# 嵌入式系统实验

## ——WRTnode 交叉编译环境

在树莓派或 Acadia 上实现一个 C 语言的交叉编译环境，能编译产生 WRTnode 用的 MIPS 程序

## 实验目的

1. 熟悉 WRTnode 交叉编译环境配置；
2. 掌握 Acadia 平台的交叉编译环境配置

## 实验器材

### 硬件

- pcDuino v2 板一块；
- 5V/1A 电源一个；
- microUSB 线一根；
- USB-TTL 串口线一根（FT232RL 芯片或 PL2303 芯片）。

以下为自备（可选）器材：

- PC（Windows/Mac OS/Linux）一台；
- 以太网线一根（可能还需要路由器等）。

### 软件

- PC 上的 USB-TTL 串口线配套的驱动程序；
- PC 上的串口终端软件，如 minicom、picocom、putty 等；
- PC 上的 SSH 软件，如 putty 等。

### 实验步骤

**1. 配置依赖环境；**

下载编译运行 WRTnode 交叉编译环境所需的依赖环境(如下)，包括 gawk 查看替换文本工具

```
root@Acadia:~# sudo apt-get install build-essential subversion libncurses5-dev g
awk gcc-multilib flex git-core gettext
```

## 2. 下载交叉编译源代码

创建工程文件夹如下：

```
root@Acadia:~# mkdir openwrt
root@Acadia:~# cd openwrt
```

从 wrtnode 官网下载 wrtnode 交叉编译环境源代码，如下，得到 sdk.tar.bz2 源代码压缩文件

```
root@Acadia:~/openwrt# wget http://d.wrtnode.com/sdk/sdk.tar.bz2
--1970-01-01 13:20:49--  http://d.wrtnode.com/sdk/sdk.tar.bz2
Resolving d.wrtnode.com (d.wrtnode.com)...
```

## 3. 挂载 TF 卡

由于交叉编译中间过程生成的总的文件大小在 7G 左右，因此，pcduino 的剩余空间不足以进行直接编译，这里，插入 TF 卡在其中编译环境

利用 fdisk -l 命令查看插入的 TF 卡设备为/dev/mmcblk1

```
Disk /dev/mmcblk0boot0 doesn't contain a valid partition table

Disk /dev/mmcblk1: 16.0 GB, 16021192704 bytes
4 heads, 16 sectors/track, 488928 cylinders, total 31291392 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x00000000

Disk /dev/mmcblk1 doesn't contain a valid partition table
```

以 ext4 文件系统格式格式化该设备

```
Mkfs.ext4 /dev/mmcblk1
```

挂载该设备至/wrtnode

```
mount /dev/mmcblk1 wrtnode
```

由于 wrtnode 交叉编译工具的编译过程需要非 root 用户执行，因此，这里，将该设备的所有权交给 rhb 用户

```
Chown rhb wrtnode
```

## 4. 编译交叉编译源代码

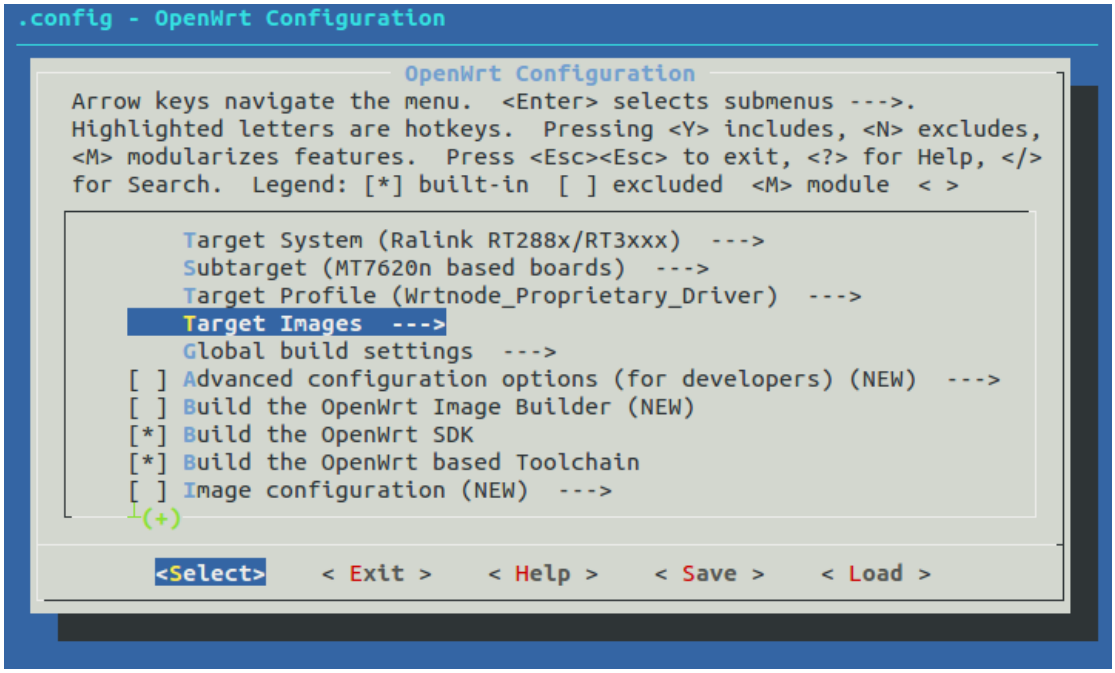
拷贝源代码文件至 wrtnode 目录下，对其进行解压缩

```
root@Acadia:~/openwrt# tar -jxvf sdk.tar.bz2
wrtnode-sdk/
wrtnode-sdk/feeds/
wrtnode-sdk/feeds/packages.tmp/
wrtnode-sdk/feeds/packages.tmp/.host.mk
wrtnode-sdk/feeds/packages.tmp/info/
wrtnode-sdk/feeds/packages.tmp/info/.packageinfo-net_umurmur
wrtnode-sdk/feeds/packages.tmp/info/.packageinfo-lang_luaexpat
wrtnode-sdk/feeds/packages.tmp/info/.packageinfo-net_knot
wrtnode-sdk/feeds/packages.tmp/info/.packageinfo-net_git
wrtnode-sdk/feeds/packages.tmp/info/.packageinfo-utils_attr
```

进入解压缩得到的 wrtnode-sdk 目录，对配置文件进行配置

Make menuconfig

勾选创建 OpenWrt 的交叉编译选项




在当期目录下，执行如下命令，进行编译

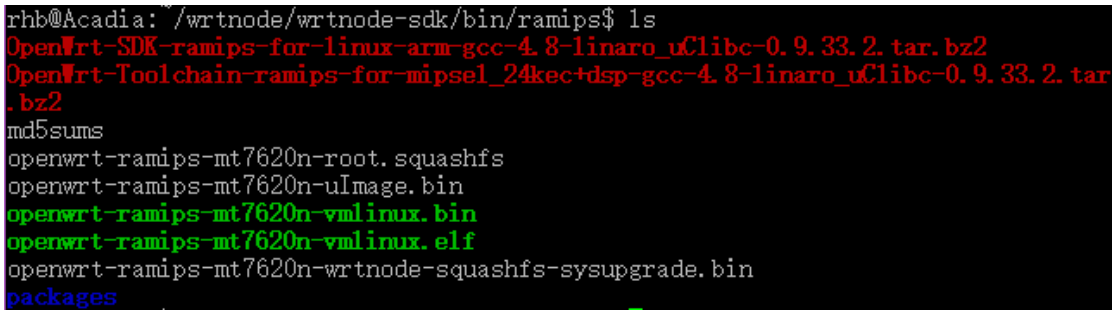
```
rhb@Acadia:~/wrtnode-sdk$ make V=s
```

```
checking for C/C++ restrict keyword... __restrict
checking for struct timeval... yes
checking for wide-enough struct timeval.tv_sec member... yes
checking whether gettimeofday is declared without a macro... yes
checking for nl_langinfo and CODESET... yes
checking whether getc_unlocked is declared... yes
checking whether we are using the GNU C Library >= 2.1 or uClibc... yes
checking whether lstat correctly handles trailing slash... yes
checking whether malloc, realloc, calloc are POSIX compliant... yes
checking for stdlib.h... (cached) yes
checking for GNU libc compatible malloc... yes
checking for unsigned long long int... yes
checking for long long int... yes
checking for mbstate_t... yes
checking for a traditional japanese locale... none
checking for a transitional chinese locale... none
checking for a french Unicode locale... none
checking for mmap... yes
checking for MAP_ANONYMOUS... yes
checking whether memchr works... yes
checking whether <limits.h> defines MIN and MAX... no
checking whether <sys/param.h> defines MIN and MAX... yes
checking for stdbool.h that conforms to C99... yes
```

其编译得到可在 ARM 处理器下运行的交叉编译工具

## 5. 测试交叉编译工具

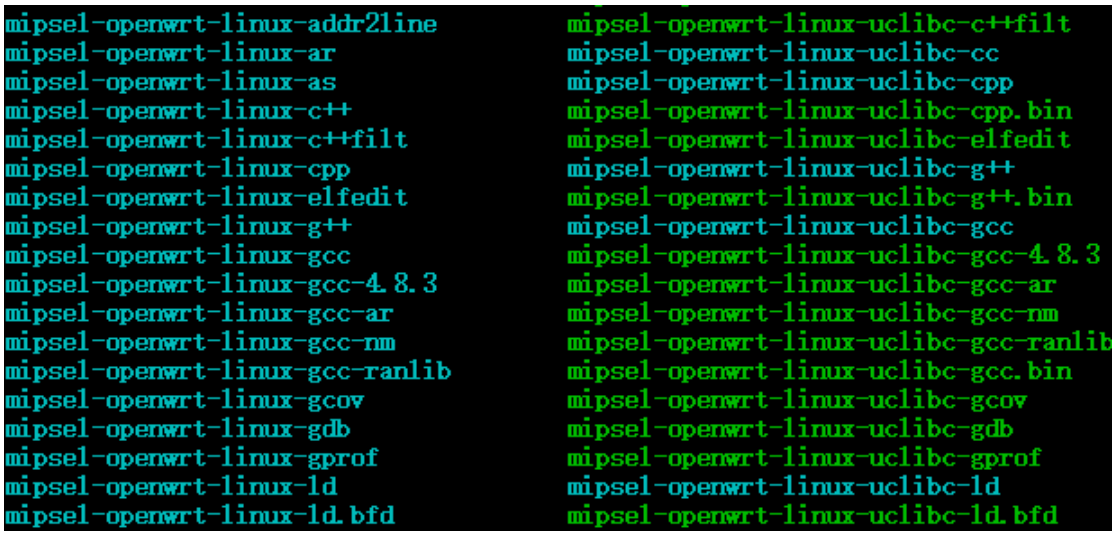
在经过较长时间的编译之后，在/wrtnode-sdk/bin/ramips 下可以得到编译获得的交叉编译工具固件




对其进行解压缩

```
tar -jxvf OpenWrt-Toolchain-ramips-for-mipsel_24kec+dsp-gcc-4.8-linaro_uClibc-0.9.33.2.tar.bz2
```

进入解压缩得到的目录，在/OpenWrt-Toolchain-ramips-for-mipsel_24kec+dsp-gcc-4.8-linaro_uClibc-0.9.33.2/toolchain-mipsel_24kec+dsp_gcc-4.8-linaro_uClibc-0.9.33.2/bin 目录下可以找到交叉编译所用的工具




编写测试代码

```
#include <stdio.h>
int main(void)
{
   printf("hello world");
   return 0;
}
```

运行 mipsel-openwrt-linux-gcc 命令编译该测试代码

```
./mipsel-openwrt-linux-gcc -o hello hello.c
```

得到可执行文件 hello，利用 file 命令查看其文件类型，可以得到其是在 MIPS 环境下的可执行文件

```
ro_uClibc-0.9.33.2/bin$ file hello
hello: ELF 32-bit LSB executable, MIPS, MIPS32 rel2 version 1 (\001), dynamica11
unknown capability 0x70100 = 0x3040000, not stripped
```

将该文件拷贝至 OpenWRT 平台下运行，运行结果如下：

```
root@OpenWrt:~# chmod +x hello
root@OpenWrt:~# ./hello
hello worldroot@OpenWrt:~#
```

因此，成功配置 OpenWRT 交叉编译环境