AN0042
应用笔记

AT32 SPIM Application Note

前言

这篇应用笔记描述了怎么使用AT32 MCU的SPIM作为外部存储器的扩展功能。

支持型号列表：

支持型号	AT32F403 系列 AT32F403A 系列 AT32F407 系列 AT32F413 系列 AT32A403A 系列

目录

表目录

图目录

1 概述

SPIM (External SPI FLASH memory interface)最大地址段为0x08400000 - 0x093FFFFF(16MB)，是AT32 MCU独有的一种Flash访问方式。用户可以使用自己的Flash作为AT32 MCU的外挂Flash。该方式有别于片上Bank1/Bank2，用户可根据具体需求选择是否开启。开启SPIM后可以作为Flash扩展实现如下功能：

- 在SPIM地址存放用户执行程序，类似于Bank1/Bank2一样执行程序。
- 用户程序直接访问SPIM地址，作为存储字体库，图片等存储器使用。

2 SPIM 配置

SPIM仅允许按字(32bit)或者半字(16bit)操作，在执行读、编程、擦除SPIM前，必须首先执行初始化及解锁操作。

SPIM在AT32不同系列产品及不同封装上，可能使用的pin有所不同，具体请参考对应型号产品的RM和DS，下面以AT32F403A系列为例，描述操作步骤

2.1 初始化及解锁操作

初始化及解锁的步骤，在AT32的BSP中已经封装成库函数，用户可以直接调用

1) 使能GPIOA、GPIOB和IOMUX CRM时钟。
2) 配置对应pin PA8、PA11、PA12、PB1、PB6、PB7为推挽复用输出模式。
3) IOMUX_REMAP2寄存器中使能SPIM接口。
4) 设置FLASH_SELECT寄存器选择配置SPIM Flash的类型。
5) 使用KEY解锁SPIM：写FLASH_UNLOCK3寄存器2次，按顺序分别写0x45670123和0xCDEF89AB。
6) 检查SPIM是否解锁成功，读取FLASH_CTRL3寄存器的OPLK位，如果被清除为0，则可以开始操作SPIM。

2.2 Flash 型号选择

SPIM可以配置支持不同型号的spi Flash，支持的指令集如下表，更详细描述可以看参考手册FlASH章节。

表 1. SPIM 支持的指令集

指令名称	指令码	FLASH_SELECT 寄存器配置	补充说明
Write Enable	0x06	0x1/0x2	2 类型号闪存均需要支持 0x06 指令
Quad Page Program	0x32	0x1/0x2	2 类型号闪存均需要支持 0x32 指令
Sector Erase	0x20	0x1/0x2	2 类型号闪存均需要支持 0x20 指令
Chip Erase	0xC7	0x1/0x2	2 类型号闪存均需要支持 0xC7 指令
Read Status Register	0x05	0x1/0x2	2 类型号闪存均需要支持 0x05 指令
Quad I/O Read	0xEB	0x1/0x2	2 类型号闪存均需要支持 0xEB 指令 24bit Addr + 6 个 Dummy cycle
Volatile status Register write enable	0x50	0x1	3 条指令用于当选择型号 1 闪存时，硬体自动发送指令配置闪存 Status Register 中的 Quad Enable（QE 位） 型号 1 闪存需要支持： 0x50 与 0x01 或是支持 0x50 与 0x31
Write Status Register-1	0x01		
Write Status Register-2	0x31		

2.3 读操作

直接按字(32bit)或者半字(16bit)访问需要读取数据的地址段：0x08400000 – 0x093FFFFF

2.4 编程操作

编程操作步骤，在AT32的BSP中已经封装成库函数，用户可以直接调用

1) 打开编程操作，FLASH_CTRL3寄存器FPRGM位置1
2) 直接按字(32bit)或者半字(16bit)在需要编程的地址写入数据
3) 检查是否写入完成读取FLASH_STS3寄存器的OBF位是否清除，如果清除表示写入完成
4) 关闭编程操作，FLASH_CTRL3寄存器FPRGM位置0
5) 检查是否写入成功，读取FLASH_STS3寄存器的PRGMERR和EPPERR位，如果都为0则表示写入成功

2.5 擦除操作

SPIM擦除分为Mass Erase和Sector Erase，每个sector固定为4KB，在AT32的BSP中已经封装成库函数，用户可以直接调用

Mass Erase

1) 使能擦除，FLASH_CTRL3寄存器CHPERS位置1
2) 开始擦除，FLASH_CTRL3寄存器ERSTR位置1
3) 关闭擦除，FLASH_CTRL3寄存器CHPERS位置0

Sector Erase

1) 使能擦除，FLASH_CTRL3寄存器SECERS位置1
2) 选择擦除扇区地址，FLASH_ADDR3寄存器写入需要擦除扇区的地址
3) 开始擦除，FLASH_CTRL3寄存器ERSTR位置1
4) 关闭擦除，FLASH_CTRL3寄存器SECERS位置0

2.6 加密操作

因为SPIM电路裸露在MCU芯片外部，为防止存储在SPIM Flash里边的数据被外界直接读取，SPIM提供了加密功能，将原始数据通过特有算法进行加密操作后再写入Flash，AT32 MCU读取SPIM数据时会先进行解密得到原始数据，然后才使用，保证数据安全。加密算法所用的SPIM scrambled KEY为用户系统数据区的地址0x1FFFF820-0x1FFFF82F范围内数据。

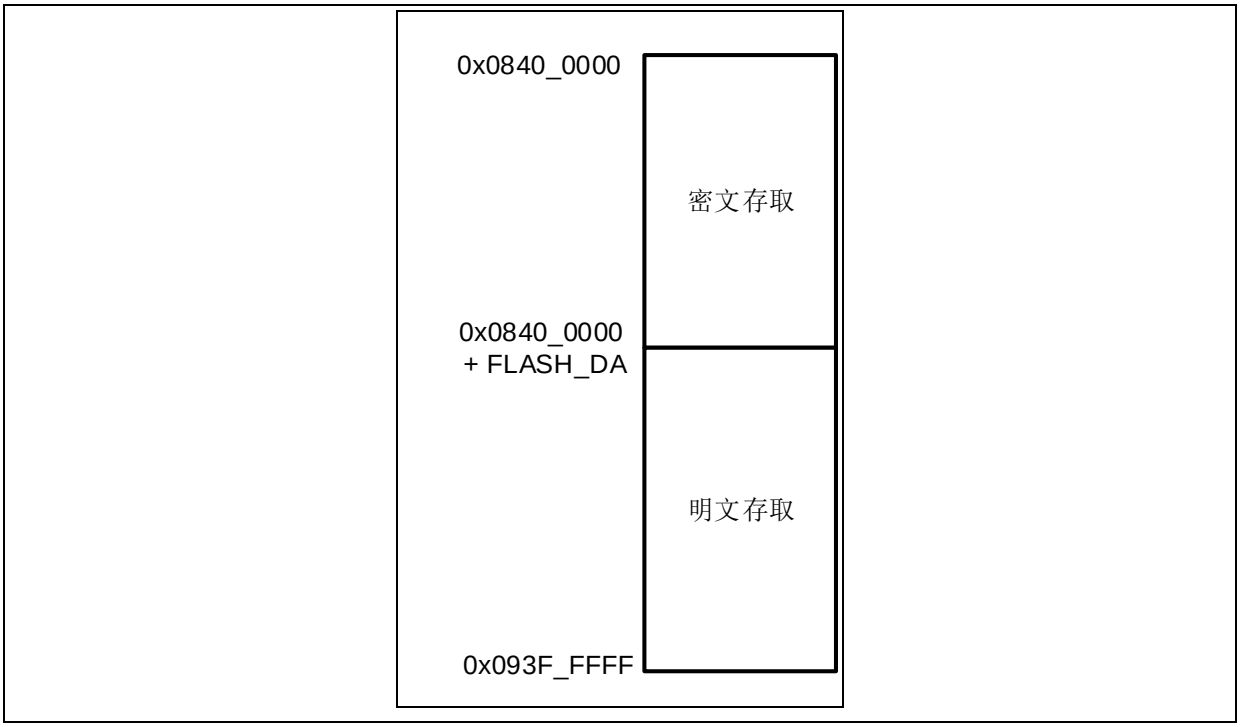
- 当SPIM scrambled KEY值全为0xFFFFFFFF时，加密功能关闭。
- 当SPIM scrambled KEY值不全为0xFFFFFFFF时，加密功能开启。AT32 MCU根据FLASH_DA寄存器值作为加密范围分界，小于0x08400000+FLASH_DA地址段的数据为密文，其余范围数据仍然采用明文存储。

注意：数据写入时的加密状态必须和读取时的加密状态保持一致，否则可能导致读取的数据是乱码而无法正确使用或者运行。即写入时如果配置了 SPIM scrambled KEY 及 FLASH_DA，那么在读取时也必须配置相同的 SPIM scrambled KEY 及 FLASH_DA。

图 1. SPIM scrambled KEY 存放地址

0x1FFF_F820	nBANK3KEY1	BANK3KEY1	nBANK3KEY0	BANK3KEY0
0x1FFF_F824	nBANK3KEY3	BANK3KEY3	nBANK3KEY2	BANK3KEY2
0x1FFF_F828	nBANK3KEY5	BANK3KEY5	nBANK3KEY4	BANK3KEY4
0x1FFF_F82C	nBANK3KEY7	BANK3KEY7	nBANK3KEY6	BANK3KEY6

图 2. SPIM 地址范围

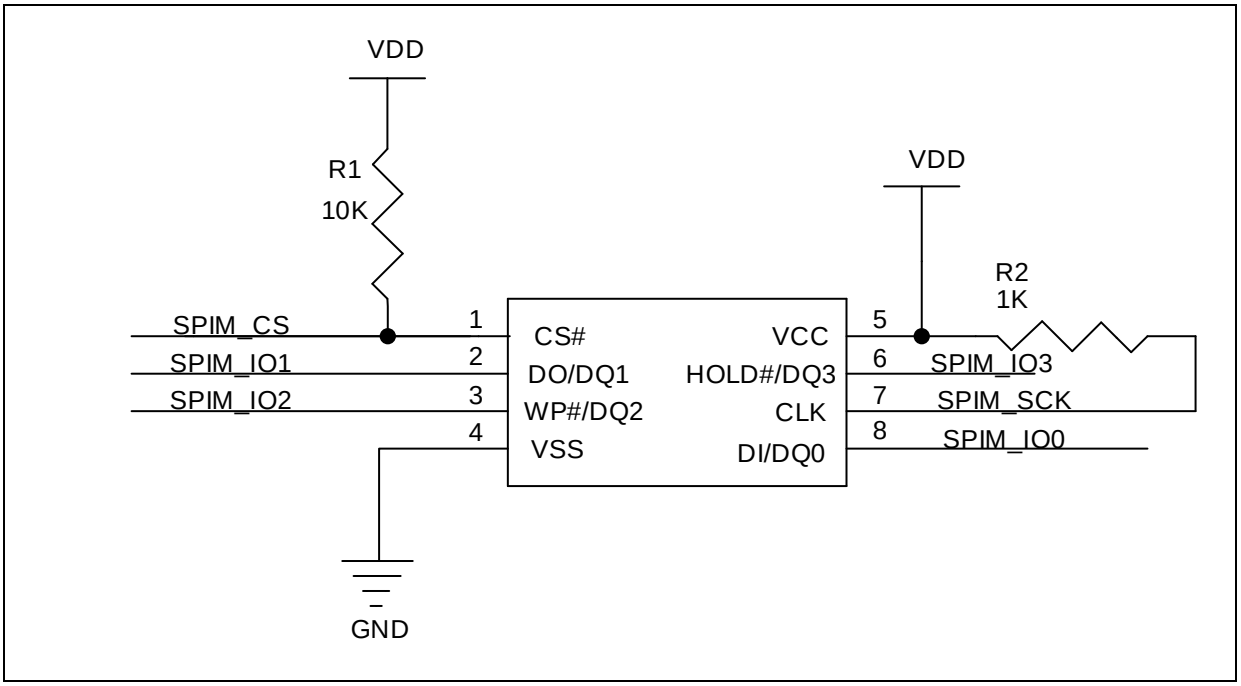


2.7 硬件电路

因为SPIM连接在外部电路，受环境影响较大，为保证电路稳定，需尽量减少PCB布线长度。

注意：SPIM 运行频率为 MCU 的 AHB 时钟频率的 1/2，所以当开启 SPIM 时，MCU 对应的运行 AHB 频率有最大限制值，不同型号 MCU 在 SPIM 开启时运行的最大频率值请用户参考 DS 的通用工作条件章节描述。

图 3. 参考电路



2.8 IO 复用

要使用SPIM，需要注意跟其他外设IP的IO复用问题。

当SPIM使用的IO还有对应的其他外设使能的话，有些IO即使其他外设应用中没有用到，但也可能会占用。

例如：XMC和SPIM同时使用时，在F403A上PB7用作SPIM的IO2，但是如果配置使能了XMC，即使XMC_NADV功能没有使用，则PB7也会被XMC_NADV默认开启占用，导致SPIM工作异常。此时需手动配置IOMUX_REMAP2寄存器关闭XMC_NADV功能，调用库函数即可，如下

gpio_pin_remap_config (XMC_NADV_MUX, TRUE)

图 4. XMC 和 SPIM 共用引脚

PB7	I2C1_SDA(7) / XMC_NADV / SPIM_IO2 / TMR4_CH2(7)	USART1_RX / SPI4_SCK / I2S4_CK

图 5. 单独关闭 XMC_NADV 功能

位 10	XMC_NADV_MUX	0x0	rw	XMC_NADV_MUX：XMC NADV 连接。 选择是否使用 XMC_NADV 信号。 0：XMC_NADV 连接到 pin。（默认） 1：XMC_NADV 不使用，对应的 pin 可被其他外设使用。

3 demo 示例

AT32的BSP中，有两个例程operate_spim和run_in_spim演示了如何使用SPIM。

3.1 用户程序访问 SPIM 存储区

如果通过用户程序去执行读、编程及擦除操作，只需要按照正常配置初始化及解锁后就可以实现。BSP中examples\flash\operate_spim执行了SPIM的初始化、擦除、编程和读取流程，并通过LED显示执行结果。

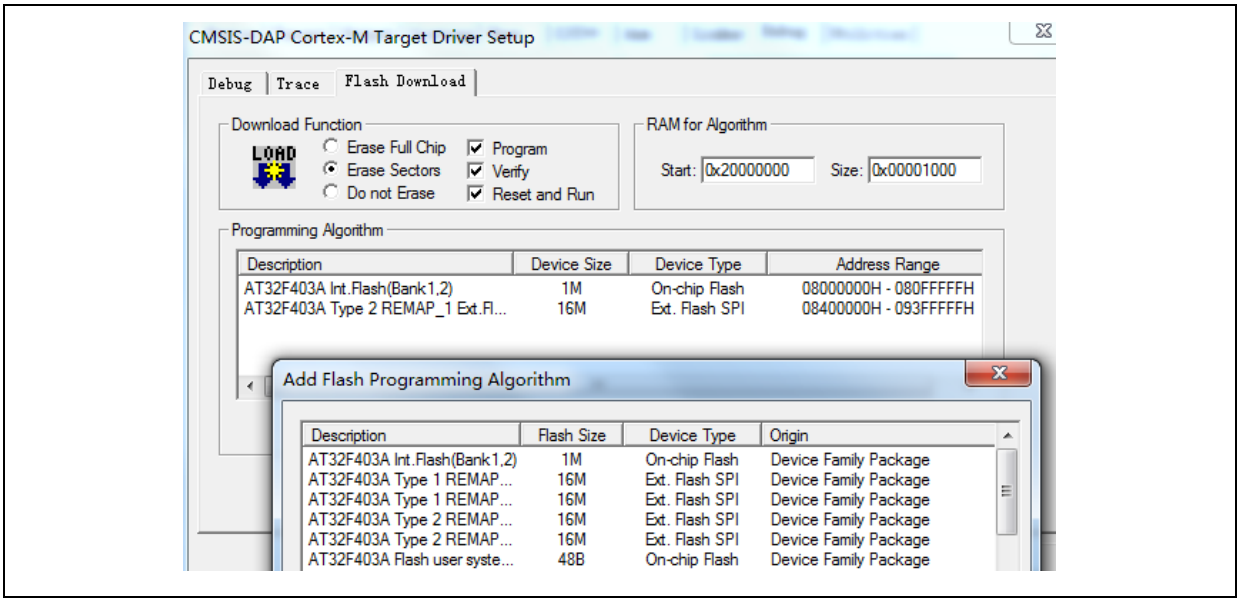
3.2 下载用户程序到 SPIM 或者用户程序在 SPIM 执行

如果用户程序通过keil下载到SPIM中，或者想程序直接在SPIM执行，需要执行一些额外的操作。

附件工程run_in_spim通过LED的闪烁，简单的演示了代码如何在SPIM上运行。

1) Options-Debug-Settings-Flash Download中选择添加外挂flash类型。

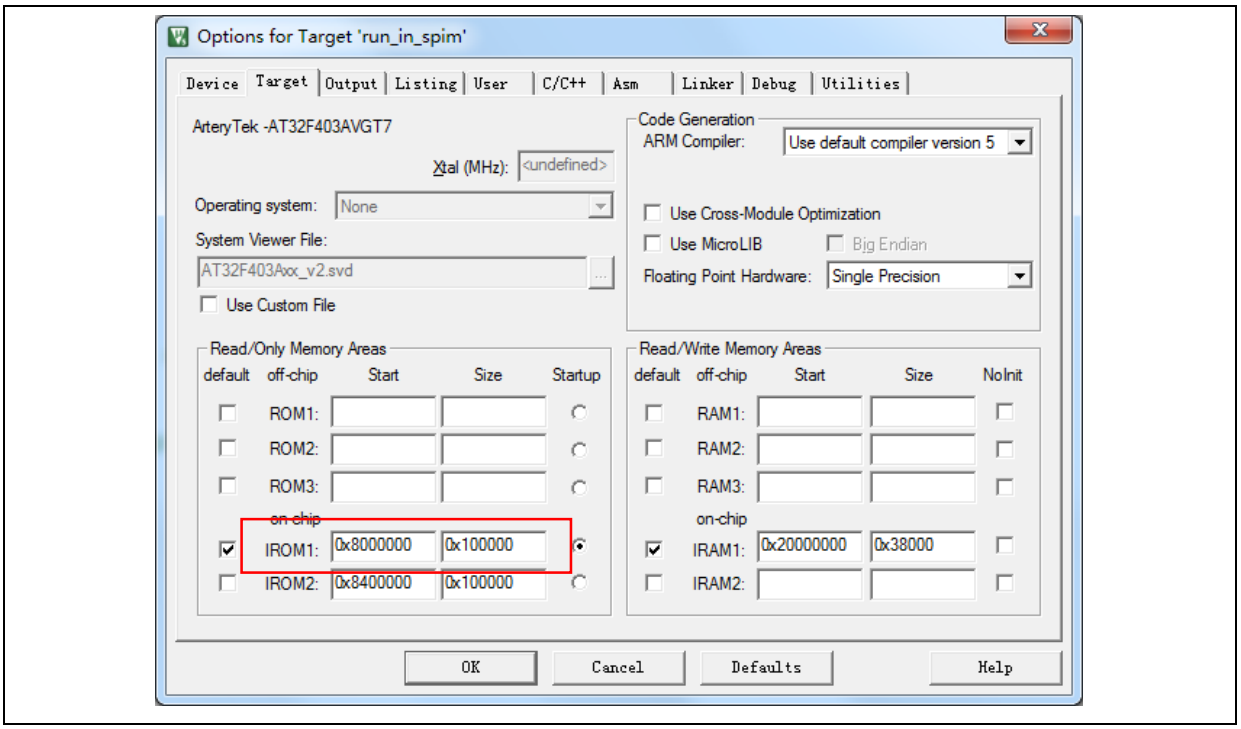
图 6. 配置 SPIM Flash 类型



2) Options-Target中添加SPIM起始地址和容量大小，但不勾选。Demo中将SPIM定义在IROM2位置，用于存储C文件

注意：如果勾选则 KEIL 在编译时可能会将其余不需要运行在 SPIM 的函数编译到 SPIM 地址段。

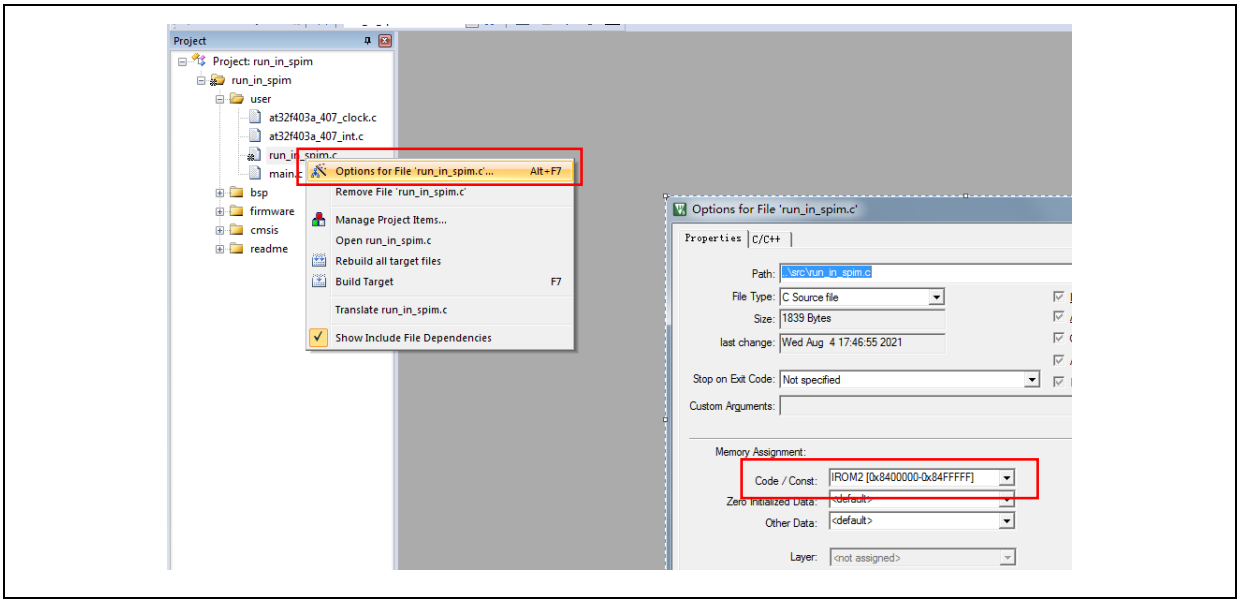
图 7. 配置 SPIM 起始地址和容量



3) Project中选择需要运行在SPIM对应位置的C文件，鼠标右键选Options进入将Memory Assignment的code地址改为对应地址段。

注意：如果工程中函数还有更多指定地址分段编译的需求，可以继续添加到对应 ROM1/2/3 等，也可以手动修改 sct 文件

图 8. 配置需运行在 SPIM 第一部分代码



4) 编译时勾选自动生成sct文件的选项，编译完成后浏览sct文件可以发现需要运行在SPIM区域的函数已被正确编译到该区域。

图 9 勾选自动生成 sct 文件

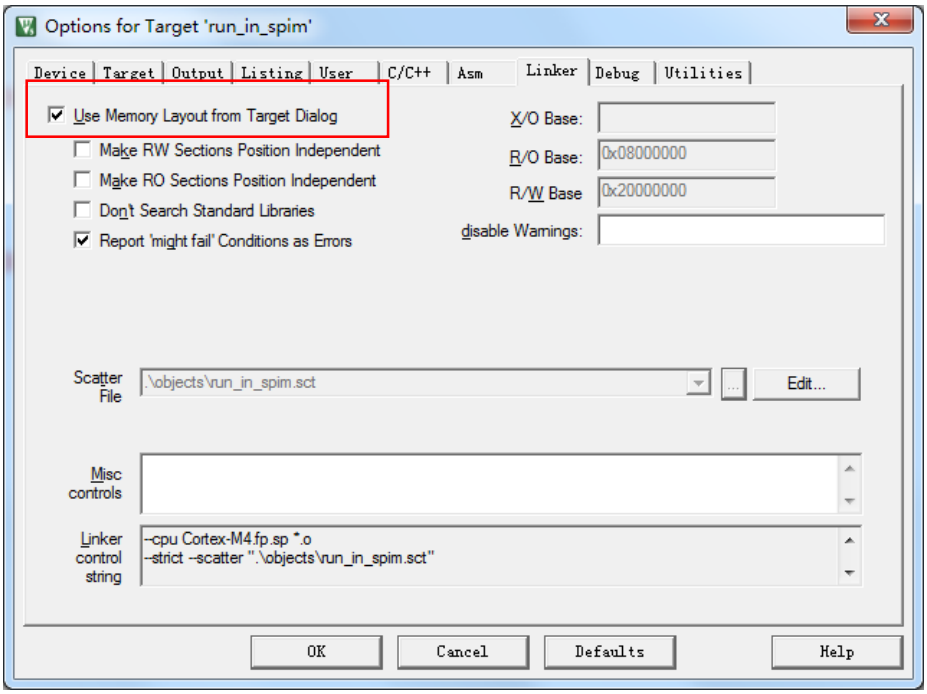


图 10 自动生成的 sct 文件

```
; *************************************************************
; *** Scatter-Loading Description File generated by uVision ***
; *************************************************************

LR_IROM1 0x08000000 0x00100000  {    ; load region size_region
  ER_IROM1 0x08000000 0x00100000  {  ; load address = execution address
   *.o (RESET, +First)
   *(InRoot$$Sections)
   .ANY (+RO)
   .ANY (+XO)
  }
  RW_IRAM1 0x20000000 0x00038000  {  ; RW data
   .ANY (+RW +ZI)
  }
}

LR_IROM2 0x08400000 0x00100000  {
  ER_IROM2 0x08400000 0x00100000  {  ; load address = execution address
    run_in_spim.o (+RO)
  }
}
```

注意：整个代码的启动必须从 bank1 开始，须保证 SPIM flash 的初始化代码在程序执行到 SPIM 前运行。

4 版本历史

表 2. 文档版本历史

日期	版本	变更
2022.01.19	2.0.0	最初版本

重要通知 – 请仔细阅读

买方自行负责对本文所述雅特力产品和服务的选择和使用，雅特力概不承担与选择或使用本文所述雅特力产品和服务相关的任何责任。

无论之前是否有过任何形式的表示，本文档不以任何方式对任何知识产权进行任何明示或默示的授权或许可。如果本文档任何部分涉及任何第三方产品或服务，不应被视为雅特力授权使用此类第三方产品或服务，或许可其中的任何知识产权，或者被视为涉及以任何方式使用任何此类第三方产品或服务或其中任何知识产权的保证。

除非在雅特力的销售条款中另有说明，否则，雅特力对雅特力产品的使用和/或销售不做任何明示或默示的保证，包括但不限于有关适销性、适合特定用途（及其依据任何司法管辖区的法律的对应情况），或侵犯任何专利、版权或其他知识产权的默示保证。

雅特力产品并非设计或专门用于下列用途的产品：（A）对安全性有特别要求的应用，例如：生命支持、主动植入设备或对产品功能安全有要求的系统；（B）航空应用；（C）航天应用或航天环境；（D）武器，且/或（E）其他可能导致人身伤害、死亡及财产损害的应用。如果采购商擅自将其用于前述应用，即使采购商向雅特力发出了书面通知，风险及法律责任仍将由采购商单独承担，且采购商应独力负责在前述应用中满足所有法律和法规要求。

经销的雅特力产品如有不同于本文档中提出的声明和/或技术特点的规定，将立即导致雅特力针对本文所述雅特力产品或服务授予的任何保证失效，并且不应以任何形式造成或扩大雅特力的任何责任。