1. 概述

本文介绍UC8288/UC8088系列的软件架构及使用方式，以帮助客户快速进行二次开发。

2. 工程描述

2.1 源码

终端侧代码：https://github.com/ucchip/wiota_dev_customer 。
基站侧代码：https://github.com/ucchip/wiota_ap_customer 。

2.2 源码目录结构

2.2.1 终端侧

```
wiota_dev_customer
├───app
|   ├───codec
|   |   ├───cbor
|   |   ├───cJSON
|   |   ├───coding
|   |   └───fastlz
|   ├───custom_data
|   ├───custom
|   ├───manager
|   └───platfrom
├───applications
├───bin
├───board
├───boot
├───doc
├───libraries
|   ├───HAL_Drivers
|   └───UC8288_HAL_Driver
├───packages
├───PS
|   ├───app
|   └───at
└───rt-thread
```

- [app] 应用demo，demo演示相关代码，包括频点管理，配对，状态上报，属性上报等。
  - [codec] demo编解码相关，包括app协议编解码、数据解压缩、cJSON。
    - [cbor] 暂时未使用。
    - [cJSON] cJSON相关代码。
    - [coding] app协议编解码。
    - [fastlz] 数据解压缩。
  - [custom_data] 用户数据。
  - [custom] 包含用户管理，配对请求和灯控场景、开关场景两个demo示例。
  - [manager] 包含配置管理，频点管理等。
  - [platfrom] demo通用函数。

- [applications] 应用层，包括main函数入口和watchdog。
- [board] 板级支持包。
- [boot] Ymodem下载支持等。
- [libraries] 驱动层，包括串口、I2C、SPI等等。
- [packages] 软件包信息。
- [PS]
  - [app] 二次开发提供的接口文件，还包括一个test文件作为使用示例。
    - [include] api头文件。
    - [lib] WIoTa协议静态库。
    - [test] api接口使用示例。
  - [at] WIoTa的AT指令。
- [rt-thread] 此目录是rt-thread系统文件，包括调度器、时钟、内存管理等等。
- [bin] 镜像文件。
- [doc] IOTE配套文档。

2.2.2 基站侧

```
wiota_ap_customer
├───app
|   ├───codec
|   |   ├───cbor
|   |   ├───cJSON
|   |   ├───coding
|   |   └───fastlz
|   ├───custom_data
|   ├───net_passthrough
|   ├───peripherals_manager
|   ├───platfrom
|   └───wiota_manager
|   |   ├───manager_iote_data
|   |   ├───manager_network_data
|   |   ├───manager_wiota
|   |   ├───manager_wiota_base
|   |   └───manager_wiota_to_network
├───applications
├───bin
├───board
├───boot
├───doc
├───libraries
|   ├───HAL_Drivers
|   └───UC8088_HAL_Driver
├───Other
├───packages
|   ├───at_device
|   └───uc_wolfMQTT
├───PS
|   ├───app
|   ├───at
|   └───lib
└───rt-thread
```

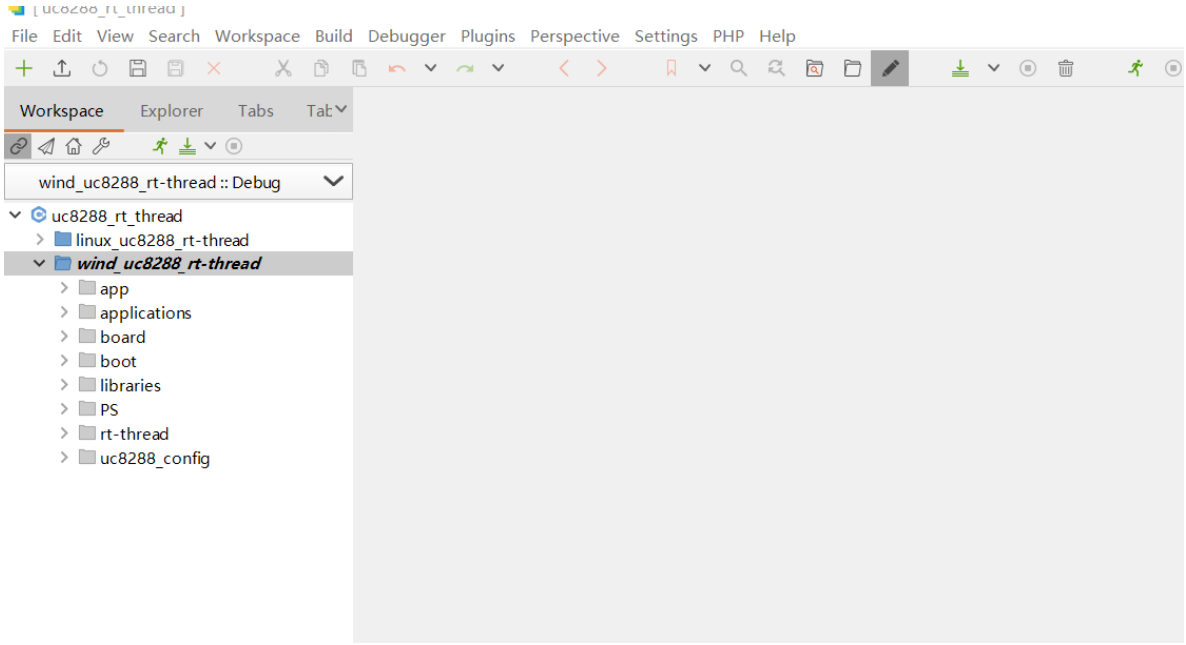
- [app] 应用demo层，demo演示相关代码，包括频点管理，消息转发，状态上报，id管理，网关透传，外设管理等。
    - [codec] demo编解码相关，包括app协议编解码、数据解压缩、cJSON。
        - [cbor] 暂时未使用。
        - [cJSON] cJSON相关代码。
        - [coding] app协议编解码。
        - [fastlz] 数据解压缩。
    - [custom_data] 用户数据。
    - [net_passthrough] 网关透传，通过MQTT与服务器交互。
    - [peripherals_manager] 外设管理。
    - [platfrom] demo通用函数。
    - [wiota_manager] 管理相关，包括终端上报数据管理，网络下发的数据管理等。
        - [manager_iote_data] 终端上报数据管理。
        - [manager_network_data] 网络下发数据管理。
        - [manager_wiota] 终端id管理。
        - [manager_wiota_base] 基础功能管理。
        - [manager_wiota_to_network] AP自身管理。
- [applications] 应用层，包括main函数入口。
- [board] 板级支持包。
- [boot] Ymodem下载支持、通过8088SPI烧写8288等。
- [libraries] 驱动层，包括串口、I2C、SPI等等。
- [Other] 包含软件包信息。
- [PS]
    - [app] 二次开发提供的接口文件，还包括一个test文件作为使用示例。
    - [at] WIoTa的AT指令。
    - [lib] WIoTa协议静态库文件。
- [rt-thread] 此目录是rt-thread系统文件，包括调度器、时钟、内存管理等等。
- [bin] 刷机镜像。
- [doc] AP配套文档。

2.3 使用方式

2.3.1 打开工程

使用codelite软件打开后缀是workspace的文件，基站侧是uc8088_wiota_ap.workspace，终端侧是uc8288_rt-thread.workspace。**codelite的安装及使用方式请查看**（按住ctrl，选中链接点击鼠标左键可直接打开）：https://mkdocs.ucthings.com/ucchip_env/ucchip_ide_install/

以终端侧工程为例，打开工程之后可看到如下两个项目：



linux开头的用于在Linux系统下编译，wind开头的在Windows系统下编译。加粗的是当前激活的项目，只会编译到加粗的项目。双击项目名即可加粗。

2.3.2 编译工程

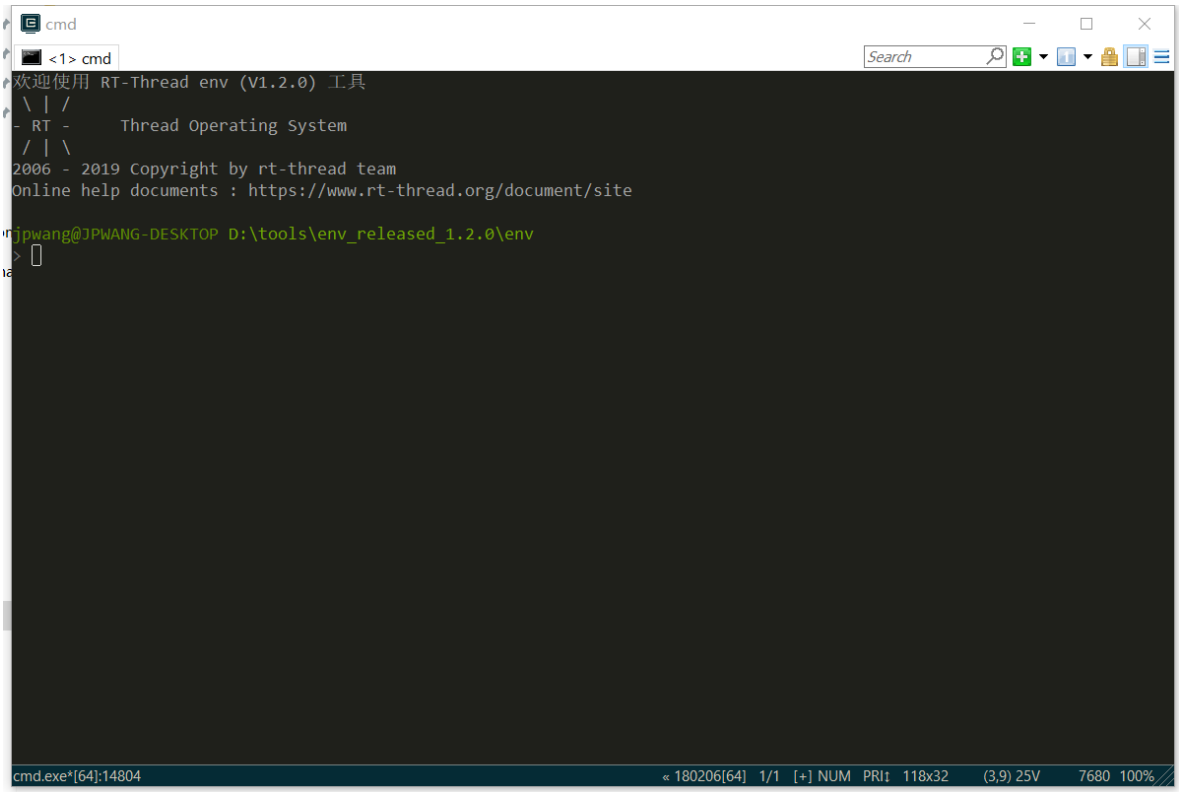
注：v0.12及之后的版本均采用rt-thread的编译脚本进行编译，因此需先安装rt-thread的环境。

2.3.2.1 RT-Thread env 工具下载与安装

RT-Thread env 工具下载链接（按住ctrl，选中链接点击鼠标左键可直接打开）：https://www.rt-thread.org/page/download.html

点击网站下载，下载完成后，解压后在env目录下打开env.exe，rt-thread的控制窗口就打开了，最好按照env目录下Add_Env_To_Right-click_Menu.png图片的指示配置下右键快捷方式，这样在任意目录

都能通过鼠标右键呼出rt-thread的控制窗口了。



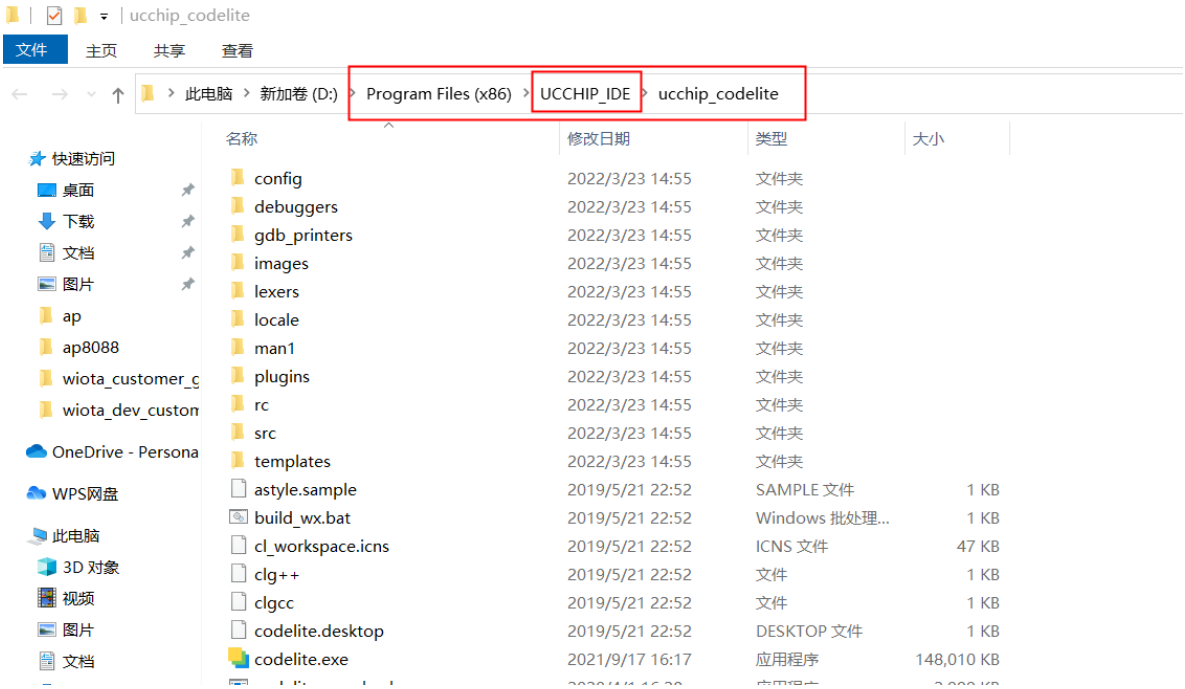
2.3.2.2 更改编译配置文件

在代码根目录下找到rtconfig.py文件，然后打开配置工具链路径，将下图红框所示的路径改为自己的工具链路径，如果相同则不需修改，该路径不正确无法编译。

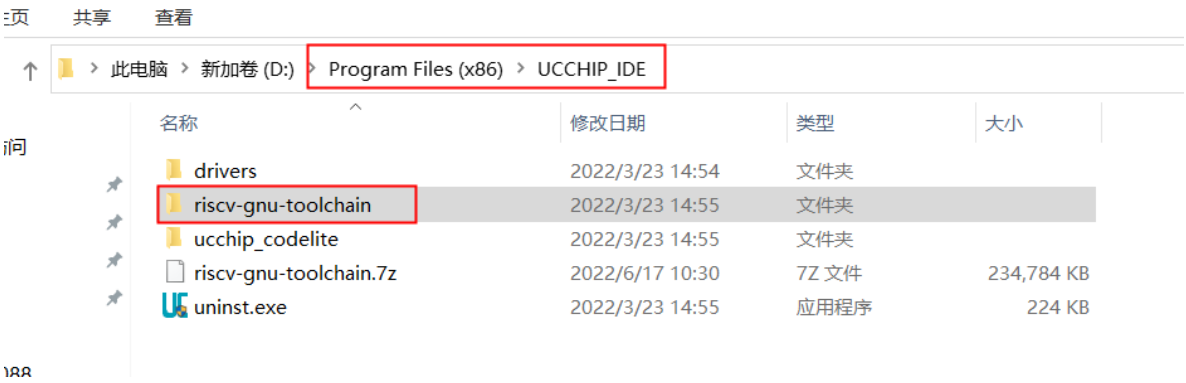
```
if   CROSS_TOOL == 'gcc':
     PLATFORM    = 'gcc'
     EXEC_PATH   = r'D:/Program Files (x86)/UCCHIP IDE/riscv-gnu-toolchain/bin/'
else:
     print('Please make sure your toolchains is GNU GCC!')
     exit(0)
```

工具链路径查找：

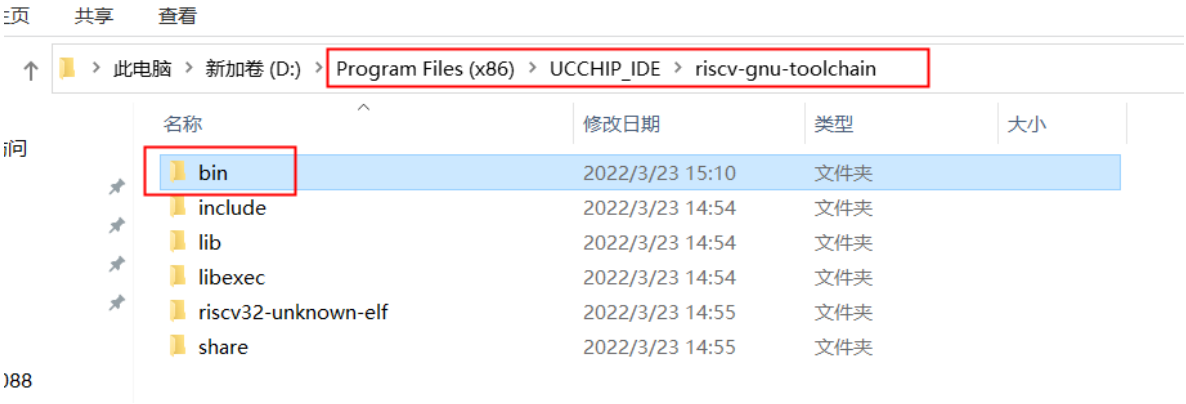
如果已经正确按照2.3.1的步骤安装了codelite，则在电脑桌面会看到UCCIP_IDE的图标，鼠标右键点击改图标选择“打开文件所在的位置”，如下图：



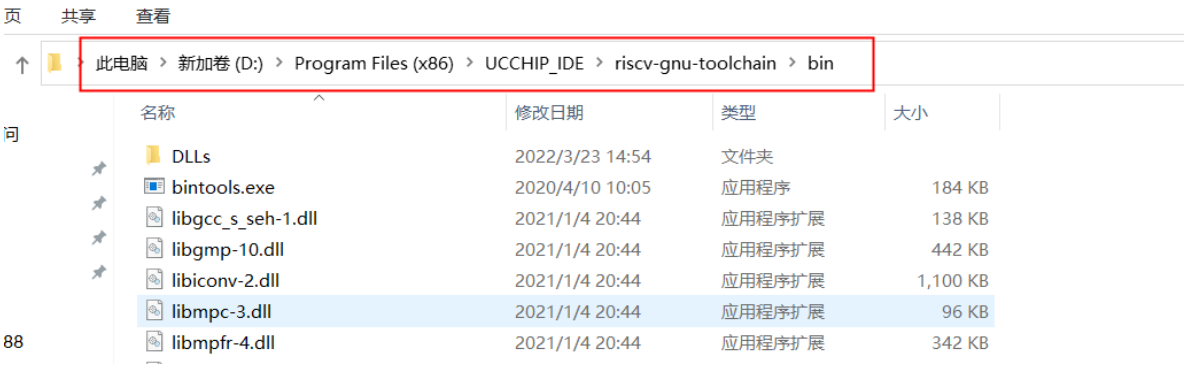
点击红框所示的“UCCHIP_IDE”，也就是当前目录的上级目录，如下图：



双击红框目录“riscv-gun-toolchain”进入，如下图：



双击红框目录“riscv-gun-toolchain”进入，如下图：



该目录即为工具链的路径，用鼠标左键单击红框右侧即可复制路径。

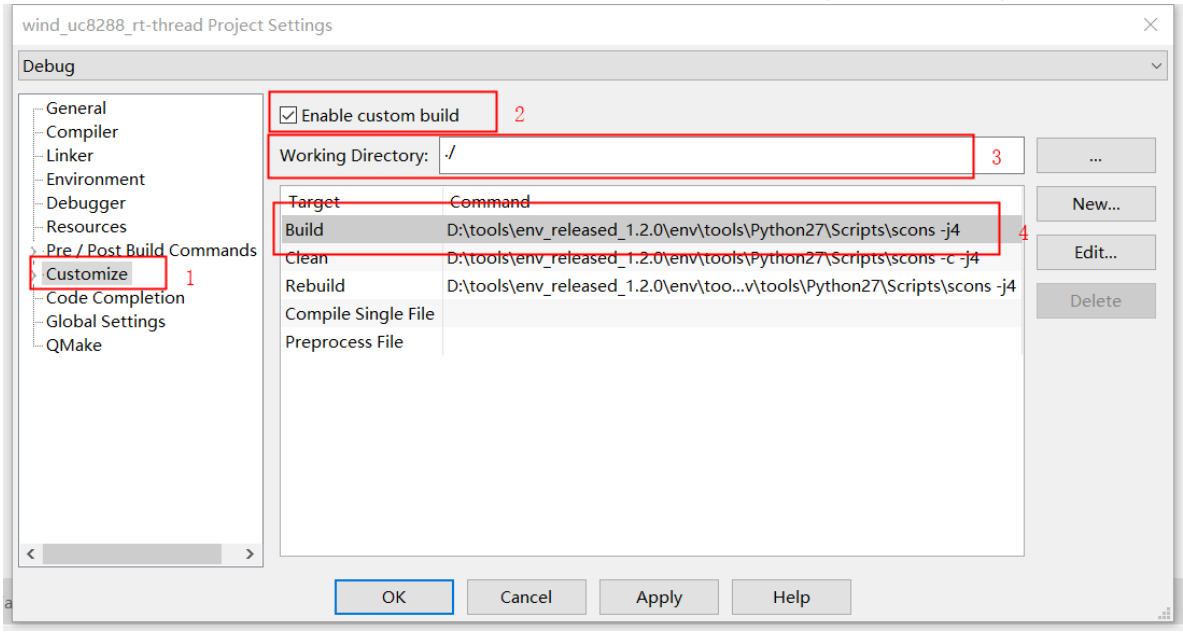
2.3.2.2 用命令行编译工程

在代码根目录下打开rt-thread的控制窗口，输入编译命令scons -j4即可编译，编译完成后会在代码根目录生成rtthread.bin、rtthread.elf，该文件即为该工程的镜像文件和符号文件。

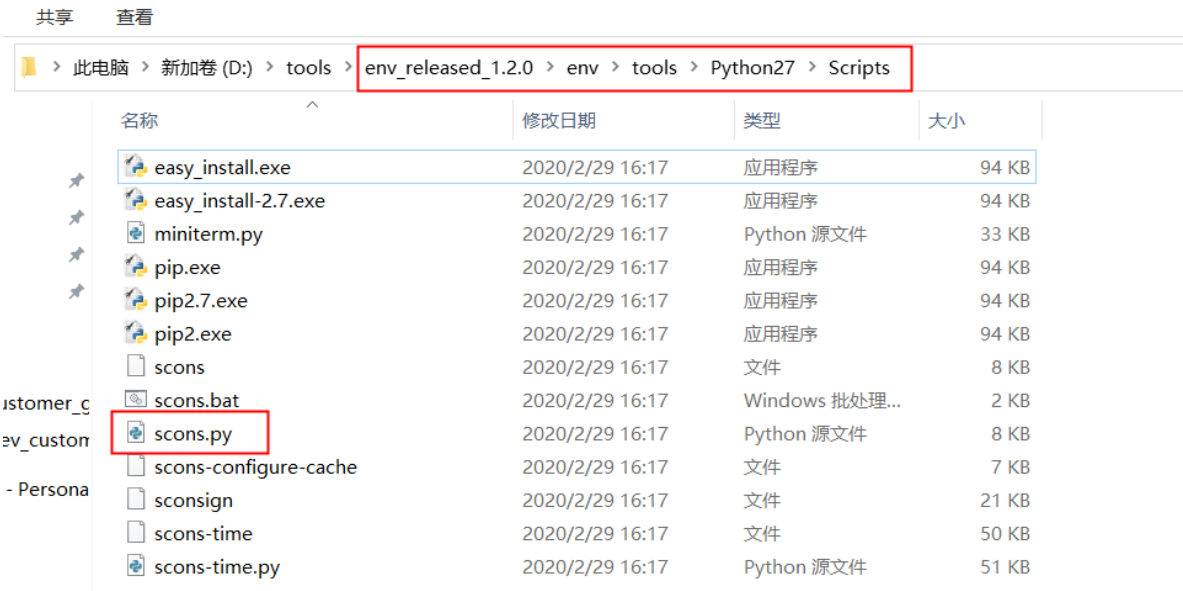
如需清除编译则输入：scons -c -j4该命令会清除执行scons -j4时生成的临时文件和目标文件。

2.3.2.3 用codelite编译工程

打开codelite并打开工程，选择windows工程，鼠标右键点击选择Settings，打开Setting窗口如下：



按照序号操作，选择Customize--->勾选Enable custom build--->Working Directory填写当前路径./--->双击编辑Build选择scons脚本路径（env_released_1.2.0\env\tools\Python27\Scripts\scons）。
例：
Build：D:\tools\env_released_1.2.0\env\tools\Python27\Scripts\scons -j4
Clean：D:\tools\env_released_1.2.0\env\tools\Python27\Scripts\scons -c -j4
Rebuild：D:\tools\env_released_1.2.0\env\tools\Python27\Scripts\scons -c -j4;D:\tools\env_released_1.2.0\env\tools\Python27\Scripts\scons -j4（重新构建：即先清除，后构建）



2.4 scons编译用户添加的文件

下面以在wiota_dev_customer的根目录下添加hello_wiota模块为例：
首先在根目录下新建文件夹并重命名为hello_wiota，然后在hello_wiota下新建一个hello_wiota.c、hello_wiota.h和Sconscript文件（该文件的文件名必须为Sconscript且无后缀格式），Sconscript文件内容如下：

```
from building import *

cwd          = GetCurrentDir()    # 获取当前脚本路径
include_path = [cwd]              # 将当前路径加入构建搜索的头文件路径
```

```
src          = Split('''
hello_wiota.c
''')                                # 源文件
# 也可等效成 src = ['hello_wiota.c'] 或 src= Glob('*.c')

# 使用DefineGroup创建一个名为hello_wiota的组
# 该Group是否被添加到工程中，参与编译，取决于depend的宏是否在Kconfig中被使能。为空则不依赖宏
group = DefineGroup('hello_wiota', src, depend = [''], CPPPATH = include_path)

Return('group')                     # 将当前脚本指定的构建对象返回上级SCons脚本
```

完成上述步骤再次执行scons -j4时新添加的hello_wiota就会被编译了。**关于更多scons相关的请查看rt-thread官网进行学习**https://www.rt-thread.org/document/site/#/development-tools/build-config-system/summary

2.5 内部已使用外设

2.5.1 终端侧

- RTC：定时，闹钟。
- ADC：温度获取。
- uart0： AT cmd。
- uart1： 协议栈LOG。
- timer0： 系统时钟。
- GPIO：index 2/3/7/16/17，用作协议栈状态展示灯。

2.5.2 基站侧

- uart0： AT cmd。
- uart1： 协议栈LOG。
- timer0： 系统时钟。
- timer1： 校准时钟。