RTL8720D 开发环境搭建

修改记录

类型	修改内容	修改人	日期	软件版本
A	初版	杨宾	2019/06/04	-
M	增加硬件连接和代码结构简要说明	杨宾	2019/12/12	-

类型：A-新增 M-修改 D-删除

1. 环境搭建注意事项

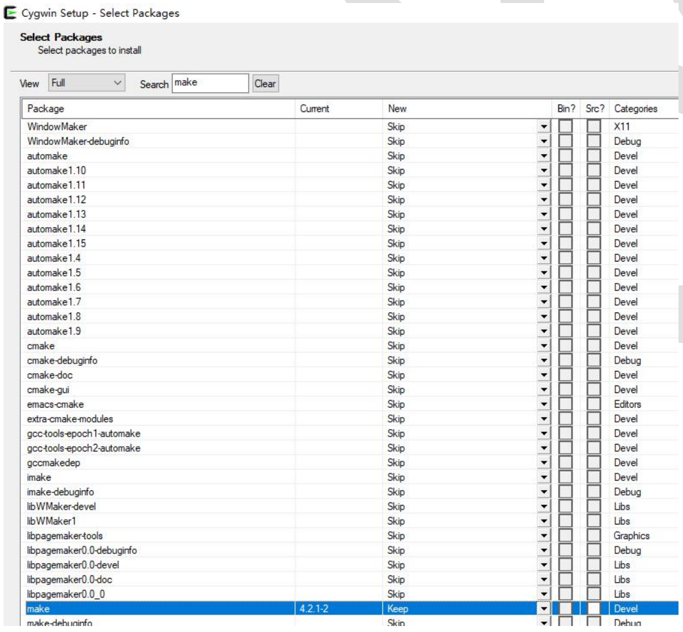
Windows 使用 Cygwin 环境，必须使用 Cygwin　32 位版本(64 位 windows 也要使用 32 位的 Cygwin)
本教程开发环境使用的是 32 位的 Cygwin。

2. 编译环境搭建

编译环境主要有两种，一种是 window 下的 Cygwin 或者使用 ubuntu（测试环境使用的 ubuntu1604）环境搭建（Cygwin 搭建后实际上就是一个 linux 环境，后续操作和 ubuntu 操作基本一致）

2.1 安装 Cygwin(Cygwin 和 unbunt 环境选一个即可)

安装 cygwin，在软件包的地方选择安装 make，bc，gawk 工具

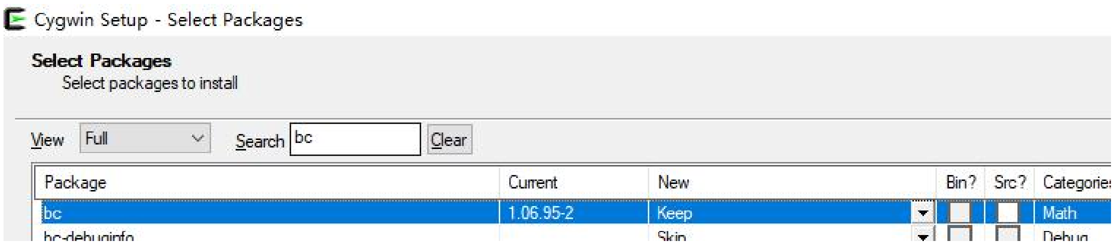


Cygwin Setup - Select Packages

Select Packages
Select packages to install

View: Full　Search: make　Clear

Package	Current	New	Bin?	Src?	Categories
WindowMaker		Skip			X11
WindowMaker-debuginfo		Skip			Debug
automake		Skip			Devel
automake1.10		Skip			Devel
automake1.11		Skip			Devel
automake1.12		Skip			Devel
automake1.13		Skip			Devel
automake1.14		Skip			Devel
automake1.15		Skip			Devel
automake1.4		Skip			Devel
automake1.5		Skip			Devel
automake1.6		Skip			Devel
automake1.7		Skip			Devel
automake1.8		Skip			Devel
automake1.9		Skip			Devel
cmake		Skip			Devel
cmake-debuginfo		Skip			Debug
cmake-doc		Skip			Devel
cmake-gui		Skip			Devel
emacs-cmake		Skip			Editors
extra-cmake-modules		Skip			Devel
gcc-tools-epoch1-automake		Skip			Devel
gcc-tools-epoch2-automake		Skip			Devel
gccmakedep		Skip			Devel
imake		Skip			Devel
imake-debuginfo		Skip			Debug
libWMaker-devel		Skip			Libs
libWMaker1		Skip			Libs
libpagemaker-tools		Skip			Graphics
libpagemaker0.0-debuginfo		Skip			Debug
libpagemaker0.0-devel		Skip			Libs
libpagemaker0.0-doc		Skip			Libs
libpagemaker0.0_0		Skip			Libs
make	4.2.1-2	Keep			Devel
make-debuginfo		Skip			Debug

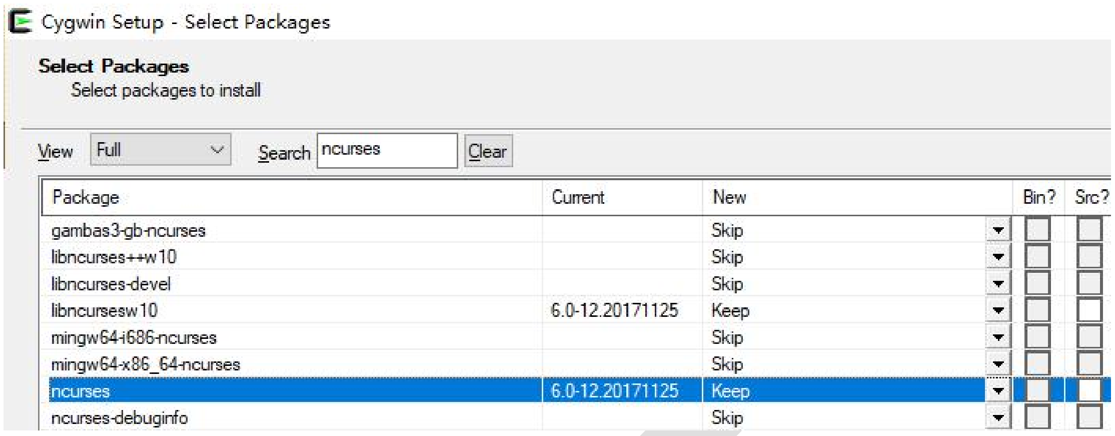


Cygwin Setup - Select Packages

Select Packages
Select packages to install

View: Full　Search: bc　Clear

Package	Current	New	Bin?	Src?	Categories
bc	1.06.95-2	Keep			Math
bc-debuginfo		Skip			Debug


Cygwin Setup - Select Packages

Select Packages
Select packages to install

View: Full   Search: ncurses   Clear

Package	Current	New	Bin?	Src?
gambas3-gb-ncurses		Skip	☐	☐
libncurses++w10		Skip	☐	☐
libncurses-devel		Skip	☐	☐
libncursesw10	6.0-12.20171125	Keep	☐	☐
mingw64-i686-ncurses		Skip	☐	☐
mingw64-x86_64-ncurses		Skip	☐	☐
ncurses	6.0-12.20171125	Keep	☐	☐
ncurses-debuginfo		Skip	☐	☐

2.2 安装 ubuntu 环境(Cygwin 和 unbunt 环境选一个即可)

(1) 安装 ubuntu 或者虚拟机（详细步骤可以参考网上教程），这里使用的是 ubuntu1604
(2) Ubuntu 命令行执行下面的指令安装依赖包

```
sudo apt-get install libc6-i386 lib32ncurses5 make bc gawk ncurses
```

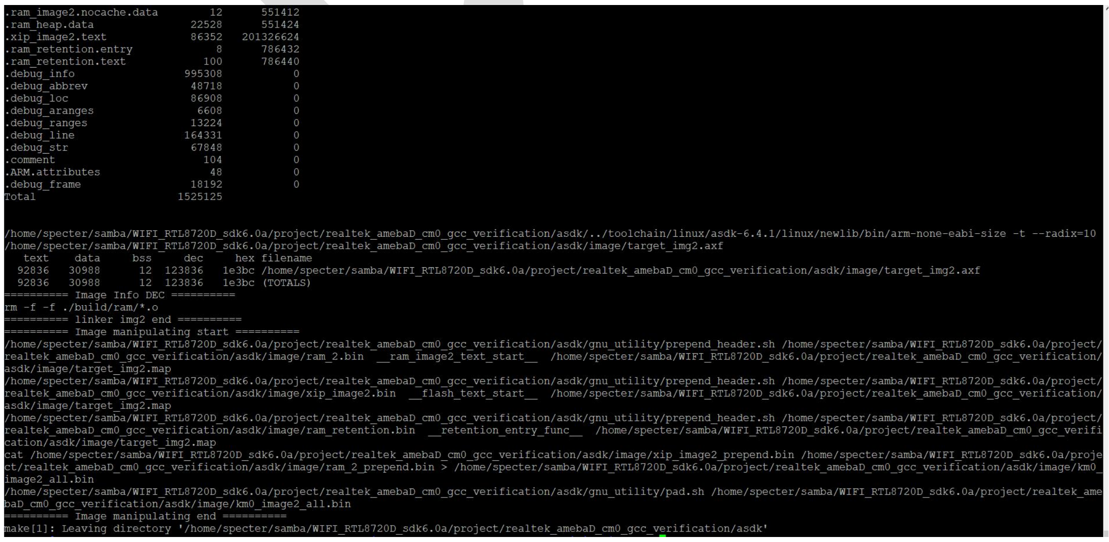
2.3 拷贝代码

拷贝并解压 sdk 代码

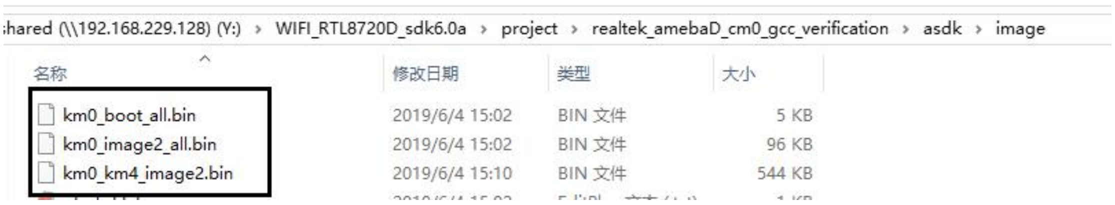
2.4 编译

2.4.1 编译 km0

进入 sdk/project/realtek_amebaD_cm0_gcc_verification 执行 make all
编译成功结果如下


```
.ram_image2.nocache.data        12      551412
.ram_heap.data               22528      551424
.xip_image2.text             86352   201326624
.ram_retention.entry             8      786432
.ram_retention.text            100      786440
.debug_info                 995308           0
.debug_abbrev                48718           0
.debug_loc                   86908           0
.debug_aranges                6608           0
.debug_ranges                13224           0
.debug_line                 164331           0
.debug_str                   67848           0
.comment                       104           0
.ARM.attributes                 48           0
.debug_frame                 18192           0
Total                      1525125


/home/specter/samba/WIFI_RTL8720D_sdk6.0a/project/realtek_amebaD_cm0_gcc_verification/asdk/../toolchain/linux/asdk-6.4.1/linux/newlib/bin/arm-none-eabi-size -t --radix=10
/home/specter/samba/WIFI_RTL8720D_sdk6.0a/project/realtek_amebaD_cm0_gcc_verification/asdk/image/target_img2.axf
   text    data     bss     dec     hex filename
  92836   30988      12  123836   1e3bc /home/specter/samba/WIFI_RTL8720D_sdk6.0a/project/realtek_amebaD_cm0_gcc_verification/asdk/image/target_img2.axf
  92836   30988      12  123836   1e3bc (TOTALS)
========== Image Info DEC ==========
rm -f -f ./build/ram/*.o
========== linker img2 end ==========
========== Image manipulating start ==========
/home/specter/samba/WIFI_RTL8720D_sdk6.0a/project/realtek_amebaD_cm0_gcc_verification/asdk/gnu_utility/prepend_header.sh /home/specter/samba/WIFI_RTL8720D_sdk6.0a/project/
realtek_amebaD_cm0_gcc_verification/asdk/image/ram_2.bin  __ram_image2_text_start__  /home/specter/samba/WIFI_RTL8720D_sdk6.0a/project/realtek_amebaD_cm0_gcc_verification/
asdk/image/target_img2.map
/home/specter/samba/WIFI_RTL8720D_sdk6.0a/project/realtek_amebaD_cm0_gcc_verification/asdk/gnu_utility/prepend_header.sh /home/specter/samba/WIFI_RTL8720D_sdk6.0a/project/
realtek_amebaD_cm0_gcc_verification/asdk/image/xip_image2.bin  __flash_text_start__  /home/specter/samba/WIFI_RTL8720D_sdk6.0a/project/realtek_amebaD_cm0_gcc_verification/
asdk/image/target_img2.map
/home/specter/samba/WIFI_RTL8720D_sdk6.0a/project/realtek_amebaD_cm0_gcc_verification/asdk/gnu_utility/prepend_header.sh /home/specter/samba/WIFI_RTL8720D_sdk6.0a/project/
realtek_amebaD_cm0_gcc_verification/asdk/image/ram_retention.bin  __retention_entry_func__  /home/specter/samba/WIFI_RTL8720D_sdk6.0a/project/realtek_amebaD_cm0_gcc_verifi
cation/asdk/image/target_img2.map
cat /home/specter/samba/WIFI_RTL8720D_sdk6.0a/project/realtek_amebaD_cm0_gcc_verification/asdk/image/xip_image2_prepend.bin /home/specter/samba/WIFI_RTL8720D_sdk6.0a/proje
ct/realtek_amebaD_cm0_gcc_verification/asdk/image/ram_2_prepend.bin > /home/specter/samba/WIFI_RTL8720D_sdk6.0a/project/realtek_amebaD_cm0_gcc_verification/asdk/image/km0_
image2_all.bin
/home/specter/samba/WIFI_RTL8720D_sdk6.0a/project/realtek_amebaD_cm0_gcc_verification/asdk/gnu_utility/pad.sh /home/specter/samba/WIFI_RTL8720D_sdk6.0a/project/realtek_ame
baD_cm0_gcc_verification/asdk/image/km0_image2_all.bin
========== Image manipulating end ==========
make[1]: Leaving directory '/home/specter/samba/WIFI_RTL8720D_sdk6.0a/project/realtek_amebaD_cm0_gcc_verification/asdk'
```

生成文件在 sdk\project\realtek_amebaD_cm0_gcc_verification\asdk\image 目录下

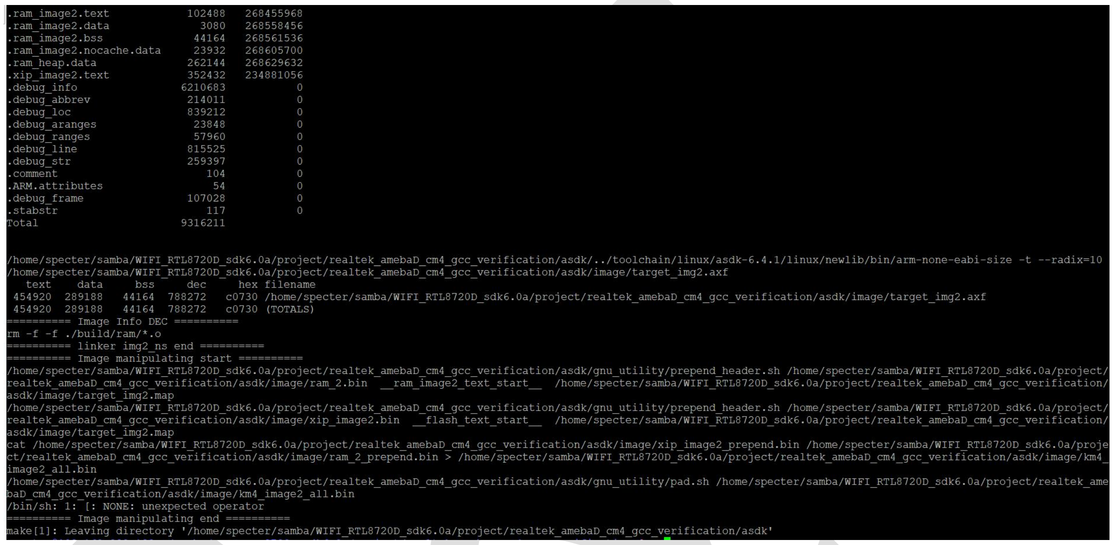


```
shared (\\192.168.229.128) (Y:) › WIFI_RTL8720D_sdk6.0a › project › realtek_amebaD_cm0_gcc_verification › asdk › image

名称                    修改日期            类型        大小
km0_boot_all.bin        2019/6/4 15:02     BIN 文件      5 KB
km0_image2_all.bin      2019/6/4 15:02     BIN 文件     96 KB
km0_km4_image2.bin      2019/6/4 15:10     BIN 文件    544 KB
```

2.4.2 编译 km4

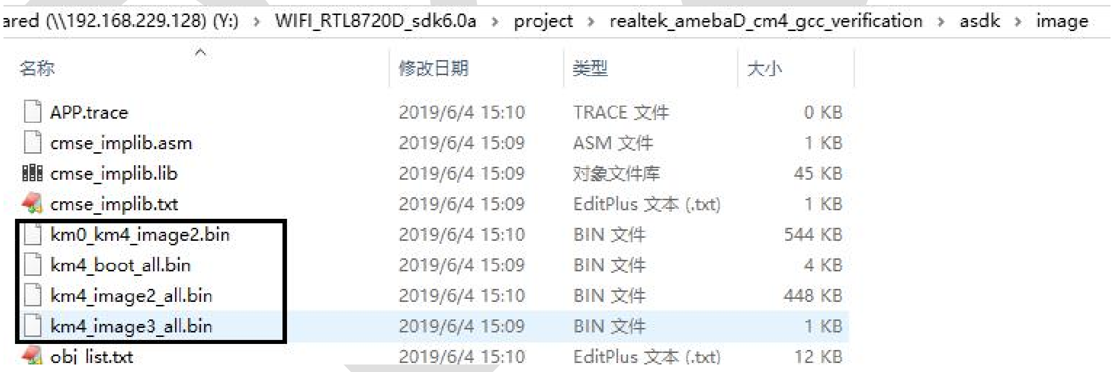
进入 sdk/project/realtek_amebaD_cm4_gcc_verification 执行 make all

编译成功结果如下



```
.ram_image2.text            102488   268455968
.ram_image2.data              3080   268558456
.ram_image2.bss              44164   268561536
.ram_image2.nocache.data     23932   268605700
.ram_heap.data              262144   268629632
.xip_image2.text            352432   234881056
.debug_info                6210683           0
.debug_abbrev               214011           0
.debug_loc                  839212           0
.debug_aranges               23848           0
.debug_ranges                57960           0
.debug_line                 815525           0
.debug_str                  259397           0
.comment                       104           0
.ARM.attributes                 54           0
.debug_frame                107028           0
.stabstr                       117           0
Total                      9316211


/home/specter/samba/WIFI_RTL8720D_sdk6.0a/project/realtek_amebaD_cm4_gcc_verification/asdk/../toolchain/linux/asdk-6.4.1/linux/newlib/bin/arm-none-eabi-size -t --radix=10
/home/specter/samba/WIFI_RTL8720D_sdk6.0a/project/realtek_amebaD_cm4_gcc_verification/asdk/image/target_img2.axf
   text    data     bss     dec     hex filename
 454920  289188   44164  788272   c0730 /home/specter/samba/WIFI_RTL8720D_sdk6.0a/project/realtek_amebaD_cm4_gcc_verification/asdk/image/target_img2.axf
 454920  289188   44164  788272   c0730 (TOTALS)
========== Image Info DEC ==========
rm -f -f ./build/ram/*.o
========== linker img2_ns end ==========
========== Image manipulating start ==========
/home/specter/samba/WIFI_RTL8720D_sdk6.0a/project/realtek_amebaD_cm4_gcc_verification/asdk/gnu_utility/prepend_header.sh /home/specter/samba/WIFI_RTL8720D_sdk6.0a/project/
realtek_amebaD_cm4_gcc_verification/asdk/image/ram_2.bin  __ram_image2_text_start__  /home/specter/samba/WIFI_RTL8720D_sdk6.0a/project/realtek_amebaD_cm4_gcc_verification/
asdk/image/target_img2.map
/home/specter/samba/WIFI_RTL8720D_sdk6.0a/project/realtek_amebaD_cm4_gcc_verification/asdk/gnu_utility/prepend_header.sh /home/specter/samba/WIFI_RTL8720D_sdk6.0a/project/
realtek_amebaD_cm4_gcc_verification/asdk/image/xip_image2.bin  __flash_text_start__  /home/specter/samba/WIFI_RTL8720D_sdk6.0a/project/realtek_amebaD_cm4_gcc_verification/
asdk/image/target_img2.map
cat /home/specter/samba/WIFI_RTL8720D_sdk6.0a/project/realtek_amebaD_cm4_gcc_verification/asdk/image/xip_image2_prepend.bin /home/specter/samba/WIFI_RTL8720D_sdk6.0a/proje
ct/realtek_amebaD_cm4_gcc_verification/asdk/image/ram_2_prepend.bin > /home/specter/samba/WIFI_RTL8720D_sdk6.0a/project/realtek_amebaD_cm4_gcc_verification/asdk/image/km4_
image2_all.bin
/home/specter/samba/WIFI_RTL8720D_sdk6.0a/project/realtek_amebaD_cm4_gcc_verification/asdk/gnu_utility/pad.sh /home/specter/samba/WIFI_RTL8720D_sdk6.0a/project/realtek_ame
baD_cm4_gcc_verification/asdk/image/km4_image2_all.bin
/bin/sh: 1: [: NONE: unexpected operator
========== Image manipulating end ==========
make[1]: Leaving directory '/home/specter/samba/WIFI_RTL8720D_sdk6.0a/project/realtek_amebaD_cm4_gcc_verification/asdk'
```

生成文件在 sdk\project\realtek_amebaD_cm4_gcc_verification\asdk\image 目录下



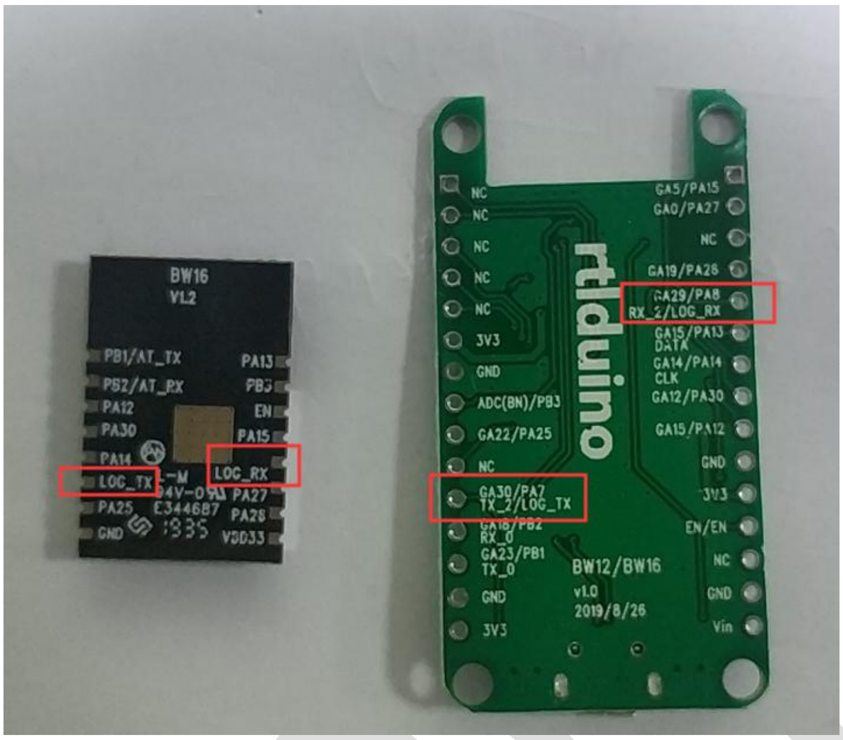
```
ared (\\192.168.229.128) (Y:) › WIFI_RTL8720D_sdk6.0a › project › realtek_amebaD_cm4_gcc_verification › asdk › image

名称                    修改日期            类型                  大小
APP.trace               2019/6/4 15:10     TRACE 文件              0 KB
cmse_implib.asm         2019/6/4 15:09     ASM 文件                1 KB
cmse_implib.lib         2019/6/4 15:09     对象文件库              45 KB
cmse_implib.txt         2019/6/4 15:09     EditPlus 文本 (.txt)    1 KB
km0_km4_image2.bin      2019/6/4 15:10     BIN 文件              544 KB
km4_boot_all.bin        2019/6/4 15:09     BIN 文件                4 KB
km4_image2_all.bin      2019/6/4 15:10     BIN 文件              448 KB
km4_image3_all.bin      2019/6/4 15:09     BIN 文件                1 KB
obj_list.txt            2019/6/4 15:10     EditPlus 文本 (.txt)   12 KB
```

3. 串口下载

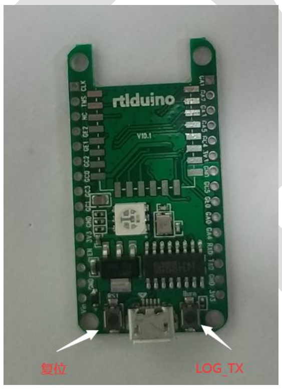
将 KM0 和 KM4 都编译完成后可以使用串口将编译好的固件下载到模块。

串口下载软件使用 sdk\tools\AmebaZ\Image_Tool\ImageTool.exe 工具

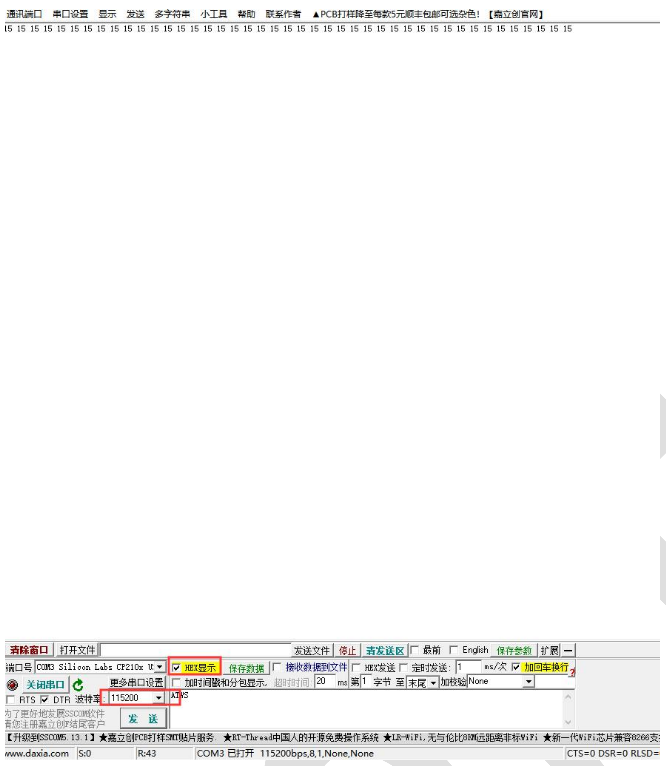
硬件需要用 USB 转 TLL 串口连接模块的 log 串口（LOG_TX(PA7),LOG_RX(PA8)）进行下载，接口如下，左侧为模块，右侧为开发底板（开发底板上有两个丝印的看斜杠右侧的丝印）



串口下载需要模块进入下载模式，进入下载模式的方法如下
如果使用开发底板
(1) 连接好 VCC/GND 和 LOG_TX/LOG_RX
(2) 按住右边的按键不要松开
(3) 按下左边的复位按键
(4) 松开右边的 LOG_TX，此时模块进入烧录模式



(5) 此时为了检查模块是否处于下载模式可以打开串口工具,波特率 115200,8,N,1，此时用 16 进制显示，如果看到串口如下图一样，不断接收到数据，则表示进入的烧录模式，如果模块打印正常的启动 log，这表示没有进入烧录模式，需要重复上述操作，直到进入烧录模式。



```
15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15
```

端口号 COM3 Silicon Labs CP210x U　☑ HEX显示　☐ HEX发送　☐ 定时发送：1 ms/次　☑ 加回车换行
☐ 加时间戳和分包显示，超时时间：20 ms 第 1 字节 至 末尾 加校验 None
☐ RTS ☑ DTR 波特率：115200

```
ATWS
```

【升级到SSCOM5.13.1】★嘉立创PCB打样SMT贴片服务. ★RT-Thread中国人的开源免费操作系统 ★LR-WiFi,无与伦比8KM远距离非标WiFi ★新一代WiFi芯片兼容8266支

www.daxia.com　S:0　R:43　COM3 已打开　115200bps,8,1,None,None　CTS=0 DSR=0 RLSD=

如果使用模块

(1) 首先将 LOG_TX 用 2k 电阻下拉到地（如果不用电阻下拉部分串口会卡死导致无法同分，部分串口无影响，串口卡死后需要松开 LOG_TX 然后重新插拔串口）

(2) 保持 LOG_TX 下拉，给模块上电或者按下复位键（丝印为 EN，低电平触发）

(3) 松开 LOG_TX

(4) 检查模块是否进入烧录模式和上面使用开发底板的方法相同

当模块进入下载模式之后打开烧录软件 sdk\tools\AmebaZ\Image_Tool\ImageTool.exe

(1) 点击 Chip Select->AmebaD(8721D) 选择芯片

(2) 选择要烧录的文件

烧录文件有 3 个

KM0boot：

sdk\project\realtek_amebaD_cm0_gcc_verification\asdk\image\km0_boot_all.bin

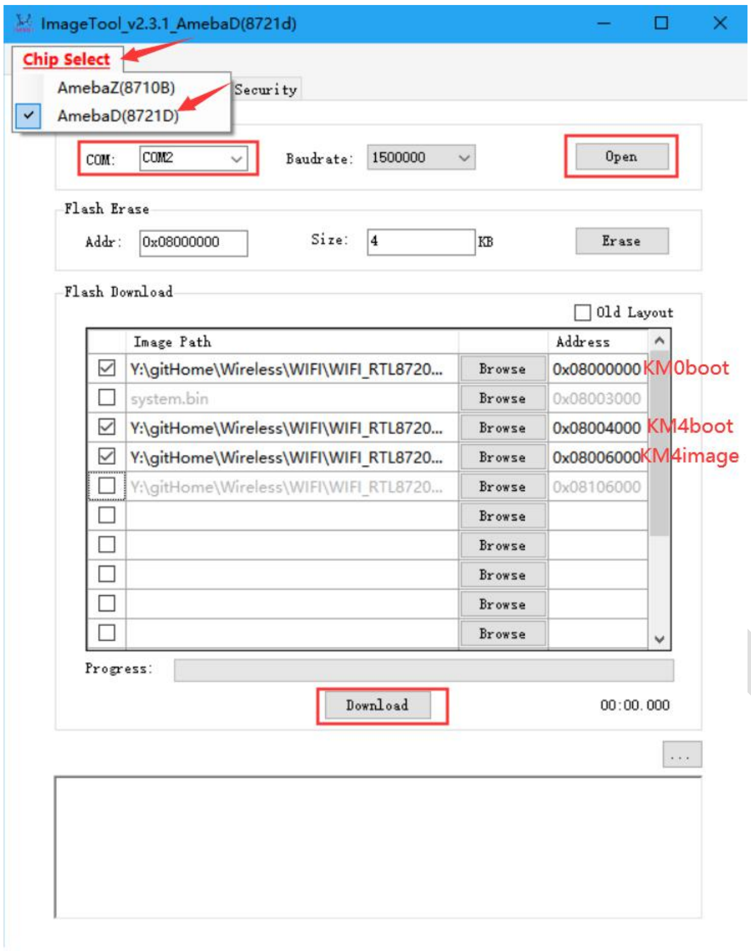
KM4boot：

sdk\project\realtek_amebaD_cm4_gcc_verification\asdk\image\km4_boot_all.bin

KM4image：

sdk\project\realtek_amebaD_cm4_gcc_verification\asdk\image\km0_km4_image2.bin

(3) 选择串口然后点击 OPEN 打开串口，然后点击 download 下载



ImageTool_v2.3.1_AmebaD(8721d)

Chip Select: AmebaZ(8710B) / [✓ AmebaD(8721D)]

COM: COM2 Baudrate: 1500000

Flash Erase — Addr: 0x08000000 Size: 4 KB

Flash Download — Old Layout: unchecked

	Image Path		Address
✓	Y:\gitHome\Wireless\WIFI\WIFI_RTL8720...	Browse	0x08000000 KM0boot
	system.bin	Browse	0x08003000
✓	Y:\gitHome\Wireless\WIFI\WIFI_RTL8720...	Browse	0x08004000 KM4boot
✓	Y:\gitHome\Wireless\WIFI\WIFI_RTL8720...	Browse	0x08006000 KM4image
	Y:\gitHome\Wireless\WIFI\WIFI_RTL8720...	Browse	0x08106000
		Browse	
		Browse	
		Browse	
		Browse	
		Browse	

Progress:

00:00.000

(4) 下载 log 如下

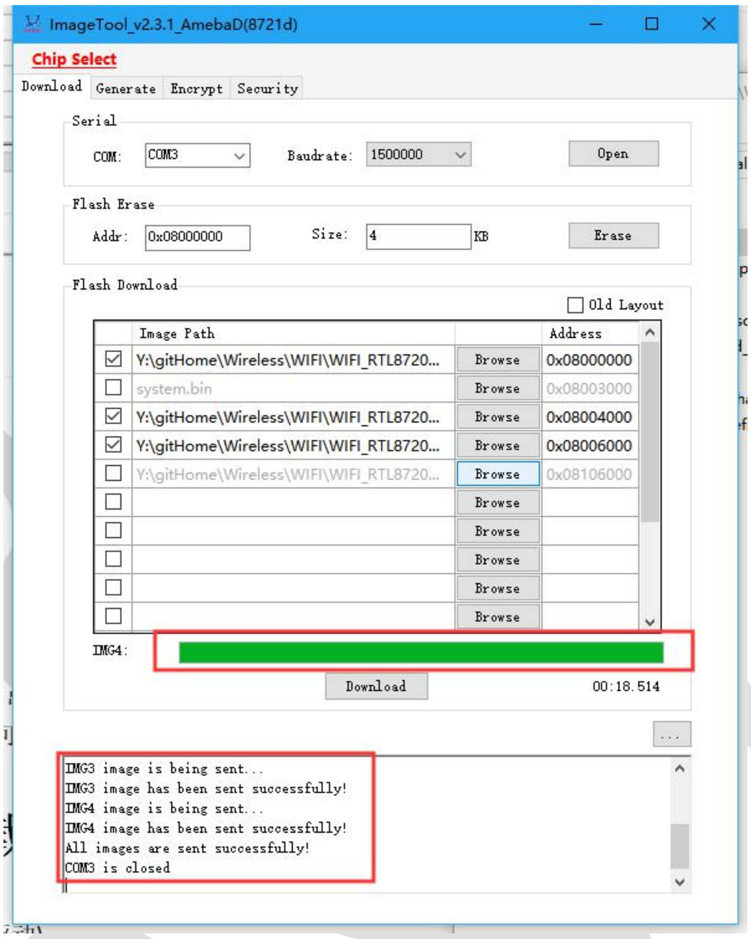


	Image Path		Address
☑	Y:\gitHome\Wireless\WIFI\WIFI_RTL8720...	Browse	0x08000000
☐	system.bin	Browse	0x08003000
☑	Y:\gitHome\Wireless\WIFI\WIFI_RTL8720...	Browse	0x08004000
☑	Y:\gitHome\Wireless\WIFI\WIFI_RTL8720...	Browse	0x08006000
☐	Y:\gitHome\Wireless\WIFI\WIFI_RTL8720...	Browse	0x08106000

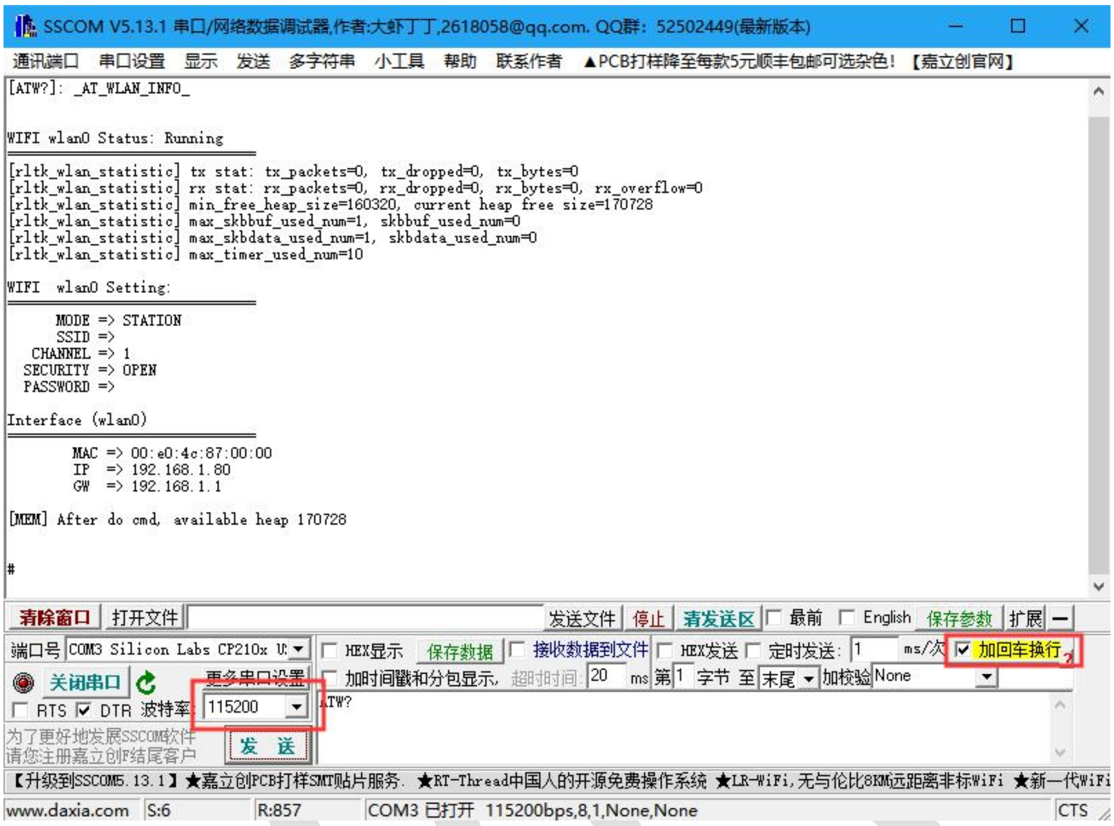
IMG4: 🟩

```
IMG3 image is being sent...
IMG3 image has been sent successfully!
IMG4 image is being sent...
IMG4 image has been sent successfully!
All images are sent successfully!
COM3 is closed
```

如果 log 卡死在 Uart download server has started...
这个一般是因为没有进入串口下载模式的原因，首先确认串口是否进入了串口下载模式。

4. 测试

下载完成后接上 log 串口（LOG_TX,LOG_RX，波特率 115200）可以正常打印 log，测试 ATW?指令测试指令是否可以正常执行，注意 AT 指令以回车换行结束。


SSCOM V5.13.1 串口/网络数据调试器,作者:大虾丁丁,2618058@qq.com. QQ群：52502449(最新版本)
通讯端口 串口设置 显示 发送 多字符串 小工具 帮助 联系作者 ▲PCB打样降至每款5元顺丰包邮可选杂色！【嘉立创官网】
清除窗口 打开文件 发送文件 停止 清发送区 最前 English 保存参数 扩展
端口号 COM3 Silicon Labs CP210x U HEX显示 保存数据 接收数据到文件 HEX发送 定时发送: 1 ms/次 [✓ 加回车换行]
关闭串口 更多串口设置 加时间戳和分包显示, 超时时间: 20 ms 第1 字节 至 末尾 加校验 None
RTS [✓ DTR] 波特率: 115200
为了更好地发展SSCOM软件 请您注册嘉立创F结尾客户
发 送
【升级到SSCOM5.13.1】★嘉立创PCB打样SMT贴片服务. ★RT-Thread中国人的开源免费操作系统 ★LR-WiFi,无与伦比8KM远距离非标WiFi ★新一代WiFi
www.daxia.com S:6 R:857 COM3 已打开 115200bps,8,1,None,None CTS

```
[ATW?]: _AT_WLAN_INFO_


WIFI wlan0 Status: Running
==============================
[rltk_wlan_statistic] tx stat: tx_packets=0, tx_dropped=0, tx_bytes=0
[rltk_wlan_statistic] rx stat: rx_packets=0, rx_dropped=0, rx_bytes=0, rx_overflow=0
[rltk_wlan_statistic] min_free_heap_size=160320, current heap free size=170728
[rltk_wlan_statistic] max_skbbuf_used_num=1, skbbuf_used_num=0
[rltk_wlan_statistic] max_skbdata_used_num=1, skbdata_used_num=0
[rltk_wlan_statistic] max_timer_used_num=10

WIFI  wlan0 Setting:
==============================
      MODE => STATION
      SSID =>
   CHANNEL => 1
  SECURITY => OPEN
  PASSWORD =>

Interface (wlan0)
==============================
        MAC => 00:e0:4c:87:00:00
        IP  => 192.168.1.80
        GW  => 192.168.1.1

[MEM] After do cmd, available heap 170728


#
```

```
ATW?
```

5. J-link 下载

(1) 安装 jlink 驱动(官方驱动)

(2) 指定 jlink 路径

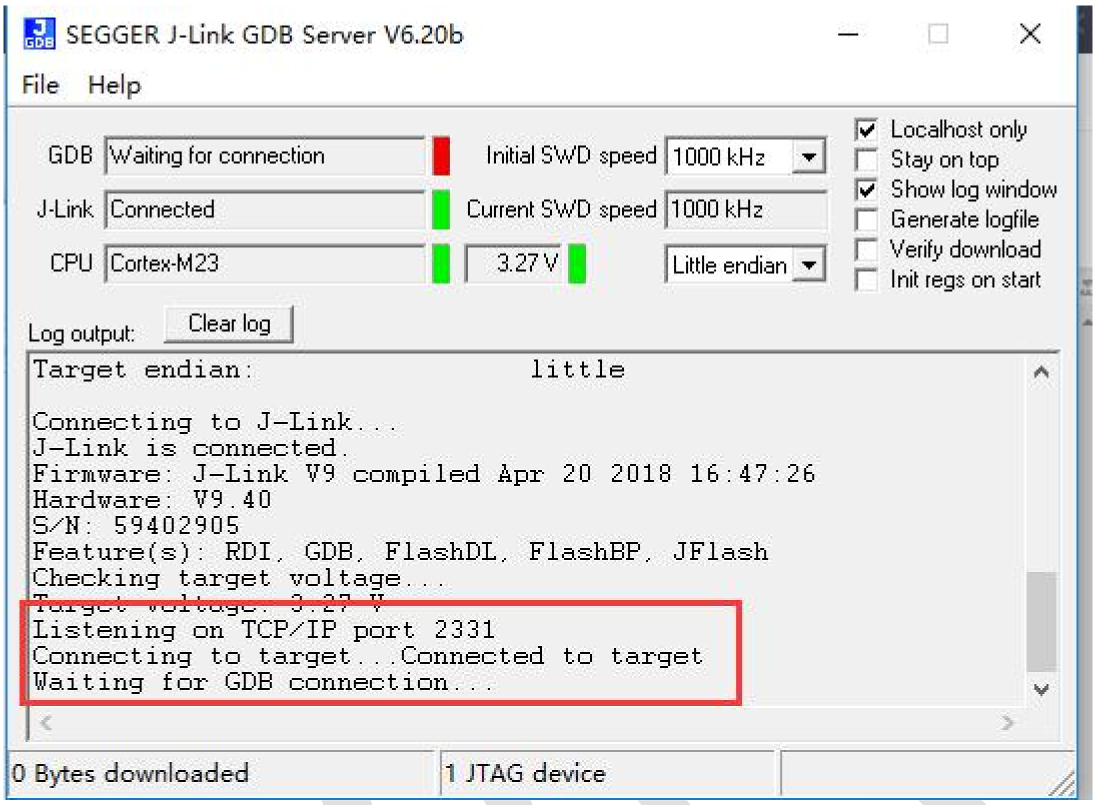
修改 sdk/project/realtek_amebaD_cm0_gcc_verification/jlink_script/cm0_jlink.bat 和 Sdk/project/realtek_amebaD_cm4_gcc_verification/jlink_script/cm4_jlink.bat 中的 jlink 路径为自己安装的路径。

(3) 设置使用 jlink 调试

分别进入 sdk/project/realtek_amebaD_cm0_gcc_verification 和 Sdk/project/realtek_amebaD_cm4_gcc_verification 执行 `make setup GDB_SERVER=jlink` 设置使用 `jlink` 为 gdb 调试工具

(4) 硬件连接将 JLINK 的 swd 接线连接好

(5) 执行 dk/project/realtek_amebaD_cm0_gcc_verification/jlink_script/cm0_jlink.bat 连接成功可以看到如下结果



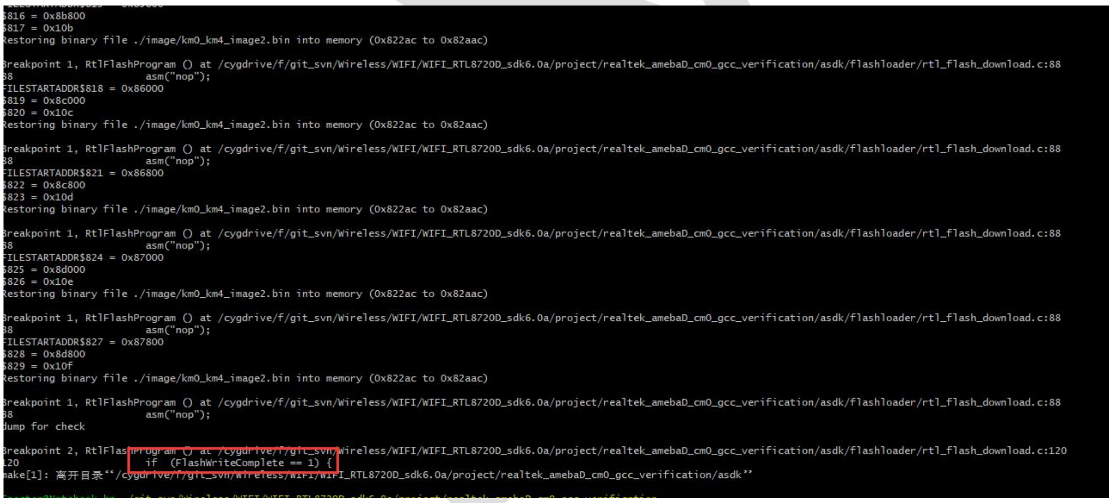
GDB Waiting for connection 🟥
J-Link Connected 🟩
CPU Cortex-M23 🟩 3.27 V 🟩
Initial SWD speed 1000 kHz
Current SWD speed 1000 kHz
Little endian
[✓] Localhost only
[ ] Stay on top
[✓] Show log window
[ ] Generate logfile
[ ] Verify download
[ ] Init regs on start

Log output:

```
Target endian:                 little

Connecting to J-Link...
J-Link is connected.
Firmware: J-Link V9 compiled Apr 20 2018 16:47:26
Hardware: V9.40
S/N: 59402905
Feature(s): RDI, GDB, FlashDL, FlashBP, JFlash
Checking target voltage...
Target voltage: 3.27 V
Listening on TCP/IP port 2331
Connecting to target...Connected to target
Waiting for GDB connection...
```

0 Bytes downloaded | 1 JTAG device

(6) 进入 sdk/project/realtek_amebaD_cm0_gcc_verification 目录执行 `make flash` 此时开始下载 km0 代码，下载成功后可以看到如下结果

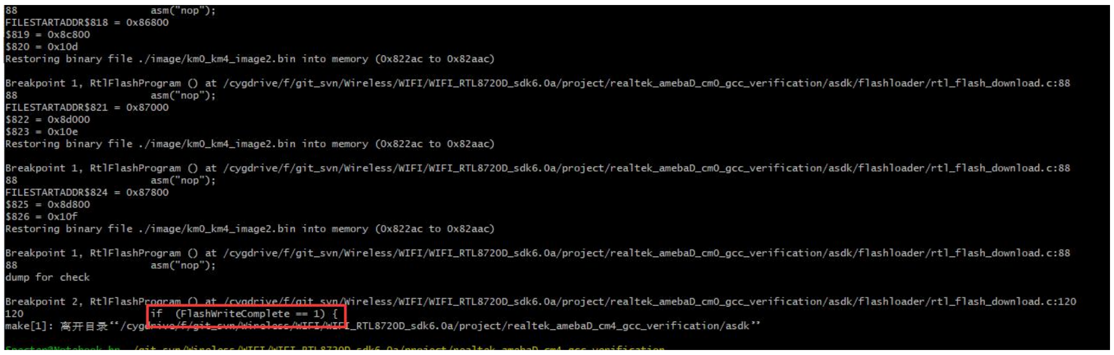


```
$816 = 0x8b800
$817 = 0x10b
Restoring binary file ./image/km0_km4_image2.bin into memory (0x822ac to 0x82aac)

Breakpoint 1, RtlFlashProgram () at /cygdrive/f/git_svn/Wireless/WIFI/WIFI_RTL8720D_sdk6.0a/project/realtek_amebaD_cm0_gcc_verification/asdk/flashloader/rtl_flash_download.c:88
88                          asm("nop");
FILESTARTADDR$818 = 0x86000
$819 = 0x8c000
$820 = 0x10c
Restoring binary file ./image/km0_km4_image2.bin into memory (0x822ac to 0x82aac)

Breakpoint 1, RtlFlashProgram () at /cygdrive/f/git_svn/Wireless/WIFI/WIFI_RTL8720D_sdk6.0a/project/realtek_amebaD_cm0_gcc_verification/asdk/flashloader/rtl_flash_download.c:88
88                          asm("nop");
FILESTARTADDR$821 = 0x86800
$822 = 0x8c800
$823 = 0x10d
Restoring binary file ./image/km0_km4_image2.bin into memory (0x822ac to 0x82aac)

Breakpoint 1, RtlFlashProgram () at /cygdrive/f/git_svn/Wireless/WIFI/WIFI_RTL8720D_sdk6.0a/project/realtek_amebaD_cm0_gcc_verification/asdk/flashloader/rtl_flash_download.c:88
88                          asm("nop");
FILESTARTADDR$824 = 0x87000
$825 = 0x8d000
$826 = 0x10e
Restoring binary file ./image/km0_km4_image2.bin into memory (0x822ac to 0x82aac)

Breakpoint 1, RtlFlashProgram () at /cygdrive/f/git_svn/Wireless/WIFI/WIFI_RTL8720D_sdk6.0a/project/realtek_amebaD_cm0_gcc_verification/asdk/flashloader/rtl_flash_download.c:88
88                          asm("nop");
FILESTARTADDR$827 = 0x87800
$828 = 0x8d800
$829 = 0x10f
Restoring binary file ./image/km0_km4_image2.bin into memory (0x822ac to 0x82aac)

Breakpoint 1, RtlFlashProgram () at /cygdrive/f/git_svn/Wireless/WIFI/WIFI_RTL8720D_sdk6.0a/project/realtek_amebaD_cm0_gcc_verification/asdk/flashloader/rtl_flash_download.c:88
88                          asm("nop");
dump for check

Breakpoint 2, RtlFlashProgram () at /cygdrive/f/git_svn/Wireless/WIFI/WIFI_RTL8720D_sdk6.0a/project/realtek_amebaD_cm0_gcc_verification/asdk/flashloader/rtl_flash_download.c:120
120                         if  (FlashWriteComplete == 1) {
make[1]: 离开目录“/cygdrive/f/git_svn/Wireless/WIFI/WIFI_RTL8720D_sdk6.0a/project/realtek_amebaD_cm0_gcc_verification/asdk”
```

(7) 重启模块！！！ 注意，这里如果不重新芯片下面的 jlink server 将无法连接，出现闪退

(8) 执行 Sdk/project/realtek_amebaD_cm4_gcc_verification/jlink_script/cm4_jlink.bat(注意：此时 cm0_jlink.bat 也不能关闭，要保持开启)

(9) 进入 Sdk/project/realtek_amebaD_cm4_gcc_verification 目录执行 `make flash` 此时开始下载 km4 代码，下载成功后可以看到如下结果



```
88                      asm("nop");
FILESTARTADDR$818 = 0x86800
$819 = 0x8c800
$820 = 0x10d
Restoring binary file ./image/km0_km4_image2.bin into memory (0x822ac to 0x82aac)

Breakpoint 1, RtlFlashProgram () at /cygdrive/f/git_svn/Wireless/WIFI/WIFI_RTL8720D_sdk6.0a/project/realtek_amebaD_cm0_gcc_verification/asdk/flashloader/rtl_flash_download.c:88
88                      asm("nop");
FILESTARTADDR$821 = 0x87000
$822 = 0x8d000
$823 = 0x10e
Restoring binary file ./image/km0_km4_image2.bin into memory (0x822ac to 0x82aac)

Breakpoint 1, RtlFlashProgram () at /cygdrive/f/git_svn/Wireless/WIFI/WIFI_RTL8720D_sdk6.0a/project/realtek_amebaD_cm0_gcc_verification/asdk/flashloader/rtl_flash_download.c:88
88                      asm("nop");
FILESTARTADDR$824 = 0x87800
$825 = 0x8d800
$826 = 0x10f
Restoring binary file ./image/km0_km4_image2.bin into memory (0x822ac to 0x82aac)

Breakpoint 1, RtlFlashProgram () at /cygdrive/f/git_svn/Wireless/WIFI/WIFI_RTL8720D_sdk6.0a/project/realtek_amebaD_cm0_gcc_verification/asdk/flashloader/rtl_flash_download.c:88
88                      asm("nop");
dump for check

Breakpoint 2, RtlFlashProgram () at /cygdrive/f/git_svn/Wireless/WIFI/WIFI_RTL8720D_sdk6.0a/project/realtek_amebaD_cm0_gcc_verification/asdk/flashloader/rtl_flash_download.c:120
120                     if  (FlashWriteComplete == 1) {
make[1]: 离开目录"/cygdrive/f/git_svn/Wireless/WIFI/WIFI_RTL8720D_sdk6.0a/project/realtek_amebaD_cm4_gcc_verification/asdk"
```

(10) 此时代码就已经全部下载完成了，下载完成后重启芯片就可以正常执行了

6. SDK 目录结构简介

RTL8720D 的内核有两个，一个是 KM0，一个是 KM4，KM0 一般不会修改，我们应用逻辑一般都是添加在 KM4 上的。

以下是一些比较重要的目录（不常用的目录已经删减）

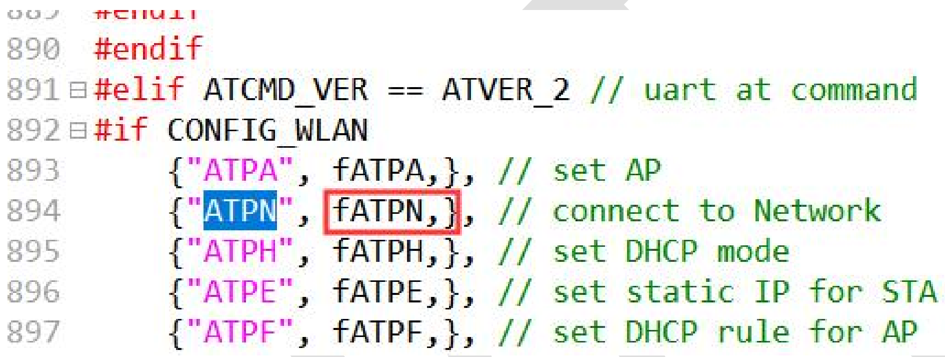
```
sdk
 ├── component
 |    ├── common
 |    |    ├── api
 |    |    |    └── at_cmd   //AT 指令相关代码
 |    |    ├── application //一些第三方应用工具
 |    |    |    ├── baidu
 |    |    |    ├── google
 |    |    |    ├── mqtt
 |    |    |    ├── wigadget
 |    |    |    └── xmodem
 |    |    └── example      //应用层示例代码(应用开发主要参考这个和 AT 指令的实现)
 |    └── os   //操作系统（freeRtos）
 └── project    //工程入口
      ├── realtek_amebaD_cm0_gcc_verification      //KM0 工程（一般不会修改这个目录下的
文件）
      |    └── asdk
      |         └── image   //编译完成后的 KM0 镜像在这个目录中
      └── realtek_amebaD_cm4_gcc_verification  //KM4 工程
             ├──src
      |      └── main.c     //KM4 镜像 main 函数
           ├── asdk
           |    └── image   //编译完成后的 KM4 镜像在这个目录中
           └── example_sources      //硬件驱动 demo 目录（目录中 mbed 和 raw 是用的两个库
函数实现，用其中一个就好）
```

KM0 的镜像我们一般不会修改，我们的驱动和应用代码一般都是在 KM4 内核上开发的，KM4 内核代码的 main 函数位于 sdk\project\realtek_amebaD_cm4_gcc_verification\src\main.c 中，任务调度系统使用的是 FreeRTOS，在 main 函数最后一行启动的任务调度器，我们可以根据自己的需要创建任务，添加自己的代码。

硬件驱动主要参考 sdk\project\realtek_amebaD_cm4_gcc_verification\example_sources 中的例子。

应用层主要参考 sdk\component\common\example 和 sdk\component\common\api\at_cmd 中的例子，其中前者是一些 demo 的最简 demo，后者是现有 AT 指令集的实现。

参考 AT 指令集的实现可以首先查看 AT 指令手册，例如我们要实现连接 wifi 功能，就可以参考 AT 指令中的 ATPN，然后我们搜索 ATPN 就可以找到如下内容，这个表示 ATPN 对应的实现函数就是 fATPN。



```
#endif
#elif ATCMD_VER == ATVER_2 // uart at command
#if CONFIG_WLAN
	{"ATPA", fATPA,}, // set AP
	{"ATPN", fATPN,}, // connect to Network
	{"ATPH", fATPH,}, // set DHCP mode
	{"ATPE", fATPE,}, // set static IP for STA
	{"ATPF", fATPF,}, // set DHCP rule for AP
```

然后我们就可以参 fATPN 的实现或者直接调用这个函数也可以，函数的参数就是我们输入的 AT 指令的参数。

例如 AT 指令执行 ATPN=test01,123456789 ，那么就可以直接 d 调 fATPN(”test01,123456789”); 实现相应的 AT 指令功能。