GigaDevice Semiconductor Inc.

GD32E230 系列移植到 GD32F3x0 系列

应用笔记

AN046

目录

图索引

表索引

1. 前言

本应用笔记旨在帮助您快速将应用程序从 GD32E230xx 系列控制器移植到 GD32F3x0 系列控制器。

为了更好的利用本应用笔记中的信息，您需要从官网 www.GD32MCU.com 下载 GD32 各系列微控制器资料，如 Datasheet、用户手册、官方例程及各种开发工具等。

2. 硬件差异介绍

GD32E230 系列的封装类型有：TSSOP20、LGA20、QFN28、QFN32、LQFP32、LQFP48；GD32F3x0 系列的封装类型有：TSSOP20（仅 GD32F330/F310 系列）、QFN28、QFN32、LQFP32（仅 GD32F310 系列）、LQFP48、LQFP64（仅 GD32F330/F350 系列），两个系列相同封装的芯片引脚是兼容的，见*图 2-1. GD32F3x0 系列及 GD32E230 系列 LQFP48 封装对比图*，*图 2-2. GD32F3x0 系列及 GD32E230 系列 QFN32 封装对比图*，*图 2-3. GD32F3x0 系列及 GD32E230 系列 QFN28 封装对比图*，*图 2-4. GD32F330/F310 系列及 GD32E230 系列 TSSOP20 封装对比图*，*图 2-5. GD32F310 系列及 GD32E230 系列 LQFP32 封装对比图*。

注意：

1. TSSOP20 和 QFN28 的封装中，GD32E230 系列 PA9、PA10 可以映射为 PA11、PA12，GD32F3x0 系列不具备此功能。
2. LQFP48 封装管脚 1 在 GD32E230 系列上面是 VDD，在 GD32F3x0 上是 VBAT，也就是说 E230 不支持掉电运行 RTC。

图 2-1. GD32F3x0 系列及 GD32E230 系列 LQFP48 封装对比图

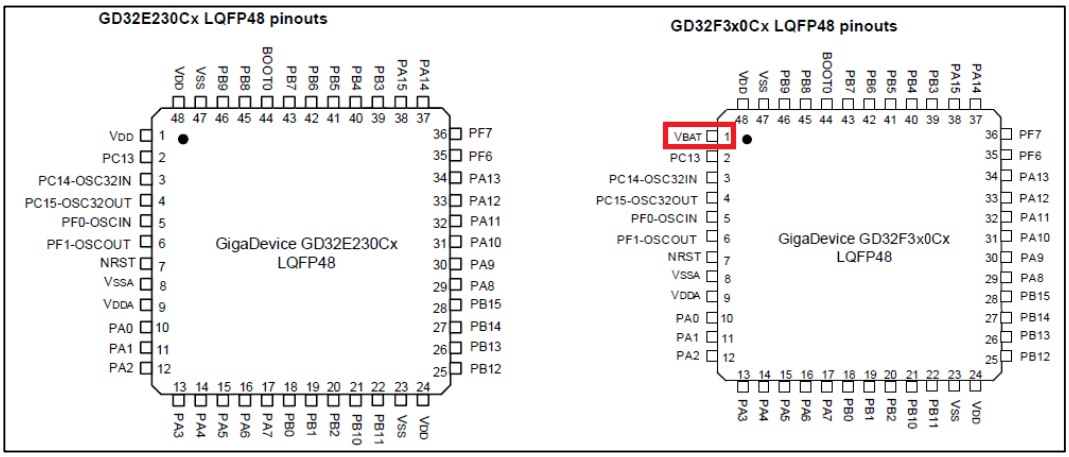


图 2-2. GD32F3x0 系列及 GD32E230 系列 QFN32 封装对比图

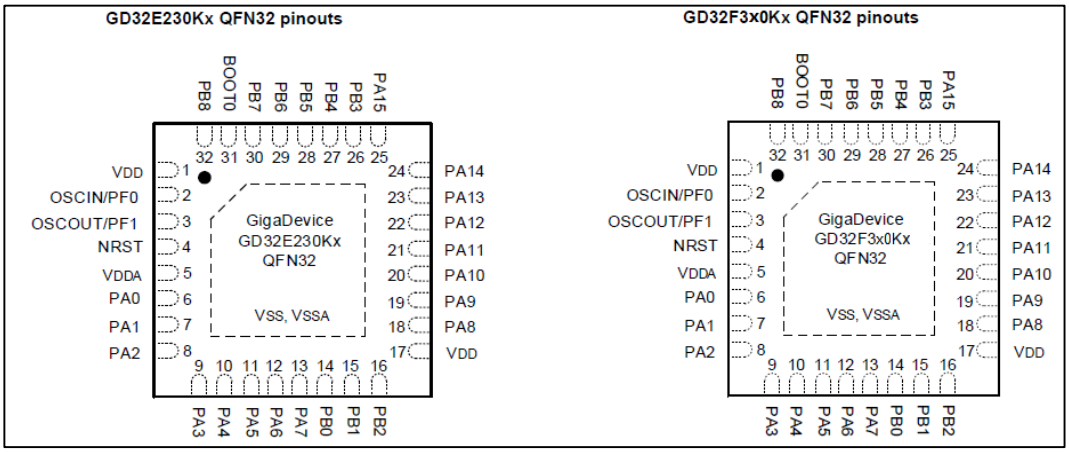


图 2-3. GD32F3x0 系列及 GD32E230 系列 QFN28 封装对比图

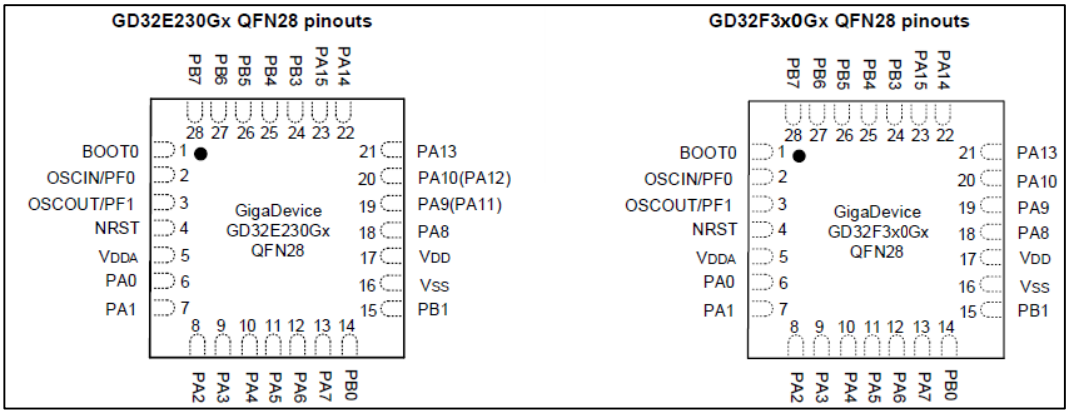


图 2-4. GD32F330/F310 系列及 GD32E230 系列 TSSOP20 封装对比图

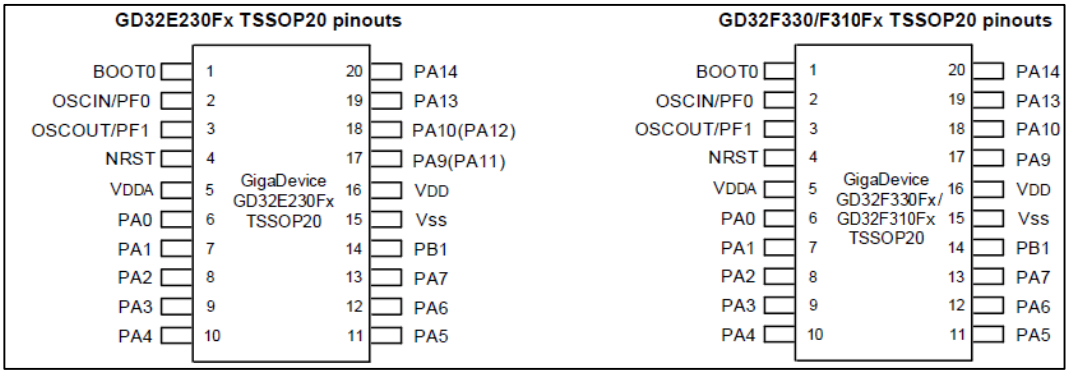
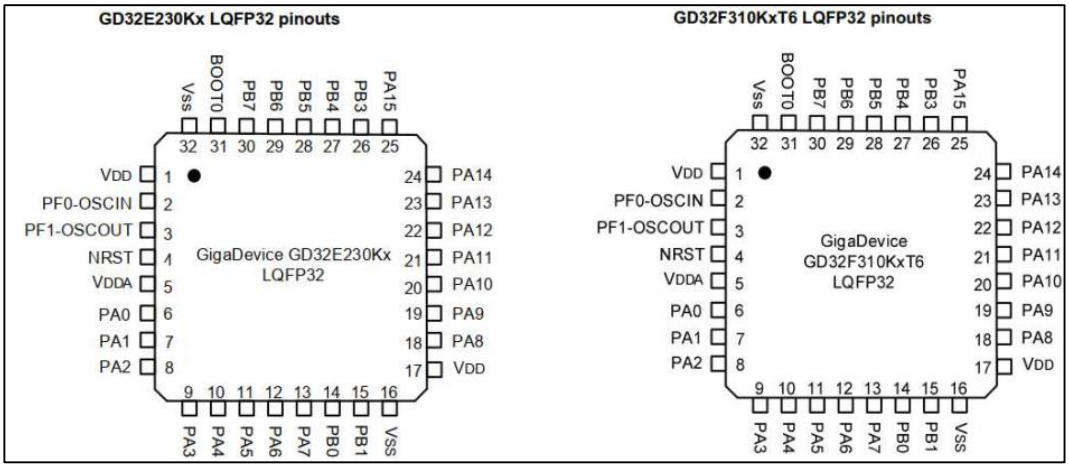


图 2-5. GD32F310 系列及 GD32E230 系列 LQFP32 封装对比图



3. 资源及外设地址对比介绍

GD32F3x0 与 GD32E230 的资源有细微的差别：

1. GD32F3x0 具有 TIMER1 外设，但没有 TIMER5（GD32F350 保留有此外设）外设，GD32E230 不具有 TIMER1 外设，但有 TIMER5 外设；
2. GD32E230 系列有一路比较器，GD32F330/F310 没有该外设，GD32F350 具有此外设；
3. GD32E230 系列新增了 1K 的 OTP 区域，GD32F3x0 没有该资源。
4. GD32F350 具有 USBFS、HDMI-CEC、DAC、TSI 功能，GD32F330/F310、GD32E230 没有这些外设。

详情请查询*表 3-1. GD32F3x0 系列及 GD32E230 系列资源对比总览*及*表 3-2. GD32F3x0 系列及 GD32E230 系列外设地址对比总览*。

表 3-1. GD32F3x0 系列及 GD32E230 系列资源对比总览

Peripheral	GD32F310系列	GD32F330系列	GD32F350系列	GD32E230系列
Core	Cortex-M4	Cortex-M4	Cortex-M4	Cortex-M23
Flash	16K-64K	16K-128K	16K-128K	16K-64K
RAM	4K-8K	4K-16K	4K-16K	4K-8K
主频	72MHz	84MHz	108MHz	72MHz
GPTM(32bit)	0	1	1	0
GPTM(16bit)	4/5	4/5	5	4/5
AdvTM	1	1	1	1
BaseTM	0	0	1	1
U(S)ART	1/2	1/2	1/2	1/2
I2C	1/2	1/2	1/2	1/2
SPI	1/2	1/2	1/2	1/2
I2S	1	0	1	1
USBFS	0	0	1	0
HDMI-CEC	0	0	1	0
TSI	0	0	1	0
COMP	0	0	2	1
ADC	1(9)/1(10)	1(9)/1(10)/1(16)	1(10)/1(16)	1(9)/1(10)
DAC	0	0	1	0
Operating Voltage	2.6-3.6V	2.6-3.6V	2.6-3.6V	1.8-3.6V
Temperature Range	-40-85℃	-40-85℃	-40-85℃	-40-85℃

注：以上“/”代表有多种情况，需要根据具体芯片型号区分。

表 3-2. GD32F3x0 系列及 GD32E230 系列外设地址对比总览

Peripheral	BUS	GD32F3x0系列	GD32E230系列
GPIOF	AHB2	0X48001400	0X48001400
GPIOD		0X48000C00	-
GPIOC		0X48000800	0x48000800

GPIOB		0X48000400	0X48000400
GPIOA		0X48000000	0X48000000
USBFS	AHB1	0X50000000	-
TSI		0X40024400	-
CRC		0X40023000	0X40023000
FMC		0X40022000	0X40022000
RCU		0X40021000	0X40021000
DMA		0X40020000	0X40020000
DBG	APB2	0xE0042000	0X40015800
TIMER16		0X40014800	0X40014800
TIMER15		0X40014400	0X40014400
TIMER14		0X40014000	0X40014000
USART0		0X40013800	0X40013800
SPI0/I2S0		0X40013000	0X40013000
TIMER0		0X40012C00	0X40012C00
ADC		0X40012400	0X40012400
EXTI		0X40010400	0X40010400
SYSCFG+CMP		0X40010000	0X40010000
CTC	APB1	0X4000C800	-
CEC		0X40007800	-
DAC		0X40007800	-
PMU		0X40007000	0X40007000
I2C1		0X40005800	0X40005800
I2C0		0X40005400	0X40005400
USART1		0X40004400	0X40004400
SPI1		0X40003800	0X40003800
FWDGT		0X40003000	0X40003000
WWDGT		0X40002C00	0X40002C00
RTC		0X40002800	0X40002800
TIMER13		0X40002000	0X40002000
TIMER5		0X40001000	0X40001000
TIMER2		0X40000400	0X40000400
TIMER1		0X40000000	-
SRAM		0x20000000	0x20000000
Option Byte		0x1FFFF800	0x1FFFF800
Main Flash		0x08000000	0x08000000
System Memory		0x1FFFEC00	0x1FFFEC00
OTP		-	0x1FFF7000

4. 开发工具对比

GD32F3x0可使用MDK for ARM的Keil 4及Keil 5进行开发，使用Keil 4建议安装4.74及以上；使用Keil 5建议安装5.20以上版本。也可以使用IAR for ARM开发，建议安装IAR 6.3及以上版本，如*表4-1. GD32F3x0系列及GD32E230系列IDE环境对比*介绍。

表 4-1. GD32F3x0 系列及 GD32E230 系列 IDE 环境对比

芯片系列	GD32F3x0系列	GD32E230系列
KEIL	KEIL4或者Keil5均支持	KEIL 5.25及以上版本
IAR	IAR 6.3及以上版本	IAR 8.1及以上版本

GD32F3x0 可以使用 JLINK、ULINK、GDLINK 等调试工具进行开发，如*表 4-2. GD32F3x0 系列及 GD32E230 系列调试工具对比*介绍。

表 4-2. GD32F3x0 系列及 GD32E230 系列调试工具对比

芯片系列	GD32F3x0系列	GD32E230系列
JLINK	JLINK OB、V8、V9等均支持	仅JLINK V9及以上版本支持
ULINK	支持	支持
GDLINK	支持	支持

5. 软件环境设置

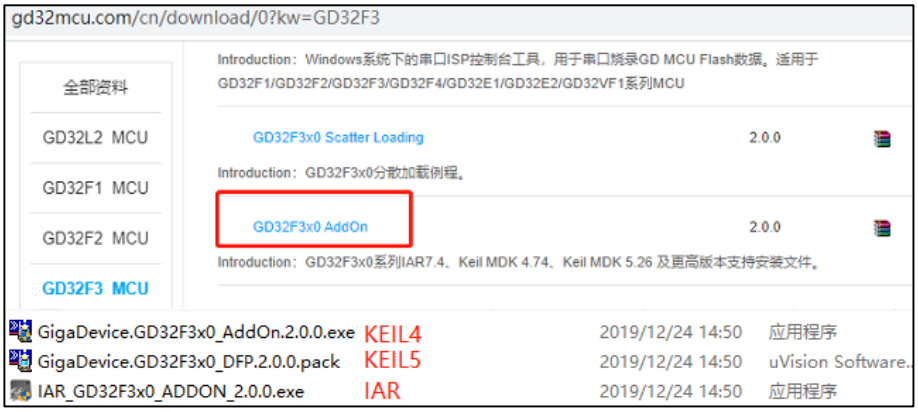
5.1. 使用 Keil 开发 GD32F3x0

目前市面通用的MDK for ARM版本有Keil 4和Keil 5：使用Keil 4建议安装4.74及以上，使用Keil 5建议安装5.20以上版本。

5.1.1. 在 Keil4 中添加 GD32F3x0 MCU Device

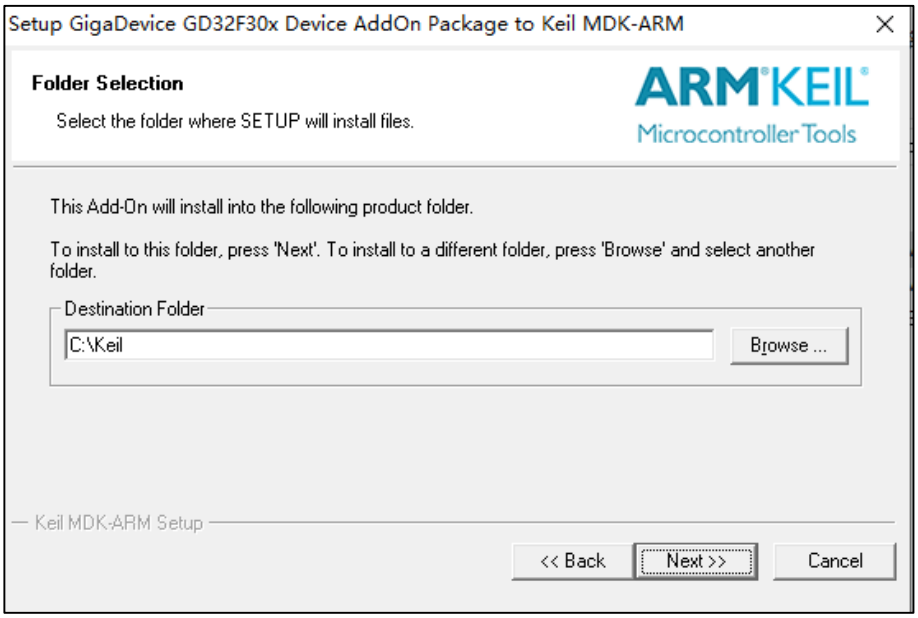
1. 从GD32MCU官网下载相关的GD32F3x0系列插件。

图 5-1. GD32F3x0 pack 包明细



2. 双击安装文件，把插件安装至Keil 4的目录，一般都会默认选择，如若同时安装了Keil 4和Keil 5才需要手动选择。

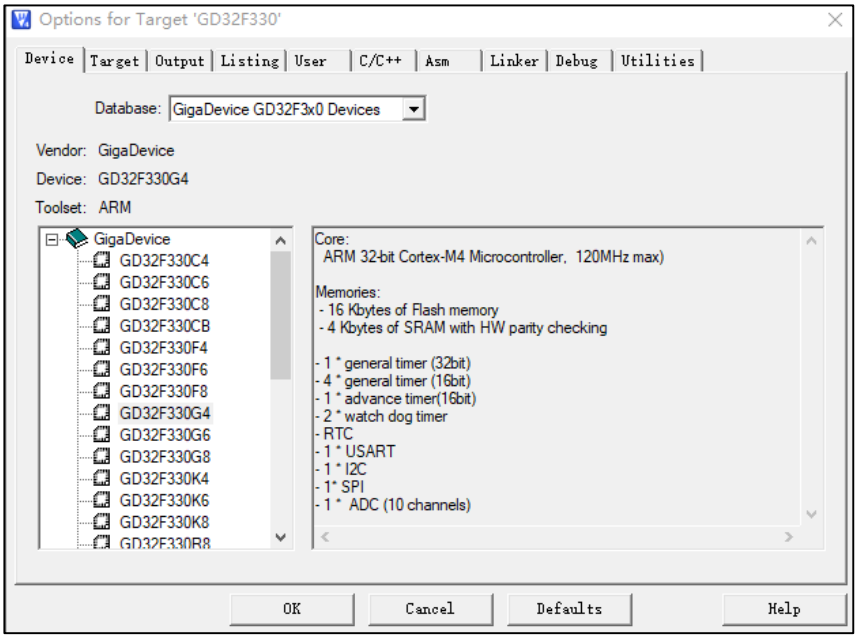
图 5-2. GD32F3x0 系列 MCU Pack 包安装示意图（Keil4）



3. 安装成功后，重新打开Keil 4，则可以在“File->Device Database”中出现“Gigadevice”

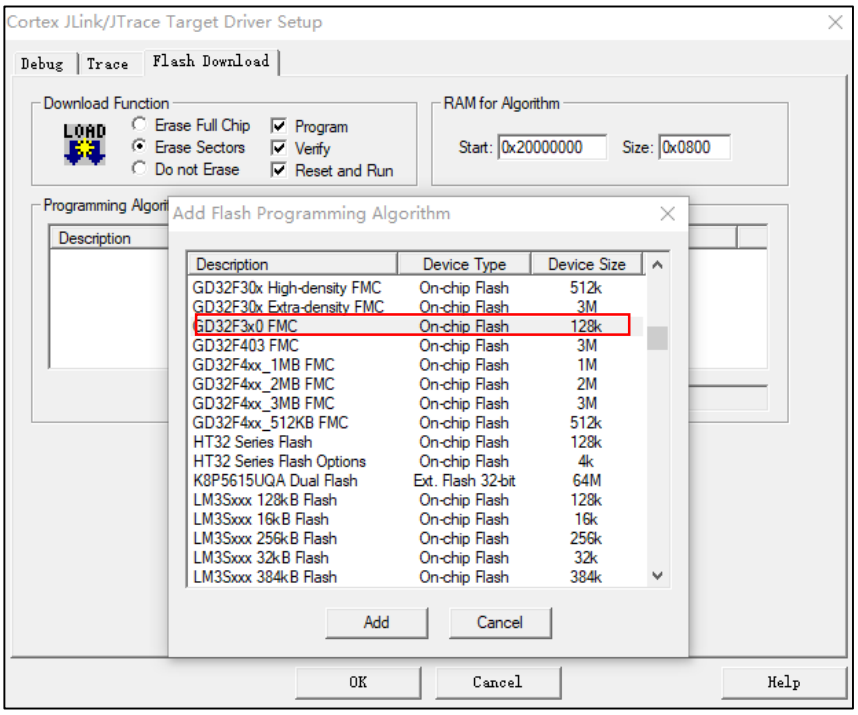
的下拉选项，点击可以查看到相应的型号。

图 5-3. GD32F3x0 系列 MCU Pack 包成功安装示意图（Keil4）



4. 为了后续debug工作的顺利进行，建议检查一下安装路径下是否有下载算法，可以通过如下方式查看：打开一个工程，将型号选为GD32F3x0的型号，然后“Options for Target -> Debug ->Settings -> Flash Download-> Add”，如果下拉选项中有GD32F3x0的Flash下载算法则表示安装成功。

图 5-4. GD32F3x0 系列 Flash 算法文件选择示意图（Keil4）



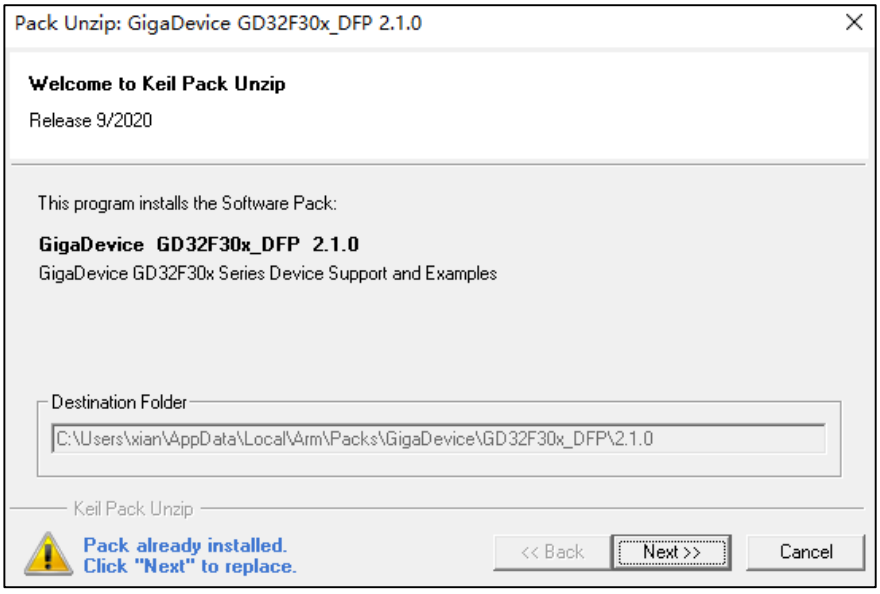
5.1.2. 在 Keil5 中添加 GD32F3x0 MCU Device

1. 从GD32MCU官网下载相关的GD32F3x0系列插件。

图 5-5. GD32F3x0 pack 包明细

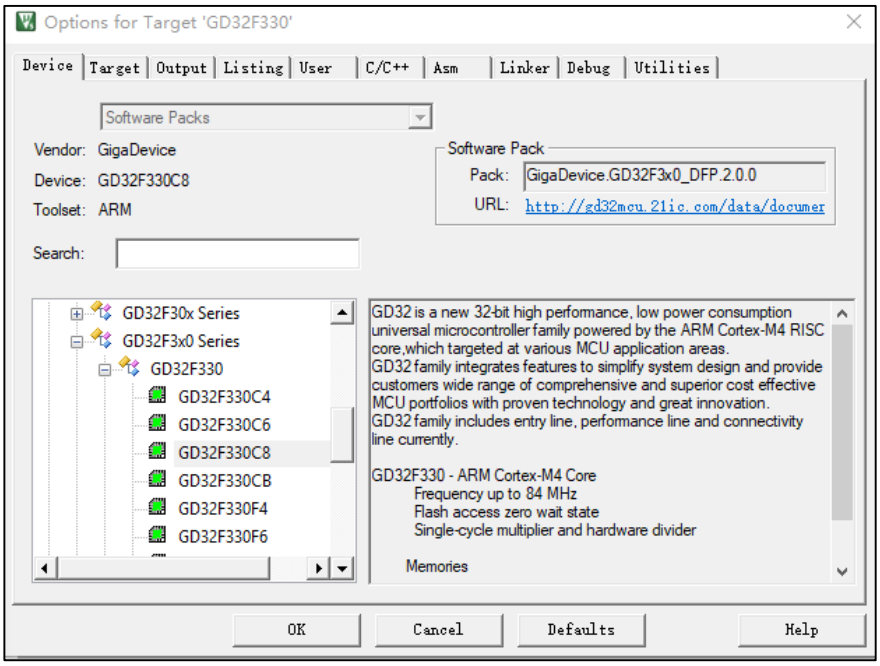
2. 解压并安装至Keil 5的目录，一般都会默认选择。

图 5-6. GD32F3x0 系列 MCU Pack 包安装示意图（Keil5）



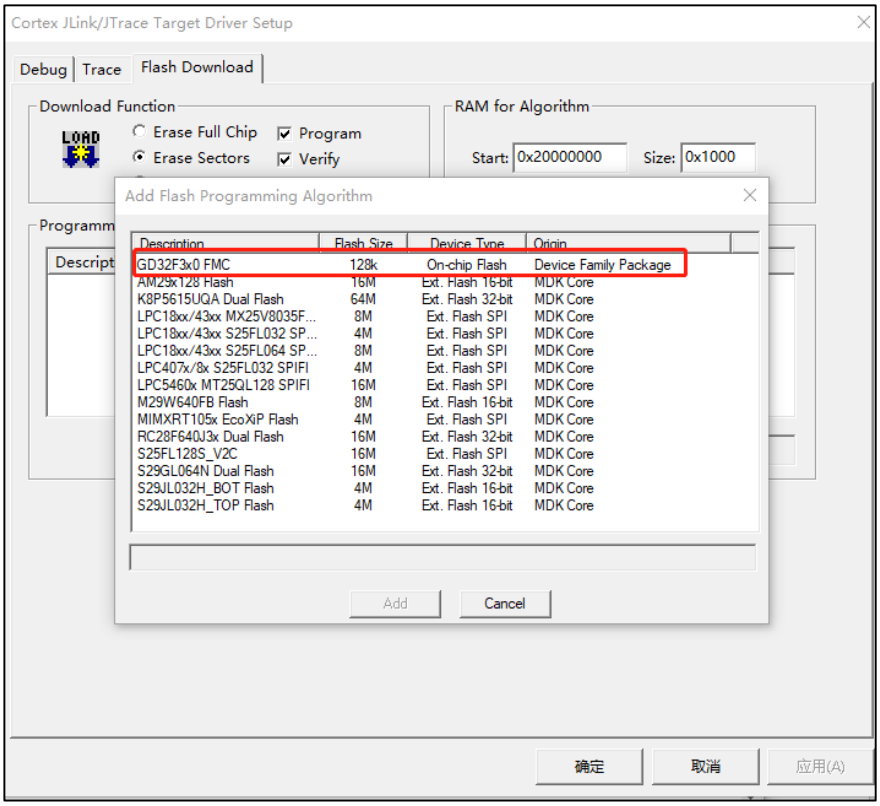
3. 安装完后重新打开Keil 5工程，即可在Device中出现Gigadevice的型号。

图 5-7. GD32F3x0 系列 MCU Pack 包安装成功示意图（Keil5）



4. 在“Options for Target -> Debug ->Settings ->Flash Download”中添加Flash算法，会出现GD32F3x0的算法，即说明安装成功。根据相应的芯片选择合适的算法，即可下载仿真。

图 5-8. GD32F3x0 系列 Flash 算法文件添加示意图（Keil5）



5. Keil 4打开Keil 5工程

如果没有安装Keil 5，也是能够使用Keil 4来编译Keil 5的工程，具体做法就是修改工程的后缀名，将Keil 5工程的后缀名“xxxx.uvprojx”修改为“xxxx.uvproj”，即可使用Keil 4来查看编译了。

6. Keil 5打开Keil 4工程

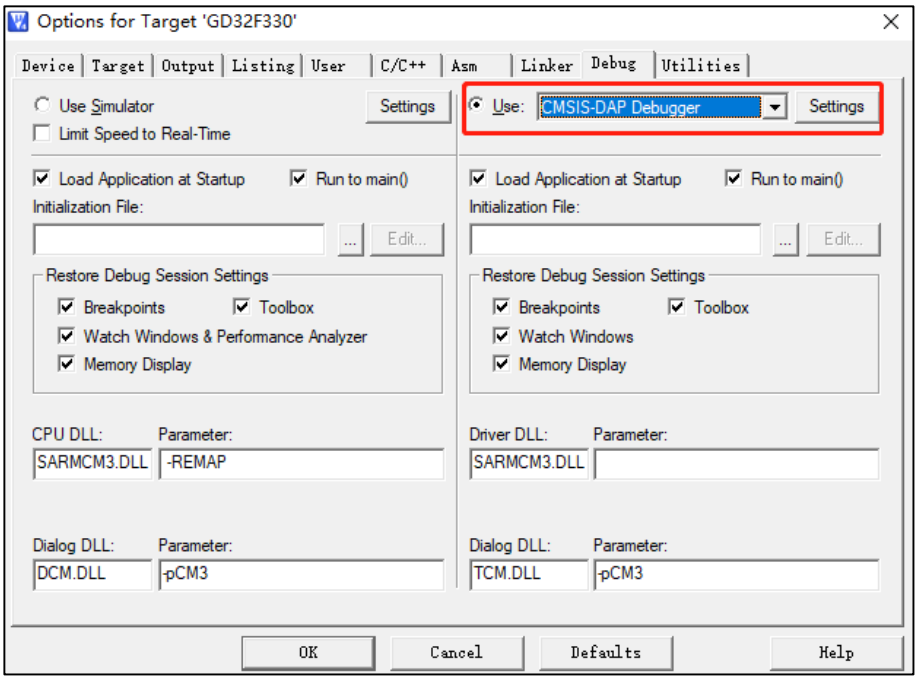
   如果使用Keil 5打开Keil 4工程，打开时会遇到找不到MCU器件的情况，这种可以直接将Keil 4工程的后缀名“xxxx.uvproj修改为xxxx.uvprojx”，即可正常使用Keil 5来查看编译了。

5.2. 使用 GD-Link 工具开发 GD32F3x0

使用GD-Link工具来debug GD MCU，硬件上需要用GD-Link工具连接开发板，软件上具体配置如下：

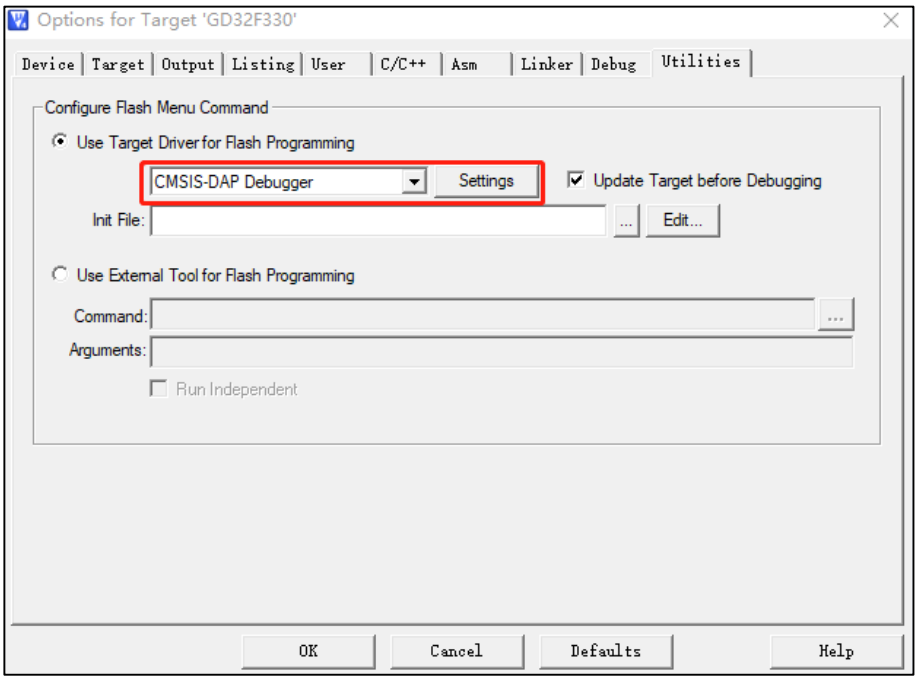
1. 打开一个GD32F3x0的工程，在“Options for Target -> Debug”中选择“CMSIS-DAP Debugger”，部分客户反馈找不到这一驱动器选项，那是因为MDK版本过低，只有Keil4.74以上的版本和Keil 5才支持“CMSIS-DAP Debugger”选项。

图 5-9. Debug 界面中选择“CMSIS-DAP Debugger”选项（Keil4）



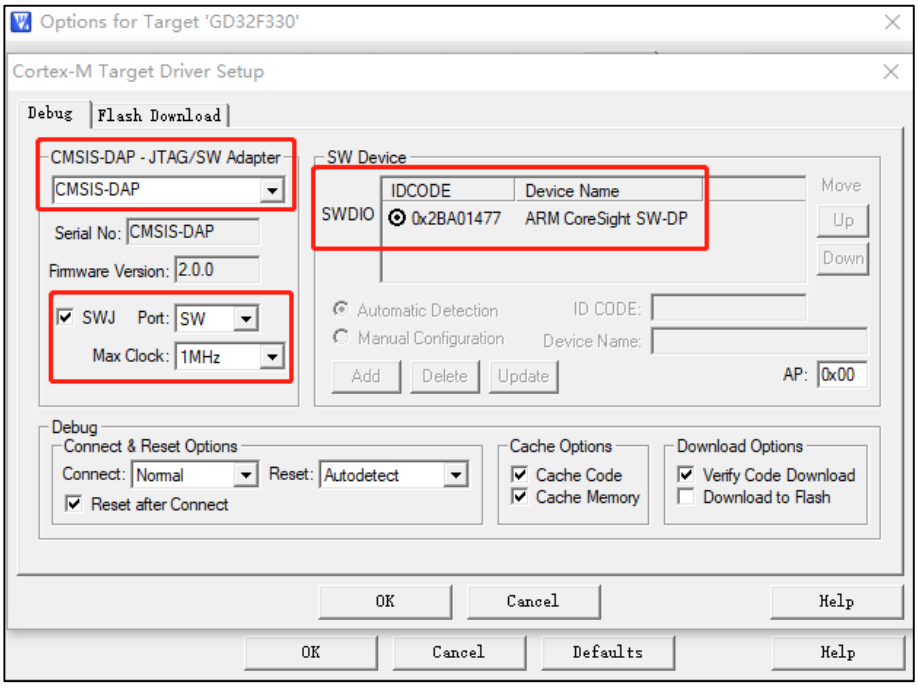
2. 在“Options for Target -> Utilities”，也需选择“CMSIS-DAP Debugger”。

图 5-10. Utilities 界面中选择“CMSIS-DAP Debugger”选项（Keil4）



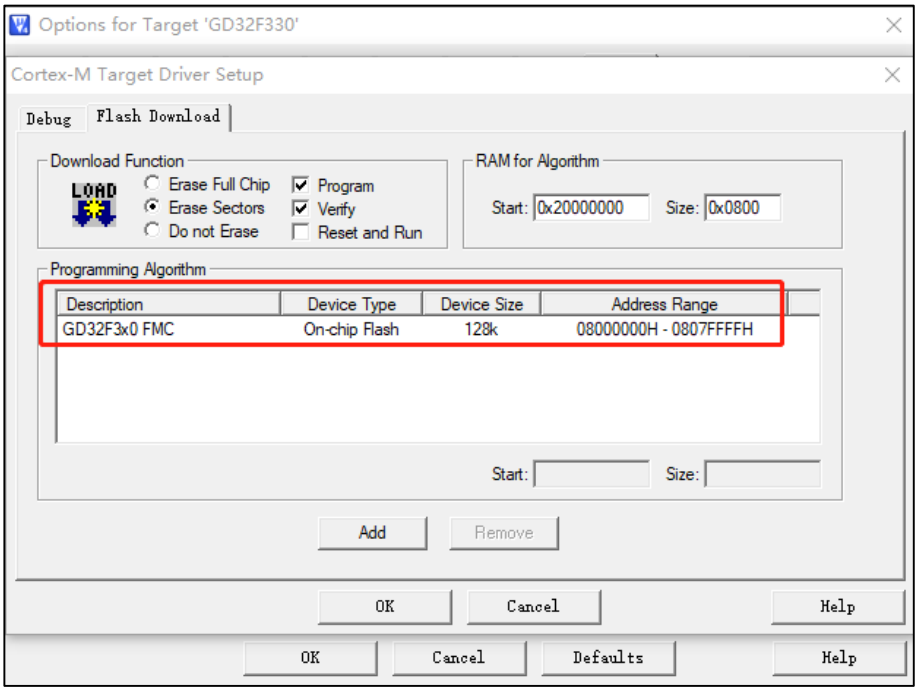
3. 在“Options for Target -> Debug ->Settings”勾选SWJ、Port选择 SW。右框IDcode会出现“0xXBAXXXXX”。

图 5-11. GD-Link 成功连接目标芯片示意图（Keil4）



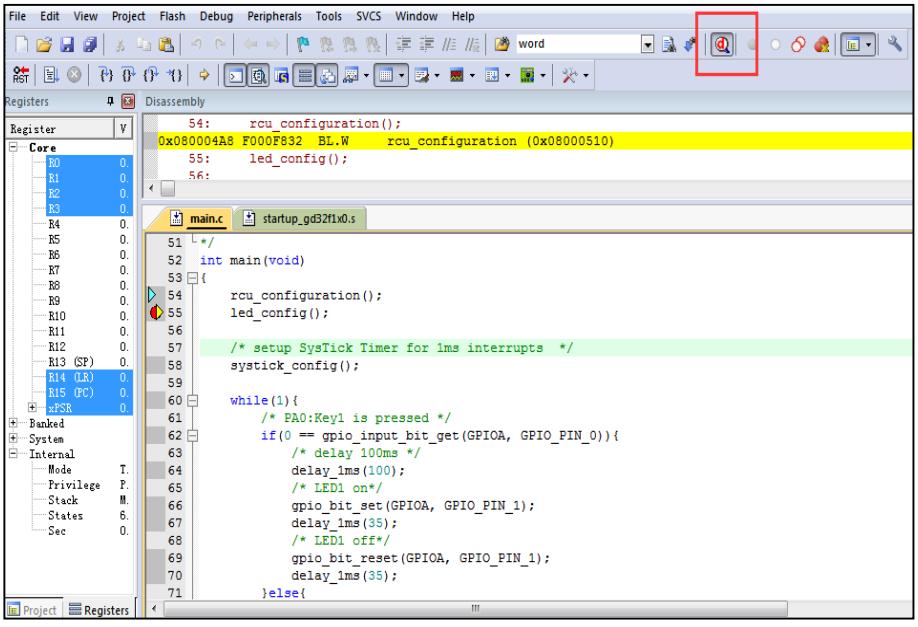
4. 在“Options for Target -> Debug ->Settings -> Flash Download”中添加GD32的Flash算法。

图 5-12. 添加 GD32F3x0 Flash 算法文件示意图（Keil4）



5. 单击*图5-13. GD32F3x0工程GD-Link仿真示意图（Keil4）*的红框的快捷方式debug，即可使用GD-Link进行仿真。

图 5-13. GD32F3x0 工程 GD-Link 仿真示意图（Keil4）

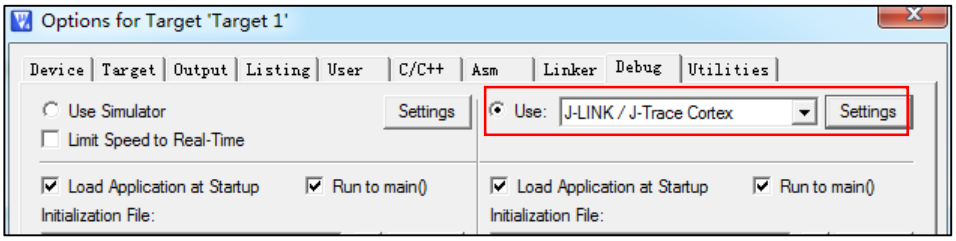


5.3. 使用 J-Link 工具开发 GD32F3x0

使用J-Link工具来debug GD MCU，硬件上需要用J-Link工具连接开发板，软件上具体配置如下：

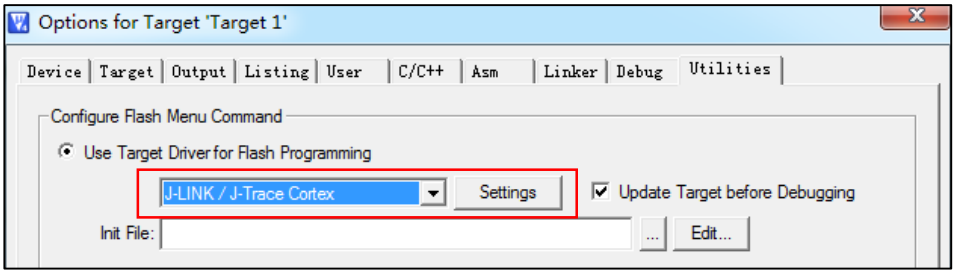
1. 打开一个GD32F3x0的工程，在“Options for Target -> Debug中”选择“J-LINK/J-Trace Cortex”。

图 5-14. Debug 界面中选择“J-LINK/J-Trace Cortex”选项（Keil4）



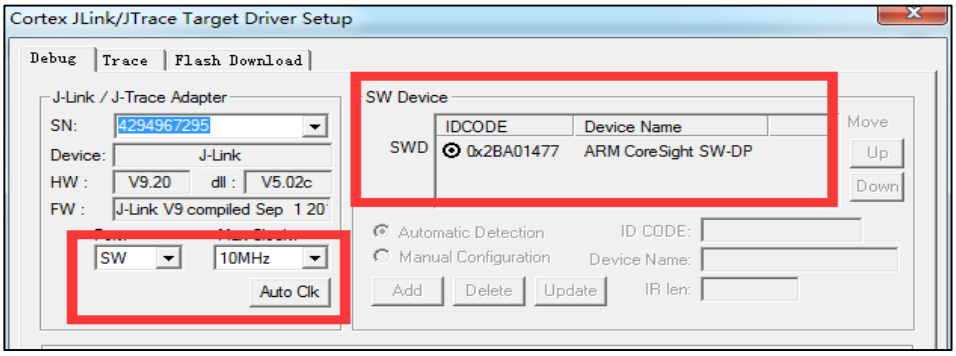
2. 在Options for Target -> Debug ->Utilities，也要选择“J-LINK/J-Trace Cortex”。

图 5-15. Utilities 界面中选择“J-LINK/J-Trace Cortex”类型（Keil4）



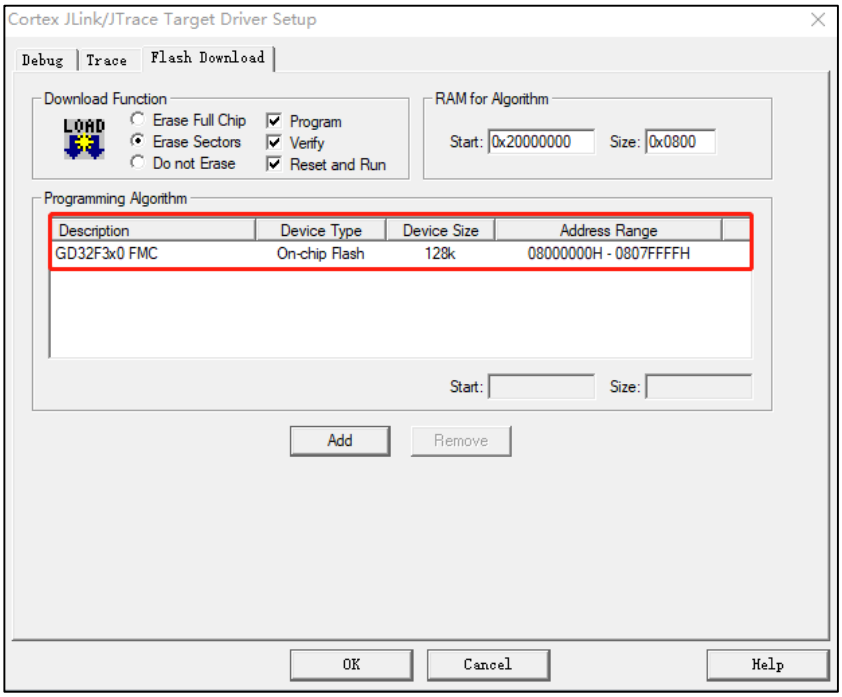
3. 在“Options for Target -> Debug ->Settings”，Port选择SW。右框IDCODE会出现“0xXBAXXXXX”，表示成功连接目标芯片。

图 5-16. J-Link 成功连接芯片示意图（Keil4）



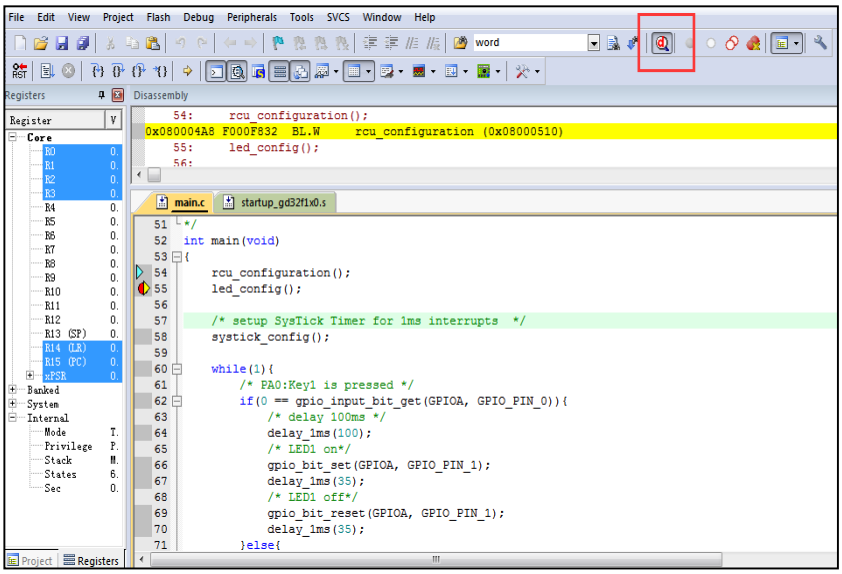
4. 在“Options for Target -> Debug ->Settings -> Flash Download”中添加GD32的Flash算法。

图 5-17. 添加 GD32F3x0 Flash 算法文件示意图（Keil4）



5. 单击如*图5-18. GD32F3x0工程J-Link仿真示意图（Keil4）*所示的快捷方式debug，即可使用J-Link进行仿真。

图 5-18. GD32F3x0 工程 J-Link 仿真示意图（Keil4）



5.4. 使用 IAR 开发 GD32F3x0

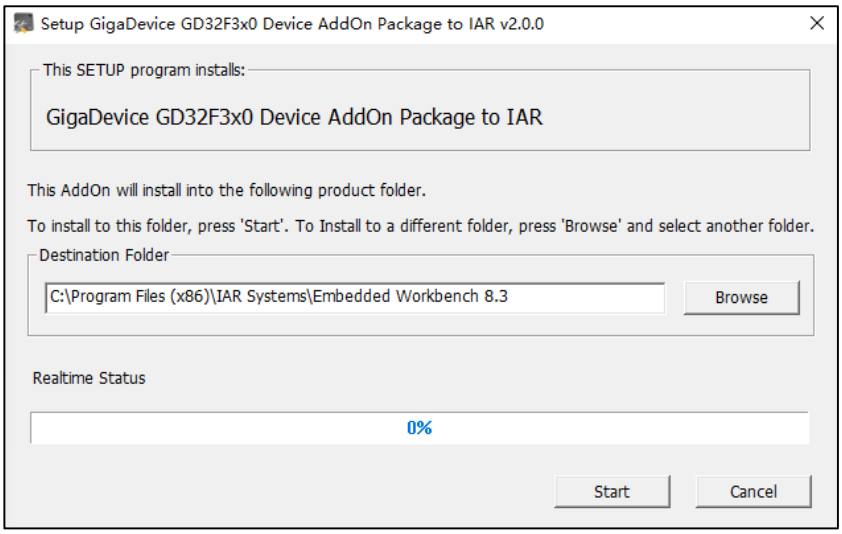
IAR版本众多，版本之间的兼容性并不好，如果初次使用建议安装7.3以上的版本,安装好IAR以后再根据该文档来添加GD的器件型号，进行相关的debug工作。

5.4.1. 在 IAR 中添加 GD32F3x0 MCU Device

1. 从相关网站下载相应的GD32F3x0系列插件：IAR_GD32F3x0_ADDON_2.0.0.exe；

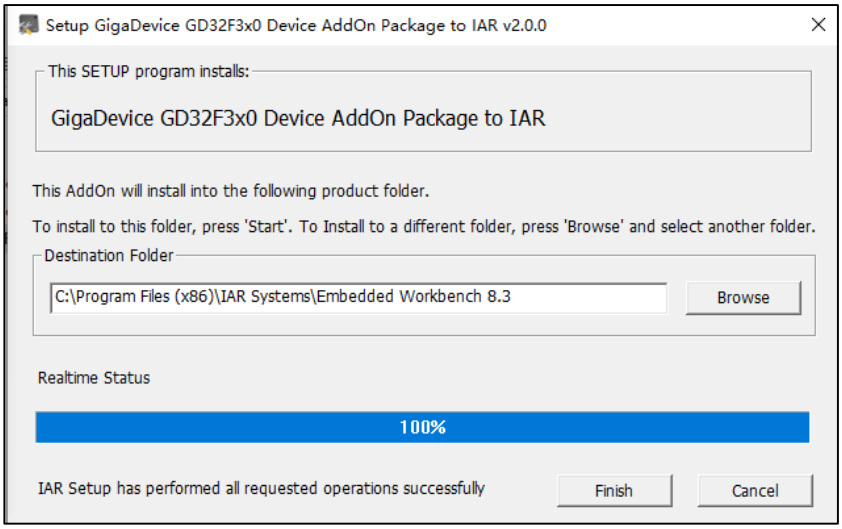
2. 运行IAR_GD32F3x0_ADDON_2.0.0.exe，单击Start开始安装插件。

图 5-19. GD32F3x0 系列 MCU Pack 包安装示意图（IAR）



3. 安装成功后单击Finish，结束插件安装。

图 5-20. GD32F3x0 系列 MCU Pack 包安装完成示意图（IAR）

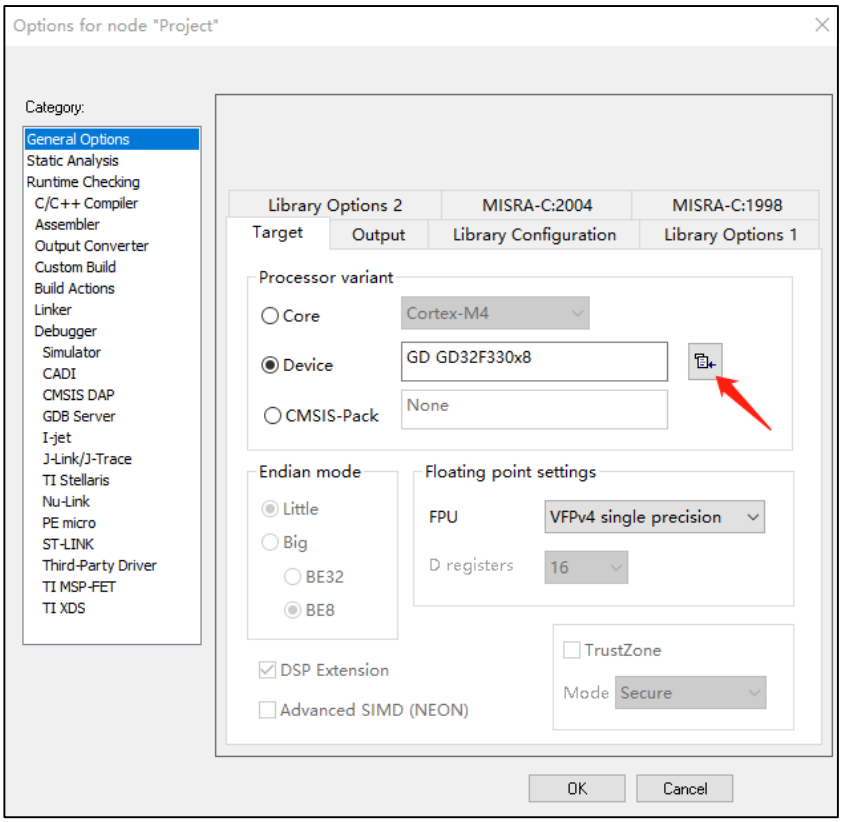


5.4.2. 在 IAR 中编译调试 GD32F3x0

在上一小节中我们已经添加了GD32F3x0系列的插件，这一小节我们介绍应如何使用它。

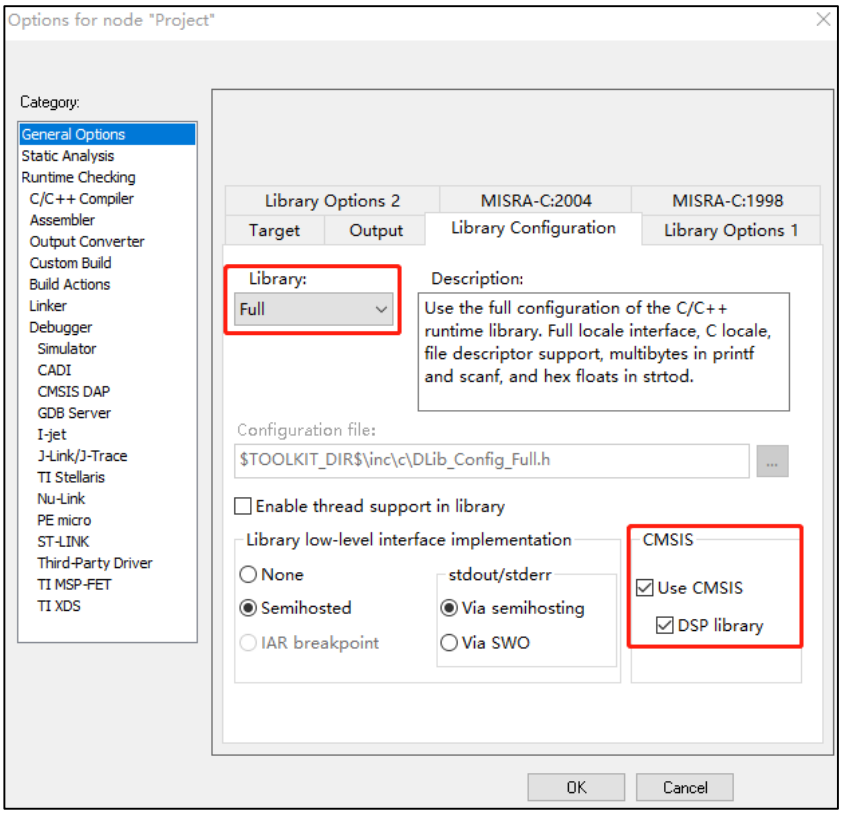
1. 使用IAR编译GD的型号，有两个办法，一种是使用现有的工程进行修改，另一种就是重新建立工程，这里就不细说具体工程应该如何建立，GD的工程建立和别的平台都一致，建立工程时选择GD的相应型号即可。

图 5-21. 在 IAR “Options”界面中选择芯片型号示意图



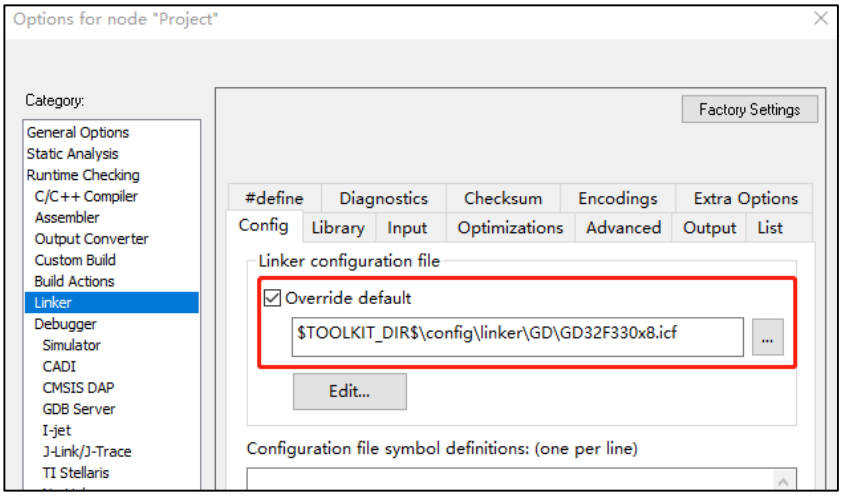
2. 6.1版本以后的IAR不需要添加CMSIS文件（core_cm3.c和core_cm3.h），但是需要勾选“General Options->Library Configuration”的Use CMSIS，如果软件代码有使用到printf函数，还需要修改Library为FULL。

图 5-22. 在 IAR “Options”界面中添加 CMSIS 文件示意图



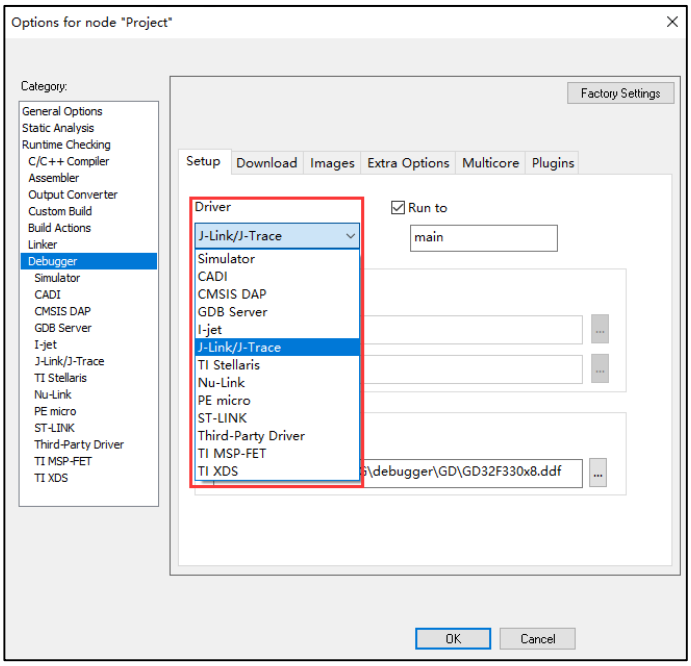
3. 芯片的Link文件建立工程时会默认根据型号选定，但是编译前还是要有检查的习惯，检查一下ICF文件是否有配置，是否正确。

图 5-23. 在 IAR “Options”界面中添加 ICF 文件示意图



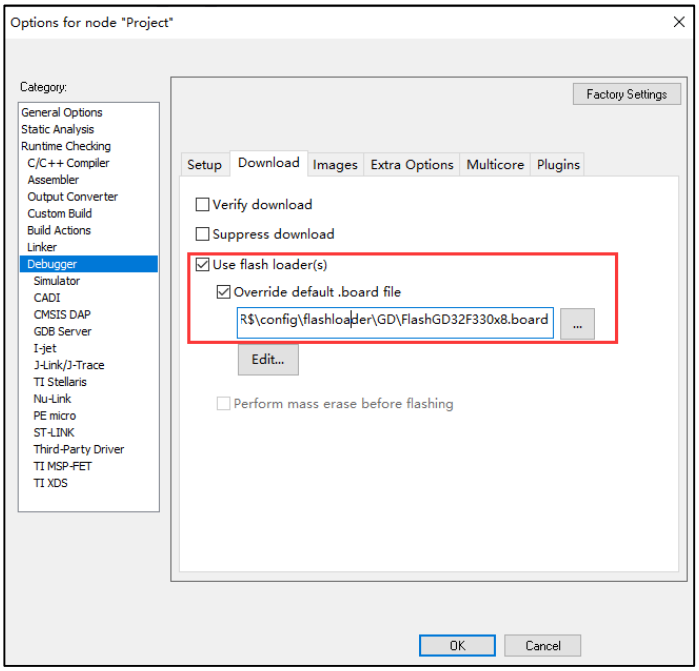
4. 配置“Debugger->Setup”选项，新建立的工程默认是Simulator模拟，如果需要调试那么需要根据实际情况来选择：使用GD-Link选择CMSIS DAP（兼容性不好，不建议在IAR下使用）或使用J-Link选择J-Link/J-Trace。

图 5-24. 在 IAR “Options”界面选择 Debugger 工具示意图



5. 配置“Debugger->Download”选项，新建的工程有可能没有配置download选项，如果我们需要调试代码那么务必要勾选User flash loader选项，且保证board file准确，否则程序无法正常下载至芯片内部。

图 5-25. 在 IAR “Options”界面配置 flash loader 示意图

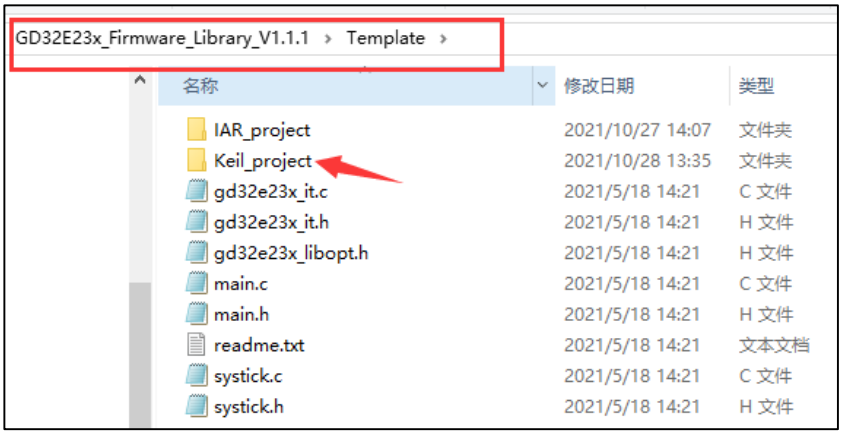


6. GD32E23x 固件库适配 GD32F3x0 系列 MCU 步骤

本章将使用GD32E23x_Firmware_Library_V1.1.1固件库文件Template里的工程做示例，介绍如何适配GD32F3x0系列芯片。

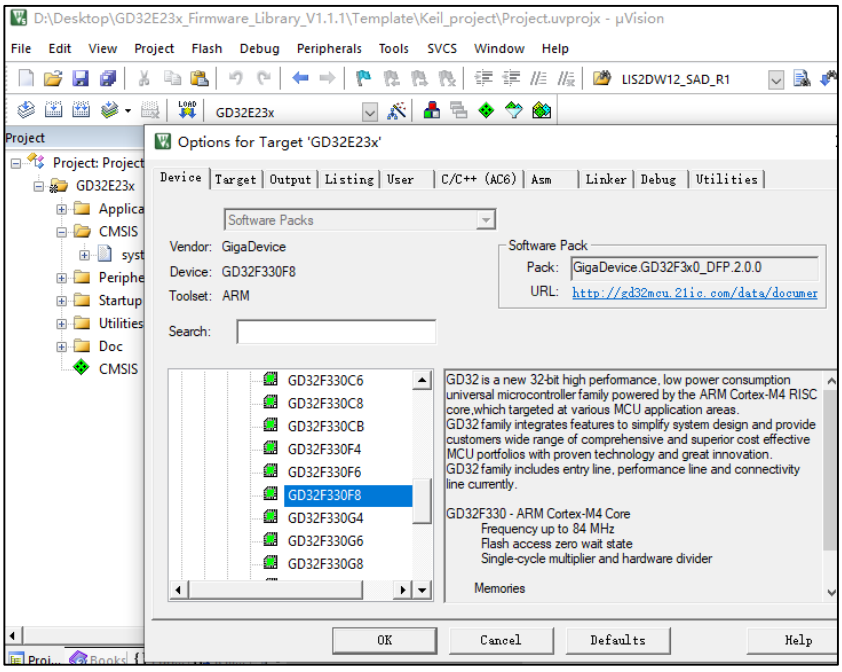
1. 打开Keil工程

图 6-1. 打开 GD32E23x Keil 工程示意图



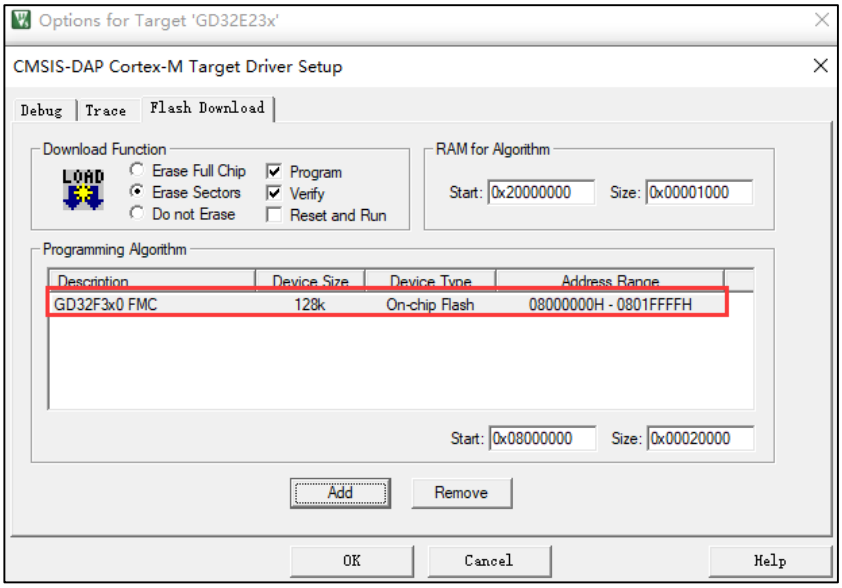
2. 打开工程后，“Options for Target -> Device”，选择对应的GD32F3x0型号。

图 6-2. 在 GD32E23x 工程中选择 GD32F3x0 芯片型号示意图



3. 在“Options for Target -> Debug ->Settings -> Flash Download”中添加GD32F3x0的Flash算法。

图 6-3. 在 GD32E23x 工程中添加 GD32F3x0 的 Flash 算法示意图



4. 拷贝Cortex M4 内核支持文件至:

x:\ GD32E23x_Firmware_Library_V1.1.1\Firmware\CMSIS。

图 6-4. 在 GD32E23x 固件库文件中添加 Cortex M4 内核文件示意图

5. 修改GD32E23x固件库中“gd32e23x.h”头文件的内容。

图 6-5. 修改 GD32E23x 固件库中“gd32e23x.h”头文件的内容

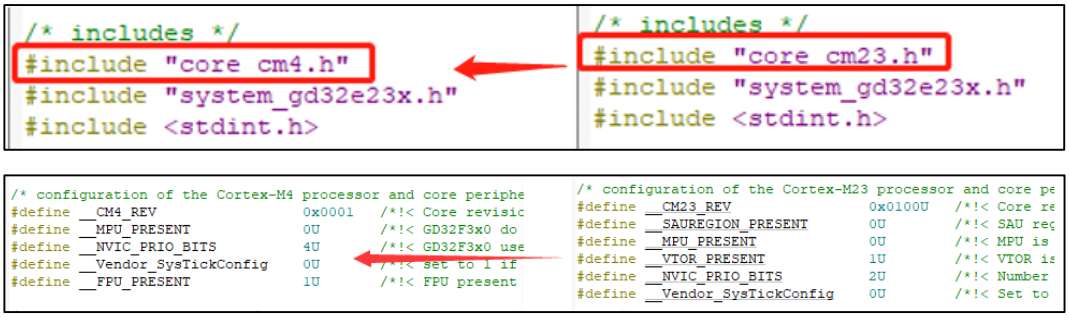


表 6-1. 修改 GD32E23x 固件库中“gd32e23x.h”头文件的内容

修改后	修改前
#include "core_cm23.h"	#include "core_cm23.h"
#define __CM4_REV 0x0001	#define __CM23_REV 0x0100U

#define __MPU_PRESENT 0U	#define __SAUREGION_PRESENT 0U
#define __NVIC_PRIO_BITS 4U	#define __MPU_PRESENT 0U
#define __Vendor_SysTickConfig 0U	#define __VTOR_PRESENT 1U
#define __FPU_PRESENT 1U	#define __NVIC_PRIO_BITS 2U
	#define __Vendor_SysTickConfig 0U

6. GD32E230不支持中断分组，所以固件库中没有“void nvic_priority_group_set(uint32_t nvic_prigroup)”函数，需要在固件库添加相应的内容。

表 6-2. 修改 GD32E23x 固件库中“gd32e23x_misc.h”头文件的内容

```
 /* priority group - define the pre-emption priority and the subpriority */
#define NVIC_PRIGROUP_PRE0_SUB4      ((uint32_t)0x00000700)
#define NVIC_PRIGROUP_PRE1_SUB3      ((uint32_t)0x00000600)
#define NVIC_PRIGROUP_PRE2_SUB2      ((uint32_t)0x00000500)
#define NVIC_PRIGROUP_PRE3_SUB1      ((uint32_t)0x00000400)
#define NVIC_PRIGROUP_PRE4_SUB0      ((uint32_t)0x00000300)

 /* set the priority group */
void nvic_priority_group_set(uint32_t nvic_prigroup);
```

表 6-3. 修改 GD32E23x 固件库中“gd32e23x_misc.c”头文件的内容

```
void nvic_priority_group_set(uint32_t nvic_prigroup)
{
    /* set the priority group value */
    SCB->AIRCR = NVIC_AIRCR_VECTKEY_MASK | nvic_prigroup;
}
```

7. GD32E230仅支持4级抢占优先级，不支持子优先级，GD32F3x0既支持抢占优先级也支持子优先级，需要在固件库里修改相应的内容。

表 6-4. 修改 GD32E23x 固件库中“gd32e23x_misc.h”头文件的内容

```
 /* enable NVIC request */
void nvic_irq_enable(uint8_t nvic_irq, uint8_t nvic_irq_pre_priority, uint8_t
nvic_irq_sub_priority);
```

表 6-5. 修改 GD32E23x 固件库中“gd32e23x_misc.c”文件的内容

```
void nvic_irq_enable(uint8_t nvic_irq,
                     uint8_t nvic_irq_pre_priority,
                     uint8_t nvic_irq_sub_priority)
{
    uint32_t temp_priority = 0x00U, temp_pre = 0x00U, temp_sub = 0x00U;
    /* use the priority group value to get the temp_pre and the temp_sub */
    switch ((SCB->AIRCR) & (uint32_t)0x700U) {
    case NVIC_PRIGROUP_PRE0_SUB4:
        temp_pre = 0U;
        temp_sub = 0x4U;
```

```
        break;
    case NVIC_PRIGROUP_PRE1_SUB3:
        temp_pre = 1U;
        temp_sub = 0x3U;
        break;
    case NVIC_PRIGROUP_PRE2_SUB2:
        temp_pre = 2U;
        temp_sub = 0x2U;
        break;
    case NVIC_PRIGROUP_PRE3_SUB1:
        temp_pre = 3U;
        temp_sub = 0x1U;
        break;
    case NVIC_PRIGROUP_PRE4_SUB0:
        temp_pre = 4U;
        temp_sub = 0x0U;
        break;
    default:
        nvic_priority_group_set(NVIC_PRIGROUP_PRE2_SUB2);
        temp_pre = 2U;
        temp_sub = 0x2U;
        break;
    }
    /* get the temp_priority to fill the NVIC->IP register */
    temp_priority = (uint32_t)nvic_irq_pre_priority << (0x4U - temp_pre);
    temp_priority |= nvic_irq_sub_priority &(0x0FU >> (0x4U - temp_sub));
    temp_priority = temp_priority << 0x04U;
    NVIC->IP[nvic_irq] = (uint8_t)temp_priority;
    /* enable the selected IRQ */
    NVIC->ISER[nvic_irq >> 0x05U] = (uint32_t)0x01U << (nvic_irq & (uint8_t)0x1FU);
}
```

8. GD32F3x0的Flash是零等待的，GD32E230系列需要配置Flash插入等待周期，因此可去掉插入等待周期的函数。

表 6-6. 去掉 GD32E23x 工程中插入等待周期的函数

```
FMC_WS = (FMC_WS & (~FMC_WS_WSCNT)) | WS_WSCNT_2;
```

9. GD32E230的Flash支持32位和64位编程，GD32F3x0的Flash支持32位字和半字编程。如过应用代码中使用了64位编程需要修改成32位字或半字编程，GD32E230固件库中需要添加半字编程的内容。

表 6-7. 在 GD32E23x 固件库中“gd32e23x_fmc.h”文件中添加半字编程的内容

```
/* FMC program a half word at the corresponding address */
fmc_state_enum fmc_halfword_program(uint32_t address, uint16_t data);
```

表 6-8. 在 GD32E23x 固件库中“gd32e23x_fmc.c”文件中添加半字编程的内容

```
fmc_state_enum fmc_halfword_program(uint32_t address, uint16_t data)
{
    fmc_state_enum fmc_state = fmc_ready_wait(FMC_TIMEOUT_COUNT);

    if(FMC_READY == fmc_state){
        /* set the PG bit to start program */
        FMC_CTL |= FMC_CTL_PG;
        REG16(address) = data;
        /* wait for the FMC ready */
        fmc_state = fmc_ready_wait(FMC_TIMEOUT_COUNT);
        /* reset the PG bit */
        FMC_CTL &= ~FMC_CTL_PG;
    }
    /* return the FMC state */
    return fmc_state;
}
```

10. 如项目中使用到TIMER5定时器，由于GD32F3x0剪裁掉此定时器（GD32F350保留），则相关TIMER5的代码需要更改为其它未使用的定时器。

11. 编译GD32E23x工程，至此，即可在GD32F3x0系列芯片中使用修改好的GD32E23x固件库进行软件开发。

7. GD32E23x 项目底层 Library 替换成 GD32F3x0 Library 步骤

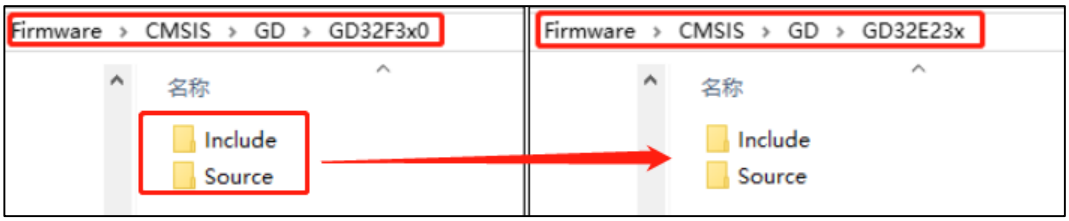
本章将使用“GD32E23x_Firmware_Library_V1.1.1\Template”里的工程以及“GD32F3x0_Firmware_Library_V2.2.0\Template”里的工程做示例。

1. 复制“GD32F3x0_Firmware_Library_V2.2.0\Firmware\CMSIS”下的.h文件替换到“GD32E23x_Firmware_Library_V1.1.1\Firmware\CMSIS”文件夹下。

图 7-1. 把 G32F3x0 固件库中 CMSIS 文件里的.h 文件复制到 GD32E23x 固件库中

2. 复制“GD32F3x0_Firmware_Library_ V2.2.0\Firmware\CMSIS\GD\GD32F3x0”里的Iclude和Source两个文件夹替换到“GD32E23x_Firmware_Library_V1.1.1\Firmware\CMSIS\GD\GD32E23x”文件夹下。

图 7-2. 把 GD32F3x0 固件库 CMSIS 下的 Include 与 Source 文件复制替换到 GD32E23x 固件库中去

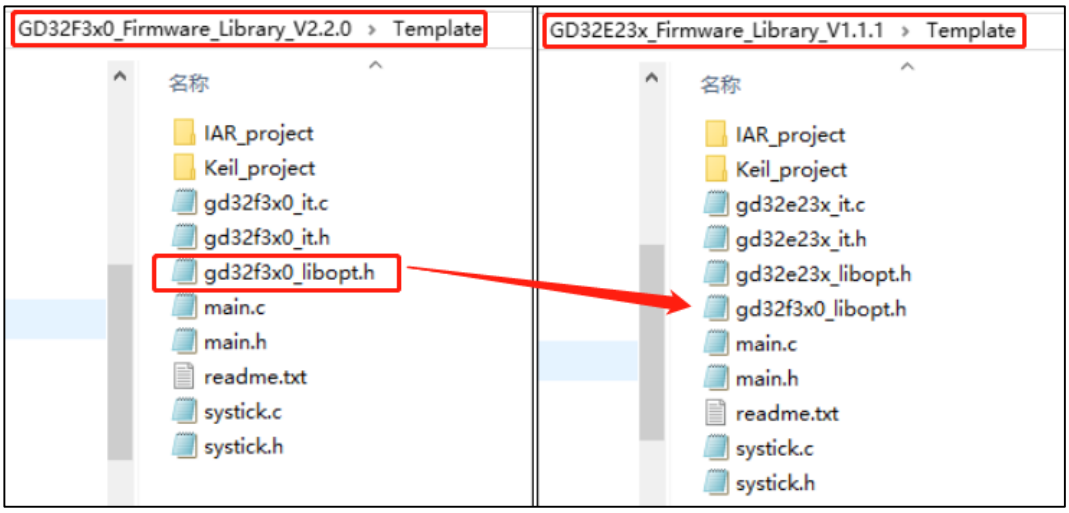


3. 复制“GD32F3x0_Firmware_Library_V2.2.0\Firmware\GD32F3x0_standard_peripheral”里的Iclude和Source两个文件夹替换到“GD32E23x_Firmware_Library_V1.1.1\Firmware\GD32E23x_standard_peripheral”文件夹下。

图 7-3. 把 GD32F3x0 固件库 standard_peripheral 下的文件复制替换到 GD32E23x 固件库中去

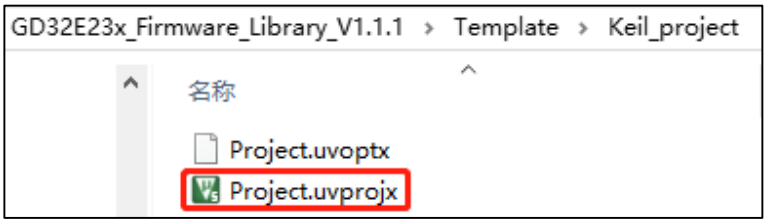
4. 复制“GD32F3x0_Firmware_Library_V2.2.0\Template”下的“gd32f3x0_libopt.h”到E230的相应文件夹路径下“GD32E23x_Firmware_Library_V1.1.1\Template”。

图 7-4. 把 GD32F3x0 固件库中的“gd32f3x0_libopt.h”文件复制到 GD32E23x 固件库中去



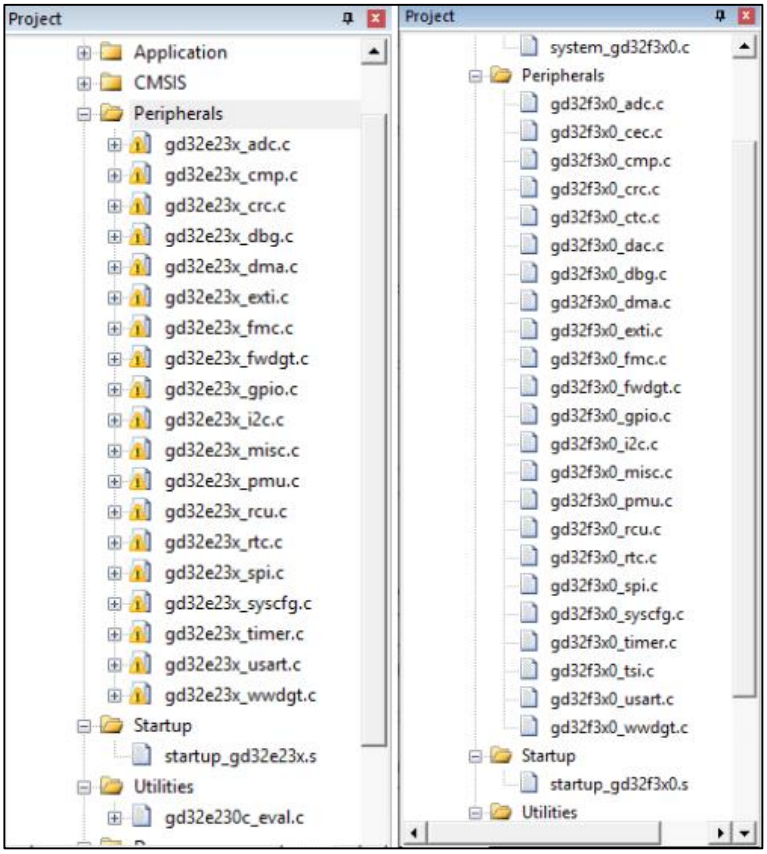
5. 至此，打开GD32E23x固件库中Template文件下的Keil工程。

图 7-5. 打开 GD32E23x 固件库中 Template 文件下的 Keil 工程



6. 工程界面上会看到左侧有黄色三角标记，表示原文件已经不存在，原因是前面的文件替换步骤已经把旧文件替换掉。此时只需要把黄色标记的文件全部移除，其中“gd32e230c_eval.c”为开发板配套配置，实际项目不使用，可以移除，然后再添加相应的GD32F3x0文件。

图 7-6. 移除黄色标记文件并添加新文件



7. 将项目应用的“main.c”“systick.c”文件里包含的头文件“#include "gd32e23x.h"”修改为“#include "gd32f3x0.h"”，并删除“#include "gd32e230c_eval.h"”。然后重新选择芯片型号，以及FLASH算法。

图 7-7. 修改“main.c” 、“systick.c”文件中的内容

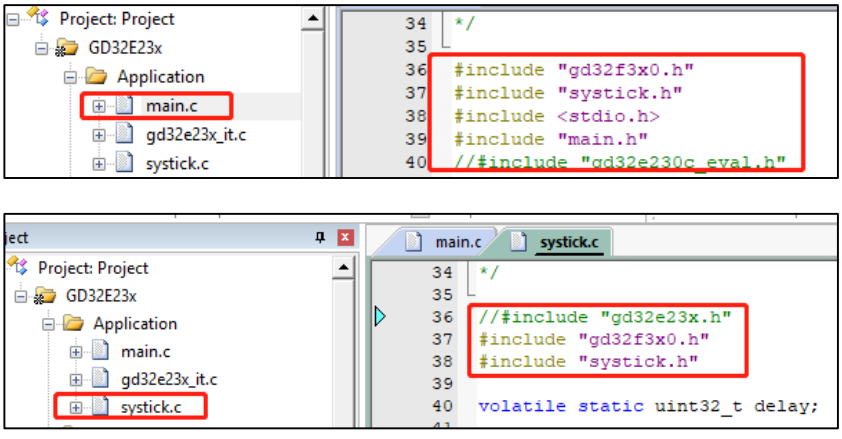


图 7-8. 重新选择 GD32F3x0 芯片型号

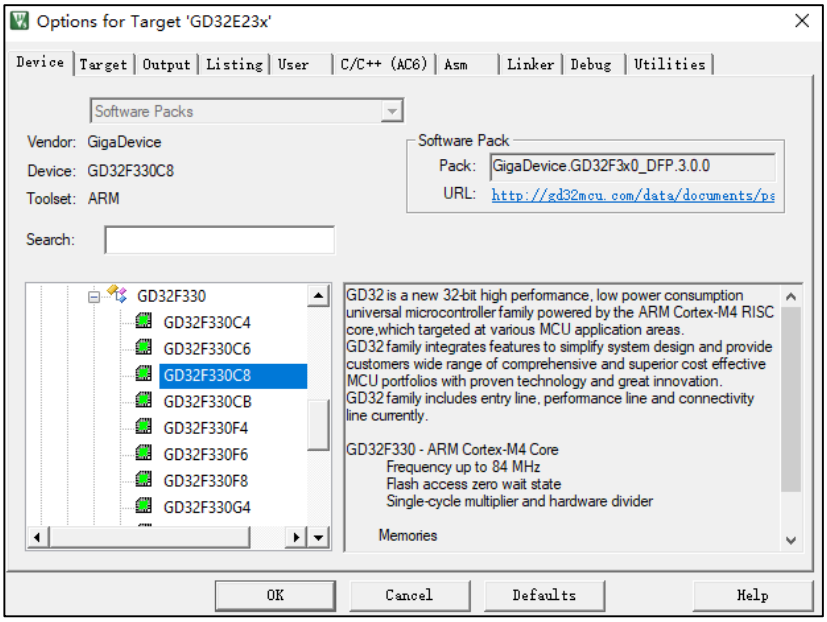
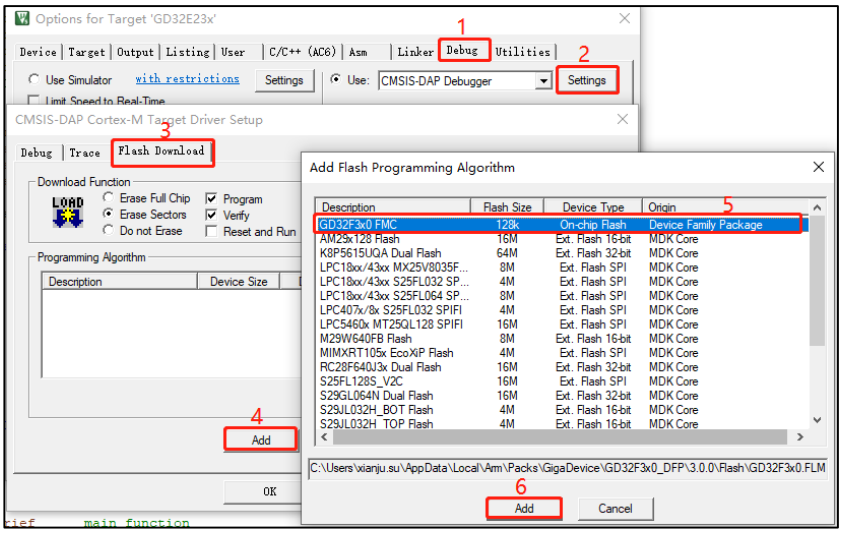


图 7-9. 选择 GD32F3x0 Flash 算法



8. 由于GD32E230不支持配置优先级组的位长度，在移植GD32F3x0库之后，当项目应用代码中有使用中断的配置时应用代码需要添加“void nvic_priority_group_set(uint32_t nvic_prigroup)”函数。

表 7-1. nvic_priority_group_set 函数

```
/* set the priority group */
void nvic_priority_group_set(uint32_t nvic_prigroup);
```

而且GD32E230仅支持4级抢占优先级，不支持子优先级，所以移植之后，中断使能函数需更改成*表7-2. nvic_irq_enable函数*所示的形式：

表 7-2. nvic_irq_enable 函数

```
/* set the priority group */
void nvic_irq_enable(uint8_t nvic_irq, uint8_t nvic_irq_pre_priority, uint8_t nvic_irq_sub_priority);
```

9. 如项目中使用到TIMER5定时器，由于GD32F3x0剪裁掉此定时器（GD32F350保留），则相关TIMER5的代码需要更改为其它未使用的定时器。
10. 编译项目，如有报错，则根据提示做修改，通常提示为项目上层应该逻辑代码的.C文件里包含的#include "gd32e23x.h"没有修改为#include "gd32f3x0.h"，根据提示修改即可。至此，项目移植成功，可进行GD32F3x0系列MCU的开发。

8. 版本历史

表 8-1. 版本历史

版本号.	说明	日期
1.0	首次发布	2022 年 3 月 15 日

Important Notice

This document is the property of GigaDevice Semiconductor Inc. and its subsidiaries (the "Company"). This document, including any product of the Company described in this document (the “Product”), is owned by the Company under the intellectual property laws and treaties of the People’s Republic of China and other jurisdictions worldwide. The Company reserves all rights under such laws and treaties and does not grant any license under its patents, copyrights, trademarks, or other intellectual property rights. The names and brands of third party referred thereto (if any) are the property of their respective owner and referred to for identification purposes only.

The Company makes no warranty of any kind, express or implied, with regard to this document or any Product, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. The Company does not assume any liability arising out of the application or use of any Product described in this document. Any information provided in this document is provided only for reference purposes. It is the responsibility of the user of this document to properly design, program, and test the functionality and safety of any application made of this information and any resulting product. Except for customized products which has been expressly identified in the applicable agreement, the Products are designed, developed, and/or manufactured for ordinary business, industrial, personal, and/or household applications only. The Products are not designed, intended, or authorized for use as components in systems designed or intended for the operation of weapons, weapons systems, nuclear installations, atomic energy control instruments, combustion control instruments, airplane or spaceship instruments, transportation instruments, traffic signal instruments, life-support devices or systems, other medical devices or systems (including resuscitation equipment and surgical implants), pollution control or hazardous substances management, or other uses where the failure of the device or Product could cause personal injury, death, property or environmental damage ("Unintended Uses"). Customers shall take any and all actions to ensure using and selling the Products in accordance with the applicable laws and regulations. The Company is not liable, in whole or in part, and customers shall and hereby do release the Company as well as it’s suppliers and/or distributors from any claim, damage, or other liability arising from or related to all Unintended Uses of the Products. Customers shall indemnify and hold the Company as well as it’s suppliers and/or distributors harmless from and against all claims, costs, damages, and other liabilities, including claims for personal injury or death, arising from or related to any Unintended Uses of the Products.

Information in this document is provided solely in connection with the Products. The Company reserves the right to make changes, corrections, modifications or improvements to this document and Products and services described herein at any time, without notice.