

---

# BK7221U SDK 快速入门

---

RT-THREAD 文档中心

上海睿赛德电子科技有限公司版权 ©2019



[WWW.RT-THREAD.ORG](http://WWW.RT-THREAD.ORG)

Monday 6<sup>th</sup> May, 2019

# 目录

目录	i
<b>1 bk7221u SDK 快速上手指南</b>	<b>1</b>
1.1 前提条件	1
1.1.1 硬件准备	1
1.1.2 上位机工具准备	1
1.1.3 RT-Thread 代码和编译环境准备	1
1.2 步骤一下载 all.bin	2
1.3 步骤二查看运行结果	3
1.4 步骤三编译	3
1.5 步骤四 OTA 固件升级	4
1.6 进一步运行 SDK 示例	6
1.6.1 Wi-Fi 接入	6
1.7 进阶教程	7
1.7.1 分区表工具使用	7
1.7.2 固件打包工具使用	8
1.7.3 其他下载程序的方式	9
1.8 Ubuntu 平台开发 RT-Thread	10
1.8.1 准备工作	10
1.8.2 编译和运行 RT-Thread	11

## 第 1 章

# bk7221u SDK 快速上手指南

文档主要针对基于 RT-Thread 操作系统的项目工程，提供快速上手的开发说明。说明了开发条件、工程编译、下载，以及查看运行情况，其中环境搭建参考文档 [RT-Thread env 工具用户手册](#)。另外，该文档还提供了该平台的几个重要上手示例。

### 1.1 前提条件

在该平台快速上手之前，有如下三个前提条件。

#### 1.1.1 硬件准备

- bk7221u 开发板
- USB 转串口设备及 PC 机
- 正确连接 TTL 串口设备与 beken 设备、TTL 串口设备与 PC 机
- 正确连接下载器与 beken 设备、下载器与 PC 机正确连接
- 给硬件设备正确供电

#### 1.1.2 上位机工具准备

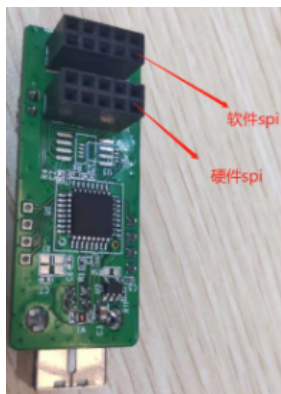
- (a) 打开串口工具软件, 配置串口波特率 115200, 数据位 8 位, 停止位 1 位, 无校验, 无流控。
- (b) 打开代码烧录工具 **Hid Download Tool V2.3.1**
- (c) 打开固件打包工具 **rt\_ota\_packaging\_tool**

#### 1.1.3 RT-Thread 代码和编译环境准备

需要使用 env 开发辅助工具，该工具提供编译构建环境、图形化系统配置及软件包管理功能，详情考文档 [RT-Thread env 工具用户手册](#)，搭建代码编译环境。

## 1.2 步骤一下载 all.bin

- (a) bk7221u 连接硬件下载器，使用硬件 spi 的方式连接，如下图所示



- (b) 通过 Hid Download Tool 工具，下载 all\_uart2.bin 文件到 Flash 。

Hid Download Tool V2.3.1--当前空闲[0]设备

下载模式 版本信息



图 1.1: hardware

注: 程序默认使用 uart2, 通过此方式下载程序比较慢, 一般是第一次拿到设备时, 或者程序死机时通过此方式下载, 以后程序的下载通过 OTA 完成。

## 1.3 步骤二查看运行结果

beken72xx 开发板的工程默认使用串口 2, 下载代码后, 连接串口, 打开终端工具, 复位开发板, 打印如下图所示:

```
\ | /
- RT -      Thread Operating System
/ | \      3.1.0 build Jun 22 2018
2006 - 2018 Copyright by rt-thread team
[FUNC]rwnxl_init
[FUNC]calibration_main
DPLL Unlock
NO TXPWR_TAB_TAB found in flash
Load default txpwr for b:0009288c
Load default txpwr for g:0009289a
fit n20 table with dist:2
Load default txpwr for n40:00091fda
init temp pwr table: mod:8, pa:8, tmp:315, idx:7, dist:0
[FUNC]ps_init
[FUNC]func_init OVER!!!

lwIP-2.0.2 initialized!
No TLV header found in flash
No TLV header found in flash
igmp_mac_filter add 224.0.0.1 01:00:5E:00:00:01
register station wlan device<sucess!
igmp_mac_filter add 224.0.0.1 01:00:5E:00:00:01
register soft-ap wlan device sucess!
beken wlan hw init
app_init finished
app_init finished
ate_start
msh />set_tmp_pwr: indx:6, mod:6, pa:8, tmp:291
```

## 1.4 步骤三编译

打开 env 工具, 切换到工程目录下, 使用命令 cd, 如下所示, 后面为工程所在文件夹

```
> cd D:\beken\bk7221u
```

按照下面流程进行 scons 编译, 输出 rtthread.elf 文件。

```
> scons
>
```

```
....  
LINK rtthread.elf  
arm-none-eabi-objcopy -O binary rtthread.elf rtthread.bin  
arm-none-eabi-size rtthread.elf  
      text    data    bss    dec    hex filename  
742494    16516   118940   877950   d657e rtthread.elf  
scons: done building targets.
```

## 1.5 步骤四 OTA 固件升级

在嵌入式设备 OTA 中，通常通过串口或者网络等方式，将升级数据包下载到 Flash，然后将下载得到的数据包搬运到 MCU 的代码执行区域进行覆盖，以完成设备固件升级更新的功能，按照如下流程进行简单使用。

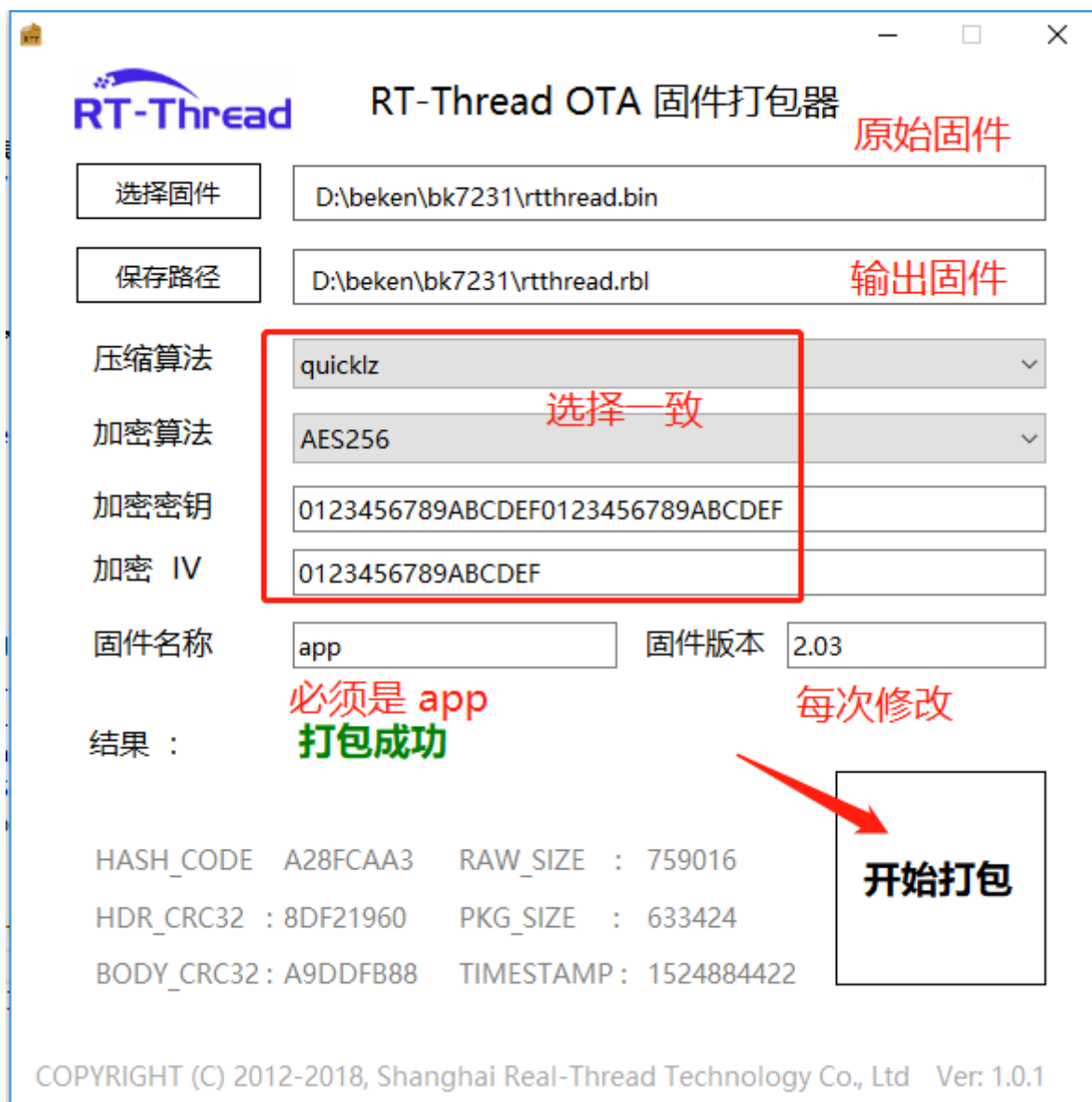
### （a）使能 OTA 功能

OTA 升级需要 Bootloader 的支持，我们首先需要将提供的 **all.bin** 固件烧录到你的设备中。**all.bin** (包含了 bootloader、app 和 download 固件)，默认提供了 Ymodem 和 HTTP OTA 功能支持。

### （b）固件打包

编译器编译出来的应用程序 **rtthread.bin** 属于原始固件，并不能用于 RT-Thread OTA 的升级固件，需要用户使用 **RT-Thread OTA 固件打包器** 打包生成 **.rb1** 后缀名的固件，然后才能进行 OTA 升级。

**RT-Thread OTA 固件打包器** 如下图所示：



(c) 升级

使用烧录了 all.bin 的设备，演示如何进行 HTTP 方式 OTA 升级。按照上图

- 按照上一小节连接 Wi-Fi，将设备与电脑通过串口连接
- 将升级固件（如 rtthread.rbl）上传到 HTTP 服务器
- 在串口中，使用命令 `http_ota [uri]` 进行 OTA 升级

将 [uri] 替换为您的 HTTP 服务器上 rtthread.rbl 固件的地址；命令示例：`http_ota http://192.168.10.135:80/rtthread.rbl`。

```
msh />
msh />http_ota http://192.168.10.135:80/rtthread.rbl
.....
[I/HTTP_OTA] Start erase flash download partition!
[I/HTTP_OTA] Erase flash (download) partition success!
```

```
[I/HTTP_OTA] Download: [=====]
100%
[I/OTA] Verify 'download' partition(fw ver: 24.4, timestamp: 1521791141) success.
reboot system
.....
```

注：示例日志中“.....”为省略日志内容，当显示第“7”行内容时，表示升级应用代码成功，接下来重启系统。

1.6 进一步运行 SDK 示例

1.6.1 Wi-Fi 接入

接入 Wi-Fi 之前，先介绍一下其接入的 API ，如下

```
wifi w0 join ssid 123456789
```

命令说明

字段	描述
wifi	有关 wifi 命令都以 wifi 开头
w0	需要操作的 wifi 设备名
join	wifi 执行连接动作
ssid	热点的名字
123456789	热点的密码，没有密码可不输入这一项

了解了上述命令，并且成功完成前面步骤，在串口中输入 `wifi w0 join ssid 123456789` ，当显示 `---- connected ----` 即表示 Wi-Fi 连接成功

```
wifi w0 join ssid 123456789
[DRV_WLAN]drivers\wlan\drv_wlan.c L629 beken_wlan_control cmd: case WIFI_INIT!
ssid:ssid key:123456789
rl_sta_start
.....
wpa_driver_associate
--- start connect ---
.....
IP up: 192.168.12.236
---- connected ----
```

- 注：“.....”部分表示省略的 log 日志。

另外，提供一个简单的测试使用命令，查询设备 IP 地址命令：ifconfig，查询设备 ip 地址。

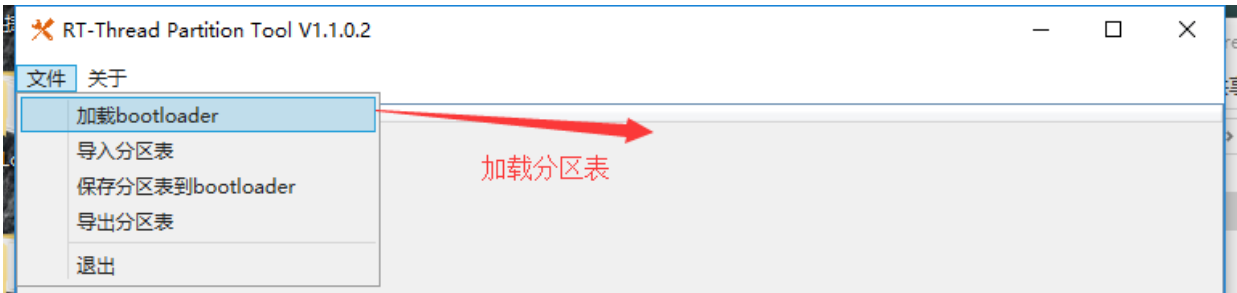


# 1.7 进阶教程

## 1.7.1 分区表工具使用

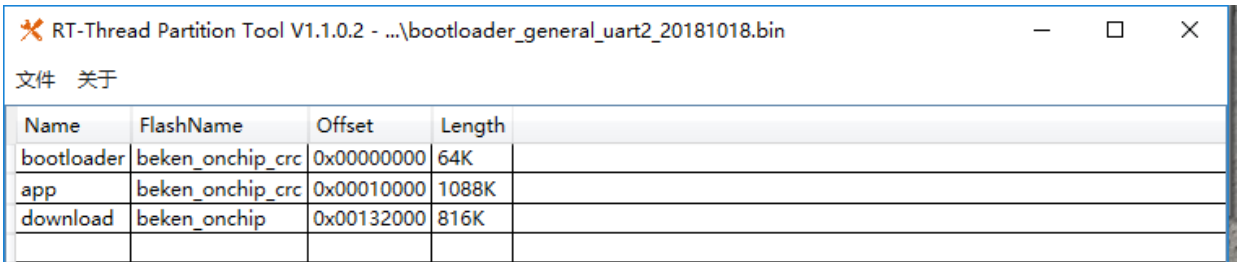
rt\_partition\_tool 是 RT\_Thread 自主研发的分区表工具，该软件用于对无分区表的 bootloader 固件进行分区表附加/修改/导入/导出操作。

- (a) 使用 rt\_partition\_tool 加载 bootloader



- (b) 为 bootloader 设置分区表

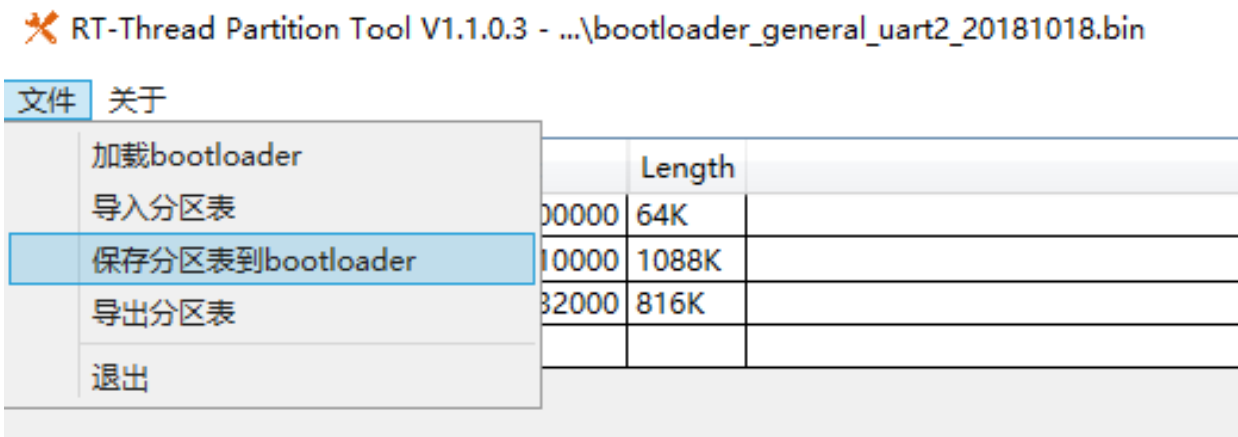
加载 bootloader 成功后，如下图所示



字段	描述
name	分区名称，固件中查找分区的依据不能重复，可以保持默认值
flashname	所在 Flash 名称，需要和 FAL 的 flash 驱动中定义的一致，可以保持默认值
offset	该分区在所在 flash 上的偏移地址，支持 xxK xxM 十六进制十进制的方式，此地址为逻辑地址
length	该分区的大小，支持 xxK xxM 十六进制十进制的方式，此处为逻辑地址

修改 offset 和 length 。

- (c) 保存分区表到 bootloader 中



此时，分区表已成功加载到 bootloader 中。

注意 事项 ： 分区表中的地址和长度都属于逻辑地址。

1.7.2 固件打包工具使用

- (a) 介绍

beken\_packager.exe 是一款由 RT-Thread 团队开发的固件打包工具，通过接读取 json 格式配置文件，根据配置信息方便快捷地将 bootloader ， application 以及 romfs 等二进制文件合成一个完整的镜像，适用于直接烧录 Flash 的场景。

- (b) 使用

使用该工具之前，首先需要根据 bootloader 中的分区表配置当前目录下的 config.json 文件。主要配置选项描述如下：

字段	描述
firmware	输入的二进制文件名称
version	版本号
partition	分区名
start_addr	分区开始地址 (此处指物理地址)
size	分区大小 (此处指物理地址)

```
{
  "firmware": "bootloader.bin",
  "version": "1.00",
  "partition": "bootloader",
  "start_addr": "0x00000000",
  "size": "68K"
},
```

- (c) 准备二进制文件

拷贝需要打包的各个二进制文件到当前目录下，如 `bootloader.bin`，`rtthread.bin`，`romfs.bin`。

- (d) 生成镜像

双击当前目录下 `beken_packager.bat`，即可自动打包，完成后将输出完整的镜像文件。将生成的 `all.bin` 下载到芯片即可。

```
partition      start_addr      size      firmware
-----
bootloader     0x00000000      64K      bootloader.bin
app            0x00010000      1792K     rtthread.bin
romfs          0x001D0000      512K      romfs_root_audio.bin

image size: 2516 KB 8. image name: all.bin 9. 10. Good bye!
```

#### 注意事项

1. 配置文件中的 `firmware` 名字需要和当前即将打包的二进制文件名保持一致，否则打包失败！
2. 谨慎修改配置文件，分区信息的配置需要和 OTA 保持一致，否则生成的镜像无法启动！
3. 分区信息支持动态增减，如 `romfs` 暂时不需要，可直接删除。
4. 如果修改了分区表中 `app` 分区的地址，还要修改 `link.lds` 中的 `app` 地址，保持和分区表中的一致，如下所示

```
MEMORY
{
    flash (rx) : ORIGIN = 0x00010000, LENGTH = 1215k /* 1216KB - 96B */
    ram (rw!x): ORIGIN = 0x00400000, LENGTH = 256k
}
```

### 1.7.3 其他下载程序的方式

除了 OTA 下载程序之外，还有其他下载程序的方式

- (a) 连接硬件下载器，通过硬件 `spi` 的方式连接
- (b) 修改链接脚本配置

一般，我们只需要修改链接脚本 `link.lds` 里中 `Flash` 段的起始地址为 `0x00000000` 即可。

以 GCC 链接脚本为例，介绍如何修改，如下所示：

```
/* Split memory into area for vectors and ram */
MEMORY
{
    flash (rx) : ORIGIN = 0x00000000, LENGTH = 2M
    ram (rw!x): ORIGIN = 0x00400000, LENGTH = 256k
}
```

- (c) 通过 `scons` 命令编译程序，生成 `rtthread.bin`。
- (d) 执行根目录下的 `encrypt.bat` 脚本，生成 `rtthread_crc.bin`。
- (e) 通过 Hid Download Tool 工具，下载 `rtthread_crc.bin` 文件到 Flash。



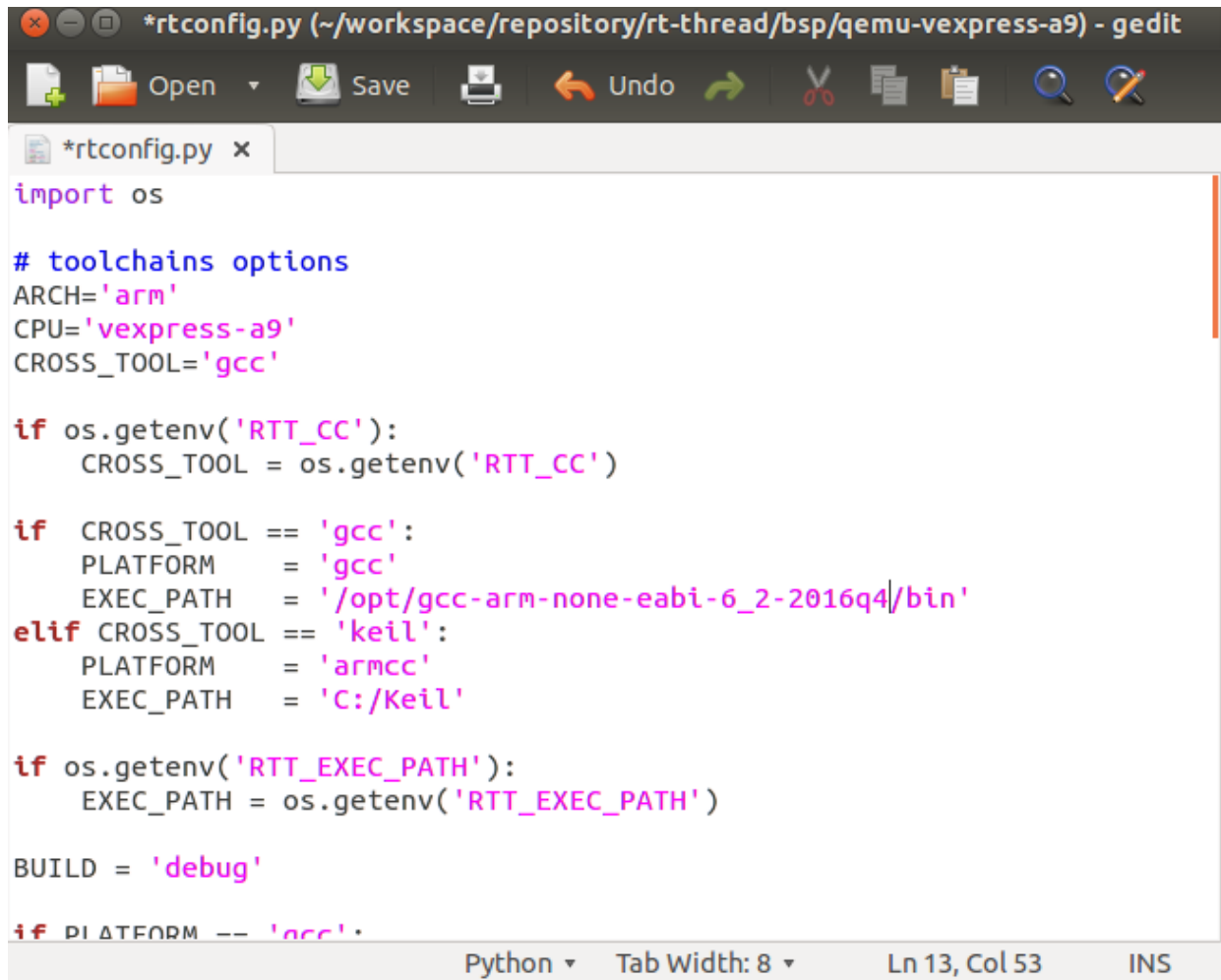
## 1.8 Ubuntu 平台开发 RT-Thread

### 1.8.1 准备工作

- 安装 Scons，使用命令：`sudo apt-get install scons`
- 安装编译器，使用 `apt-get` 命令安装的编译器版本太旧会导致编译报错，可依次使用如下命令下载安装新版本，下载链接和解压文件夹名因下载版本而异：

1. `wget https://armkeil.blob.core.windows.net/developer/Files/downloads/gnu-rm/6-2016q4/gcc-arm-none-eabi-6_2-2016q4-20161216-linux.tar.bz2`
2. `cd /opt`
3. `sudo tar xf ~/Downloads/ gcc-arm-none-eabi-6_2-2016q4-20161216-linux.tar.bz2`

编译器安装好以后需要修改 `bk7221u_release` 下面的 `rtconfig.py` 文件，修改对应路径为解压到 `opt` 目录下的编译器对应的 `bin` 目录，参考下图，目录名字因下载的编译器版本而异：



```

import os

# toolchains options
ARCH='arm'
CPU='vexpress-a9'
CROSS_TOOL='gcc'

if os.getenv('RTT_CC'):
    CROSS_TOOL = os.getenv('RTT_CC')

if CROSS_TOOL == 'gcc':
    PLATFORM = 'gcc'
    EXEC_PATH = '/opt/gcc-arm-none-eabi-6_2-2016q4/bin'
elif CROSS_TOOL == 'keil':
    PLATFORM = 'armcc'
    EXEC_PATH = 'C:/Keil'

if os.getenv('RTT_EXEC_PATH'):
    EXEC_PATH = os.getenv('RTT_EXEC_PATH')

BUILD = 'debug'

if PLATFORM == 'gcc':

```

图 1.2: 编译器路径修改

## 1.8.2 编译和运行 RT-Thread

1、在 bk7221u\_release 目录下输入 `scons` 命令编译工程:

```

lxg@lxg:~/bk7221u$ scons -j7
scons: Reading SConscript files ...
RTT_ROOT is: /home/lxg/bk7221u/rt-thread
scons: done reading SConscript files.
scons: Building targets ...
scons: building associated VariantDir targets: build
CC build/applications/main.o
CC build/applications/msh_evm.o
CC build/applications/romfs.o
CC build/beken378/app/app.o
CC build/beken378/app/config/param_config.o
CC build/beken378/app/standalone-ap/sa_ap.o
CC build/beken378/app/standalone-station/sa_station.o
CC build/beken378/demo/ieee802_11_demo.o
CC build/beken378/driver/codec/driver_codec_es8374.o

```