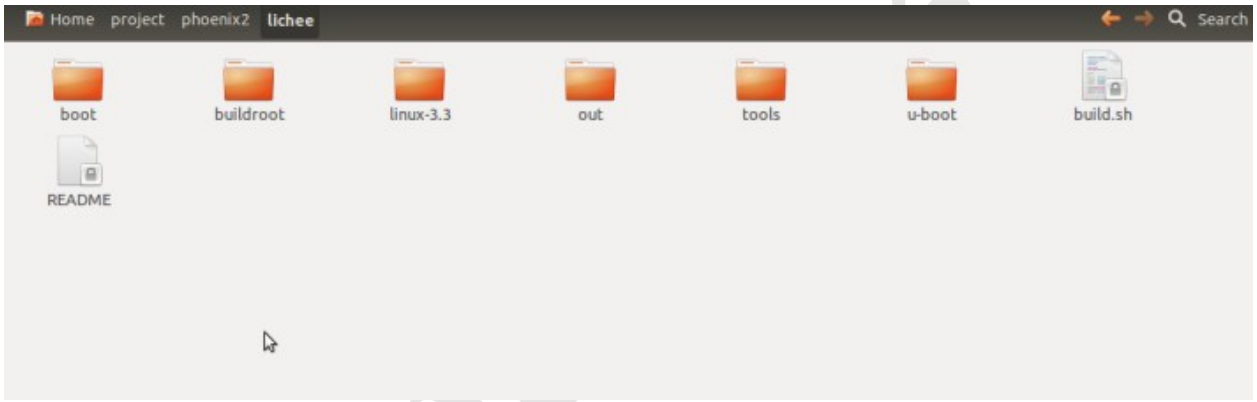
Phoenix 代码编译和固件烧写

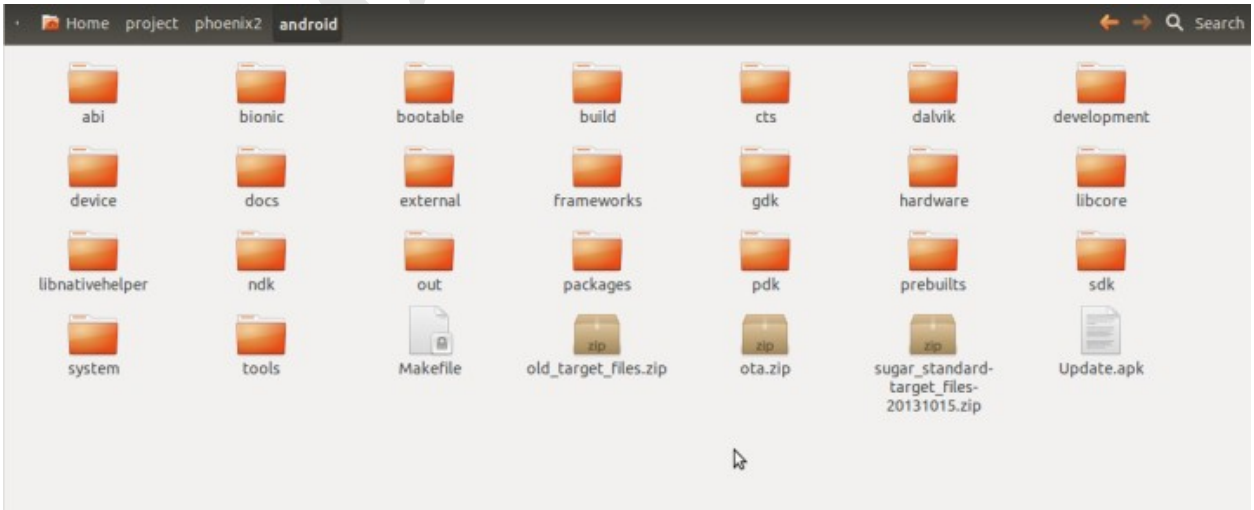
1. 代码结构介绍

Phoenix 代码下载完后，将在同一个目录中呈现出两个目录，一个是 android 目录，一个是 lichee 目录。其中 android 目录存放 android 的所有源代码，lichee 目录则存放 linux 源码以及一些工具链等。必须注意，android 和 lichee 必须放在同一个目录下，比如都放在 phoenix 目录，这个主要是编译 android 的时候需要从 lichee 中拷贝一些文件，而相关的脚本已经把拷贝文件的路径写成固定的，所以如果不放在一起的话，将导致无法从 lichee 的输出目录拷贝生成的内核文件和设备驱动，导致编译出来的 android 无法使用。

1.1 lichee 代码结构介绍



1.2 android 代码结构介绍



2. 代码编译

2.1 编译 android 代码

如果要想编译生成自定义的 android 固件，你需要同时编译 linux 和 android 的源代码，下面将详细描述如何编译这些代码，并生成固件。

1) 编译 lichee 代码

使当前目录为 lichee 的根目录。然后执行下面的命令：

```
cd lichee
./build.sh -p sun7i_android
```

或者：

```
 . buildroot/scripts/mksetup.sh     #导入环境变量，根据提示选择对应选项
   mklichee
```

即完成了一次 lichee 的编译（根据服务器配置，耗时至少 10 分钟），编译成功，屏幕上出现

```
INFO:build u-boot OK.
...
INFO:build rootfs OK.
INFO:build lichee OK.
```

2) 编译 android 代码

编译完 lichee 后，将目录切换到 android 目录，然后进行下面的操作：

```
   . build/envsetup.sh          #导入环境变量
   lunch                      #根据自己的开发平台，选择方案
   extract-bsp                  #拷贝内核和模块到 android 中
   make –j4                     #-j 开启多核编译，一般为 cpu 数量的一半
```

编译完成之后，将会在相应的产品目录（out/target/product/wing-xxx）会生成:
boot.img, recovery.img . system.img 3 个包。

3) 固件打包

至此 android 的编译就完成了，此时你可以直接调用命令

```
$ pack
```

将会在 lichee/tools/pack 目录下生成 sun7i_android_xxx.img 文件，这就是我们要的 android 固件，可以通过烧写工具直接烧写到 Phoenix 中。

2.2 编译 linux 代码

如果只需要编译 linux 固件，可以使用下面的命令：

```
cd lichee
./build.sh -p sun7i
./build.sh pack
```

根据提示选择相应的参数，即可编译打包固件。
如果某些选项没有选上，可能会导致编译出错，可输入以下命令清除，重新编译

```
./build.sh -m clean
./build.sh -p sun7i
```

2.3 编译常见问题

1) can't find lz

编译时出现以下打印信息：

```
/usr/bin/ld: cannot find -lz
collect2: ld returned 1 exit status
make: *** [out/host/linux-x86/obj/EXECUTABLES/aapt_intermediates/aapt] Error 1
```

原因是缺少 lib32z1-dev，安装即可:sudo apt-get install lib32z1-dev

2) You must install 'makeinfo' on your build machine

解决方法：sudo apt-get install texinfo

3) msgfmt not found

解决方法：sudo apt-get intall gettext

2.4 局部编译和打包

不用每次编译都对系统进行完全编译，你可以只编译有修改的部分。boot.img 镜像中包含 linux kernel 和内存盘 ramdisk，如果内核有修改，就需要重新编译内核，然后在 android 目录下执行下列命令,即可打包生成 boot.img

```
$ . build/envsetup.sh
$ lunch                    #选择 wing-xxx 产品
$ extract-bsp
$ make bootimage
```

这样就生成了 boot.img，类似的方法就可以可以重新打包生成 system.img

```
$ . build/envsetup.sh
$ lunch                    #选择 wing-xxx 产品
$ make systemimage-nodeps
```

wing-xxx 重新生成的镜像在 android/out/target/product/wing-xxxwing-xxx/ 目录下。

3. 固件烧写

至此，Phoenix 需要的固件已经编译生成完毕，下面就是要将固件下载到主板中，下载的方式有两种。一种是完全下载的方式，另外一个是局部下载方式，下面分别说明两种下载方式。

3.1 phoenixsuit 下载

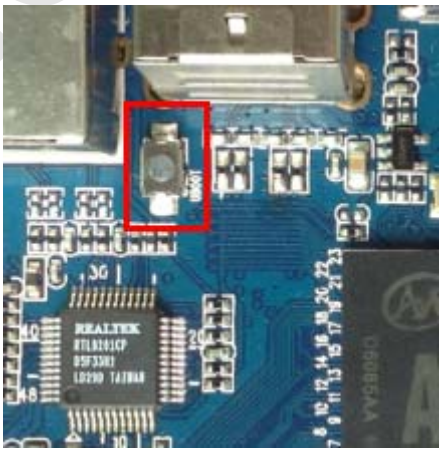
在开发环境一章中，我们已经安装了 phoenixsuit 固件下载工具，可以直接用这个工具来下载工具。具体操作如下：

1) 打开 phoenixsuit，选择需要下载的固件。

2) 将 USB 线连接到电脑。

3) Phoenix 主板断电。

4) 按住 Phoenix 主板中的 UBOOT 键后，将 USB 线的另外一端插入 Phoenix 的升级 USB 口。



UBOOT 按键位置

5) 插入后主板会自动进入升级模式，并提示是否需要格式化。

6) 根据需要选择是否格式化后，系统将进入升级状态，这个过程将持续 2 分钟左右。升级完后，工具将会提示升级成功。

7) 升级成功后，系统会自动启动，这个时候是 USB 供电，建议使用 5V 电源供电，如果需要用作移动设备，可以使用移动电源对其进行供电。

3.2 fastboot 方式烧写固件

每次使用 PhoenixSuit 升级时间会比较长，如果用户只是修改了某个 img 文件，而其他分区并没有修改，可以使用 fastboot 的方式进行固件烧写。

fastboot 是一种线刷，就是使用 USB 数据线连接手机的一种刷机模式，在 A20 主控中，可以使用 fastboot 的功能来实现局部系统的更新。

1) 进入 fastboot 模式

启动开发板，在串口界面敲任意按键，可以进入 u-boot；如果进不了 fastboot，将 lichee\tools\pack\chips\sun7i\configs\android\default\env.cfg 中 的 bootdelay=0 改 成 bootdelay=2 重新打包固件即可（需要安装 google-usb_driver 驱动）。

在串口命令行输入 fastboot 命令，进入 fastboot 模式；

通过 pc 端的 fastboot 工具烧录各个固件包（fastboot 是 windows 下的一个工具（android sdk 中有），上网自己下载一个，解压到本地，然后将 fastboot.exe 添加到 windows 环境变量）

进入在 windows 命令行：cmd 进行命令行模式，在命令行执行 fastboot 指令\

退出 fastboot 模式：ctl+c

2) 使用 fastboot 命令烧写

在 windows 命令行使用 fastboot 命令。
擦除分区命令：

```
$ fastboot erase boot #擦除 boot 分区
$ fastboot erase system #擦除 system 分区
$ fastboot erase data #擦除 data 分区
```

烧写分区命令：

```
$ fastboot flash boot boot.img #把 boot.img 烧写到 boot 分区
$ fastboot flash system system.img #把 system.img 烧写到 system 分区
$ fastboot flash data userdata.img #把 userdata.img 烧写到 data 分区
```

Fastboot 方式更新固件效率比较高，建议使用。