第6章 ARM 官方 DSP 库的移植

本期教程主要讲解 ARM 官方 DSP 库的移植和一些相关知识的介绍。

6.1 DSP 库的下载和说明

6.2 DSP 库在 MDK 上的移植

6.3 简易 DSP 库函数验证

6.4 总结

6.1 DSP 库的下载和说明

下面详细的给大家讲解一下官方 DSP 库的移植。

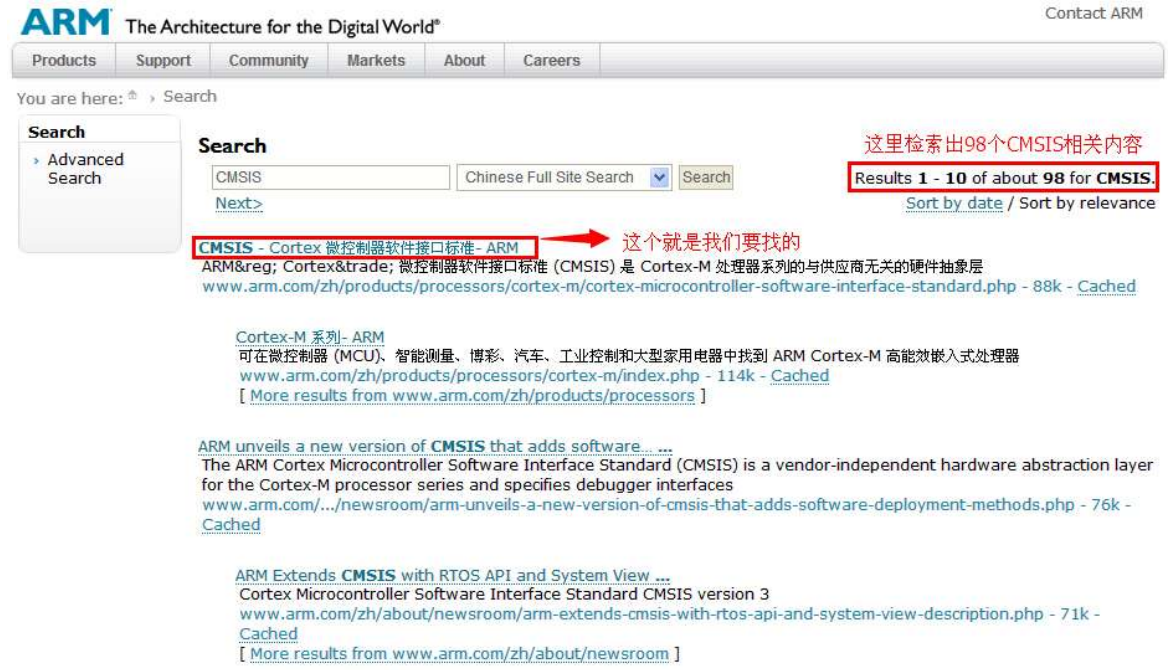
6.1.1 DSP 库的下载

DSP 库是包含在 CMSIS（Cortex Microcontroller Software Interface Standard）里面的，所以下载 DSP 库也就是下载 CMSIS。有两个地方可以下载 CMSIS，一个是 ARM 官网，一个是 ST 官网。首先说一下如何在 ARM 官网下载。

- 第一步：登陆 ARM 官方网址：http://www.arm.com/zh/
- 第二步：注册一个 ID 号，ARM 官网不像 ST 官网，不注册就可以下载大部分资料，在 ARM 官网下载资料一定要注册一个 ID。

- 第三步：具体注册过程就不多说了，按照提示步骤走即可。注册后可以简单的熟悉一下 ARM 官网的结构，然后查询我们需要下载的 CMSIS，也可以直接在这里检索 CMSIS 即可。

■ **检索后打开界面如下：**



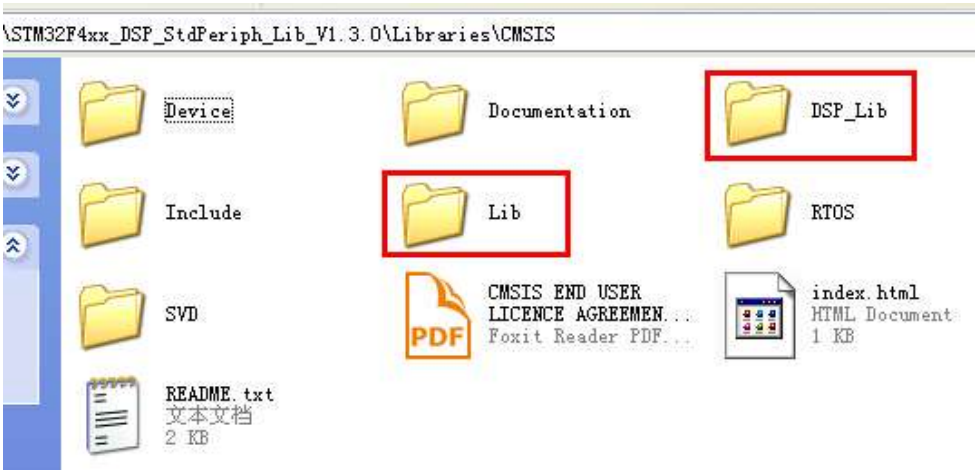
■ **点击进去后，如下就是我们要找的 CMSIS**

关于 ST 官方上 CMSIS 的下载就不在这里赘述了，在 STM32-V5 开发板用户手册第 5 章：ST 官方固件库

介绍有详细的说明。

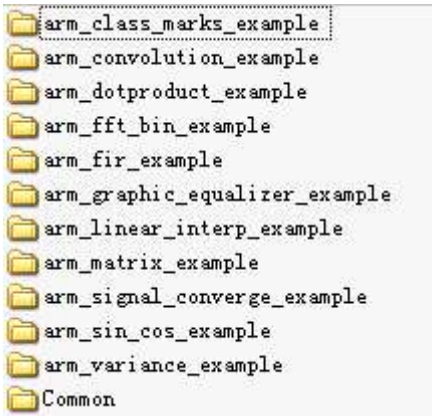
6.1.2 DSP 库的说明

这里我们以 ST 官方的 F4 系列固件库 V1.3.0 为标准进行移植。打开固件库里面的 CMSIS 文件，可以看到如下几个文件：



- 其中 DSP_Lib 中的文件如下：

Examples 中的文件如下（这些是 ARM 官方提供的 DSP 实例）：



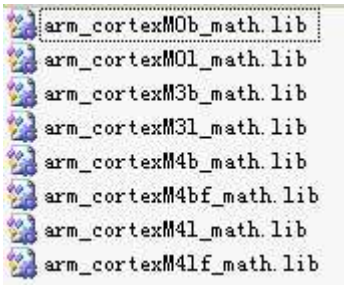
Source 中的文件如下（这些是 DSP 库的源文件）：

一般情况下不建议将源文件加到工程里面，直接将 ARM 官方整理好的 DSP 库文件加入到工程中即可。不过需要查看库文件源码实现的话，可以加入源文件。

- Lib 文件夹中就是 DSP 库文件

  打开后主要有以下三个文件夹：

  其中 ARM 文件夹中是我们可以加入到 MDK 中的 DSP 库，主要有以下几个版本：



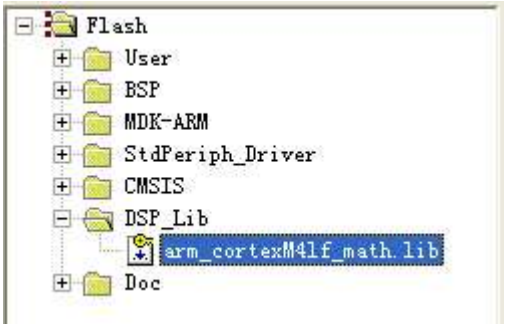
  其中最后一个 arm_cortexM4lf_math.lib 是用于 Cortex-M4 系列带 FPU 的 DSP 库文件(l 表示小端格式，b 表示大端格式)。

6.2 DSP 库在 MDK 上的移植

下面我们讲解一下如何在 MDK 上面移植 DSP 库，DSP 库的移植相对比较容易。这里我们仅介绍如何移植 DSP 库到 MDK 上面，官方没有 IAR 版本的库，所以无法提供移植（可以尝试将源码在 IAR 中进行编译）。

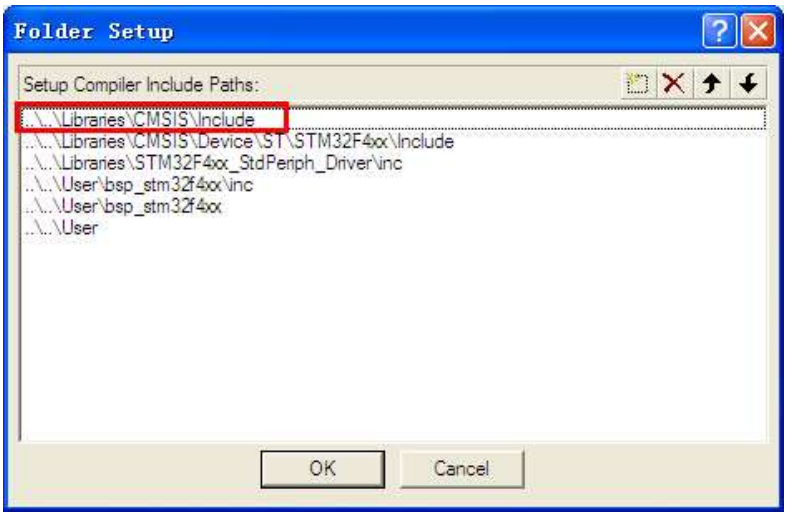
6.2.1 第一步：建立 MDK 工程并添加 DSP 库

为了方便起见，我们这里不再专门建立一个 MDK 工程了，直接以 V5 开发板中的例子：V5-101_按键检测和 LED 控制例程为模板进行添加即可。打开这个实例并在左侧添加针对 Cortex-M4F 的 DSP 库。



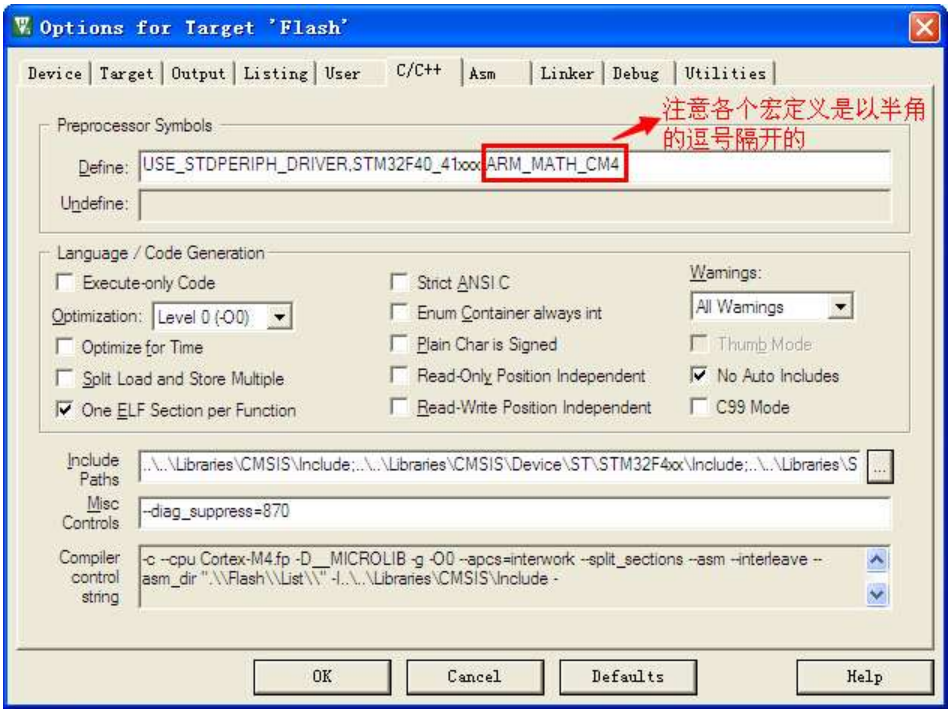
6.2.2 第二步：添加头文件路径

添加 DSP 所需的头文件路径，这个头文件路径是已经在工程中添加好的，这里只是跟大家强调一下。



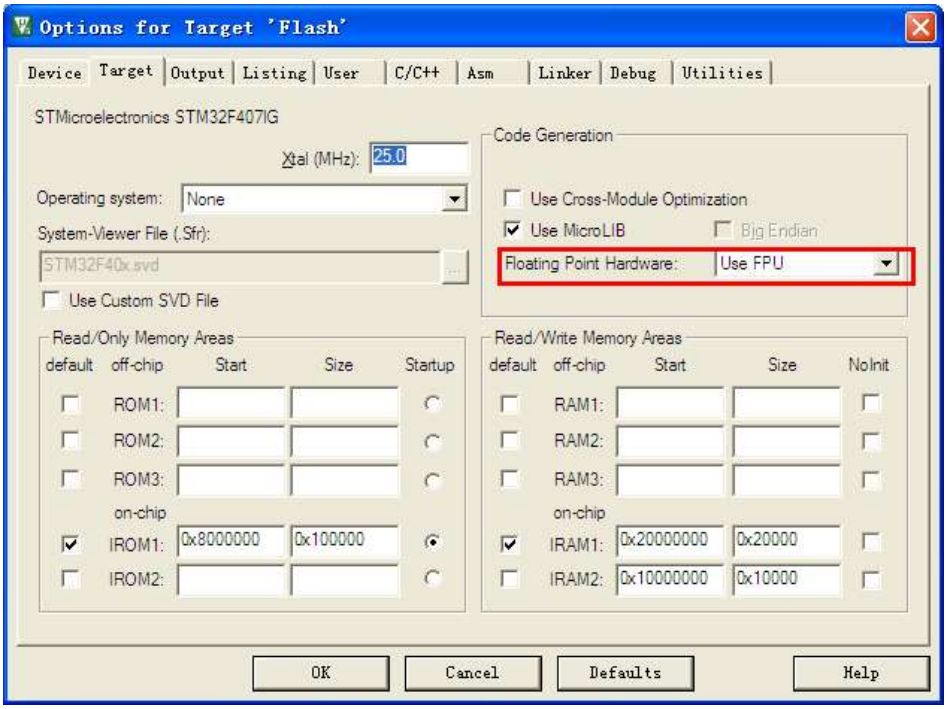
6.2.3 第三步：添加宏定义

添加宏定义如下（这里将 STM3240XX 换成了 STM32F40_41xxx）：



6.2.4 第四步：开启 FPU

需要客户通过 MDK 开启 FPU，开启方法如下：



另外根据 ARM 官方 DSP 库的要求，还需要设置宏__**FPU_PRESENT** 为 1，不过这个宏已经在文件 stm32f4xx.h 中设置了（为保险起见，建议把__FPU_PRESENT 在第三步的地方也加上，因为部分 DSP 函数会因为没有这个声明而报错）。

```
/**
 * @brief Configuration of the Cortex-M4 Processor and Core Peripherals
 */
#define __CM4_REV                 0x0001  /*!< Core revision r0p1                           */
#define __MPU_PRESENT             1       /*!< STM32F4XX provides an MPU                    */
#define __NVIC_PRIO_BITS          4       /*!< STM32F4XX uses 4 Bits for the Priority Levels */
#define __Vendor_SysTickConfig    0       /*!< Set to 1 if different SysTick Config is used  */
#define __FPU_PRESENT             1       /*!< FPU present                                  */
```

6.2.5 第五步：添加头文件 arm_math.h

用到 DSP 库函数的相应的文件得添加**#include "arm_math.h"**的支持。

按照上面五部操作即可完成 DSP 库的移植，移植好我们通过几个 DSP 库中的函数验证下是否正确。

6.3 简易 DSP 库函数验证

这里我们主要运行下函数 arm_abs_f32，arm_abs_q31，arm_abs_q15 这三个函数，以此来验证我们移植的 DSP 库是否正确。

实验目的：

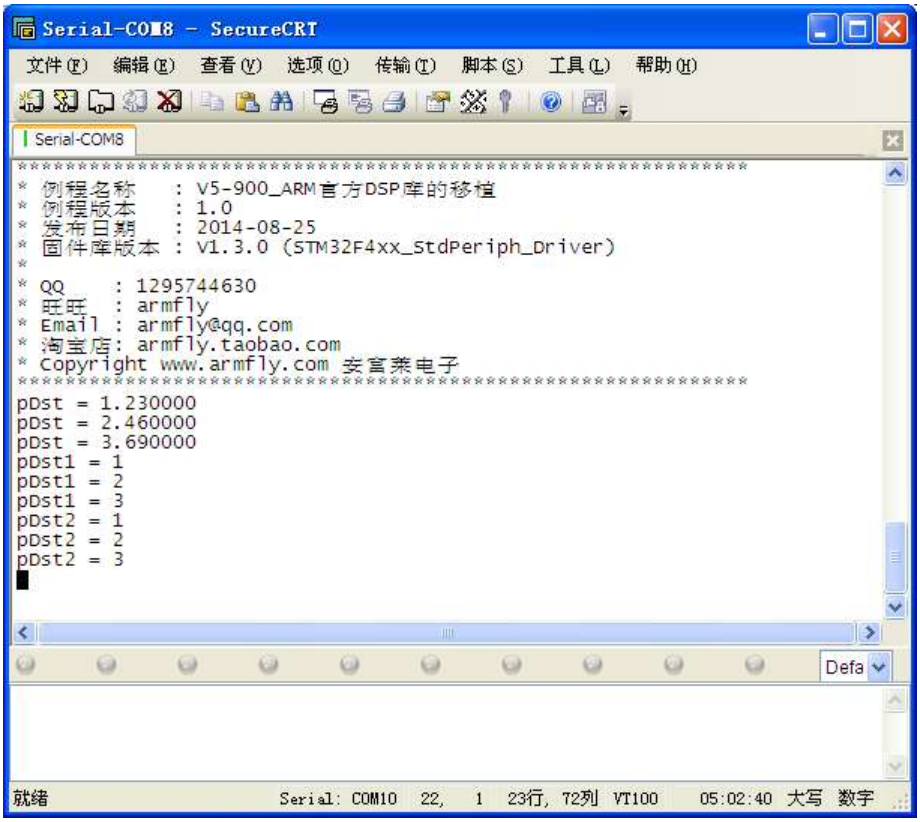
1. 学习官方 DSP 库的移植

实验内容：

1. 按下按键 K1，串口打印函数 arm_abs_f32 的输出结果
2. 按下按键 K2，串口打印函数 arm_abs_q31 的输出结果
3. 按下按键 K3，串口打印函数 arm_abs_q15 的输出结果

实验现象：

通过窗口上位机软件 SecureCRT（V5 光盘里面有此软件）查看打印信息现象如下（分别按几次 K1，K2，K3）：



程序设计：

程序的设计也比较简单，通过按下不同的按键从而打印不同的 DSP 库函数执行结果，主程序如下：

```
#include "bsp.h"                /* 底层硬件驱动 */
#include "arm_math.h"

/* 定义例程名和例程发布日期 */
#define EXAMPLE_NAME    "V5-900_ARM官方DSP库的移植"
#define EXAMPLE_DATE    "2014-08-25"
#define DEMO_VER        "1.0"

/* 仅允许本文件内调用的函数声明 */
static void PrintfLogo(void);

/*
*********************************************************************************************************
*    函 数 名: main
*    功能说明: c程序入口
*    形    参: 无
*    返 回 值: 错误代码(无需处理)
*********************************************************************************************************
*/
int main(void)
```

```
{
    uint8_t ucKeyCode;      /* 按键代码 */
    float32_t pSrc;
   float32_t pDst;
    q31_t pSrc1;
    q31_t pDst1;
    q15_t pSrc2;
    q15_t pDst2;

    bsp_Init();         /* 硬件初始化 */
    PrintfLogo(); /* 打印例程信息到串口1 */

    bsp_StartAutoTimer(0, 500); /* 启动1个500ms的自动重装的定时器 */

    /* 进入主程序循环体 */
    while (1)
    {
        bsp_Idle();         /* 这个函数在bsp.c文件。用户可以修改这个函数实现CPU休眠和喂狗 */

        if (bsp_CheckTimer(0)) /* 判断定时器超时时间 */
        {
            /* 每隔500ms 进来一次 */
            bsp_LedToggle(4);  /* 翻转LED4的状态 */
        }

        /* 按键滤波和检测由后台systick中断服务程序实现，我们只需要调用bsp_GetKey读取键值即可。 */
        ucKeyCode = bsp_GetKey();   /* 读取键值, 无键按下时返回 KEY_NONE = 0 */
        if (ucKeyCode != KEY_NONE)
        {
            switch (ucKeyCode)
            {
                case KEY_DOWN_K1:                /* K1键按下 */
                    pSrc -= 1.23f;
                    arm_abs_f32(&pSrc, &pDst, 1);
                    printf("pDst = %f\r\n", pDst);
                    break;

                case KEY_DOWN_K2:                /* K2键按下  */
                    pSrc1 -= 1;
                    arm_abs_q31(&pSrc1, &pDst1, 1);
                    printf("pDst1 = %d\r\n", pDst1);
                    break;

                case KEY_DOWN_K3:                /* K3键按下  */
                    pSrc2 -= 1;
                    arm_abs_q15(&pSrc2, &pDst2, 1);
                    printf("pDst2 = %d\r\n", pDst2);
                    break;

                default:
                    /* 其它的键值不处理 */
                    break;
            }
        }
    }
}
```

6.4 总结

本期教程主要跟大家介绍了官方 DSP 库的移植，相对来说移植也比较简单，建议初学的同学按照这个步骤移植一遍。