1. **片内flash资源划分**

扇区0：0x8000000（16K）存放boot代码，boot最好放在片上flash

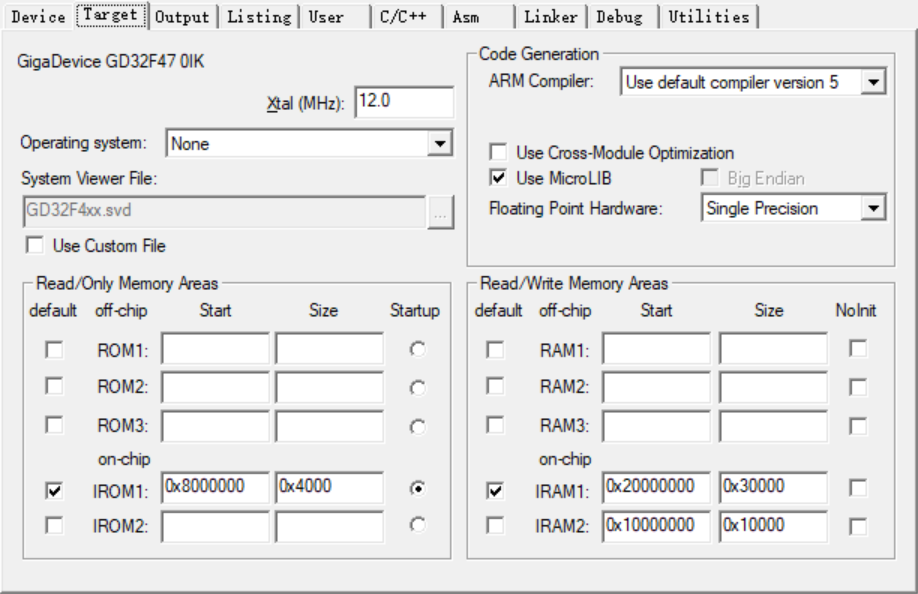
扇区n：0x8004000开始用于存放code，如果资源不够，可以存放到片外flash

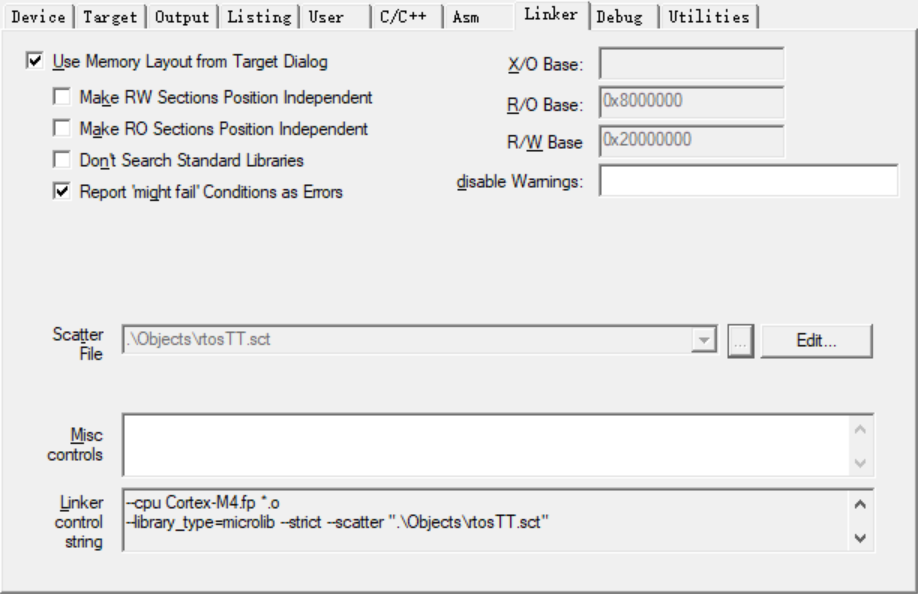
1. **bootloader工程**

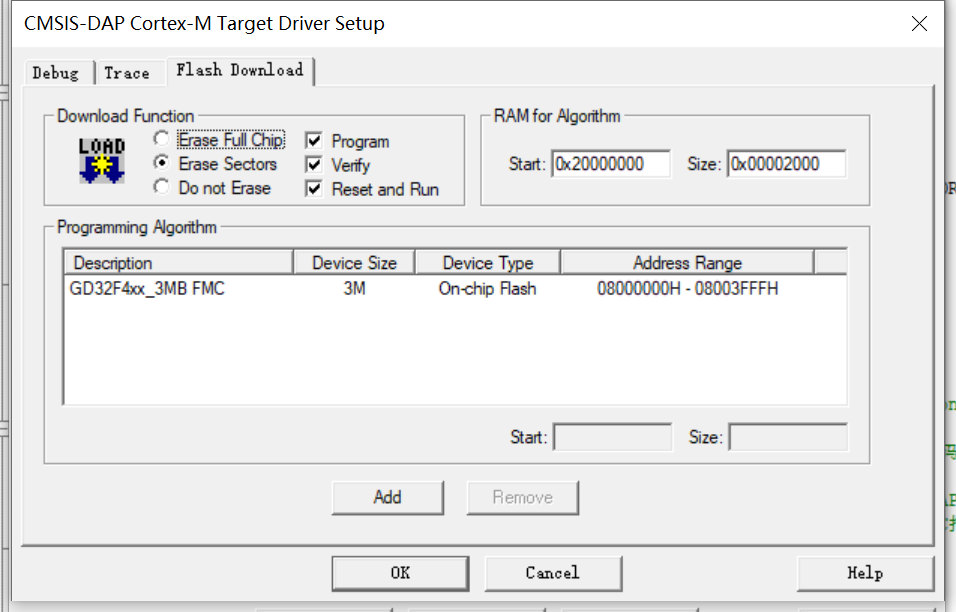
将代码下载到FLASH 之后，代码需要搬运到外部 SDRAM 才能运行，因此需要 新建 boot loder 工程，在此工程中需要完成FMC读 FLASH 和 EXMC 写 SDRAMD 驱动的初始化 ，从片上 FLASH 中将代码搬运到片外 SDRAM 位置，并将 SDRAM 地址 (0xC0000000)映射到 0x00000000 地址处以运行代码。



Keil ram和rom配置，和片上运行一样。

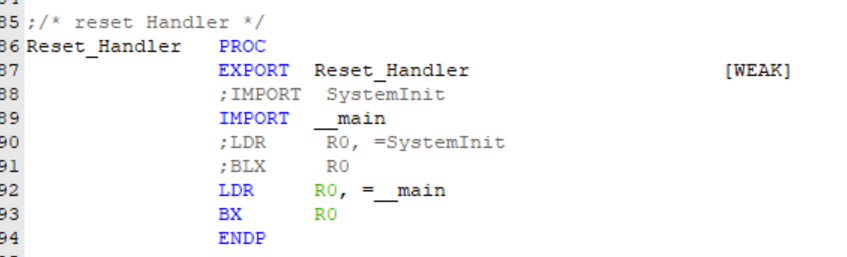




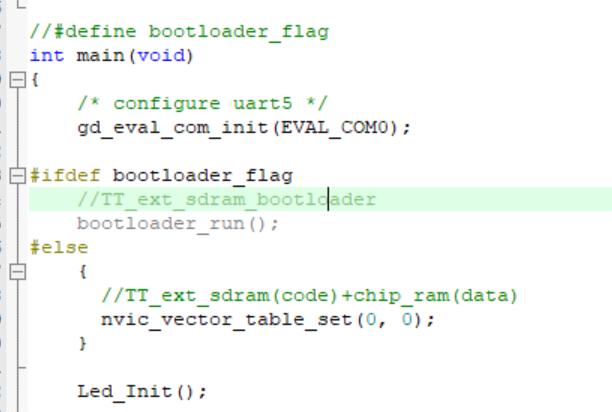


1. **APP业务工程**

启动文件修改，bootloader会完成初始化；

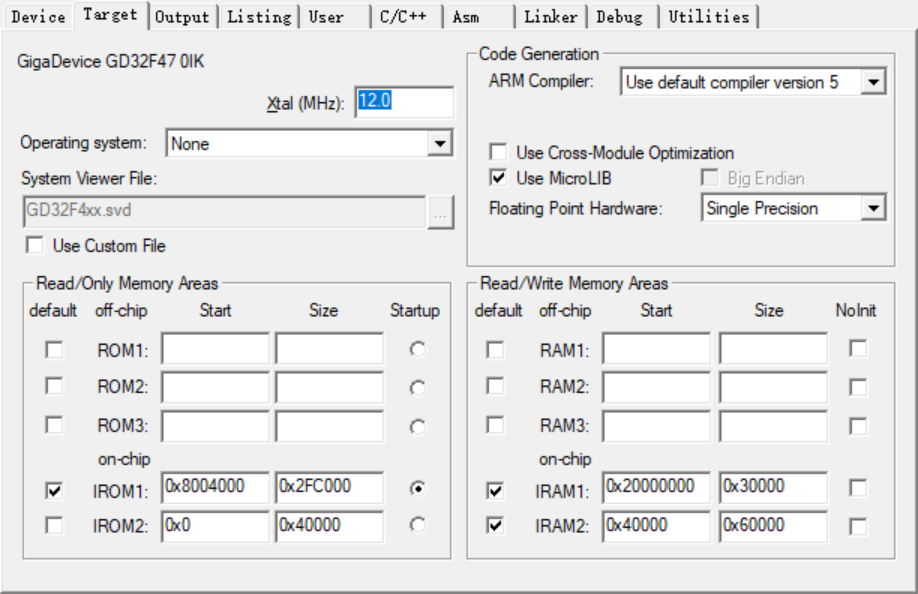


向量表重定位到0x00000000， nvic\_vector\_table\_set(0,0);

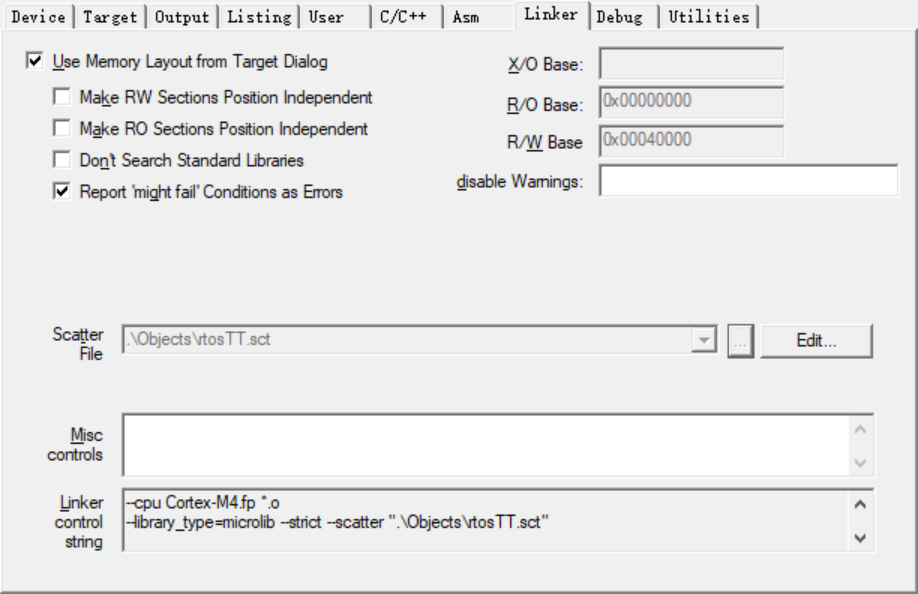


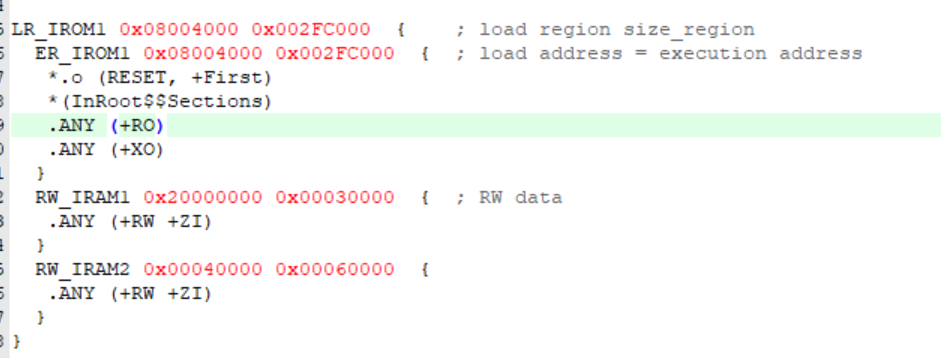
Keil ram和rom配置需要修改；IROM1用于存放code段，IRAM1用于运行keil下载算法，

IRAM2用于app ram

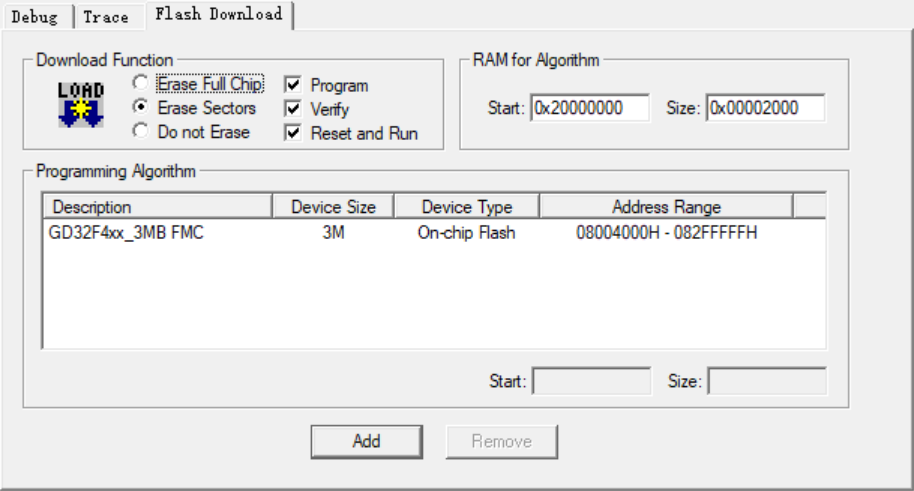


散列文件可以系统生成，也可以手动修改；



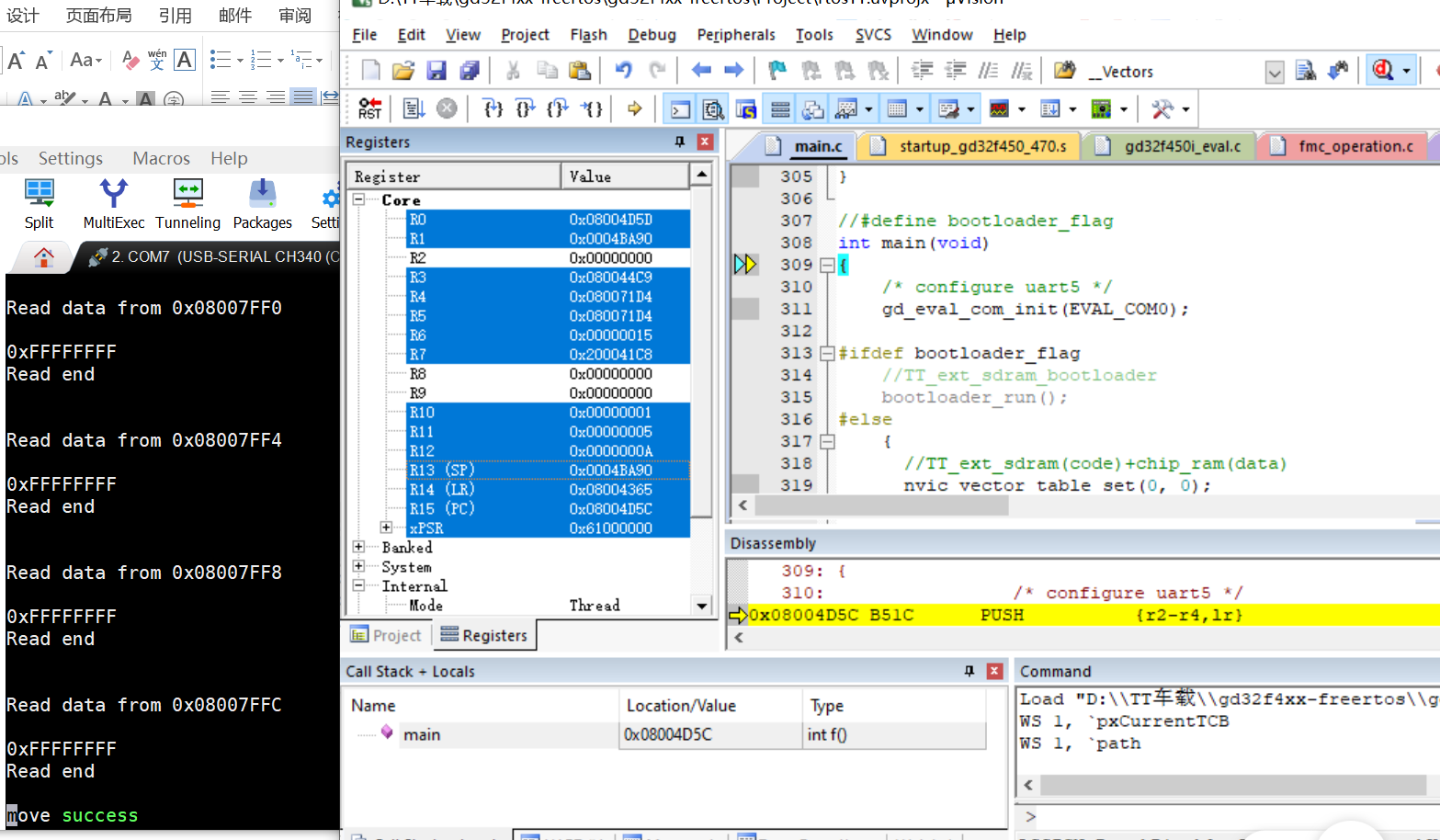


下载算法



1. **APP业务在线调试**

将BOOT工程烧写至片上，然后APP工程正常运行keil debug模式，待BOOT搬运完毕，会自动开始debug模式；



可以看到SP寄存器已经指向sdram上的\_\_initial\_sp