

ACM32F4/FP4/F3 系列芯片开发快速上手

目录

1. 开发板资源概述.....	3
1.1. 开发板板载资源	3
2. 板载硬件介绍	5
2.1. ACM32F403RET7.....	5
2.2. ACM32FP401REU7.....	6
2.3. ACM32F303RCT7	8
2.4. CMSIS-DAP 调试下载接口/虚拟 USB 转串口	9
2.5. 板载 LED	9
2.6. 按键和拨码	10
2.7. 电源输入输出	11
2.8. 板载时钟	12
2.9. IO 口.....	13
3. 开发环境搭建	14
3.1. 设备驱动安装	14
3.2. 仿真环境搭建	16
3.2.1. MCU 选择	16
3.2.2. 下载口选择	17
3.2.3. 下载算法选择	18
3.2.4. 调试模式配置	19
3.2.5. BOOT 引脚配置	19
3.3. KEIL 工程移植.....	21

图目录

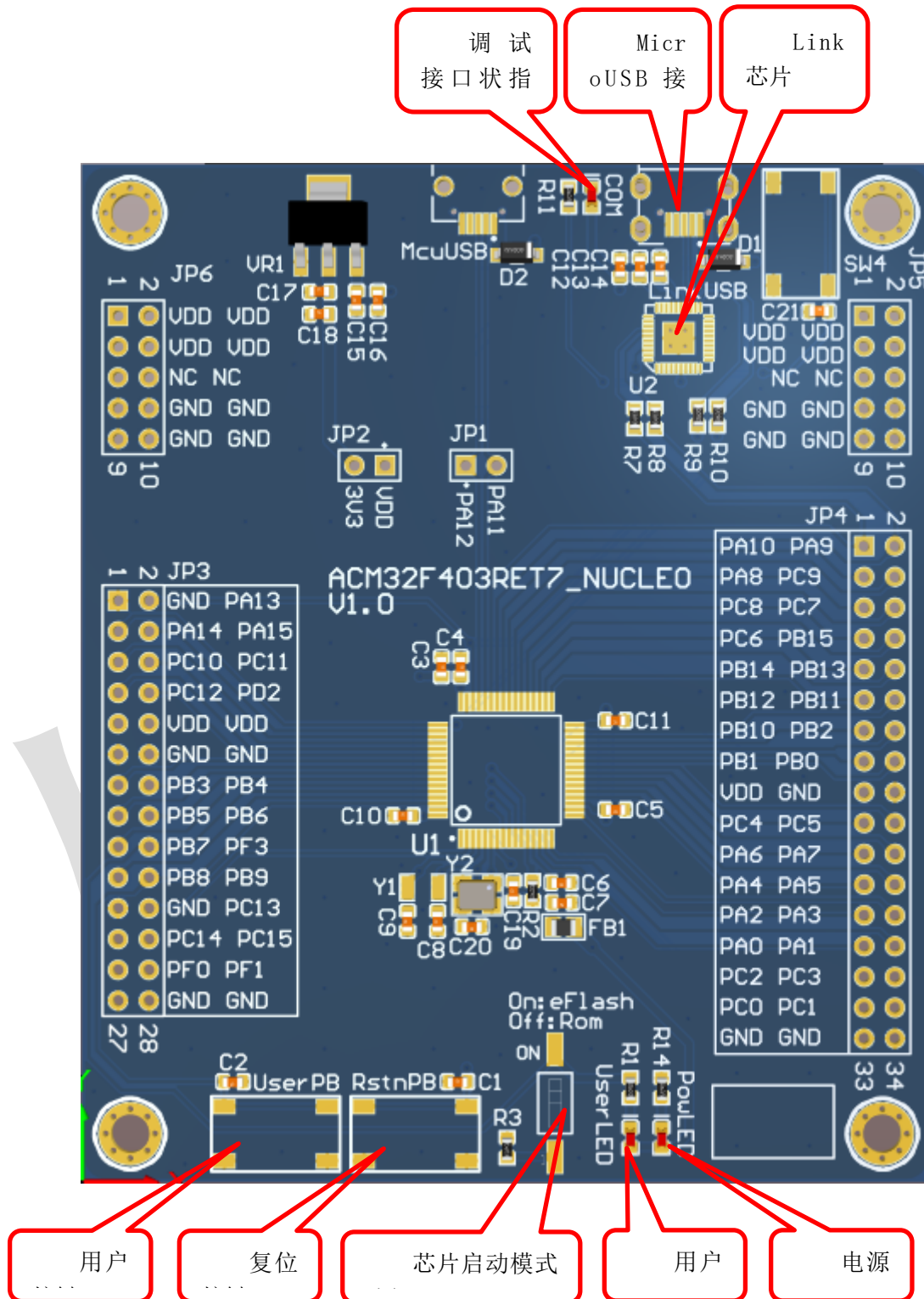
图 2-1 MCU (1)	5
图 2-2 MCU (2)	6
图 2-3 MCU (1)	7
图 2-4 MCU (2)	7
图 2-5 MCU (1)	8
图 2-6 MCU (2)	8
图 2-7 下载调试接口芯片	9
图 2-8 板载 LED(1).....	10
图 2-9 板载 LED(2).....	10
图 2-10 用户按键	11
图 2-11 复位按键	11
图 2-12 拨码开关	11
图 2-13 电源	12

图 2-14	片外时钟	12
图 2-15	扩展 IO 接口	13
图 3-1	选择驱动程序	14
图 3-2	选择驱动程序	15
图 3-3	选择驱动程序	15
图 3-4	选择驱动程序	16
图 3-5	选择驱动程序	16
图 3-6	MCU 选择	17
图 3-7	调试器类型	17
图 3-8	调试器连接情况	18
图 3-9	添加烧录插件	18
图 3-10	调试前先下载程序	19
图 3-11	航芯启动模式选择	20
图 3-13	Target 配置	23
图 3-14	User 页面	23
图 3-15	编译链接选项	24
图 3-16	链接加载	24
图 3-17	系统配置修改	24
图 3-18	STM32F103 串口初始化	25
图 3-19	ACM32F4/F3/FP4 的串口初始化	25
图 3-20	STM32F103 去除的部分	26

1. 开发板资源概述

1.1. 开发板板载资源

- MCU: ACM32F403RET7, LQFP64(10mm*10mm), FLASH: 512KB, SRAM: 192KB。
ACM32FP401REU7, QFN64(8mm*8mm), FLASH: 512KB, SRAM: 192KB。
ACM32F303RCT7, LQFP64(10mm*10mm), FLASH: 256KB, SRAM: 64KB。
- 1个电源指示灯, 2个状态指示灯。
- 1个电源供应/接入口。
- 1个启动模式选择拨码开关, 选择芯片启动模式。
- 1个系统复位按钮, 用于复位MCU。
- 1个用户功能按钮。
- 提供了CMSIS-DAP方式下载、调试, USB虚拟串口打印功能。
- 所有IO口全部引出, 包括晶振占用的IO口。



2. 板载硬件介绍

2.1. ACM32F403RET7

核心芯片(U1), 型号 ACM32F403RET7, 基于 ARMv8-M 架构, 支持 Cortex-M33 和 Cortex-M4F 指令集。最高支持 180MHz 系统工作频率, 支持浮点运算和 DSP。

该芯片内嵌高达 512KB Flash, 192KB Sram。

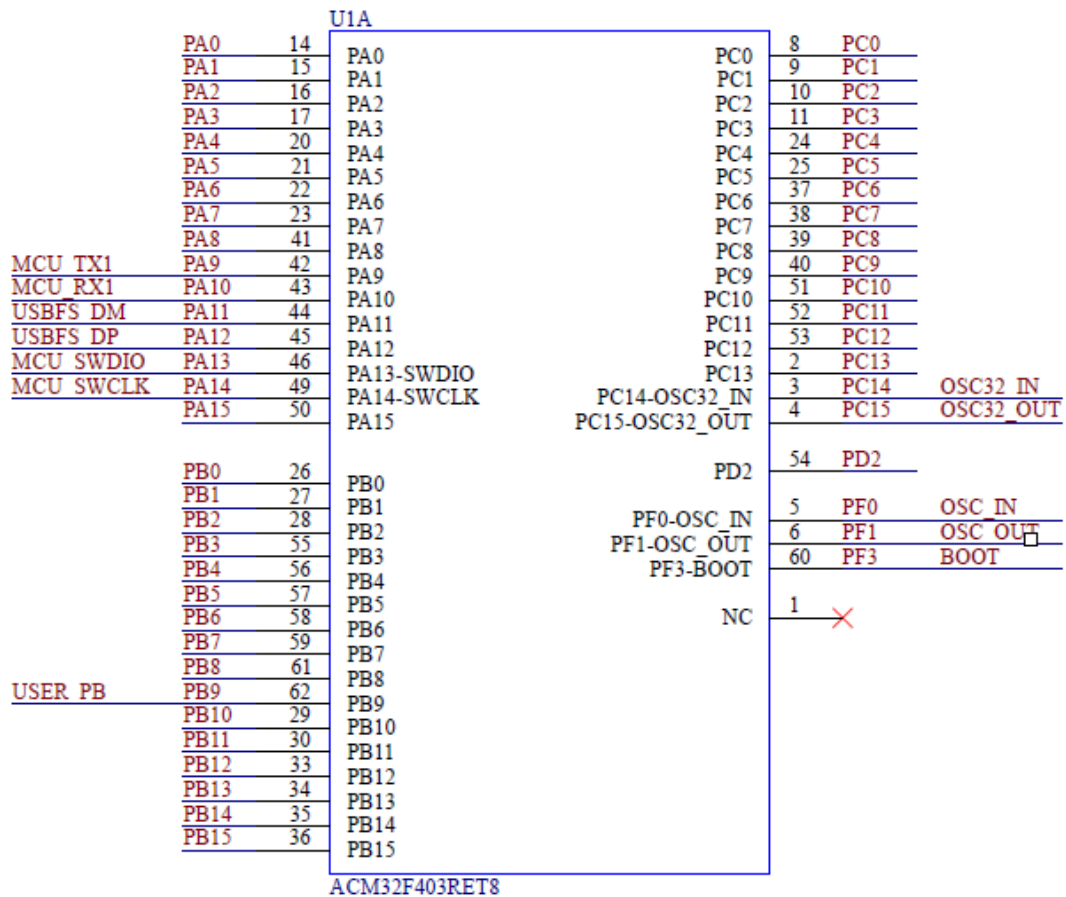


图 2-1 MCU (1)

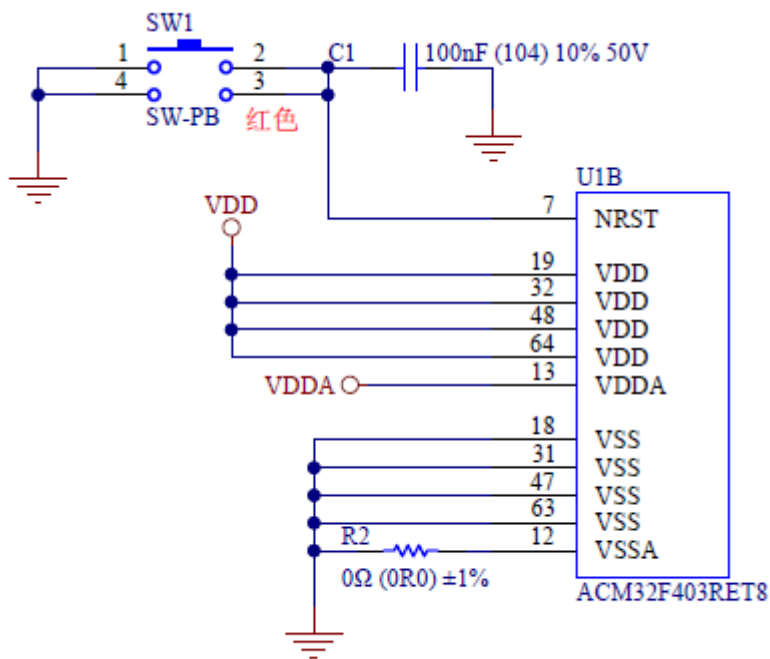


图 2-2 MCU (2)

2.2. ACM32FP401REU7

核心芯片(U1)，型号 ACM32FP401REU7，基于 ARMv8-M 架构，支持 Cortex-M33 和 Cortex-M4F 指令集。最高支持 180MHz 系统工作频率，支持浮点运算和 DSP。

该芯片内嵌高达 512KB Flash，192KB Sram。



2.3. ACM32F303RCT7

核心芯片(U1)，型号 ACM32F303RCT7，基于 ARMv8-M 架构，支持 Cortex-M33 和 Cortex-M4F 指令集。最高支持 120MHz 系统工作频率，支持浮点运算和 DSP。

该芯片内嵌高达 256KB Flash，64KB SRAM。

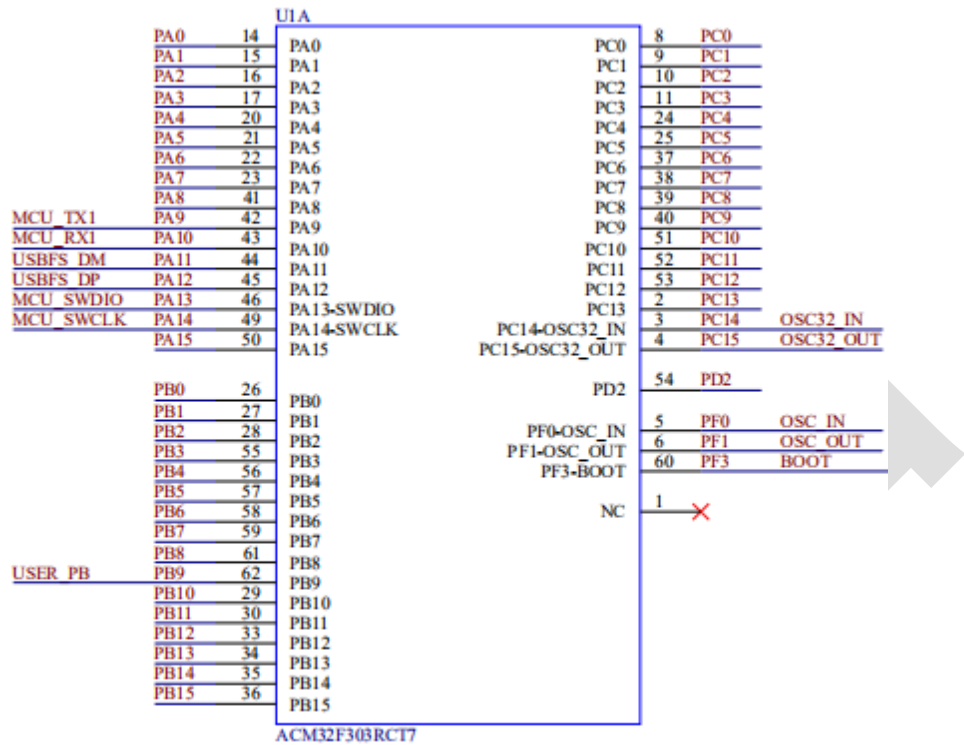


图 2-5 MCU (1)

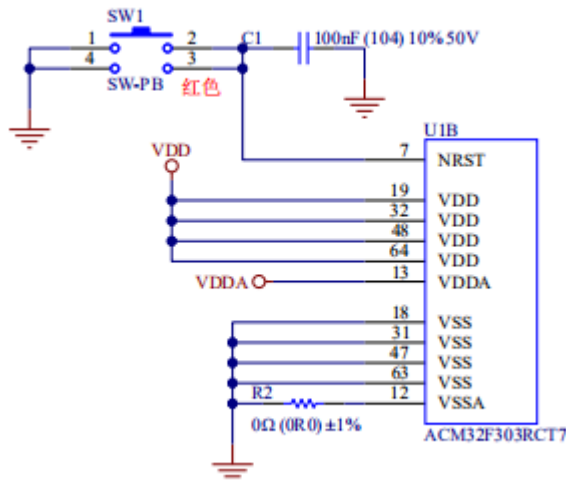


图 2-6 MCU (2)

2.4. CMSIS-DAP 调试下载接口/虚拟 USB 转串口

采用 ACL16 芯片作为下载调试/串口打印芯片，当用户通过 USB 线将开发板与 PC 机连接后，调试/下载程序时在 KEIL 中选择 CMSIS-DAP Debugger 模式，并且在设备管理器中可以找到航芯虚拟串口端口。(后续会具体介绍开发环境搭建)

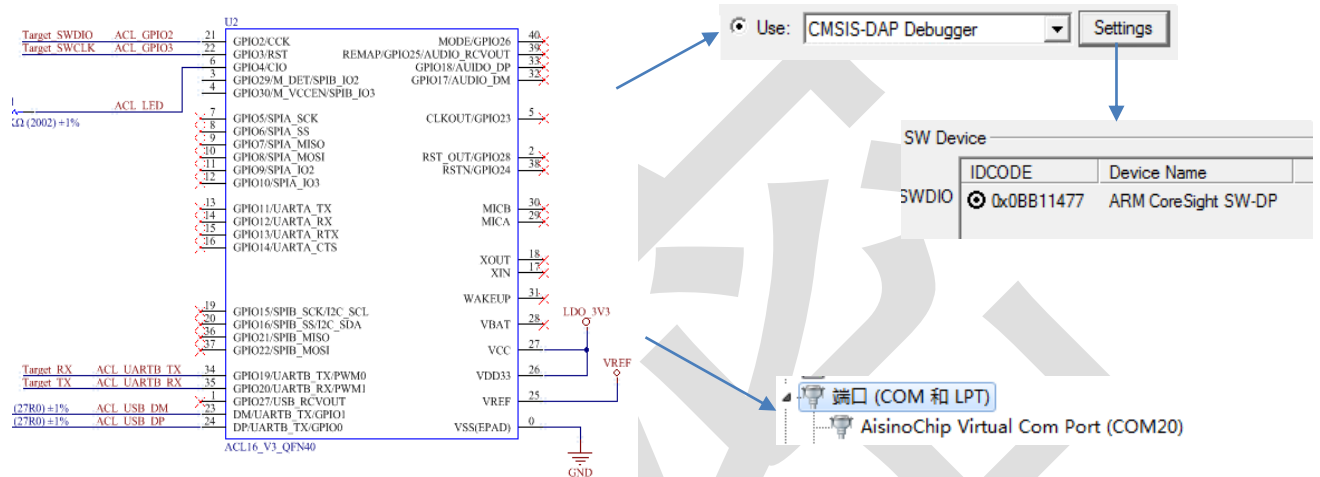


图 2-7 下载调试接口芯片

2.5. 板载 LED

用户程序灯(UserLED)，绿色，辅助用户查询/调试程序。

电源指示灯(PowLED)，绿色，MCU 上电时亮，下电时熄灭。

调试接口指示灯(COM), 红色, 反映芯片的下载/调试状态。

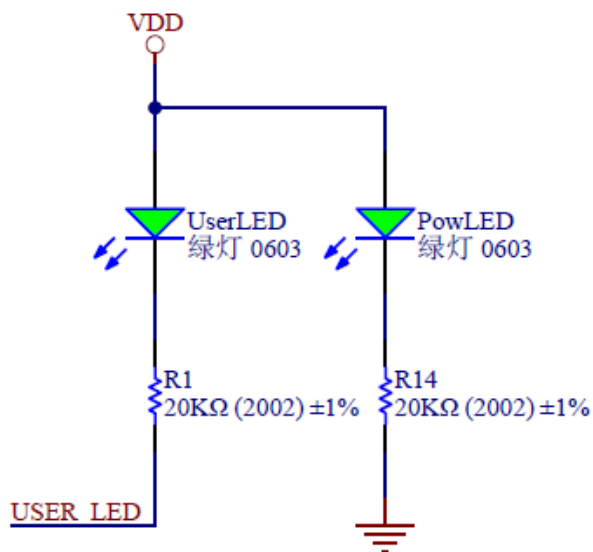


图 2-8 板载 LED(1)

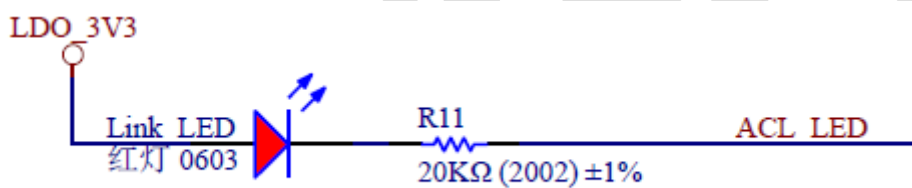


图 2-9 板载 LED(2)

2.6. 按键和拨码

MCU 复位按键(RstnPB)红色, 用于复位 ACM32F4/FP4/F3 芯片, 按下按键芯片复位。

用户按键(UserPB)黑色, 人机交互按键。

芯片启动模式选择拨码, 拨码打到 On 按复位按键, 芯片会运行用户代码 (eFlash); 拨码打到 Off 按复位按键, 芯片只会运行自有 Boot 程序(ROM 中)。

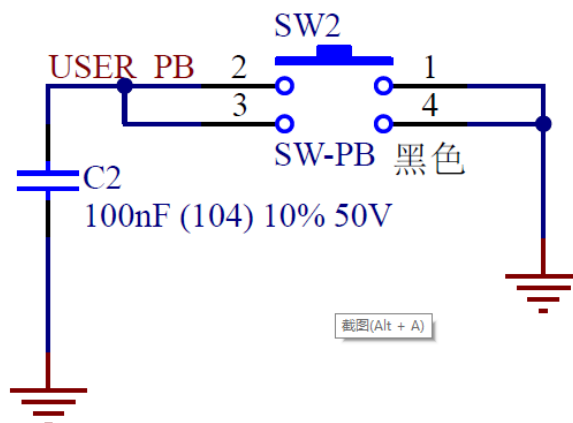


图 2-10 用户按键

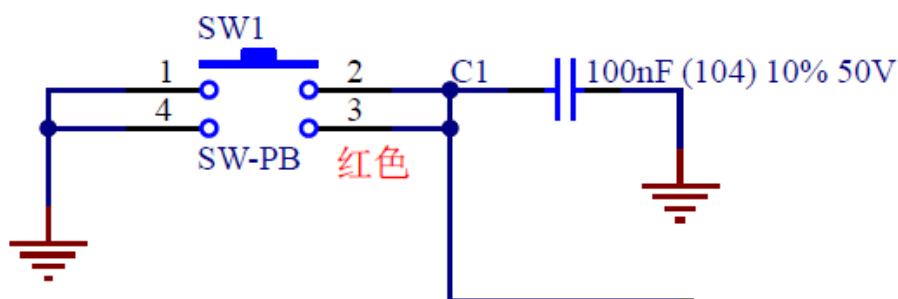


图 2-11 复位按键

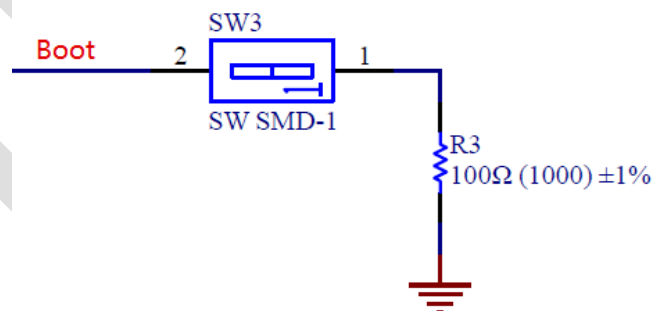
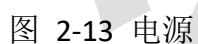


图 2-12 拨码开关

2.7. 电源输入输出

通常使用 USB 供电，板载 LDO 将 5V(USB_Vbus)转成 3.3V，JP2 默认用跳帽短接，可以通过排针 JP7 给外部提供 3.3V 电源。当不使用 USB 供电时，可以断开 JP2，通过排针 JP5 从外部接入电源给板载 MCU 供电。MCU 的 VDD 输入电压范围是 1.7V~3.63V。



板上默认有高速 12MHz 无源晶振, 低速 32.768kHz 无源晶振提供时钟提供给 ACM32 芯片。



2.9. IO 口

芯片所有 IO 引出到排针 JP3(32pin)、JP4(34pin)上。并且 JP3、JP4、JP5、JP6 上都提供电源/地，可以灵活给接插的扩展子板输入/输出电源。

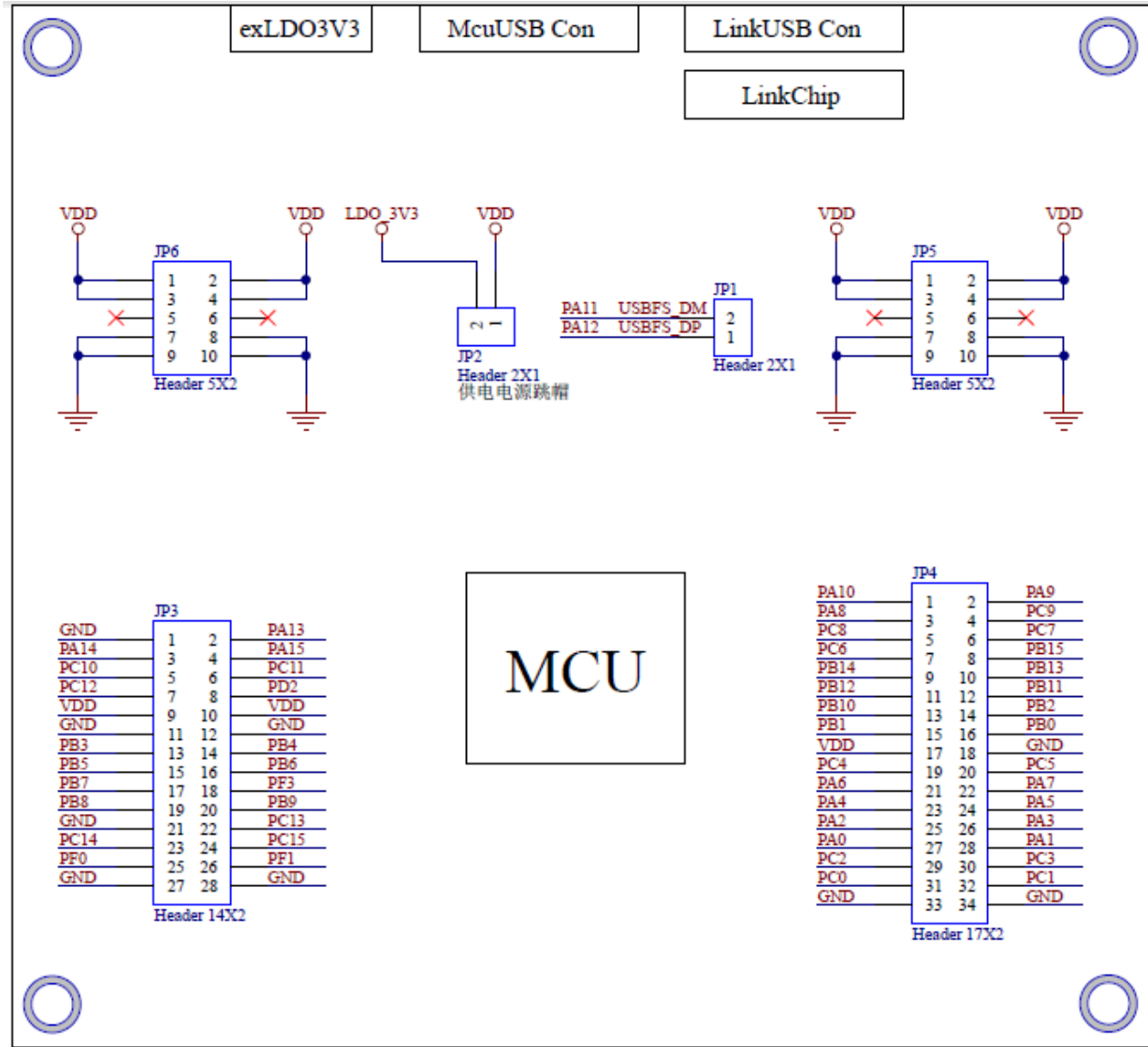
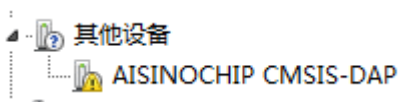


图 2-15 扩展 IO 接口

3. 开发环境搭建

3.1. 设备驱动安装

用户第一次将开发板与 PC 机的 USB 端口相连，Win7 系统用户需要安装 Link 芯片的驱动 (Win10 系统用户不需要)。



选择此设备，右键更新驱动，直接点下面的->从驱动列表中选择(COM 类)

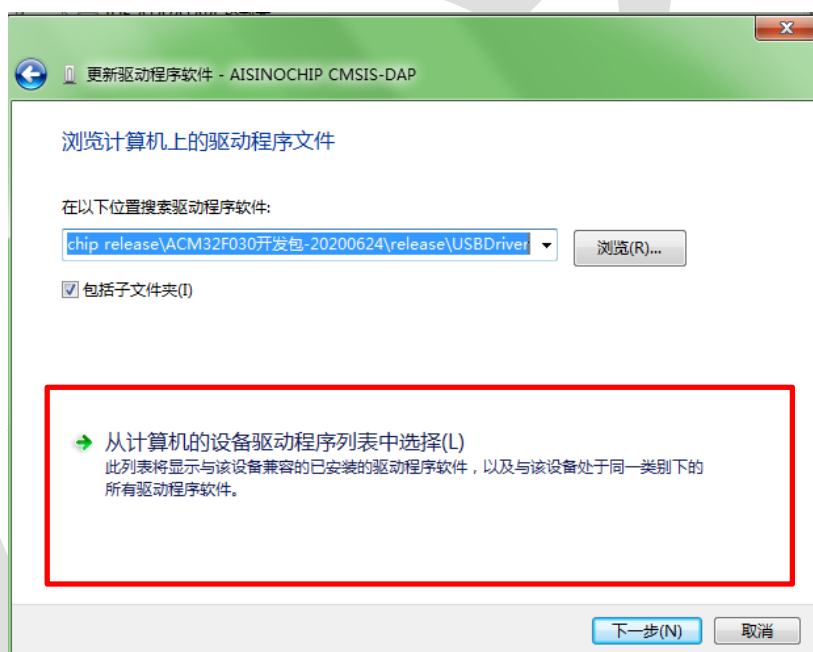


图 3-1 选择驱动程序

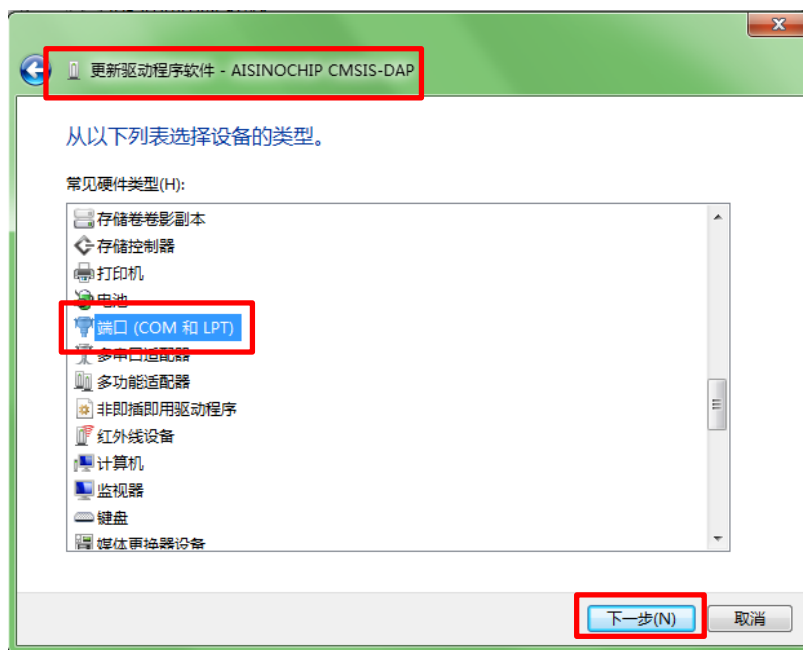


图 3-2 选择驱动程序

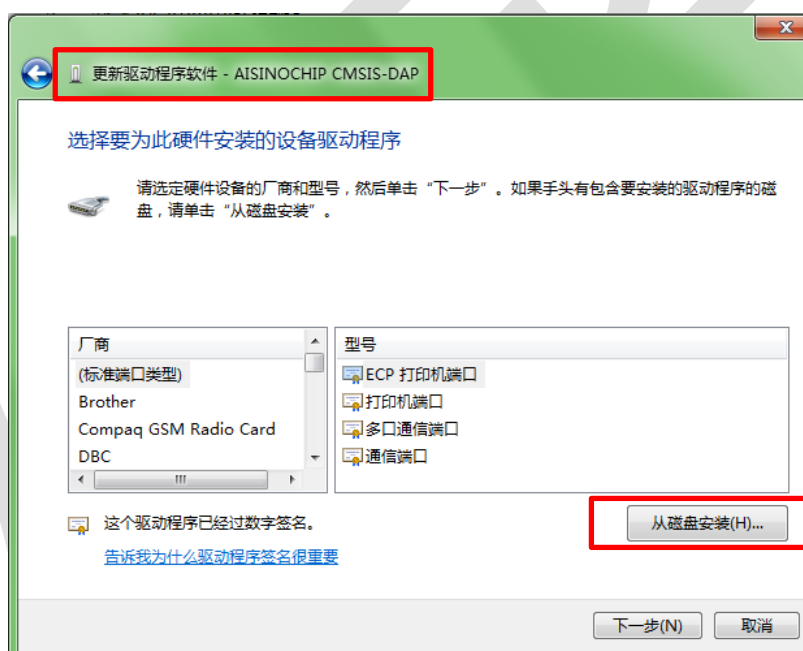


图 3-3 选择驱动程序

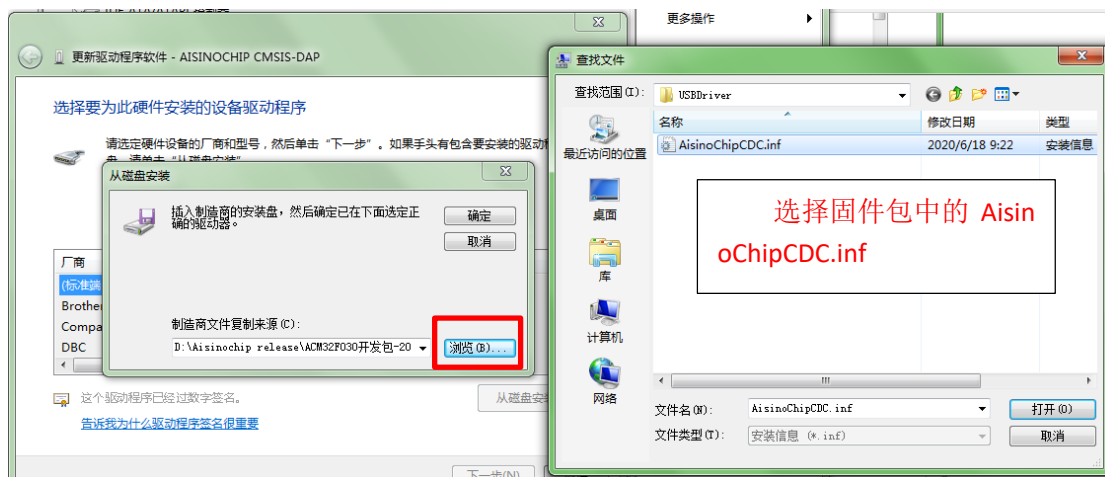


图 3-4 选择驱动程序

忽略警告提示，继续安装，安装完毕。后续调试程序可以选择此 COM 口获得打印信息。



图 3-5 选择驱动程序

3.2. 仿真环境搭建

3.2.1. MCU 选择

安装航芯开发包里的 pack 包“Aisinochip.ACM32F4/FP4/F3.1.0.0.pack”，安装完成，选择所使用的芯片具体型号，如图所示：

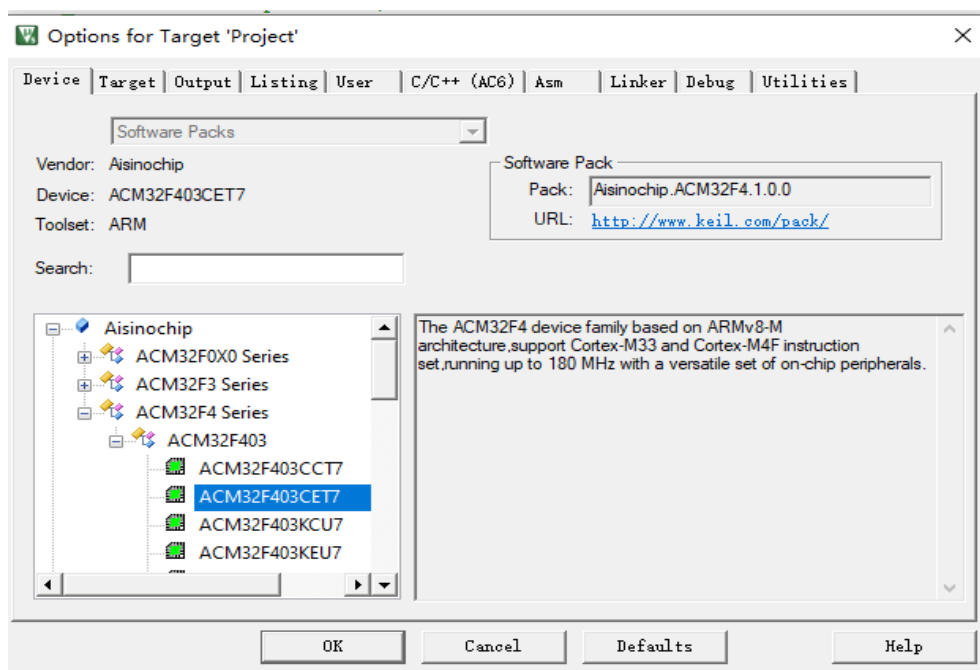


图 3-6 MCU 选择

3.2.2. 下载口选择

连接调试器：如图所示，切换到 Debug 页面后在下拉框中选择 CMSIS-DAP ARMv8-M Debugger，然后点击 Settings 按钮，显示图 3-7 的调试器连接情况。

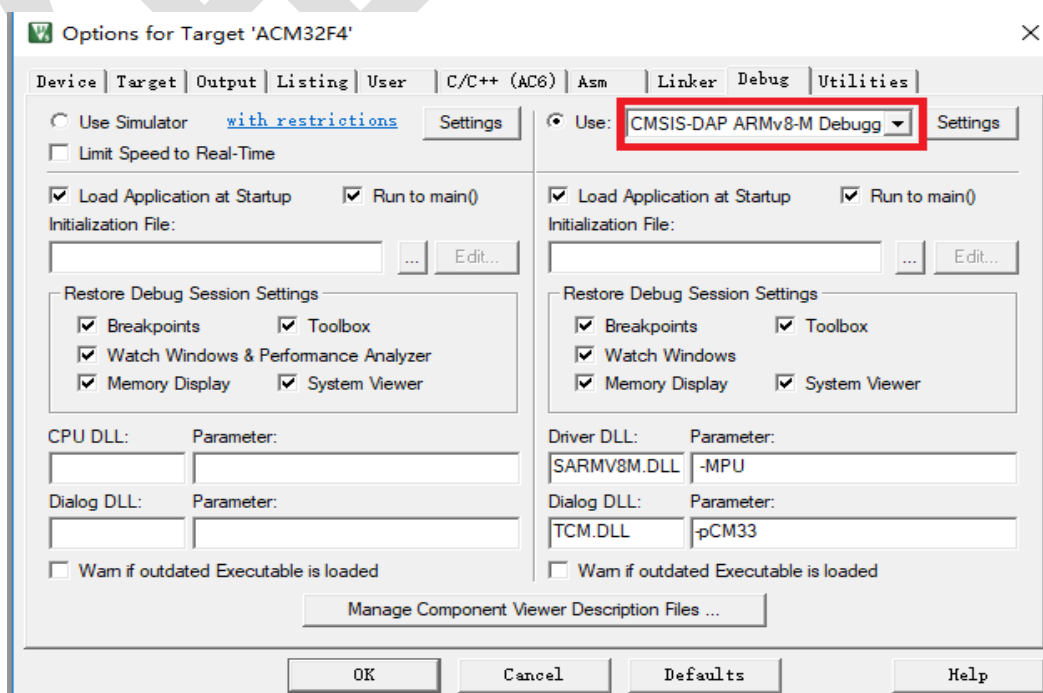


图 3-7 调试器类型

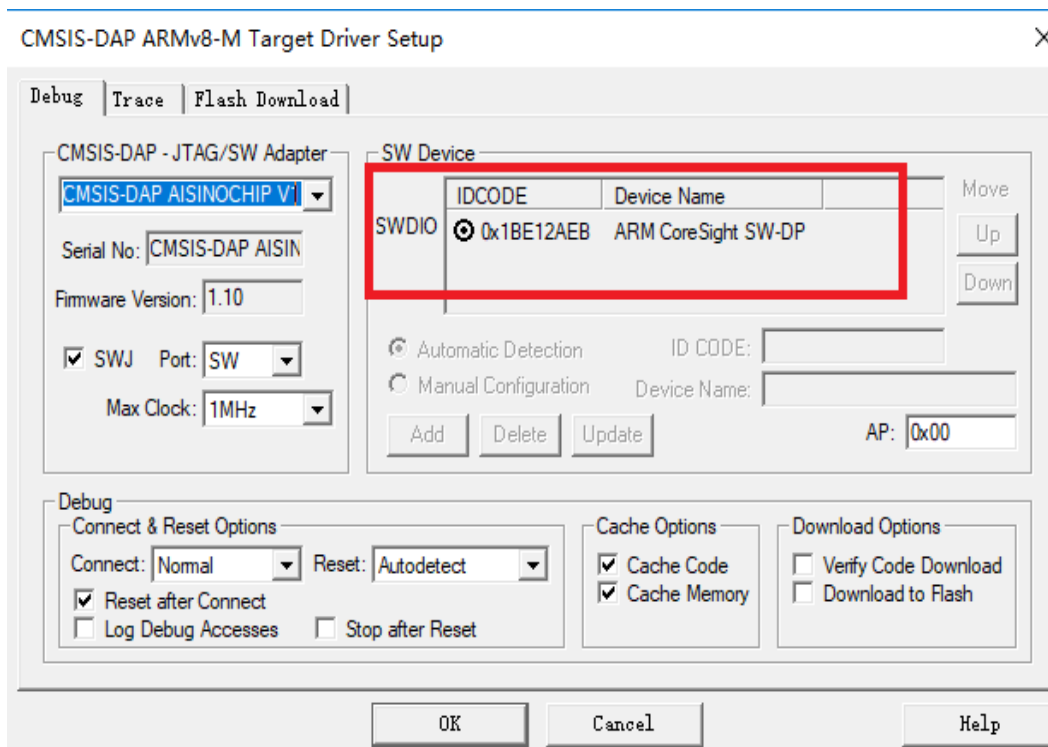


图 3-8 调试器连接情况

3.2.3. 下载算法选择

若首次使用，需要将开发包中的 ACM32F403_elflash.flm/ACM32FP4_elflash.flm /ACM32F3_elflash.flm 拷贝到 C:\KEIL\ARM\Flash 目录下。然后在图的基础上点击“Flash Download”按钮进入插件选择界面，选择 ACM32F4/FP4/F3 插件，图以 ACM32F4 为例。

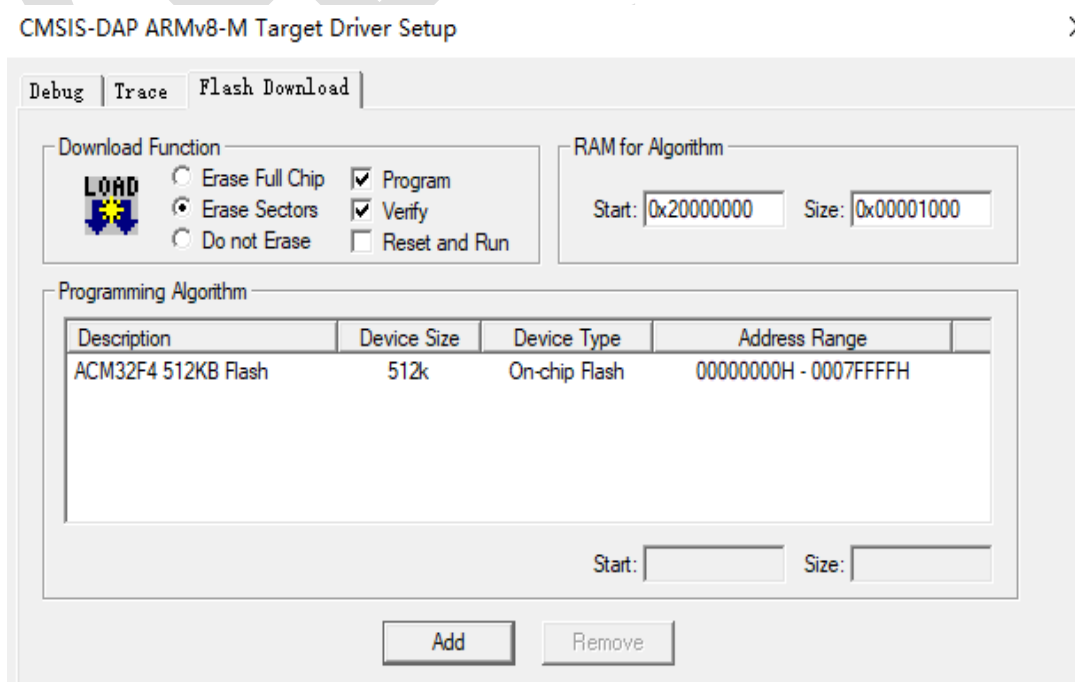


图 3-9 添加烧录插件

3.2.4. 调试模式配置

按如图所示的配置，就可以在调试前先下载程序到 eFlash 中然后开始调试程序。

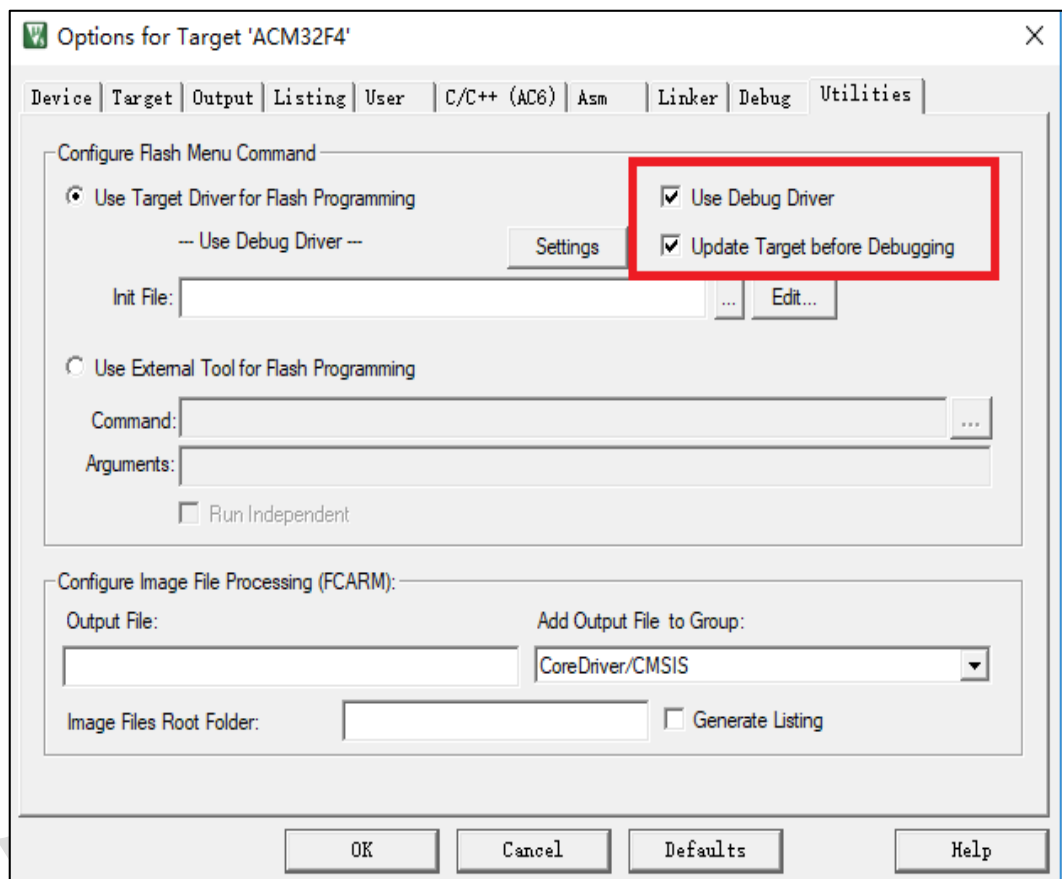


图 3-10 调试前先下载程序

3.2.5. BOOT 引脚配置

ACM32F4/FP4/F3 两种启动模式：ROM 启动和 eFlash 启动。系统上电时，芯片会读取安全序列字段和系统寄存器 WMR 的 BootMode 标志位，决定是将 eFlash 还是将 ROM 映射到 0x0 起始逻辑地址。BootMode 标志位由上电时 BOOT 引脚(PF3)的高低电平决定。图描述了芯片启动模式选择过程。

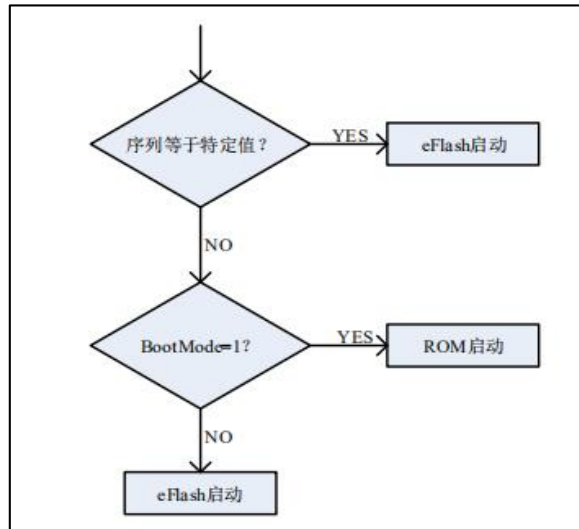


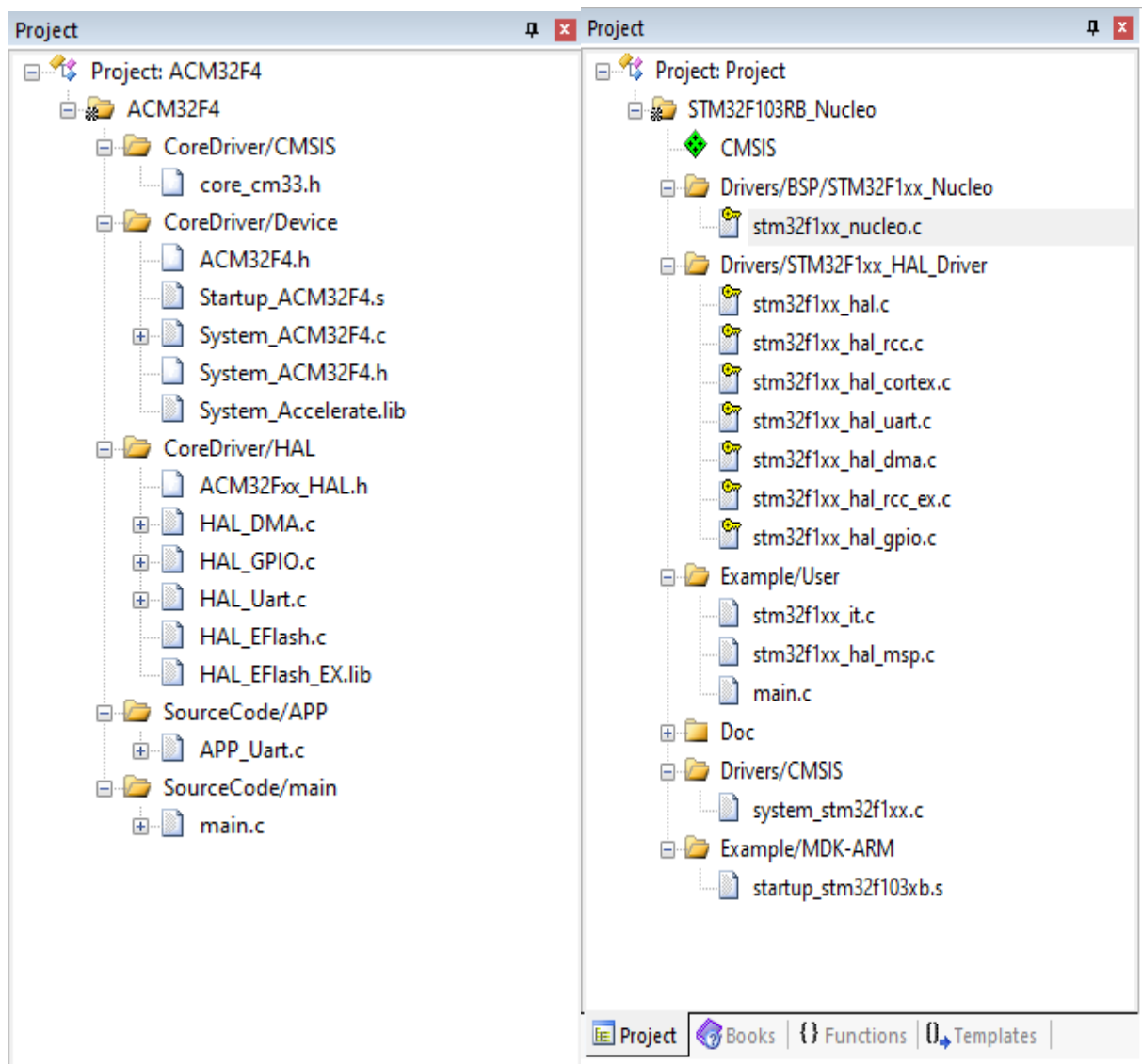
图 3-11 航芯启动模式选择

故需要在不写入安全序列的情况下，将 **BOOT** 引脚(PF3)拉低，使 ACM32F4/FP4/F3 从 eflash 启动。

3.3. KEIL 工程移植

本章节描述如何从 STM32F103 的 UART Keil 工程移植到航芯的 ACM32F4/F3/FP4 芯片上。首先，ACM32F4/F3/FP4 芯片的 Keil 版本需要 **V 5.29 及更高版本**。

其次，对比一下 STM32F103 和 ACM32F4/F3/FP4 KEIL 工程的异同，本章节中以 ACM32F4 为例，ACM32FP4/F3 同理：



再次，我们看看 STM32F103 的 UART 工程和航芯工程的对应关系以及移植方法。

- 1)、STM32F103 的 Drivers/HAL_Driver 对应航芯工程的 CoreDriver/HAL，里面放着的是芯片厂商提供的外设驱动库。
- 2)、STM32F103 中 CMSIS 对应航芯工程 CoreDriver/CMSIS 的 core_cm33.h 服务，此文件是 ARM 官方提供的。
- 3)、STM32F103 中 Example/MDK 下的.s 启动文件对应航芯工程中 CoreDriver/Device 中的.s

文件，主要是中断向量表。STM32F103 中的 `system_stm32f1xx.c` 对应的是航芯工程中的 `system_acm32f4.c`，此文件是和芯片系统相关的，`System_ACM32F4.c` 中主要是时钟和复位功能。

4)、STM32F103 中的 User 下的 `main.c` 对应航芯工程中的 `SourceCode/main.c`

和 `SourceCode/app_uart.c`，是应用层的逻辑，`app_uart.c` 实现的是串口的收发，将收到的数据全部发回。

5)、STM32F103 中的 `stm32f10x_it.c` 对应 ACM32F4 中 `SourceCode/app_uart.c` 中的 `UART1_IRQHandler`，是中断处理函数。

6)、`System_Accelerate.lib` 是 ACM32F4 的加速库，STM32F103 无此库。

如上所述，航芯的 demo 分层清晰：应用层、硬件抽象层、系统层、ARM 公共文件。应用层的 `main.c` 文件中主要是系统时钟初始化，`systick` 初始化，串口打印初始；而 `app_uart.c` 是 `uart demo` 应用层的代码，是业务相关、应用相关的逻辑处理。HAL 下的文件是硬件抽象层，如 `I2C/SPI/UART` 等的 HAL 文件都包含了外设初始化和通信服务，所有和外设通信、数据交互的服务都在 `HAL_xxx.c` 文件中提供好了。系统层是芯片系统相关的代码，如时钟和复位。ARM 公共文件则是处理器内核相关的代码，是 ARM 提供的标准化接口，如 `NVIC` 接口，`systick` 初始化等。

串口 Demo 移植介绍：

打开 ACM32F4/F3/FP4 UART demo 中的 MDK_Project 文件夹下的工程文件。

1) MCU 选择，Device 选择对应的 ACM32F4/FP4/F3。

2) 如图 3-13 所示，`IROM1` 起始地址填 0，是 `eflash` 的起始地址，大小根据产品型号的参数填写（见数据手册），最大为 512KB 即 `0x80000`。`ISRAM1` 起始地址填 `0x20000000`，为 `SRAM` 的起始地址，大小根据产品型号的参数填写，最大为 192KB 即 `0x30000`。编译器选择 **version 6**。

3) 如图 3-14 所示，可以生产 `.bin` 文件。

4) 如图 3-15 所示，将 `One ELF Section Per Function` 的 ☒ 打上，链接时未被调用到的函数不会被链接进去，能大大减少生成镜像文件的大小。编译优化选项见图 3-16，链接配置见图 3-14。

Device | Target | Output | Listing | User | C/C++ (AC6) | Asm | Linker | Debug | Utilities

ARM ARMCM33_DSP_FP

Xtal (MHz): 12.0

Operating system: None

System Viewer File:

☐ Use Custom File

Code Generation
ARM Compiler: Use default compiler version 6

☒ Use MicroLIB ☐ Big Endian

Floating Point Hardware: Single Precision

Read/Only Memory Areas

default	off-chip	Start	Size	Startup
<input type="checkbox"/>	ROM1:			<input type="radio"/>
<input type="checkbox"/>	ROM2:			<input type="radio"/>
<input type="checkbox"/>	ROM3:			<input type="radio"/>
	on-chip			
<input checked="" type="checkbox"/>	IROM1:	0x0	0x80000	<input checked="" type="radio"/>
<input type="checkbox"/>	IROM2:	0x200000	0x200000	<input type="radio"/>

Read/Write Memory Areas

default	off-chip	Start	Size	NoInit
<input type="checkbox"/>	RAM1:			<input type="checkbox"/>
<input type="checkbox"/>	RAM2:			<input type="checkbox"/>
<input type="checkbox"/>	RAM3:			<input type="checkbox"/>
	on-chip			
<input checked="" type="checkbox"/>	IRAM1:	0x20000000	0x30000	<input type="checkbox"/>
<input type="checkbox"/>	IRAM2:	0x20200000	0x20000	<input type="checkbox"/>

图 3-12 Target 配置

Device | Target | Output | Listing | User | C/C++ (AC6) | Asm | Linker | Debug | Utilities

Command Items	User Command	...	Stop on Exi...	S...
[-] Before Compile C/C++ File				
<input type="checkbox"/> Run #1		[Folder Icon]	Not Specified	<input type="checkbox"/>
<input type="checkbox"/> Run #2		[Folder Icon]	Not Specified	<input type="checkbox"/>
[-] Before Build/Rebuild				
<input type="checkbox"/> Run #1		[Folder Icon]	Not Specified	<input type="checkbox"/>
<input type="checkbox"/> Run #2		[Folder Icon]	Not Specified	<input type="checkbox"/>
[-] After Build/Rebuild				
<input checked="" type="checkbox"/> Run #1	fromelf.exe --bin --output ./Out_Files/ACM32F4...	[Folder Icon]	Not Specified	<input type="checkbox"/>
<input type="checkbox"/> Run #2		[Folder Icon]	Not Specified	<input type="checkbox"/>

图 3-13 User 页面

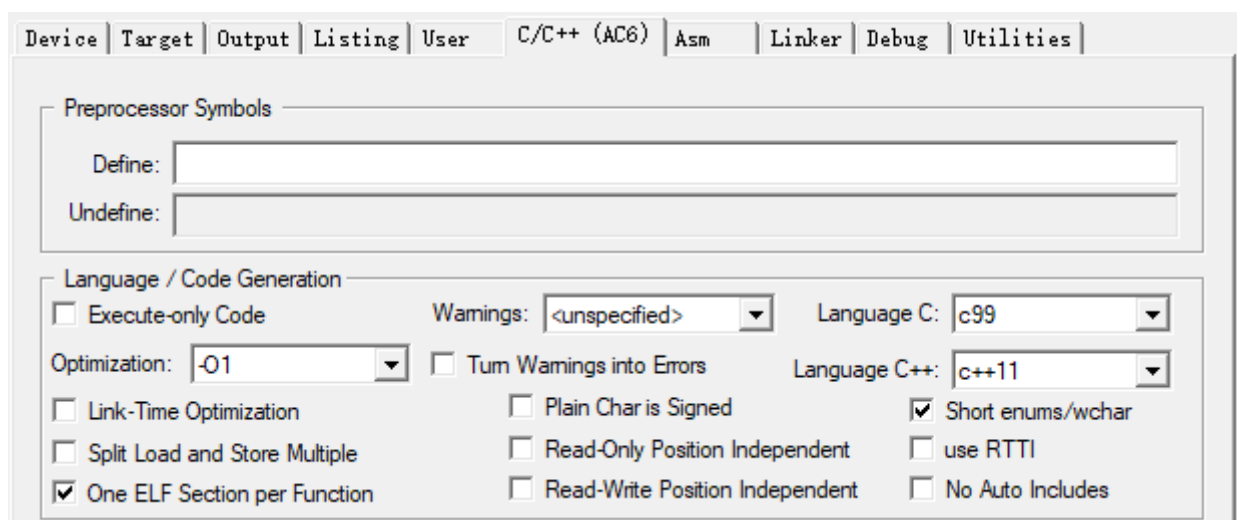


图 3-14 编译链接选项

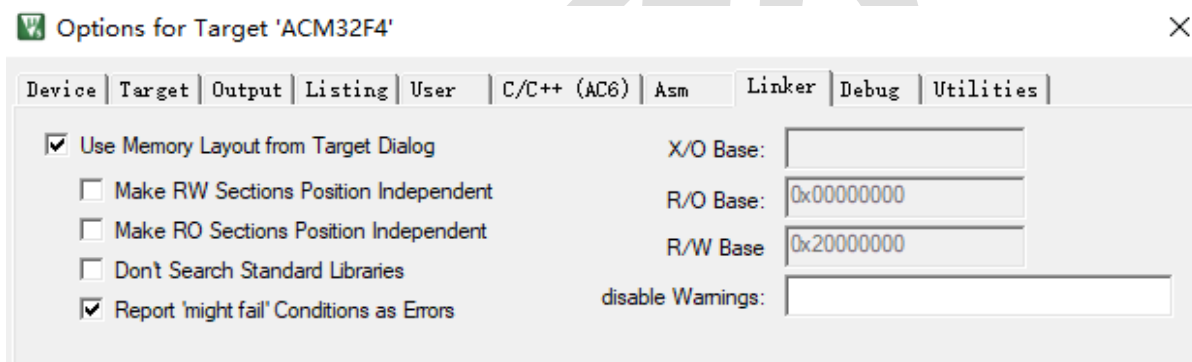


图 3-15 链接加载

5)，代码层面上，保持 CoreDriver/CMSIS、CoreDriver/Device、CoreDriver/HAL 不变，将 main 文件进行替换。系统部分的替换内容如图 3-17 所示，串口部分的替换如图 3-18 和 3-19 所示，高度相似，都是通过 HAL 接口完成。头文件使用 app_uart.h 替换 main.h。去除图 3-20 的内容。

去除 PUTCHAR_PROTOTYPE 以及之后的所有内容。



图 3-16 系统配置修改

```

/###-1- Configure the UART peripheral #####*/
/* Put the USART peripheral in the Asynchronous mode (UART Mode) */
/* UART configured as follows:
- Word Length = 8 Bits (7 data bit + 1 parity bit) : BE CAREFUL : Program 7 c
- Stop Bit     = One Stop bit
- Parity       = ODD parity
- BaudRate     = 9600 baud
Hardware flow control disabled (RTS and CTS signals) */

UartHandle.Instance      = USARTx;

UartHandle.Init.BaudRate = 9600;
UartHandle.Init.WordLength = UART_WORDLENGTH_8B;
UartHandle.Init.StopBits = UART_STOPBITS_1;
UartHandle.Init.Parity = UART_PARITY_ODD;
UartHandle.Init.HwFlowCtl = UART_HWCONTROL_NONE;
UartHandle.Init.Mode = UART_MODE_TX_RX;
if (HAL_UART_Init(&UartHandle) != HAL_OK)
{
    /* Initialization Error */
    Error_Handler();
}

```

图 3-17 STM32F103 串口初始化

```

#define UART_BAUD_RATE 115200

UART_HandleTypeDef UART1_Handle;

/*****
 * function : Uart_Init
 * Description: Uart Initiation.
 *****/
void Uart_Init(void)
{
    UART1_Handle.Instance      = UART1;
    UART1_Handle.Init.BaudRate = UART_BAUD_RATE;
    UART1_Handle.Init.WordLength = UART_WORDLENGTH_8B;
    UART1_Handle.Init.StopBits = UART_STOPBITS_1;
    UART1_Handle.Init.Parity = UART_PARITY_NONE;
    UART1_Handle.Init.Mode = UART_MODE_TX_RX_DEBUG;
    UART1_Handle.Init.HwFlowCtl = UART_HWCONTROL_NONE;

    HAL_UART_Init(&UART1_Handle);
}

```

图 3-18 ACM32F4/F3/FP4 的串口初始化

```
/* Private typedef -----*/
/* Private define -----*/
/* Private macro -----*/
/* Private variables -----*/
/* UART handler declaration */
UART_HandleTypeDef UartHandle;

/* Private function prototypes -----*/
#ifdef __GNUC__
/* With GCC, small printf (option LD Linker->Libraries->Small printf
   set to 'Yes') calls __io_putchar() */
#define PUTCHAR_PROTOTYPE int __io_putchar(int ch)
#else
#define PUTCHAR_PROTOTYPE int fputc(int ch, FILE *f)
#endif /* __GNUC__ */
void SystemClock_Config(void);
static void Error_Handler(void);

/* Private functions -----*/
```

图 3-19 STM32F103 去除的部分

联系我们

公司：上海爱信诺航芯电子科技有限公司

地址：上海市闵行区合川路 2570 号科技绿洲三期 2 号楼 702 室

邮编：200241

电话：+86-21-6125 9080

传真：+86-21-6125 9080-830

Email: Service@AisinoChip.com

Website: www.aisinochip.com

版本维护

版本	日期	作者	描述
V0.1	2021-01-07	Aisinochip	初始版
V0.2	2021-02-19	Aisinochip	F403RET8 改为 F403RET7
V0.3	2021-03-19	Aisinochip	增加KEIL工程移植章节
V0.4	2021-03-26	Aisinochip	STM32F103到航芯芯片的HAL移植

本文档的所有部分，其著作权版权归上海爱信诺航芯电子科技有限公司（简称航芯公司）所有，未经航芯公司授权许可，任何个人及组织不得复制、转载、仿制本文档的全部或部分组件。本文档没有任何形式的担保、立场表达或其他暗示，若有任何因本文档或其中提及的产品所有资讯所引起的直接或间接损失，航芯公司及所属员工恕不为其担保任何责任。除此以外，本文档所提到的产品规格及资讯仅供参考，内容亦会随时更新，恕不另行通知。