000000 的低 16 位写入 0xF5EC 实现的。向该地址中写入 0xF5EC，MCU 复位后保护生效，不能通过 DAP (Debug Access Port) 访问 MCU 的 Flash 和 RAM。

Flash 保护功能分为传统保护和带 backdoor 保护两种。需要注意的是，两种保护之间不能直接切换。使用传统保护后可以直接转成带 backdoor 的保护，但**使用带 backdoor 的保护后只能直接擦除 Flash，不能直接转成传统保护**。

2.2.1 传统保护

传统保护是指仅在地址 0x02000000 的低 16 位写入 0xF5EC 的方法，通过该方法配置后，只能用 JTAG/SWD 对 Flash 进行擦除操作才可以解除保护并恢复 DAP 访问。

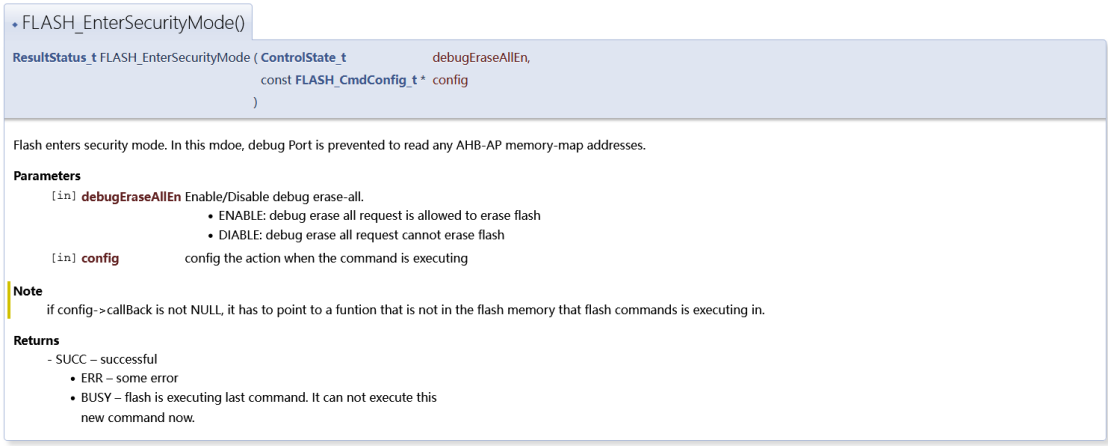
注意：如果在写入 0xF5EC 的基础上，再在地址 0x02000000 的高 16 位写入 0x3EE2，即在地址 0x02000000 写入 0x3EE2F5EC，复位后会禁止外部 DAP 对 Flash 的擦除权限，前面提到的解除保护的方法失效。此时只能使用内部预留的擦除代码擦除 Flash 上的内容，**如果没有预留代码 MCU 内的代码将彻底无法修改**。附录 B 中使用串口作为预留代码，可以通过接收串口命令擦除 Flash。

传统保护有三种实现方式，具体描述如下：

方式一

由 MCU 程序直接写入：

Z20K14xM 的 0.7 版 SDK 中有函数 FLASH_EnterSecurityMode 可直接用于 Flash 保护。



• FLASH_EnterSecurityMode()

ResultStatus_t FLASH_EnterSecurityMode (ControlState_t debugEraseAllEn,
const FLASH_CmdConfig_t * config
)

Flash enters security mode. In this mdoe, debug Port is prevented to read any AHB-AP memory-map addresses.

Parameters

[in] **debugEraseAllEn** Enable/Disable debug erase-all.
- ENABLE: debug erase all request is allowed to erase flash
- DIABLE: debug erase all request cannot erase flash

[in] **config** config the action when the command is executing

Note

if config->callBack is not NULL, it has to point to a funtion that is not in the flash memory that flash commands is executing in.

Returns

- SUCC – successful
- ERR – some error
- BUSY – flash is executing last command. It can not execute this new command now.

具体调用方法如下：

```
FLASH_EnterSecurityMode(ENABLE, &FLASHTEST_CmdExeConfig);   //Flash 命令配置，允许擦除 Flash
```

其中，配置结构体变量如下：

```
FLASH_CmdConfig_t FLASHTEST_CmdExeConfig =
{
    FLASH_CMD_ACT_WAIT,  //act，定义命令执行时的动作
                         // - FLASH_CMD_ACT_WAIT: 等待直到命令完成
    0
                         //回调函数，关闭时间件，如果“act”参数设置为 FLASH_CMD_ACT_WAIT，会在命令完成后调用该函数
};
```

方式二

在烧录程序时由下载器写入，可在 hex 文件中定义一个位于 0x02000000 处的 const，在下载程序时直接写到这个地址处，如下：

```
__root const uint16_t Securecode @0x02000000 = 0xF5EC;
```

方式三

上位机可以通过下载器直接操作 Flash controller 的 register 在 0x02000000 直接写。

一般通过下载器完成的，比如戎象下载器。

Method1：下载器通过 J-Link 直接操作寄存器，写入 0xF5EC；

Method2：下载器下载一段程序到 SRAM 运行，然后通过 SRAM 程序向 Flash 中写入 key 值；

其中，第二种方法（Method 2）会比较常用。

2.2.2 带 backdoor 的保护

相比于 Z20K11xM，Z20K14xM 的 Flash 保护多了 backdoor 功能。顾名思义，backdoor 是在 MCU 配置 Flash 保护后可以通过密码值解除保护，发起一次临时的程序访问（可以使用 IAR 中的 Attach to running target 功能）。MCU 会在复位后再次恢复为保护状态，不能再通过 DAP 调试。

在使用带 backdoor 的保护功能时，除了在 0x02000000 上写入 0xF5EC 之外，还需要在其余几个地址上写入**自定义**的密码值。例如，可以在下表列出的后三个地址中写入以下对应的密码值：

地址	数值
0x02000000	**0xFFFFF5EC**
0x02000004	**0x26332556**
0x02000008	**0x02122504**
0x0200000C	**0x029865D4**

注意：与传统保护一样，如果再在地址 0x02000000 的高 16 位写入 0x3EE2，复位后不能通过外部 DAP 访问 MCU。带 backdoor 保护后还可以通过内部软件开启 backdoor，允许临时的 DAP 访问，但仍然无法擦除 Flash，具体函数及调用方法可以参考 2.3.2 章节：带 backdoor 保护。此时只能使用内部预留的擦除代码擦写 Flash 上的内容，**如果没有预留代码 MCU 内的代码将彻底无法修改**。附录 B 中使用了串口作为接口，可以通过串口接收命令擦除 Flash。

函数 Flash_ProgramPhrase 可以用于在 Flash 上的特定地址写入数值：

• FLASH_ProgramPhrase()

```
ResultStatus_t FLASH_ProgramPhrase ( uint32_t                  flashAddr,
                                     const uint8_t *           dataP,
                                     const FLASH_CmdConfig_t * config
                                   )
```

program a flash phrase

Parameters

[in] **flashAddr** phrase start address to be programmed
[in] **dataP** points to data to be programmed into flash
[in] **config** config the action when the command is executing

Note

if config->callBack is not NULL, it has to point to a funtion that is not in the flash memory that flash commands is executing in.

Returns

- SUCC – successful
- ERR – some error
- BUSY – flash is executing last command. It can not execute this new command now.

具体调用方法如下：

```
uint32_t uData[4U];

uData[0] = 0xFFFFF5ECU;
uData[1] = 0x26332556U;
uData[2] = 0x02122504U;
uData[3] = 0x029865d4U;
FLASH_ProgramPhrase(0x02000000, (uint8_t *)uData, &FLASHTEST_CmdExeConfig);
```

2.3 解除保护

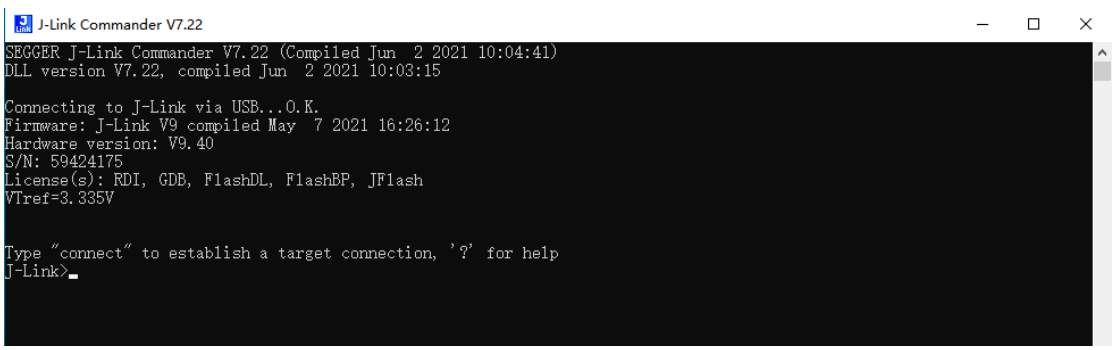
Flash 配置保护后，程序不可访问且不可通过 DAP 调试，开发者如果要再次调试程序，可使用解保护脚本将 Flash 全部擦除后再次访问，或者内部应用软件开启 backdoor 允许临时的 DAP 访问。

2.3.1 传统解除保护方式

方法：通过 J-Link Commander 刷写脚本解除保护，脚本详见附录 A

操作步骤：

1、打开 J-Link Commander

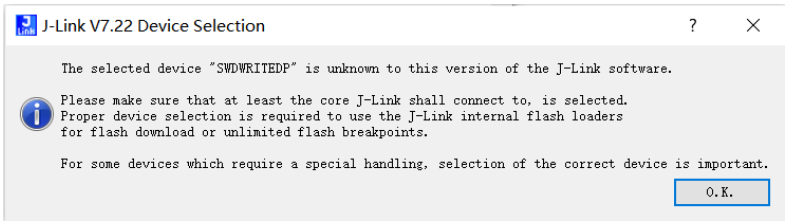


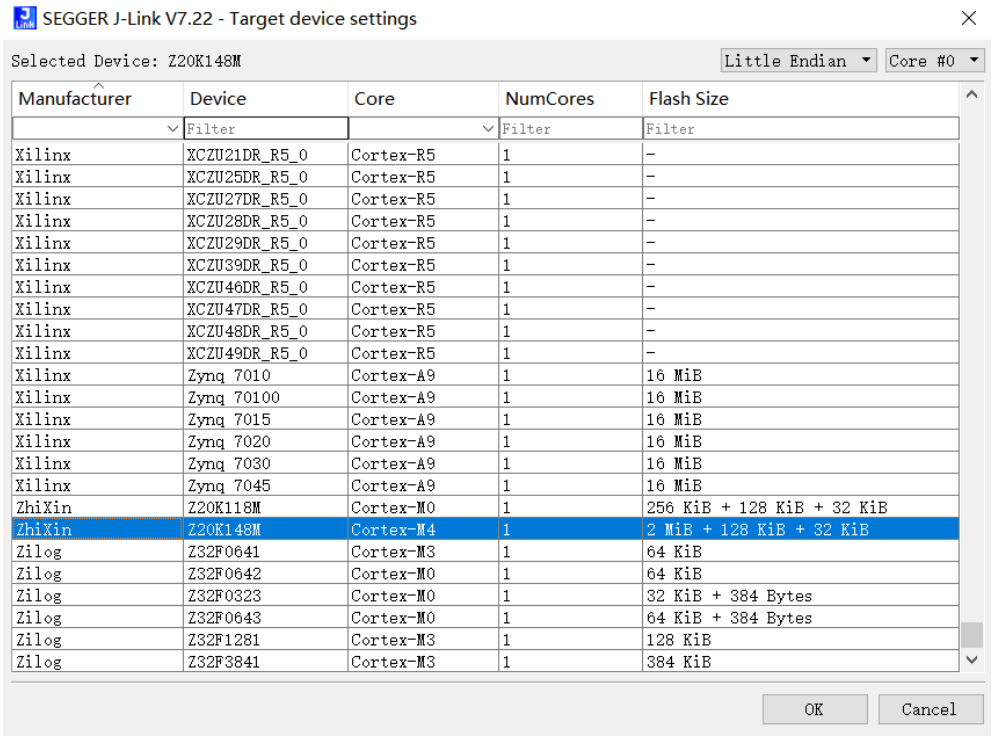
2、打开文件 unlock_cmd.txt 脚本（附录 A）

3、将 unlock_cmd.txt 中的所有指令复制

4、将上一步复制的指令粘贴到 J-Link Commander，程序会自动执行

5、程序执行过程中会跳出以下对话框用于选择芯片型号，选择“Z20K14xM”。





6、程序执行到最后时会停在“h”，按回车执行，暂停 CPU，如果程序正确执行，会出现下图结果：

```
Cortex-M4 identified.
J-Link>h
PC = FFFFFFFE, CycleCnt = 03DF9D03
R0 = 00000000, R1 = 00000000, R2 = 00000000, R3 = 00000000
R4 = 00000000, R5 = 00000000, R6 = 00000000, R7 = 00000000
R8 = 00000000, R9 = 00000000, R10= 00000000, R11= 00000000
R12= 00000000
SP(R13)= FFFFFFD8, MSP= FFFFFFD8, PSP= 00000000, R14(LR) = FFFFFFF9
XPSR = 01000003: APSR = nzcvq, EPSR = 01000000, IPSR = 003 (HardFault)
CFBP = 00000000, CONTROL = 00, FAULTMASK = 00, BASEPRI = 00, PRIMASK = 00

FPS0 = 00000000, FPS1 = 00000000, FPS2 = 00000000, FPS3 = 00000000
FPS4 = 00000000, FPS5 = 00000000, FPS6 = 00000000, FPS7 = 00000000
FPS8 = 00000000, FPS9 = 00000000, FPS10= 00000000, FPS11= 00000000
FPS12= 00000000, FPS13= 00000000, FPS14= 00000000, FPS15= 00000000
FPS16= 00000000, FPS17= 00000000, FPS18= 00000000, FPS19= 00000000
FPS20= 00000000, FPS21= 00000000, FPS22= 00000000, FPS23= 00000000
FPS24= 00000000, FPS25= 00000000, FPS26= 00000000, FPS27= 00000000
FPS28= 00000000, FPS29= 00000000, FPS30= 00000000, FPS31= 00000000
FPSCR= 00000000
J-Link>
```

7、此时输入“connect”验证芯片是否已经解除保护。如果出现以下结果，证明芯片已正确解除保护。

```
Device "Z20K148M" selected.


Connecting to target via SWD
InitTarget() start
*************************************************************
J-Link script: Z20K148M M4F core J-Link script
*************************************************************
InitTarget() end
Found SW-DP with ID 0x2BA01477
DPIDR: 0x2BA01477
Scanning AP map to find all available APs
AP[2]: Stopped AP scan as end of AP map has been reached
AP[0]: AHB-AP (IDR: 0x24770011)
AP[1]: JTAG-AP (IDR: 0x0C860000)
Iterating through AP map to find AHB-AP to use
AP[0]: Core found
AP[0]: AHB-AP ROM base: 0xE00FF000
CPUID register: 0x410FC241. Implementer code: 0x41 (ARM)
Found Cortex-M4 r0p1, Little endian.
FPUnit: 6 code (BP) slots and 2 literal slots
CoreSight components:
ROMTbl[0] @ E00FF000
ROMTbl[0][0]: E000E000, CID: B105E00D, PID: 000BB00C SCS-M7
ROMTbl[0][1]: E0001000, CID: B105E00D, PID: 003BB002 DWT
ROMTbl[0][2]: E0002000, CID: B105E00D, PID: 002BB003 FPB
ROMTbl[0][3]: E0000000, CID: B105E00D, PID: 003BB001 ITM
ROMTbl[0][4]: E0040000, CID: B105900D, PID: 000BB9A1 TPIU
ROMTbl[0][5]: E0041000, CID: B105900D, PID: 000BB925 ETM
SetupTarget() start
Init SRAM Begin
Init SRAM End
SetupTarget() end
Cortex-M4 identified.
```

8、如果在“h”之前出现以下结果，证明芯片没有正确解除保护。此时需要在回车之后重复步骤 4～6，直到“h”之前没有“ERROR”。

```
InitTarget() end
Found SW-DP with ID 0x0BC11477
DPIDR: 0x0BC11477
Scanning AP map to find all available APs
AP[2]: Stopped AP scan as end of AP map has been reached
AP[0]: AHB-AP (IDR: 0x04770031)
AP[1]: JTAG-AP (IDR: 0x0C860000)
Iterating through AP map to find AHB-AP to use
AP[0]: Skipped. Could not read CPUID register
AP[1]: Skipped. Not an AHB-AP

****** Error: Could not find core in Coresight setup

Cannot connect to target.
J-Link>usb
Disconnecting from J-Link...O.K.
Connecting to J-Link via USB...O.K.
Firmware: J-Link V9 compiled May  7 2021 16:26:12
Hardware version: V9.40
S/N: 59424175
License(s): RDI, GDB, FlashDL, FlashBP, JFlash
VTref=3.330V
J-Link>h
```

9、如果“h”之前没有“ERROR”但“h”回车之后出现了“CPU could not be halted”，需要输入“r”+回车，直到出现重置成功提示。

```
J-Link Commander V7.22
J-Link>h
CPU could not be halted
J-Link>r
Reset delay: 0 ms
Reset type NORMAL: Resets core & peripherals via SYSRESETREQ & VECTRESET bit.
Reset: Halt core after reset via DEMCR.VC_CORERESET.
Reset: Reset device via AIRCR.SYSRESETREQ.
J-Link>
```

10、此时输入“connect”，如果出现第 7 步出现的结果，证明已经解除保护成功。

2.3.2 backdoor 解除保护方式

方法：函数 FLASH_QueryKey 用于查询 backdoor 密码是否正确，此处的密码应与 bockdoor 配置保护时预先在 0x02000004~0x0200000C 地址上写的数值相同。如果输入的密码正确就会开启临时 DAP 访问。

FLASH_QueryKey()

void FLASH_QueryKey (uint8_t key[12])

Query if the key entered is correct. If it is correct, exit secure mode.

Parameters

[in] **key** it points to the 96-bit key to be entered.

Returns

none

具体调用方法如下：

```
FLASH_QueryKey((uint8_t *)key);
```

可以使用串口或按键执行此段程序，附录 B 中的例程使用了串口执行命令，详见附录 B。

操作步骤：（以附录 B 为例）

1、烧写附录 B 中的程序至 Z20K14xM demo 板，连接 demo 板上的串口至 PC，调整串口工具的 COM 口和波特率，例程中使用 COM5，波特率 9600bps 通讯。

2、启动程序会打印出以下信息，表明芯片进入主程序循环。此时 Flash 上的数值为默认的 0xFFFFFFFF。

```
[2022-10-24 11:02:30.002]
RX: FLASH_FSTAT value is 0x80
0x2000000 value is 0xffffffff
0x2000004 value is 0xffffffff
0x2000008 value is 0xffffffff
0x200000C value is 0xffffffff
```

3、例程中串口接收到“W”,执行在 Flash 的 0x02000000 上写入 0xF5EC 及自定义的 backdoor 密码值。通过串口工具发送“W”，执行带 backdoor 的 Flash 保护功能。输入“R”重启芯片，观察 0x02000000 ~ 0x0200000C 地址上的数值。此时数值变为预设值，且无法通过 J-Link 工具连上芯片，证明芯片已经正确配置 Flash 保护。

```
[2022-10-24 11:02:34.301]
TX: W

[2022-10-24 11:02:34.607]
RX: Encrypt chip with backdoor key!!!!!!!!!!!!
0x2000000 value is 0xfffff5ec
0x2000004 value is 0x26332556
0x2000008 value is 0x2122504
0x200000C value is 0x29865d4

input error cmd

[2022-10-24 11:02:38.820]
TX: R

[2022-10-24 11:02:39.136]
RX:
reset mcu
FLASH_FSTAT value is 0x80000080
0x2000000 value is 0xfffff5ec
0x2000004 value is 0x26332556
0x2000008 value is 0x2122504
0x200000C value is 0x29865d4
```

```
InitTarget() start
*************************************************************
J-Link script: Z20K148M M4F core J-Link script
*************************************************************
InitTarget() end
Found SW-DP with ID 0x2BA01477
DPIDR: 0x2BA01477
Scanning AP map to find all available APs
AP[2]: Stopped AP scan as end of AP map has been reached
AP[0]: AHB-AP (IDR: 0x24770011)
AP[1]: JTAG-AP (IDR: 0x0C860000)
Iterating through AP map to find AHB-AP to use
AP[0]: Skipped. Could not read CPUID register
AP[1]: Skipped. Not an AHB-AP
InitTarget() start
*************************************************************
J-Link script: Z20K148M M4F core J-Link script
*************************************************************
InitTarget() end
Found SW-DP with ID 0x2BA01477
DPIDR: 0x2BA01477
Scanning AP map to find all available APs
AP[2]: Stopped AP scan as end of AP map has been reached
AP[0]: AHB-AP (IDR: 0x24770011)
AP[1]: JTAG-AP (IDR: 0x0C860000)
Iterating through AP map to find AHB-AP to use
AP[0]: Skipped. Could not read CPUID register
AP[1]: Skipped. Not an AHB-AP

****** Error: Could not find core in Coresight setup

Cannot connect to target.
```

4、例程中“B”表示通过密码值开启 backdoor。通过串口发送“B”，允许临时的 DAP 访问。

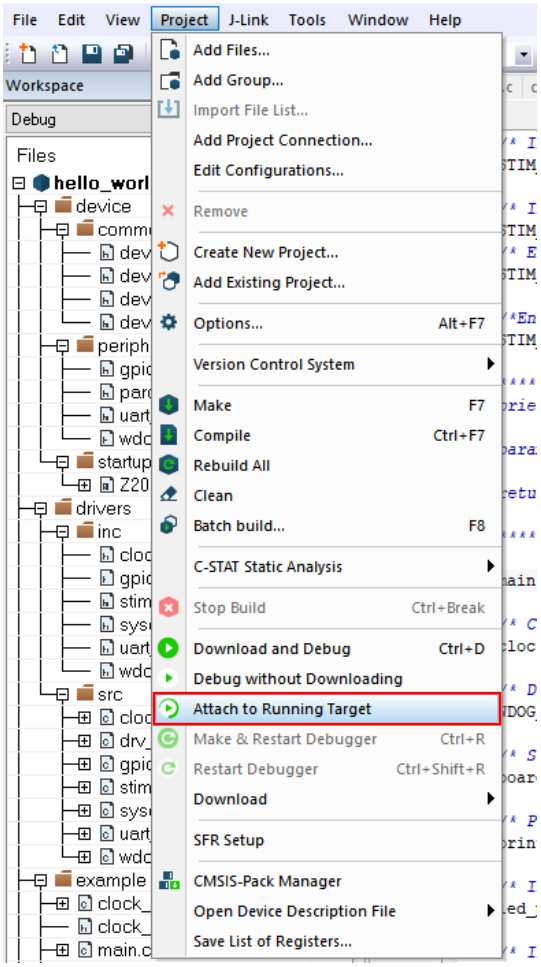
```
[2022-10-24 11:04:20.910]
TX: B

[2022-10-24 11:04:21.173]
RX: open back door
FLASH_FSTAT value is 0x880
Connect M4 chip by Jlink Command to check decrypt test result!!!!!!!!!

input error cmd
```

通过J-Link Commander发现芯片可以正确识别并连接，表明backdoor开启成功。

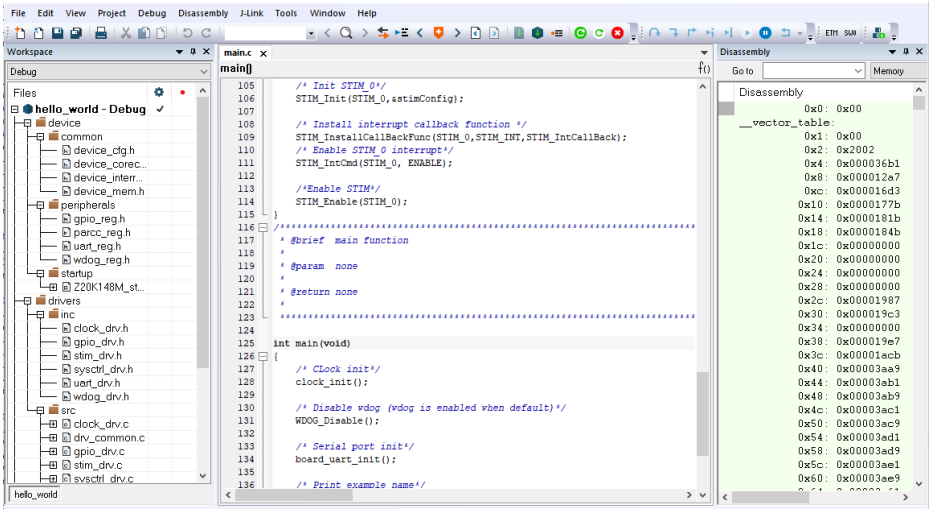
```
Device "Z20K148M" selected.


Connecting to target via JTAG
InitTarget() start
*****************************************************
J-Link script: Z20K148M M4F core J-Link script
*****************************************************
InitTarget() end
TotalIRLen = 4, IRPrint = 0x01
JTAG chain detection found 1 devices:
 #0 Id: 0x4BA00477, IRLen: 04, CoreSight JTAG-DP
DPv0 detected
Scanning AP map to find all available APs
AP[2]: Stopped AP scan as end of AP map has been reached
AP[0]: AHB-AP (IDR: 0x24770011)
AP[1]: JTAG-AP (IDR: 0x0C860000)
Iterating through AP map to find AHB-AP to use
AP[0]: Core found
AP[0]: AHB-AP ROM base: 0xE00FF000
CPUID register: 0x410FC241. Implementer code: 0x41 (ARM)
Found Cortex-M4 r0p1, Little endian.
FPUnit: 6 code (BP) slots and 2 literal slots
CoreSight components:
ROMTbl[0] @ E00FF000
ROMTbl[0][0]: E000E000, CID: B105E00D, PID: 000BB00C SCS-M7
ROMTbl[0][1]: E0001000, CID: B105E00D, PID: 003BB002 DWT
ROMTbl[0][2]: E0002000, CID: B105E00D, PID: 002BB003 FPB
ROMTbl[0][3]: E0000000, CID: B105E00D, PID: 003BB001 ITM
ROMTbl[0][4]: E0040000, CID: B105900D, PID: 000BB9A1 TPIU
ROMTbl[0][5]: E0041000, CID: B105900D, PID: 000BB925 ETM
SetupTarget() start
Init SRAM Begin
Init SRAM End
SetupTarget() end
Cortex-M4 identified.
```

除了通过 J-Link 工具验证 backdoor 之外，还可以通过 IAR 软件提供的 Attach to running target 功能进行 backdoor 验证和代码调试。

注意：选择 Project->Attach to Running Target 功能调试程序，如下图所示。如果选择 Download and Debug 或者 Debug without Downloading 会导致芯片重启，进而导致 backdoor 重新关闭。



进入 Debug 界面如下，证明 backdoor 开启成功。



5、在串口工具中发送“R”，复位 MCU。此时 Flash 的 0x02000000 地址上数值为 0xF5EC，MCU 处于 Flash 保护状态，无法通过 J-Link 工具连接 MCU。可以通过 J-Link Commander 验证。

```
[2022-10-24 11:05:57.243]
TX: R

[2022-10-24 11:05:57.567]
RX:
reset mcu
FLASH_FSTAT value is 0x80000080
0x2000000 value is 0xfffff5ec
0x2000004 value is 0x26332556
0x2000008 value is 0x2122504
0x200000C value is 0x29865d4
```

6、发送“C”，擦除 Flash 地址 0x02000000 的值，使 MCU 恢复默认值。

```
[2022-10-24 11:07:14.123]
TX: C

[2022-10-24 11:07:14.462]
RX: Erase 0x2000000 sector to remove the encryption status!!!!!!!!!
0x2000000 value is 0xffffffff
0x2000004 value is 0xffffffff
0x2000008 value is 0xffffffff
0x200000C value is 0xffffffff
```

此时 MCU 完全解除 Flash 保护，可以通过 IAR 正常烧录程序。

附录 A

解除保护脚本 unlock_cmd.txt

```
usb

si 1

swdwritedp 2 01000000

swdreadap  0

swdreadap  0

swdreadap  1

swdreadap  1

swdwriteap 1 00000018

swdwriteap 1 00000011

swdwriteap 1 00000010

sleep 1000

swdreadap  0

swdreadap  0

swdreadap  1

swdreadap  1

swdwriteap 1 00000008

swdwriteap 1 00000000

swdreadap  0

swdreadap  0

swdreadap  1

swdreadap  1

swdwritedp 2 00000000

mem32 00000000 10

usb

h
```

附录 B

带 backdoor 的 Flash 保护配置程序

```
    uint32_t key[4U];
    uint32_t uData[4U];
    uint8_t cmd;

    WDOG_Disable();
    clock_init();
    board_uart_init();

    key[0U] = 0x26332557U;
    key[1U] = 0x02122504U;
    key[2U] = 0x029865d4U;

    while (1)
    {
        cmd = getchar();
        switch (cmd)
        {
           case 'B':
                FLASH_QueryKey((uint8_t *)key);
                break;

           case 'W':
                uData[0] = 0xFFFFF5ECU;
                uData[1] = 0x26332557U;
                uData[2] = 0x02122504U;
                uData[3] = 0x029865d4U;
                FLASH_ProgramPhrase(0x02000000U, (uint8_t *)uData,
&FLASHTEST_CmdExeConfig);
                break;

            case 'R':
                NVIC_SystemReset();
                break;

            case 'C':
                FLASH_EraseSector(0x02000000U, &FLASHTEST_CmdExeConfig);
                break;
```

自定义密码值

串口接收

开启 backdoor，允许一次临时 DAP 访问

带 backdoor 保护

复位

擦除 Flash 相应地址上的值，解除保护

```
            default:
                print_log("\r\ninput error cmd\r\n");
        }
    }
```

How to Reach Us:

Home Page:

zhixin-semi.com/

Web Support:

Document Number: Z20K14xM-AN

Rev. 1.0, Nov., 2022