ZS110A 入门指南

发布 ***1.0.0***

2018 年 **11** 月 **02** 日

目录

CHAPTER 1

概述

1.1 文档目的

方便开发人员快速入门，熟悉开发环境的搭建、烧写和调试。

1.2 术语说明

表 1.1: 术语说明

术语	说明
BLE	蓝牙低功耗技术
IOT	物联网
SDK	软件开发套件
KEIL	一款单片机开发工具
DAPLINK	ARM 官方推出的开源仿真器
GATT	通用属性配置文件
PROFILE	本文特指通用属性配置文件

1.3 参考文档

- http://www.keil.com/support/man/docs/uv4/

1.4 版本历史

表 1.2: 版本历史

日期	版本	注释	作者
2018-08-30	1.0	初始版本	ZS110A 项目组

CHAPTER 2

ZS110A 简介

ZS110A 是基于炬芯科技新一代的低功耗蓝牙芯片 ATB110X 的软件开发套件，供用户快速评估学习蓝牙、IoT 等功能开发。

2.1 开发板硬件资源介绍

- 板载蓝牙天线
- 1 路 MIC
- 5 路 PWM
- 4 个 PWM 灯，一个电源指示灯
- 矩阵按键 *6、AD 按键 *3
- 1 个 USB 口
- 1 个红外发射头，1 个红外接收头
- 板载 daplink
- 1 个 reset 键，一个电源开关，一个电源选择开关
- 两种供电方式: usb、电池

2.2 软件资源介绍

ZS110A 提供完整功能的 SDK 框架供用户参考开发，并提供大量的使用示例和参考代码。

- 蓝牙语音遥控器示例
- 蓝牙 profile 示例
- 蓝牙主机、主从示例
- 各个驱动模块示例

2.3 开发板概况

开发板由核心板和底板构成，如图所示:

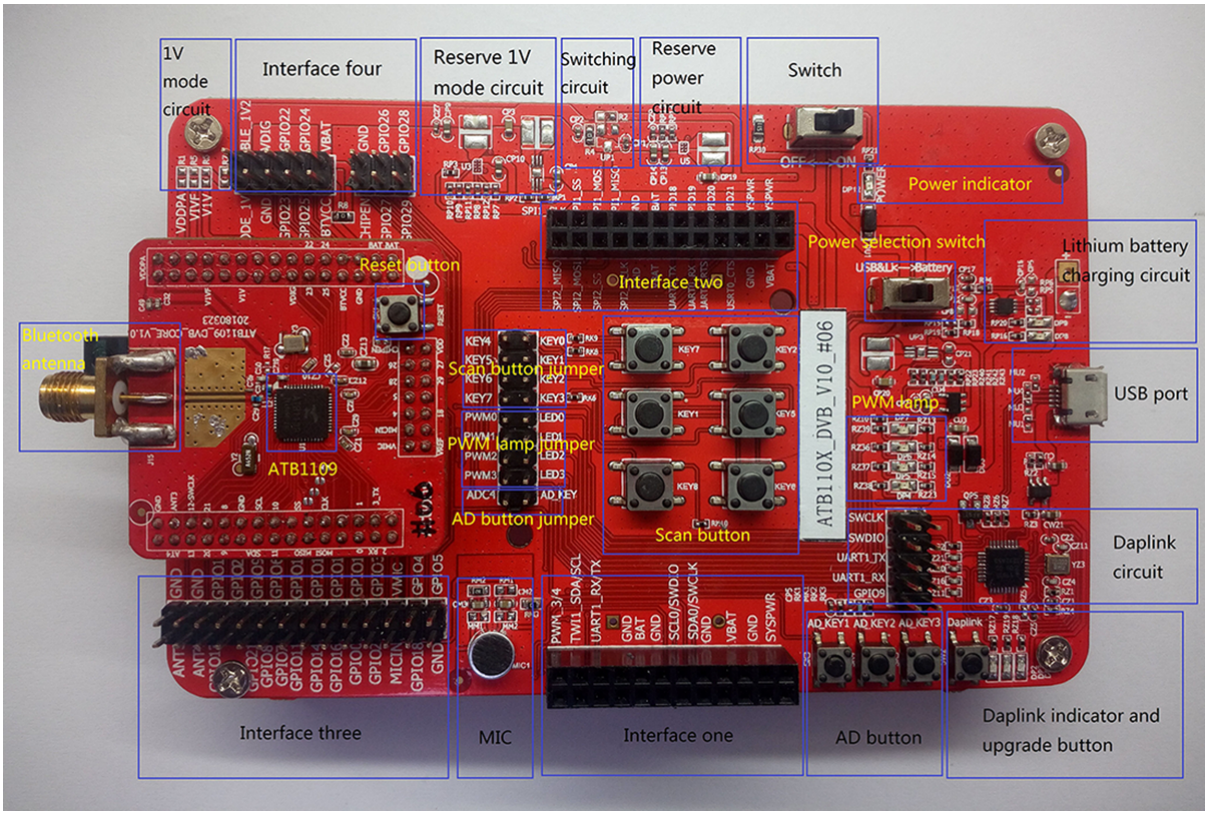

图 2.1: 开发板概况图

在使用前，需要对开发板进行一些检查并连接串口:

- Daplink: 短接 SWCLK、SWDIO。
- 电源: 电源开关选择 ON, 电源选择开关选择 USB&Li。
- UART: GPIO2 接串口 TX，GPIO3 接串口 RX (UART0)。

CHAPTER 3

开发环境搭建

3.1 开发工具

ZS110A 方案开发使用 Keil uVision5 (V5.21 及以上版本)

3.2 调试工具

ZS110A 开发板上集成了基于 LPC11U35 实现的 DAPLink (cmsis-dap)，支持以下功能：

- Debug ARM Cortex-M SOC
- usb-serial

使用 usb 线连接开发板和 PC 后，PC 上的设备管理器增加 3 个设备，如图所示:

- Mbed Serial Port
- HID-complicant device
- USB 大容量存储设备

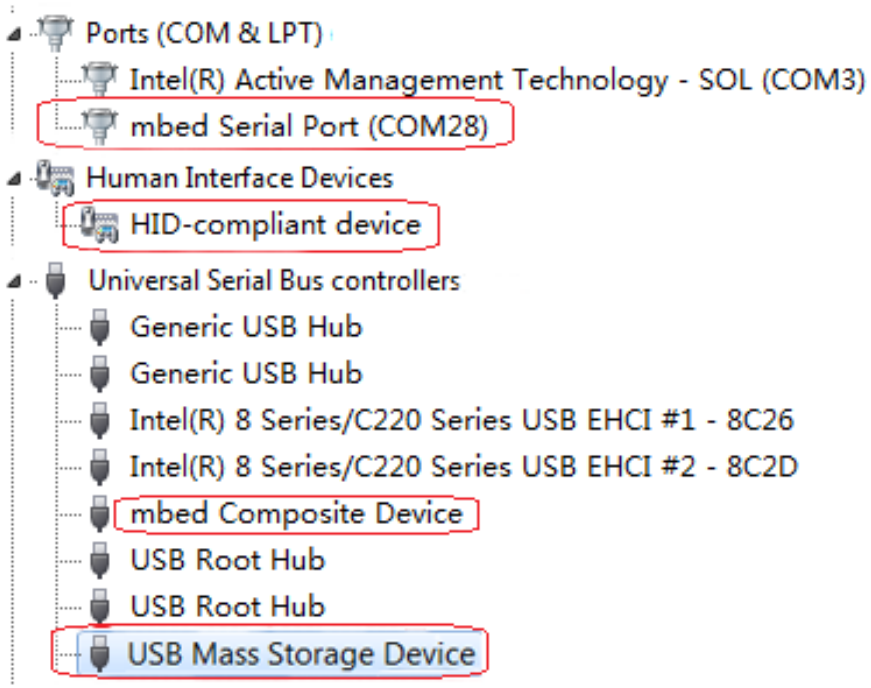

图 3.1: 设备管理器信息

USB 大容量存储设备的磁盘卷标名为”DAPLINK”，如图所示:

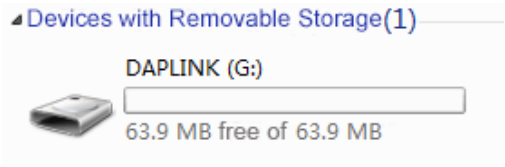

图 3.2: USB 大容量存储设备

说明: 若未检测到 med Serial Port，需安装驱动 mbedWinSerial_16466

(路径：scripts\support\actions\utils\mbed_driver\)，安装驱动时需要接上开发板。

使用 DAPLink 调试前需要在 Keil 中完成以下配置：

1. 打开 Keil Options/Debug 界面，如图所示

- Use: 选择 CMSIS-DAP Debugger
- 设置 debug 参数
  1. Load Application at Startup: 不勾选。
  2. Run to main: 不勾选。

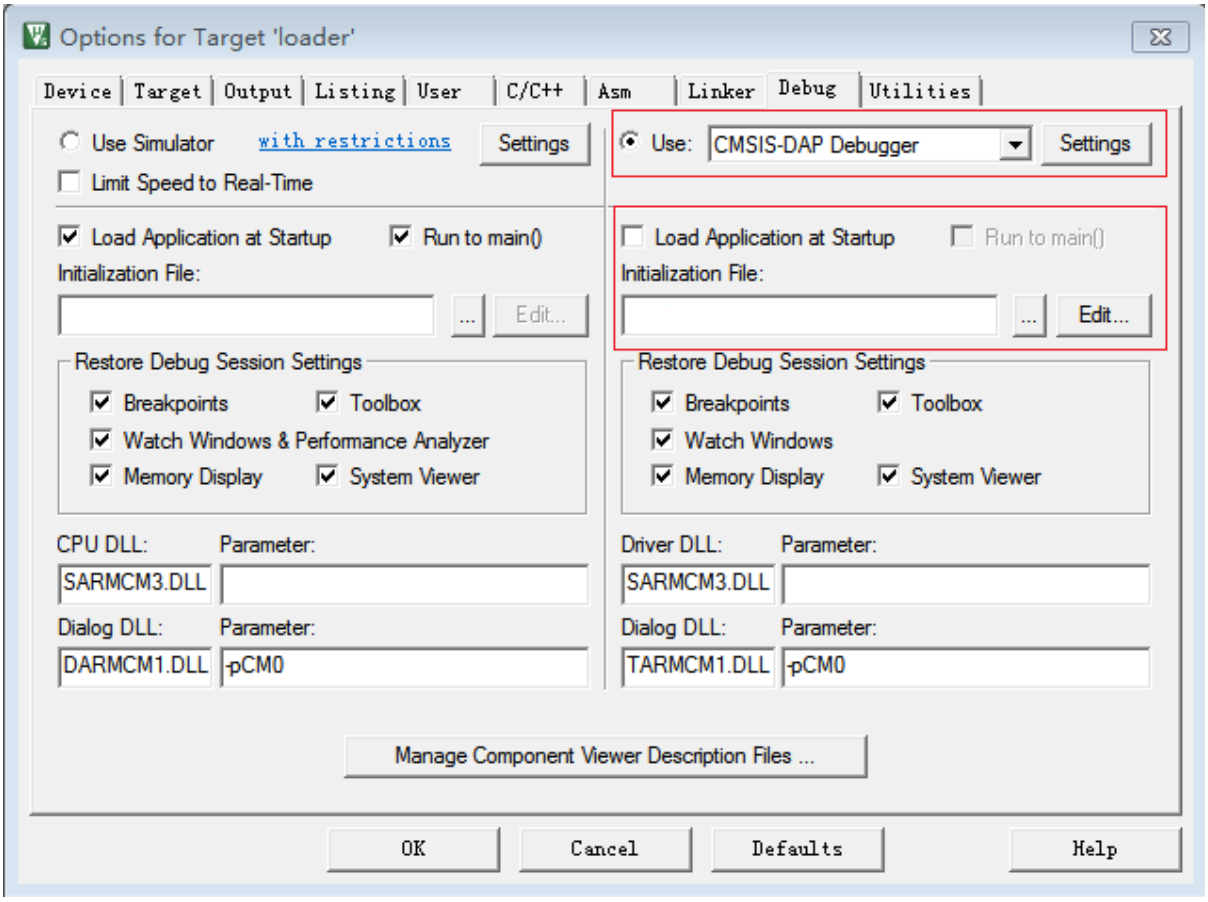

图 3.3: Keil Options/Debug 配置

2. 打开 Keil Options/Debug/Settings 界面，如图所示

- 若设备已连接，且系统未处于休眠状态：
  1. Debug 界面内发现 DAPLINK；
  2. Debug 界面内发现 Target 设备。
- Connect & Reset Options：不勾选 Reset after Connect。

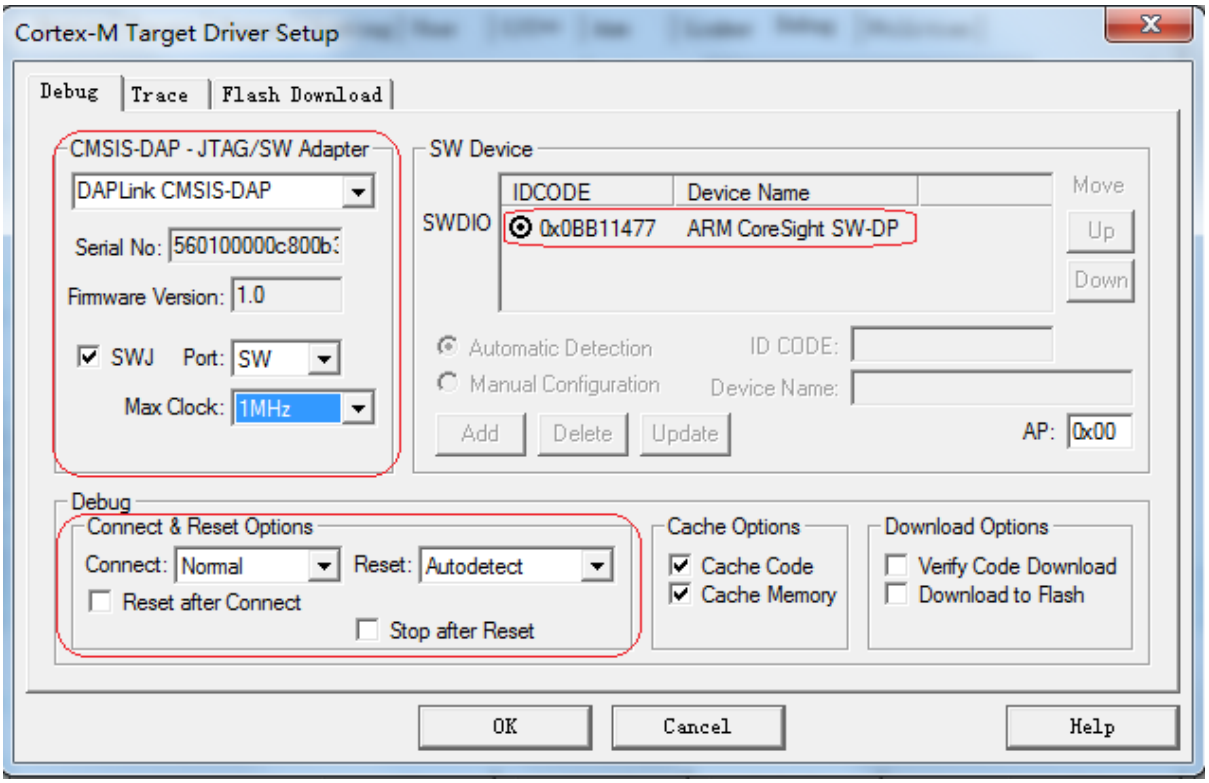

图 3.4: Keil Options/Debug/Settings 配置

3. 打开 Keil Options/Utilities 界面，如图所示

- 选择 Use Target Driver For Flash Programming
- 勾选 Use Debug Driver
- 不勾选 Update Target before Debugging

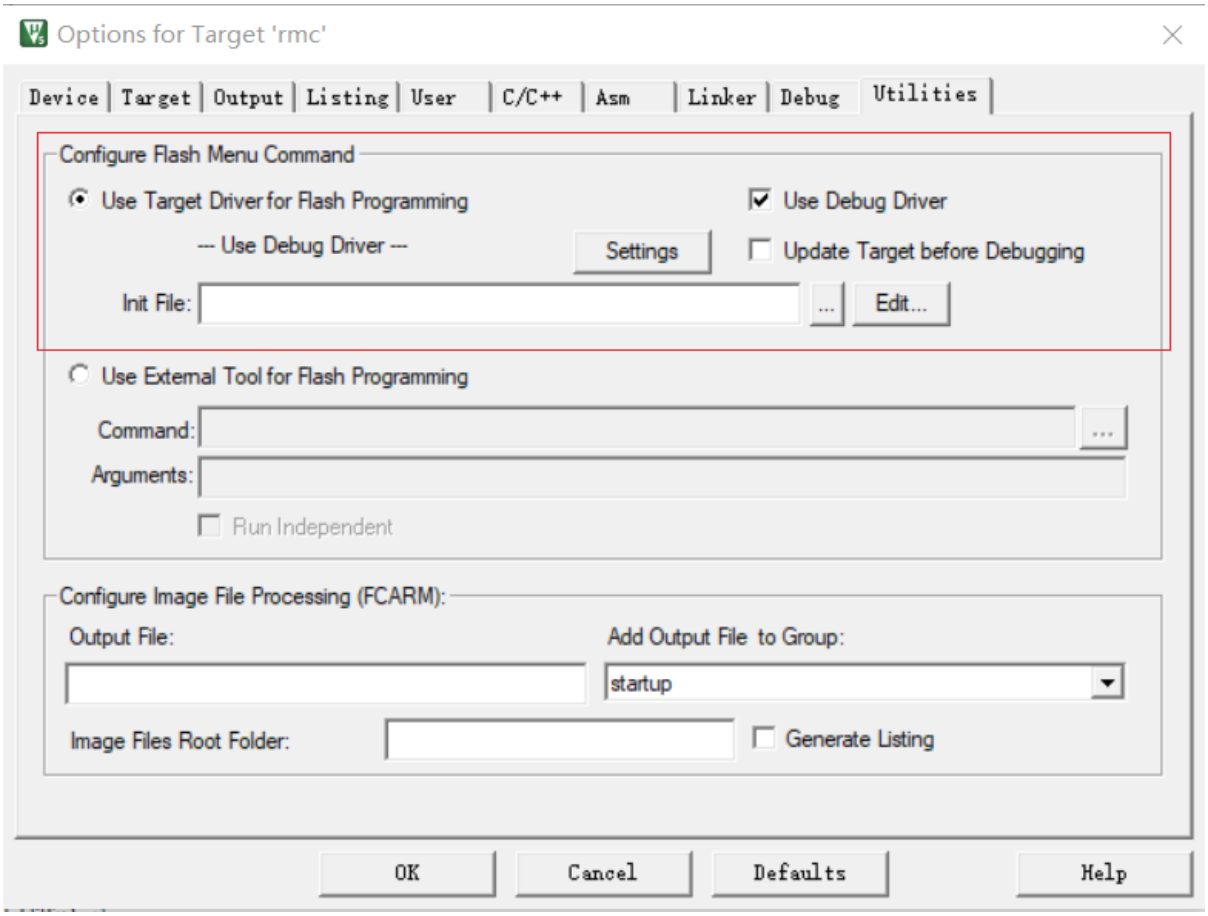

图 3.5: Keil Options/Utilities 配置

CHAPTER 4

在线调试

Keil 支持在线调试功能，通过点击”start/stop debug”即可启动在线调试。在启动调试之前，需要增改一些配置，下面以 helloworld 程序 (路径：samples\peripheral\ helloworld) 为例说明如何在 Keil 下进行调试。

1. 增加一个 ram 工程

   用 Keil 打开 Manage Project Items，在 Project Targets 一栏添加一个 ram 工程，如图所示

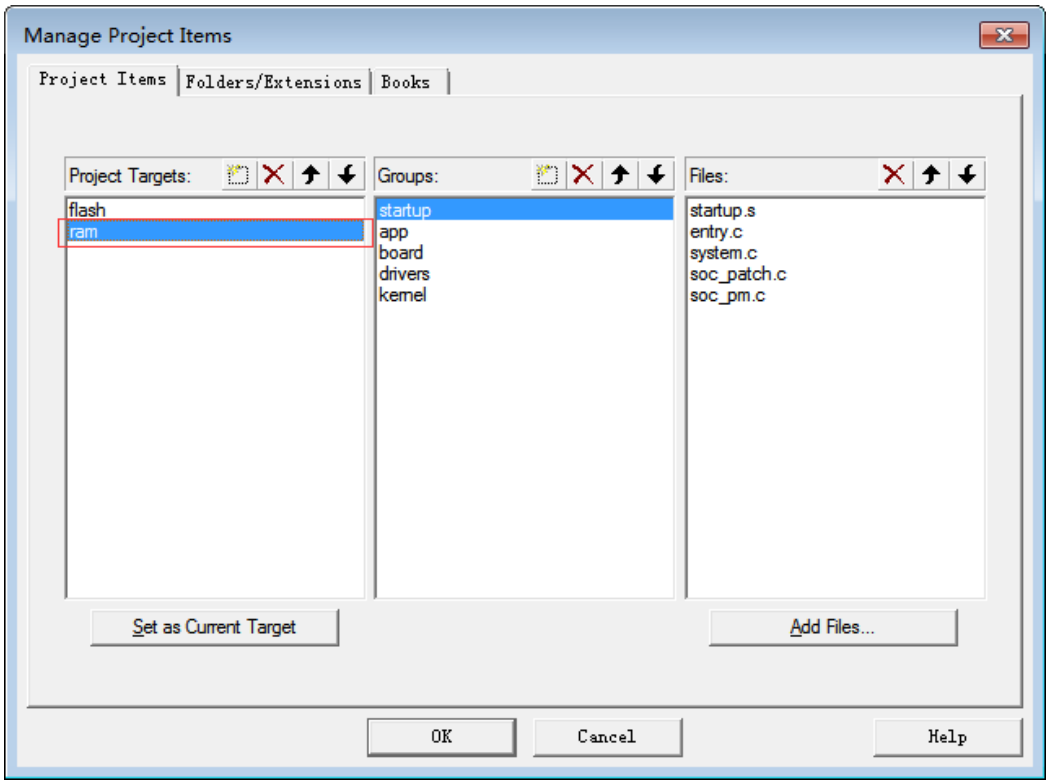

图 4.1: 增加一个 ram 工程

随后选择该工程，如图所示:

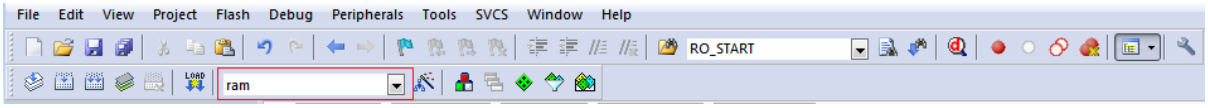

图 4.2: 选择 ram 工程

2. 配置 ram 工程

- 新建 debug.ini 文件

  在 helloworld 工程目录下新建一个名为 debug.ini 的配置文件，文件内容如下:

```
FUNC void CloseWatchDog (void)
{
  _WDWORD(0x4000401C, 0x20);
}
FUNC void Setup (void)
{
  SP = _RDWORD(0x20001000);
  PC = _RDWORD(0x20001004);
}
CloseWatchDog();
LOAD .\Objects\helloworld.axf INCREMENTAL
Setup();
g,main
```

说明: helloworld.axf 是当前工程编译产生的.axf 文件。

- 设置 debug 参数

  1. Load Application at Startup: 不勾选。
  2. Run to main: 停在 main 入口: 不勾选。
  3. Initialization File: 载入 debug.ini 文件

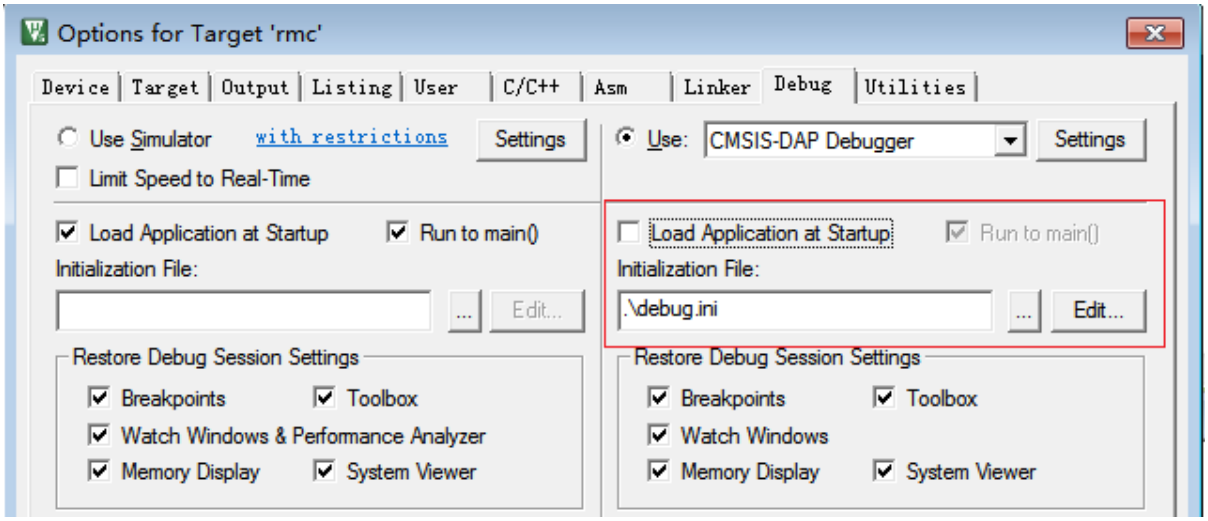

图 4.3: 设置 debug 参数

- 删除编程算法

  打开 Keil Options/Utilities/Settings，删除已有的烧写算法。点击 remove，如图所示:

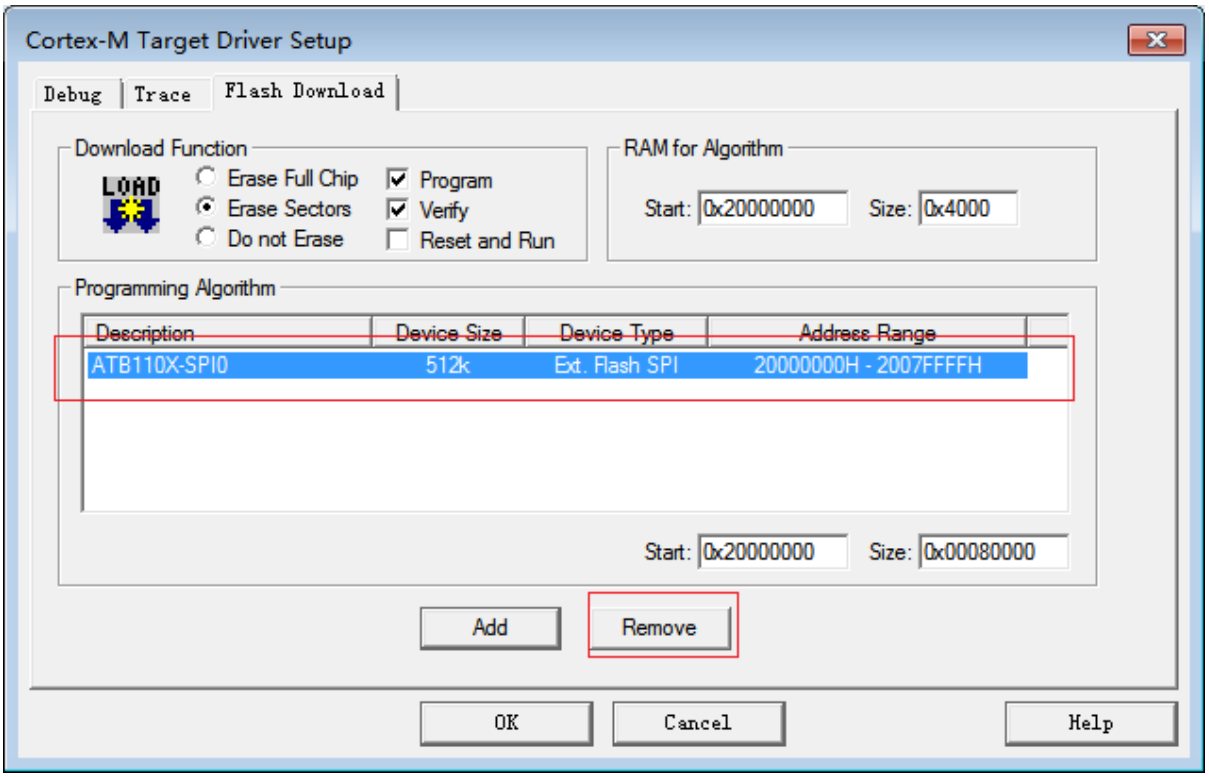

图 4.4: 修改编程算法地址

3. 编译

   点击下图中的任意一个编译按钮，即可完成编译。

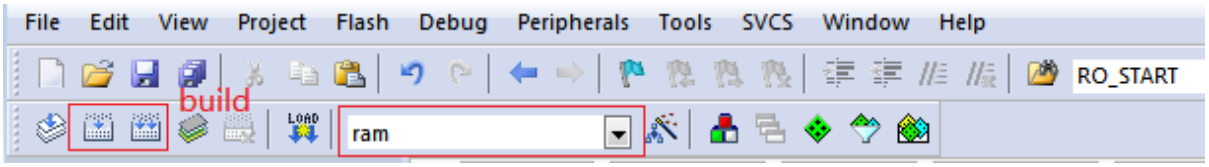

图 4.5: 编译

4. 设置断点

   为方便的控制程序的运行，可在 app_main 入口处设置一个断点，如下图所示:

```
13  #include <misc/byteorder.h>
14  #include <zephyr.h>
15
16  void app_main(void)
17  {
18      while(1) {
19          printk("Hello World!\n");
20          k_sleep(1000);
21      }
22  }
```

图 4.6: 设置断点

5. 启动调试

   点击“start/stop debug”启动调试，点击调试工具栏中的按钮单步调试。

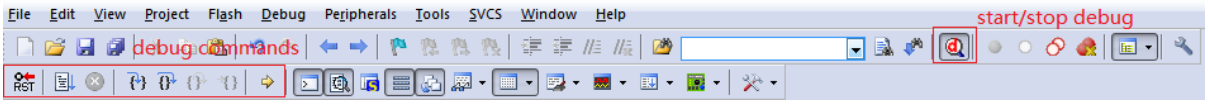

图 4.7: 启动调试

CHAPTER 5

固件烧写

编译成功后，在 Keil 工具界面点击”Load”即可完成烧写。

5.1 配置

1. 拷贝 KEIL FLM

   将 scripts\support\actions\utils\keil_flash\ATB110X_SPI0.FLM 拷贝到目录 C:\Keil\ARM\Flash\(请根据自己 Keil 实际安装路径进行调整)。

2. 选择 flash 工程，如图所示:

图 5.1: 选择 flash 工程

说明: flash 是程序烧到 flash 的默认工程名，其他工程名亦可。

3. 根据 3.2 调试工具一节配置 Keil。
4. 打开 Keil Options/Utilities/Settings，如图所示。

- Download Function：选择”Erase Sectors”,”Program”,”Verify”。
- RAM for Algorithm：起始地址 0x20000000，size 0x4000。
- Programming Algorithm：

点击”remove”删除默认的”New Device 256kB Flash”(如果有的话)，然后点击”Add”选择 ATB110X-SPI0。然后编辑 Address Range：start 配置为 0x20000000，size 采用默认值。如图所示。

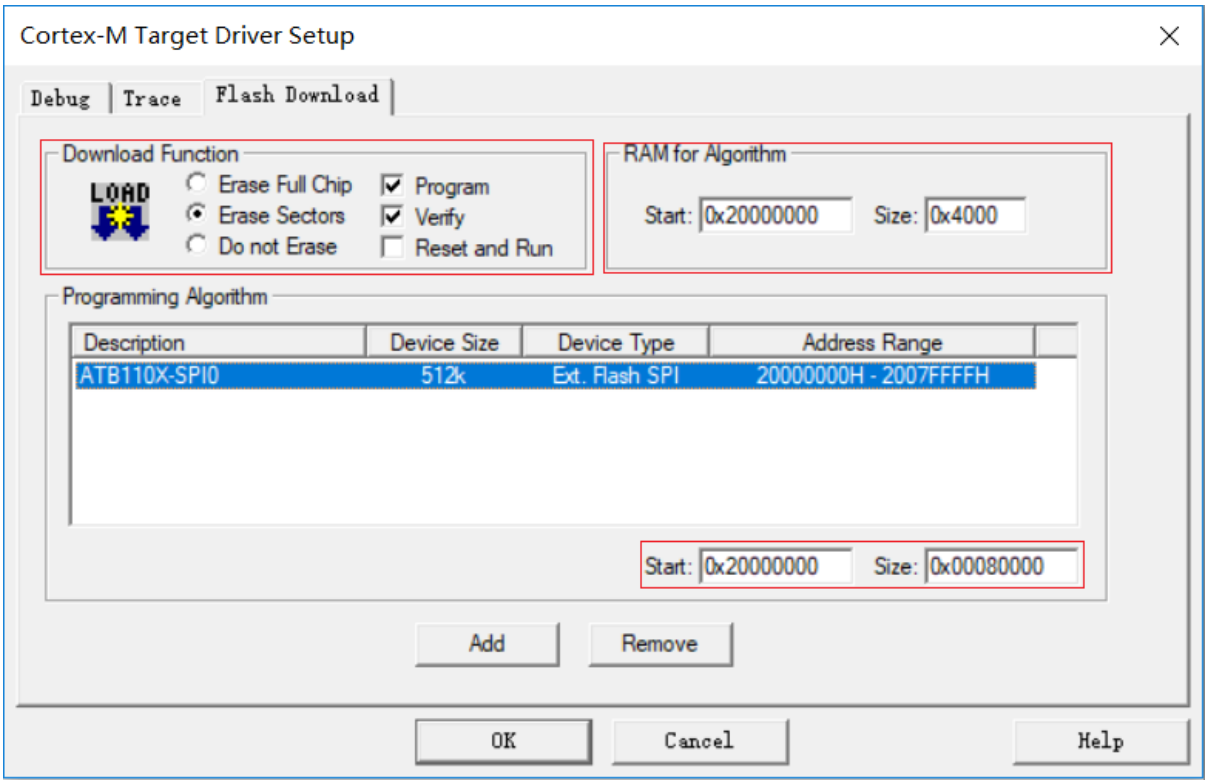

图 5.2: Options/Utilities/Settings 配置

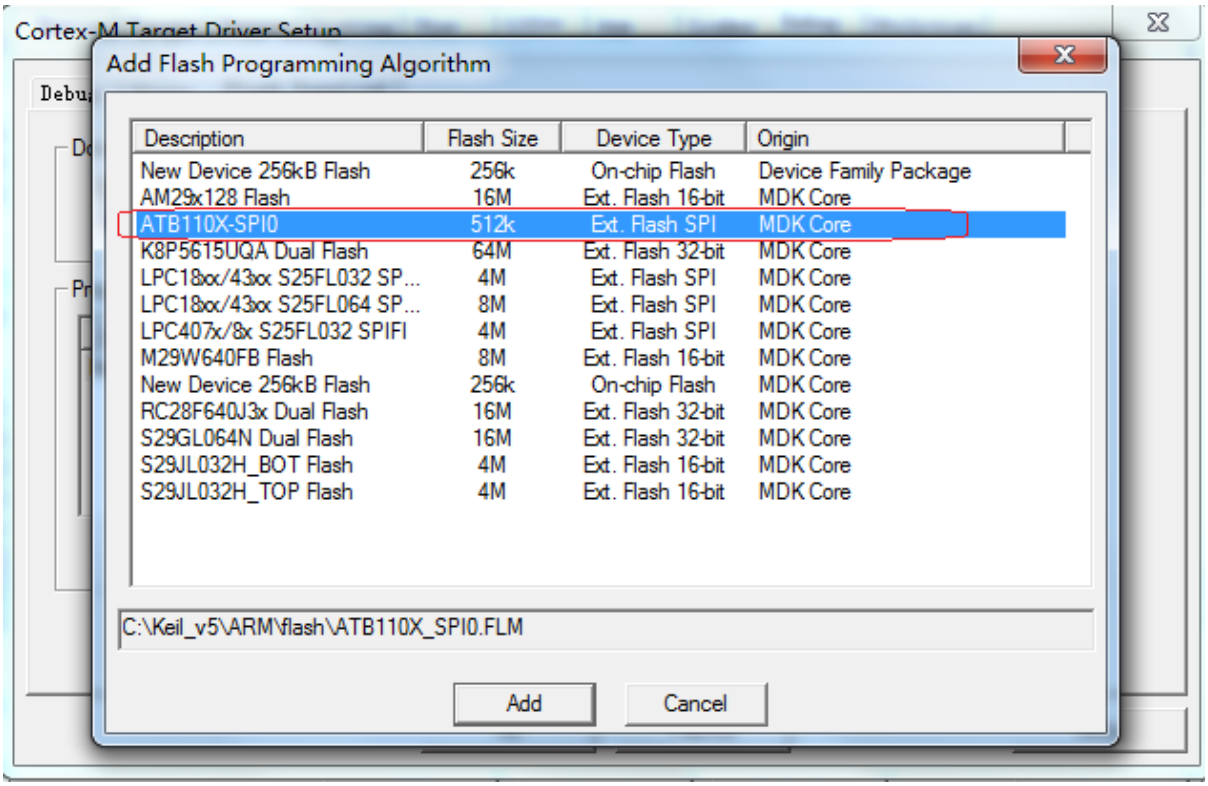

图 5.3: 增加 flash 编程算法

5.2 烧写

下面以 sample/peripheral/helloworld 工程程序为例，介绍固件的烧写。helloworld 示例程序代码如下:

```
void app_main(void)
{
    while (1) {
            printk("Hello World!\n");
            k_sleep(1000);
    }
}
```

完成 5.1 节配置，点击下图中的编译按钮编译成功后，点击 Keil 工程中的 LOAD 即可完成烧写。

图 5.4: 编译烧写方法

烧录完成后，按下 reset 按键，系统将重启，串口将循环输出 Hello World!:

```
Hello World!
Hello World!
Hello World!
```

List of Figures

List of Tables