

概述

本文主要介绍了 SPC2168 的 CAU 调试方法。

SPIN TROL

目录

1	CAU 概述	6
1.1	CAU 代码执行过程.....	6
2	双核代码烧录	7
2.1	采用 ISP Download Tool 烧录镜像	7
2.1.1	CPU 工程配置	8
2.1.2	CAU 工程配置.....	8
2.1.3	CPU 搬运 CAU 镜像.....	9
2.1.4	烧录.....	10
2.2	采用 JLINK 烧录镜像.....	11
2.2.1	CPU 工程配置.....	11
2.2.2	CAU 工程配置.....	11
2.3	采用 JLINK 调试程序.....	13
2.3.1	CPU 调试.....	13
2.3.2	CAU 调试.....	13
2.4	JLINK 调试接口说明.....	16

图片列表

图 1-1: Mailbox 架构	6
图 1-2: Main CPU 与 CAU 框图	6
图 2-1: SPC2168 内存映射图	7
图 2-2: ISP Download Tool 烧录时 CPU 工程配置	8
图 2-3: ISP Download Tool 烧录时 CAU 工程配置	8
图 2-4: CPU 搬运程序	9
图 2-5: ISP Download Tool	10
图 2-6: CAU 工程添加 Flash 算法文件	11
图 2-7: CAU 工程烧录地址及执行地址配置	12
图 2-8: CAU 工程选择 ini 文件	15
图 2-9: ini 文件内容	15
图 2-10: Debug in RAM 下“Flash Download”配置	15

表格列表

表 2-1: JTAG 设置16

SPIN TROL

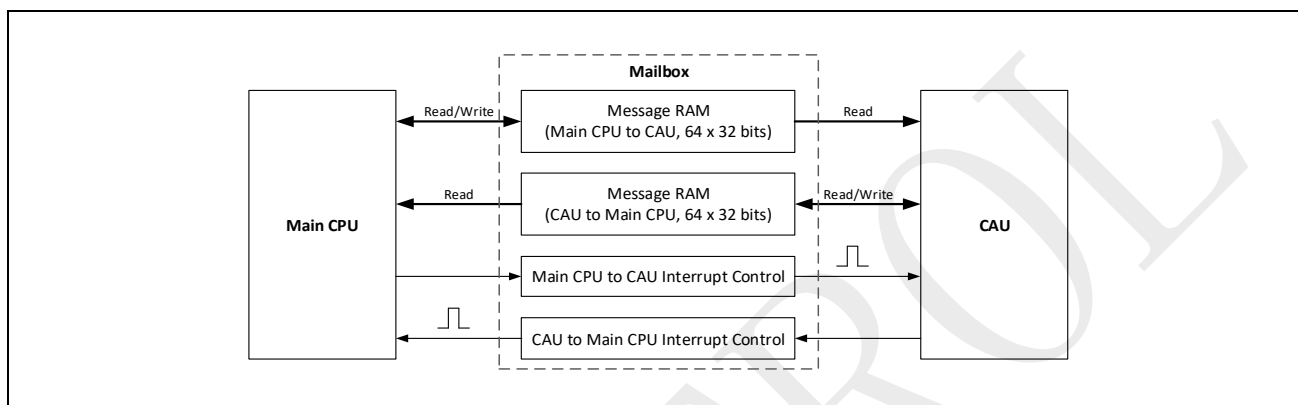
版本历史

版本	日期	作者	状态	变更
1	2022-01-04	-	Outdated	1. 初始版本。
C/0	2024-06-27	LemengZhou	Released	<ol style="list-style-type: none">1. 修改 2.2 章节为“采用 JLINK 烧录镜像”，并将本节所有 JTAG 相关修改为 JLINK，增加注意事项，以便说明烧录所需的具体 SWD 接口。2. 增加章节 2.3 采用 JLINK 调试程序。3. 原章节 2.3 变为章节 2.4，更名为“JLINK 调试接口说明”。

1 CAU 概述

SPC2168 是双核 CPU，除了主 CPU，另一颗基于 ARM CM4 的 CPU 被称为 CAU。CPU 和 CAU 之间通过 Mailbox 进行握手和共享数据，CPU 和 CAU 通过中断来相互请求，请求的消息内容通过 Message RAM 传递。Mailbox 架构和地址映射如图 1-1 所示。

图 1-1: Mailbox 架构

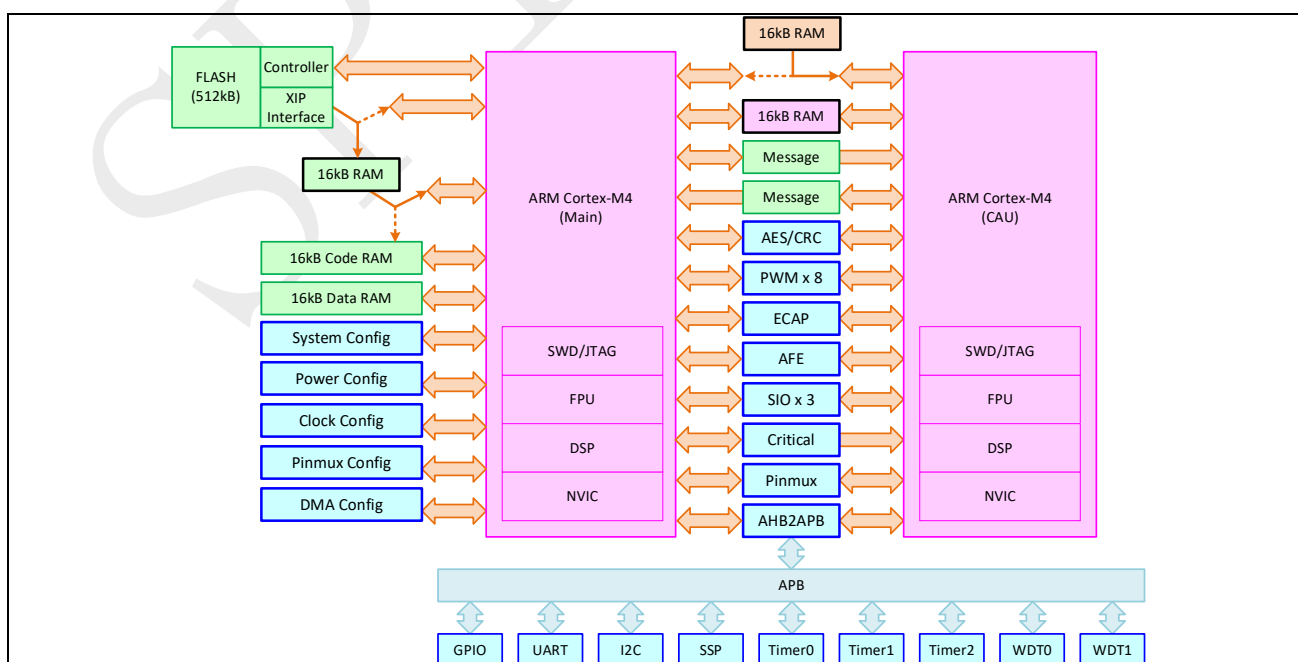


1.1 CAU 代码执行过程

如图 1-2 所示，对于存储区域而言，CAU 只能访问属于自身的一段 RAM 控件，并没有访问 Flash 的能力，所以 CAU 的代码在存储到 Flash 之后，只能依靠 CPU 将其代码搬运到 CAU 的 RAM 区域，然后有 CPU 配置 CAU 使能位，让 CAU 开始执行其代码。

本指南后续章节将对如何烧录 CAU 程序以及调试方法作详细说明。

图 1-2: Main CPU 与 CAU 框图



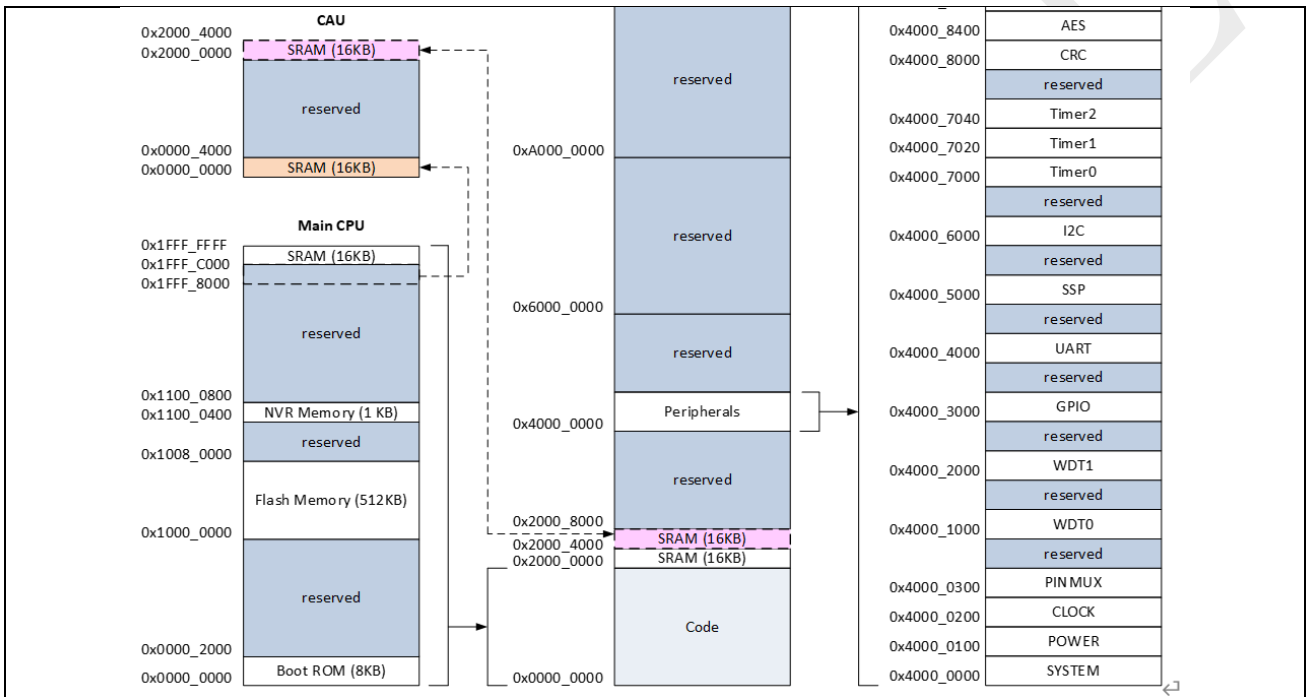
2 双核代码烧录

2.1 采用 ISP Download Tool 烧录镜像

ISP 工具是 Spintrol 开发的一款烧录工具，此工具的详细说明请参考《ISP 工具使用指南》。以下将对使用 ISP Download Tool 烧录程序时，CPU 以及 CAU 的 KEIL 工程配置情况进行说明。

范例采用 Demo Code 的 DualCore_CAU_ToggleGPIO 以及 DualCore_MainCPU_ToggleGPIO 两个示例程序，来说明 CAU 以及 CPU 的工程配置。为了方便后续说明，默认芯片的设置为使能了 Cache 以及 CAU，其内存映射如下图所示：

图 2-1: SPC2168 内存映射图

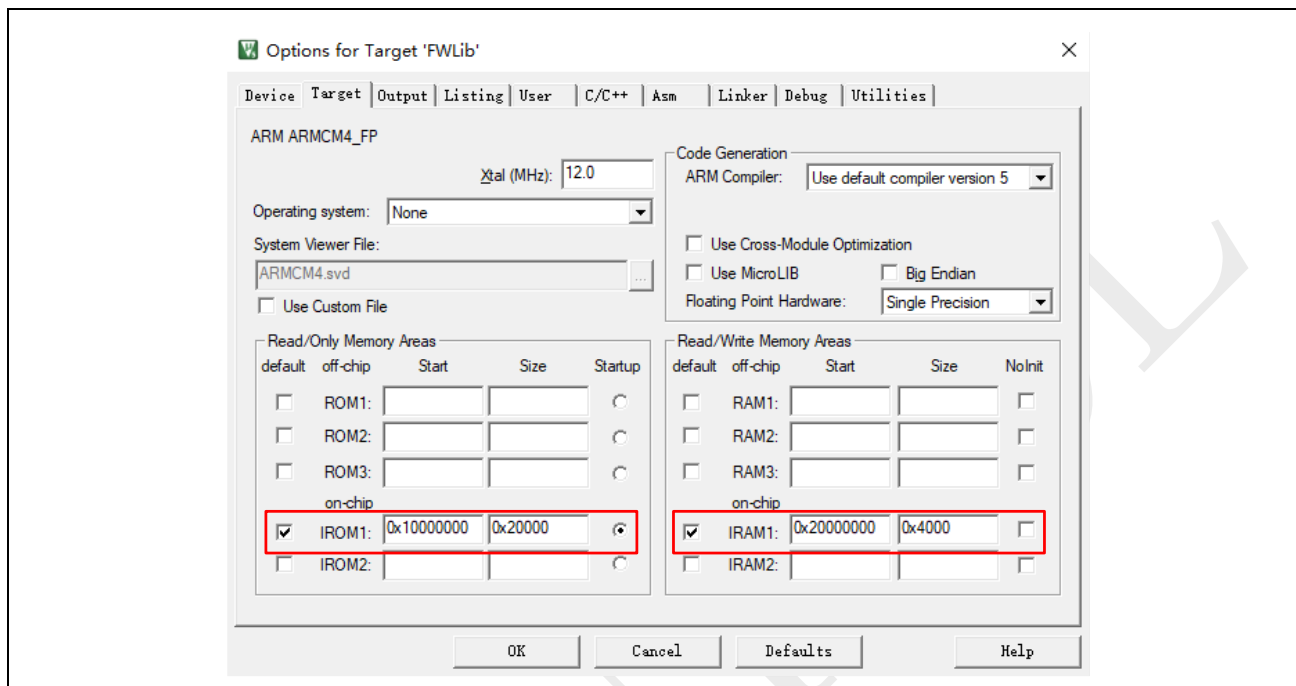


上图中，CPU 可以访问的实际物理 RAM 的地址范围为：0x1FFF8000~0x20000000（以 CPU 的视角看此地址），共 48K 大小，其中 0x1FFF8000~0x1FFFC000，大小为 16K 的地址区间与 CAU 共享；而 CAU 能够访问的实际物理 RAM 的地址范围为：0x00000000~0x4000、0x20000000~0x20004000（从 CAU 的视角看这两段 RAM 地址），共 32K 大小，其中 0x00000000~0x4000 这段 RAM 是与 CPU 共享的那段物理 RAM。

2.1.1 CPU 工程配置

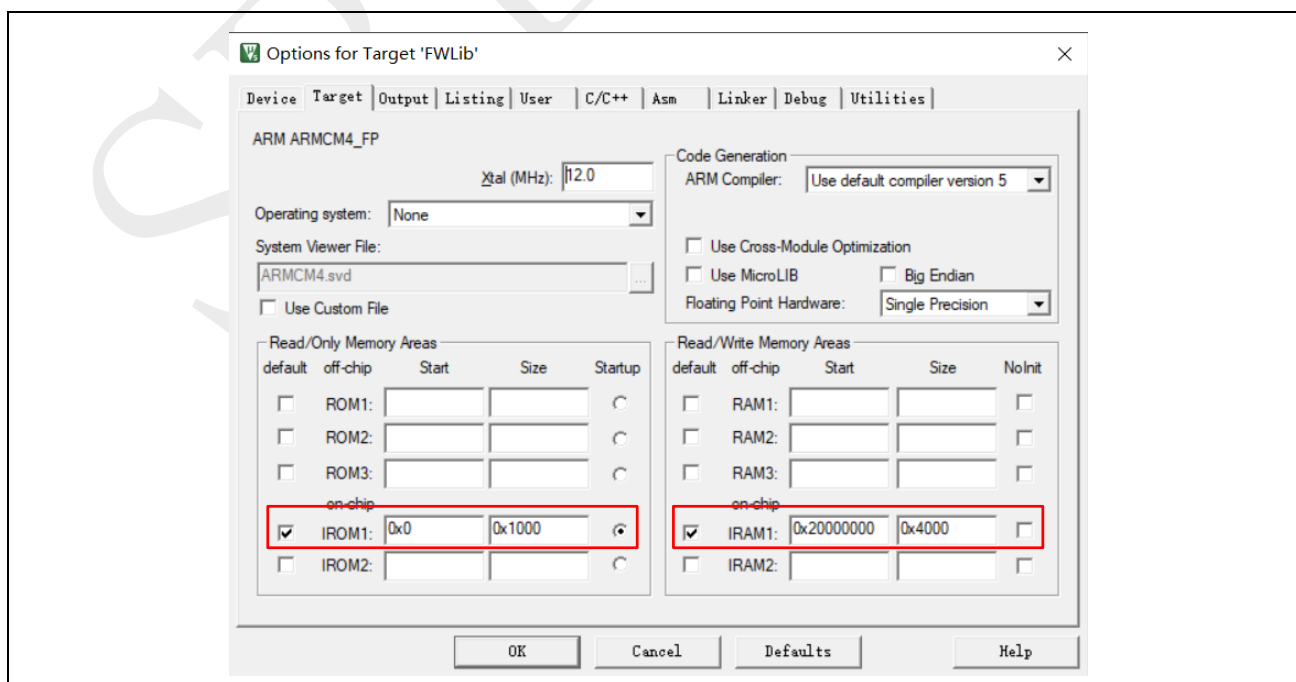
由下图可知，CPU 代码的加载域和执行域是以 0x10000000 开始的长度为 0x2000 的 Flash 区域，数据堆栈区域则为 0x20000000~0x20004000 的 DRAM 区域。

图 2-2: ISP Download Tool 烧录时 CPU 工程配置



2.1.2 CAU 工程配置

图 2-3: ISP Download Tool 烧录时 CAU 工程配置



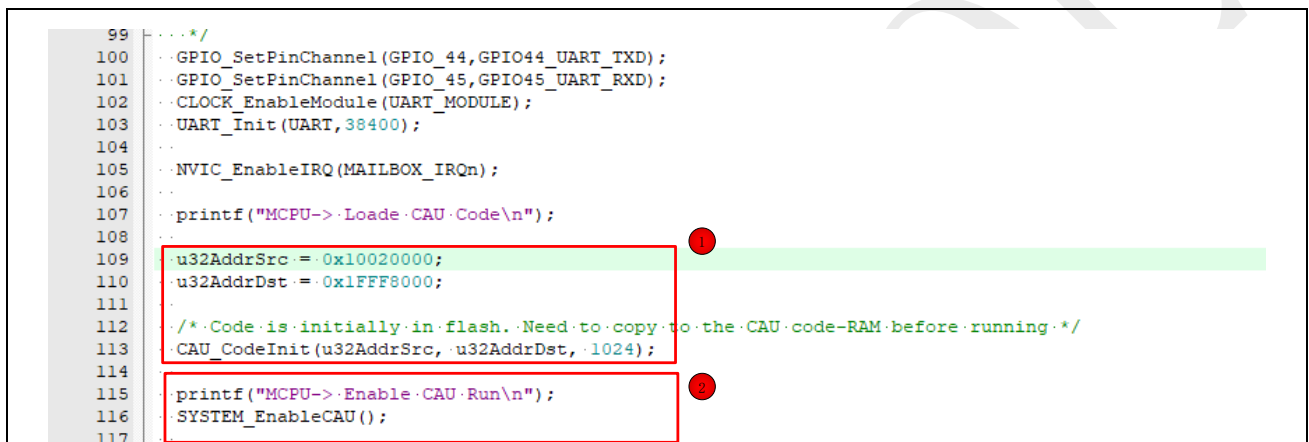
上图的配置表示：CAU 的执行域是以 CAU 的视角看到的 0x0~0x1000 的地址空间（设置的这个范围是以 CAU 的视角看到的地址范围，如果以 CPU 的视角看，则是 0x1FFF8000~0x1FFF9000），CAU 代码的加载域稍后再说明，数据堆栈区域则为 0x20000000~0x20004000。

2.1.3 CPU 搬运 CAU 镜像

如章节 1.1 描述，CAU 需要 CPU 将其代码搬运至属于自己的 RAM 控件中，在 DualCore_MainCPU_ToggleGPIO 示例程序中我们可以从代码中看到蛛丝马迹。如下图所示，● 搬运完代码之后，● 使能 CAU，此时 CAU 开始执行程序。

图 2-4: CPU 搬运程序

```
99  ...*/
100  .GPIO_SetPinChannel(GPIO_44,GPIO44_UART_TXD);
101  .GPIO_SetPinChannel(GPIO_45,GPIO45_UART_RXD);
102  .CLOCK_EnableModule(UART_MODULE);
103  .UART_Init(UART,38400);
104  .
105  .NVIC_EnableIRQ(MAILBOX_IRQn);
106  .
107  .printf("MCPU->.Loade.CAU.Code\n");
108  .
109  .u32AddrSrc = 0x10020000;
110  .u32AddrDst = 0x1FFF8000;
111  .
112  ./*Code is initially in flash. Need to copy to the CAU code-RAM before running.*/
113  .CAU_CodeInit(u32AddrSrc, u32AddrDst, 1024);
114  .
115  .printf("MCPU->.Enable.CAU.Run\n");
116  .SYSTEM_EnableCAU();
117  .
```



2.1.4 烧录

在 2.1.1 以及 2.1.2 已经介绍了 CPU 以及 CAU 的工程配置情况，在工程编译之后，就可以使用 ISP Download Tool 烧录镜像。




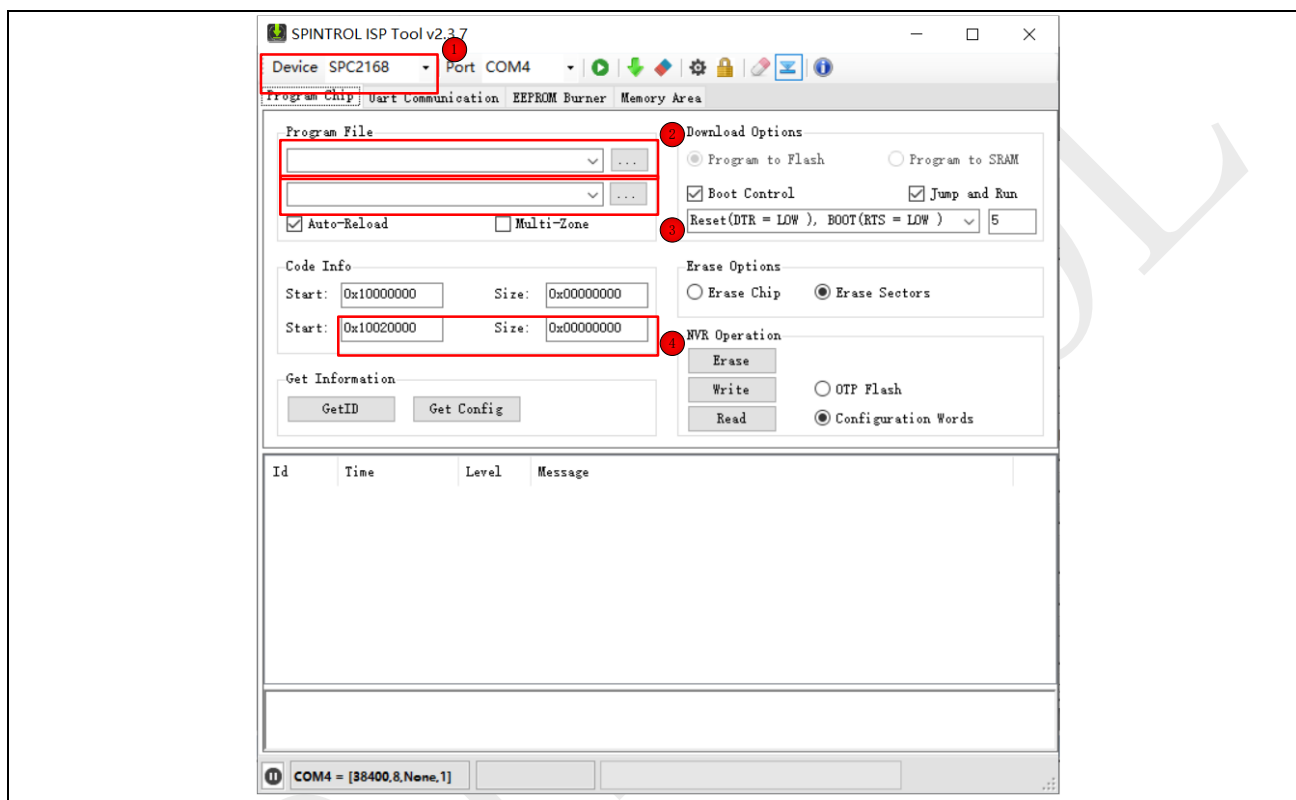
如图 2-5 所示，当选择 Decie 为 SPC2168 后，在  和  分别选择 CPU 和 CAU 工程编译之后的 hex 文件，然后在  手动更改属于 CAU 的 hex 文件的加载地址（CPU 的加载地址无需更改）。

图 2-5: ISP Download Tool



- 注意：
1. CPU 搬运代码起搬运的源地址是 2.1.4 节人为填写的 CAU 加载地址，而目的地址是从 CPU 角度看到的 CAU 0x0~0x4000 的地址。
 2. CPU 无需更改的原因是：在 CPU 的 KEIL 工程中已经设置了加载域和执行域，但是在 CAU 的工程中，设置的以 0x0 开始的地址在 ARM Cortex-M 体系结构中是不能作为加载域的，所以只能在 ISP Download Tool 中重新制定加载域。

至此，就可以点击运行按钮，使用 ISP Download Tool 烧录镜像。

2.2 采用 JLINK 烧录镜像

在开始介绍 JLINK 烧录镜像之前，建议用户首先阅读《快速上手指南》，了解 JLINK 的基本使用方法，本指南将着重介绍使用 JLINK 方式烧录的方法。

2.2.1 CPU 工程配置

对于 CPU 而言，使用 JLINK 的方式进行烧录可以完全参照《快速上手指南》中的步骤进行操作，本文不再赘述。

2.2.2 CAU 工程配置

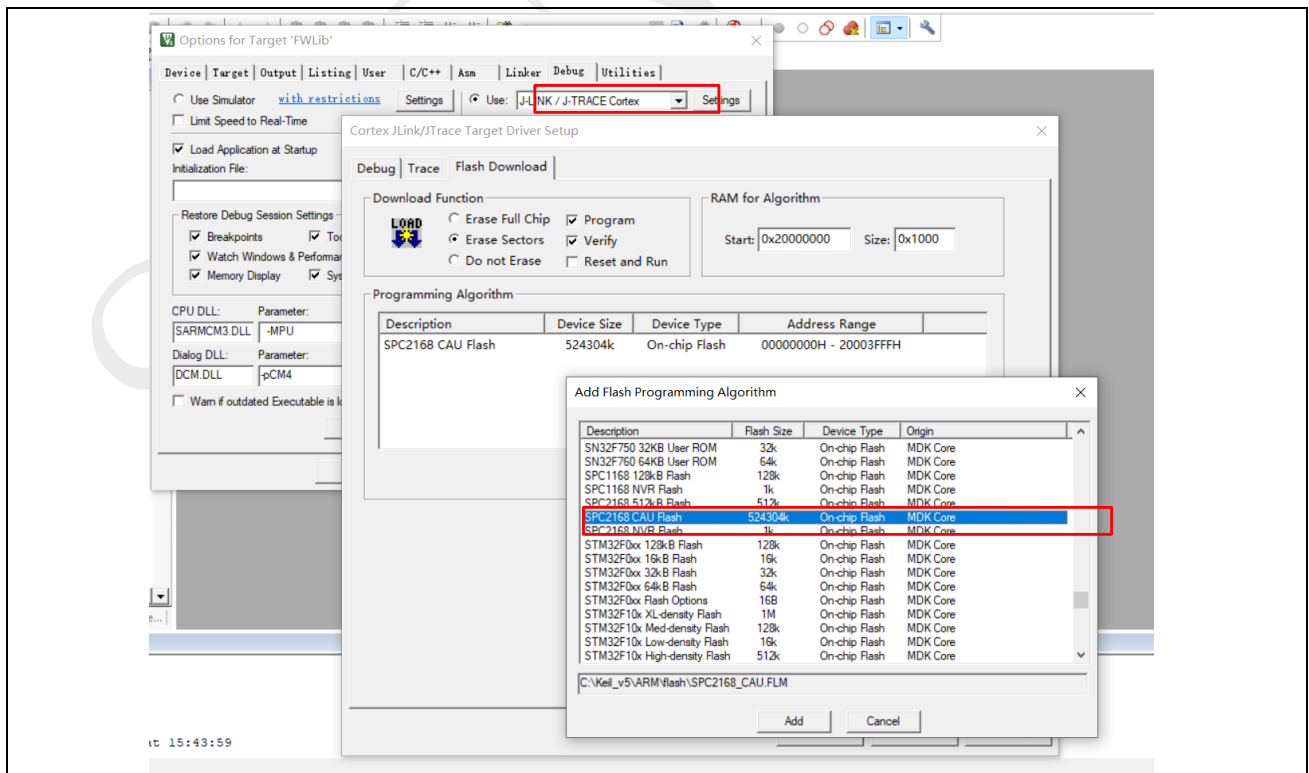
在详细介绍 CAU 的详细步骤之前，首先需要把 SPC2168_CAU.FLM 这个文件拷贝至 Keil 安装目录下的 ARM\Flash\文件夹下，另外，SPC2168_CAU.FLM 算法文件将会把 CAU 的镜像烧录到 Flash 0x10078000~0x1007FFFF 32KB 大小的区域内。

在 CAU 的工程中，在工程配置上我们需要注意以下几点：

- 添加 Flash 算法文件

如图 2-6 所示，选择正确的调试工具，然后再选择 SPC2168_CAU.FLM 算法文件，这个文件在 KEIL 中显示的 Flash Size 大小很大，这个无需担心，这个只是 KEIL IDE 的显示，并不会影响烧录过程。

图 2-6: CAU 工程添加 Flash 算法文件

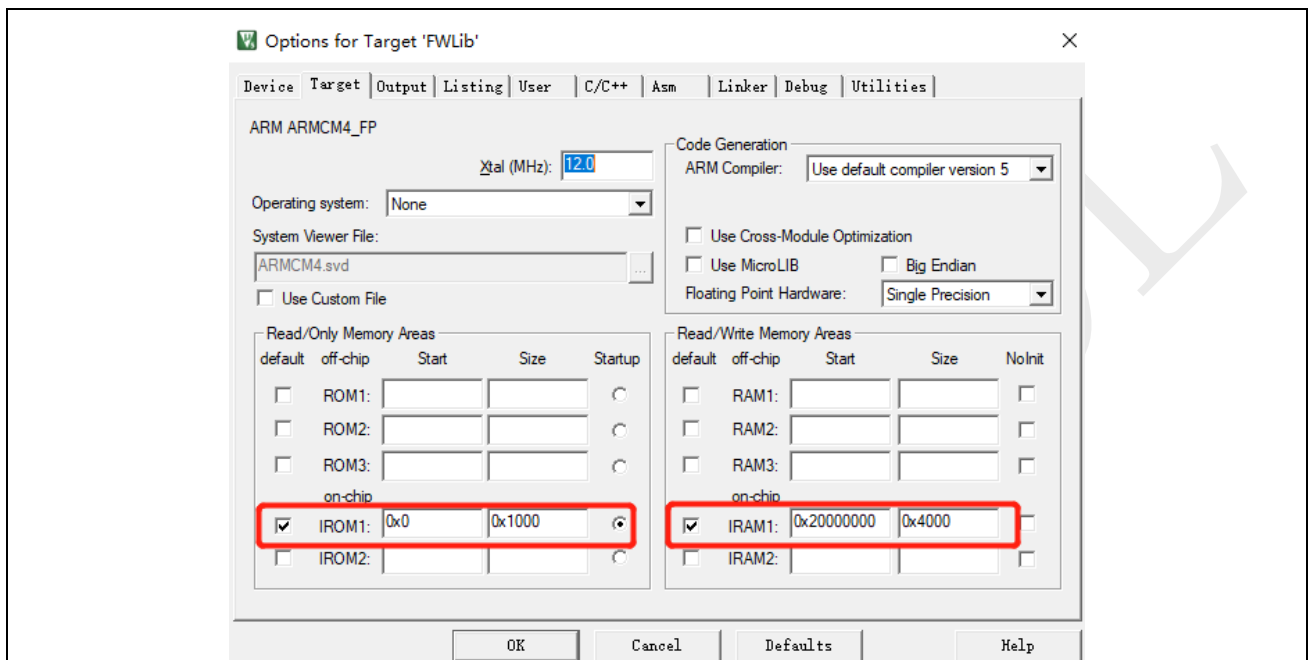


- 配置加载地址以及执行地址

CAU 工程中，加载地址必须为 0x00000000，执行地址，需要设置为 CAU 能够访问到的地址空间，具体 CAU 的访问地址空间范围，需要根据芯片的是否开启 CAU 以及是否开启 Cache 而定，具体细节请参考《技术参考手册》。

在 DualCore_CAU_ToggleGPIO 示例工程中，CAU 的加载地址以及执行地址配置如图 2-7 所示：

图 2-7: CAU 工程烧录地址及执行地址配置



当以上步骤都操作完毕，就可以使用 JLINK 的 SWD 接口烧录 CAU 镜像。

注意： 1. 因为芯片系统架构上只有 CPU 与 FLASH 连接，本章节介绍的 JLINK 烧录程序的方法中，JLINK 的 SWD 接口均需连接到芯片的 CPU 的调试接口（例如：SPC2168 CPU 的 SWD 接口为 GPIO48 GPIO49）

2.3 采用 JLINK 调试程序

在开始介绍 JLINK 烧录镜像之前，建议用户首先阅读《快速上手指南》手册，了解 JLINK 的基本使用方法。本手册将着重介绍使用 JLINK 方式调试的方法。

2.3.1 CPU 调试

对于 SPC2168 的 CPU 而言，应使用 Debug On FLASH 的方式进行调试。调试步骤为，先使用 JLINK 将程序烧录进 FLASH，再将 FLASH 中的程序进行调试。其烧录及调试可以完全参照《快速上手指南》文档中的步骤进行操作，在此就不再赘述。

2.3.2 CAU 调试

对于 SPC2168 的 CAU 而言，可以采用 Debug On FLASH 的方式，也可采用 Debug On RAM 的方式进行调试，两者的区别体现在：

- 调试接口

Debug On FLASH 的方式需要通过 CPU 的调试接口烧录程序，再通过 CAU 的调试接口调试程序。Debug On RAM 的方式只需通过 CAU 的调试接口调试程序。

- 代码是否固化

Debug On FLASH 方式调试的程序会固化到 FLASH 上。Debug On RAM 的代码，不会固化在 FLASH 上，芯片掉电后，RAM 中的程序会消失。

对于两种 CAU 的调试方法，对于应用开发人员，有如下使用建议：

- CAU 代码无需频繁修改

对于只是监测代码逻辑以及变量值，CAU 代码无需频繁修改的情况，应使用 Debug On FLASH 的方式进行调试。

- CAU 代码需频繁修改

对于需要根据每次调试不同反馈状态，进行 CAU 代码频繁修改的情况，应使用 Debug On RAM 的方式进行调试。

-
- 注意：
1. Debug On RAM 方式下，调试成功的代码，需进行烧录，才会固化到 FLASH 中。
 2. SPC2168 中 CAU 的调试接口为 GPIO50、GPIO51。
-

2.3.2.1 Debug On Flash 的方式进行 CAU 调试

- 前提条件

确保当前 CPU 烧录的程序中，具有如下代码动作：

示例代码

```
SYSTEM_EnableDebugForCAU() 或 SYSTEM_EnableDebugForDualCore()  
GPIO_SetPinChannel(GPIO_50,GPIO50_CAU_SWCK)  
GPIO_SetPinChannel(GPIO_51,GPIO51_CAU_SWL)  
SYSTEM_EnableCAU()
```

- 程序烧录

如本文 2.2.2 章节所述，通过 CPU 的调试接口，将程序烧录到 FLASH 中。

- 程序调试

如《快速上手指南》中的 3.3.1 章节所述，通过 CAU 的调试接口，进行 FLASH 中程序的调试工作。

2.3.2.2 Debug On RAM 的方式进行 CAU 调试

- 前提条件

确保当前 CPU 烧录的程序中，具有如下代码动作：

示例代码

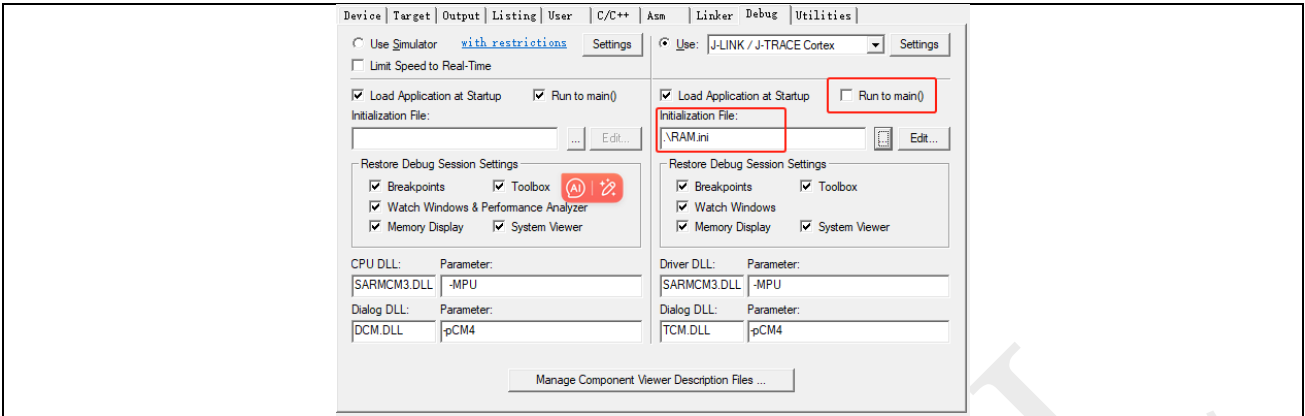
```
SYSTEM_EnableDebugForCAU() 或 SYSTEM_EnableDebugForDualCore()  
GPIO_SetPinChannel(GPIO_50,GPIO50_CAU_SWCK)  
GPIO_SetPinChannel(GPIO_51,GPIO51_CAU_SWL)  
SYSTEM_EnableCAU()
```

- 配置修改

在本文 2.2.2 章节所述基础上，进行如下修改。

第一步，配置 ini 文件，选择“Debug”配置项，在红色框部分输入 ini 文件的路径，如下图所示：

图 2-8: CAU 工程选择 ini 文件



第二步，点击“Edit...”按钮，在打开的文件中添加如下所示内容：

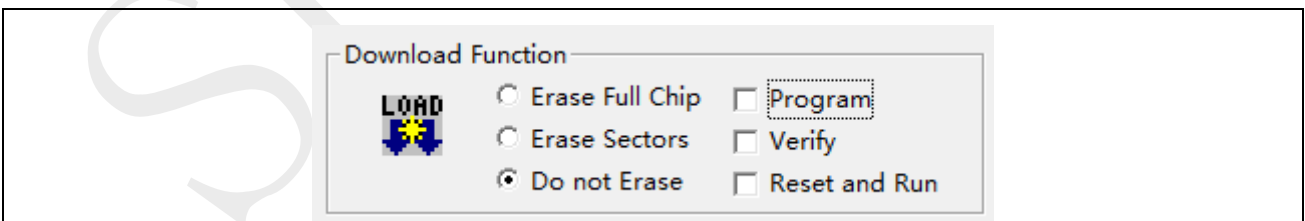
图 2-9: ini 文件内容

```

FUNC void Setup (void)
{
    SP = _RDWORD(0x0000);
    PC = _RDWORD(0x0004);           // Set Program Counter
    _WDWORD(0xE000ED08, 0x0000); // Set Vector Table Offset Register
}
LOAD %L INCREMENTAL              // Download to RAM
Setup();                          // Setup for debug running
g, main
    
```

第三步，取消 Flash 配置，选择“Flash Download”选项，在“Download Function”功能栏中选择“Do not Erase”选项，并取消“Program”、“Verify”、“Reset and Run”选项，如下所示：

图 2-10: Debug in RAM 下“Flash Download”配置



– 程序调试

上述前提条件及配置修改正确操作后，点击“Debug”按钮，可以进行 Debug On RAM 方式下的 CAU 程序调试。

2.4 JLINK 调试接口说明

SPC2168 用于调试的 PIN 只有 4 个，分别是：GPIO48 (TCK/SWCK)、GPIO49 (TMS/SWD)、GPIO50 (TDI/CAU_SWCK)、GPIO51 (TDO/CAU_SWCK)，通过 SYSTEM->CAUCTL 寄存器设置 JTAG 的功能。

表 2-1: JTAG 设置

CAUCTL 寄存器数值	DEBUG 接口功能
0x0	JTAG/SWD 只能由 CPU 使用
0xACCE51	JTAG/SWD 只能由 CAU 使用
0xACCE52	作为两个 SWD 功能，其中 GPIO48 和 GPIO49 是 CPU 的 SWD 接口，GPIO50 和 GPIO51 是 CAU 的 SWD 接口