[**啊胖的简单生活**](http://www.cnblogs.com/lisaac/)

[**openwrt x86 编译部署**](http://www.cnblogs.com/lisaac/p/4065076.html)

[TOC]

著作权归作者所有。商业转载请联系作者获得授权，非商业转载请注明出处。  
本文链接：<http://blog.ick.im/posts/2014-10-31-openwrt-x86-compile.html>

**0.背景**

前段时间看到C1037u多网口版本，很是喜欢，替换的4530r，发现玩openwrt x86的人比较少，写一篇日记，以备后用，有错误之处请指出。

**0.配置编译环境**

Linux

**1.获取源代码**

svn co svn://svn.openwrt.org/openwrt/trunk/

#svn co svn://svn.openwrt.org/openwrt/branches/barrier\_breaker -r42625

svn update

#cp feeds.conf.default feeds.conf

./scripts/feeds update -a

./scripts/feeds install -a

**2.一些问题**

* config文件位置：

openwrt config文件：.config  
kernel config:build\_dir/target-xxxxxxx/linux-x86\_generic/linux-x.xx.xx/.config

* 内核版本更改：

支持内核版本列表： include/kernel-version.mk  
更改内核版本位置： target/linux/XXX/Makefile.

**3.make menuconfig**

这里给出一个基本配置，不过其中Busybox配置，正常使用默认配置即可，不需要Customize busybox options，我为了在日后的initramfs中加入一些overlayfs的功能自己加进去了配置，另外的配置基本是x86必须的。

Target System (x86) #目标平台选择

Target Images --->

[\*] ext4 #生成.EXT4.IMG文件

Base system --->

<\*> block-mount

<\*> busybox ---> #用于今后`initramfs`支持，可以将所有lib编译到busybox

[\*] Customize busybox options

Busybox Settings --->

General Configuration --->

[\*] Support --install [-s] to install applet links at runtime

[\*] Don't use /usr

Linux Module Utilities --->

[\*] modinfo

[\*] Simplified modutils

[\*] Accept module options on modprobe command line

[\*] Skip loading of already loaded modules

(/lib/modules) Default directory containing modules

Linux System Utilities --->

[\*] mdev

[\*] Support /etc/mdev.conf

[\*] Support subdirs/symlinks

[\*] Support regular expressions substitutions when renaming

[\*] Support command execution at device addition/removal

[\*] Support loading of firmwares

[\*] findfs

[\*] blkid

[\*] Print filesystem type

[\*] losetup

[\*] lspci

[\*] lsusb

[\*] mount

[\*] Support specifying devices by label or UUID

Filesystem/Volume identification --->

[\*] Ext filesystem

[\*] fat filesystem

Networking Utilities --->

[\*] ftpd

[\*] Enable upload commands

[\*] Enable workaround for RFC-violating clients

[\*] inetd

[\*] telnetd

[\*] Support standalone telnetd (not inetd only)

[\*] tcpsvd

[\*] udpsvd

`kernel` modules --->

Block Devices --->

<\*> kmod-ata-core

<\*> kmod-ata-ahci

<\*> kmod-loop

-\*- kmod-scsi-core

<\*> kmod-scsi-generic

Filesystems --->

<\*> kmod-fs-ext4

<\*> kmod-fs-ntfs

<\*> kmod-fs-vfat

Input modules --->#键盘

-\*- kmod-hid

<\*> kmod-hid-generic

-\*- kmod-input-core

-\*- kmod-input-evdev

Native Language Support --->

<\*> kmod-nls-cp437 #vfat需要这个

<\*> kmod-nls-iso8859-1

<\*> kmod-nls-utf8

Network Devices ---> #网卡驱动

<\*> kmod-macvlan

<\*>kmod-e1000e

USB Support --->

-\*- kmod-usb-core

<\*> kmod-usb-hid #usb键盘

<\*> kmod-usb-ohci

<\*> kmod-usb-storage

<\*> kmod-usb2

<\*> kmod-usb3

Wireless Drivers --->#wifi卡驱动

Network --->

<\*> hostapd #wifi ap模式

<\*> hostapd-common

<\*> hostapd-utils

<\*> wpa-supplicant

Utilities --->#自选 fdisk等

**4.make kernel\_menuconfig  
目的是为了加入x86的多核心以及大内存支持**

Processor type and features --->

[\*] Symmetric multi-processing support

Processor family (Core 2/newer Xeon) --->#自行选择处理器平台

[\*] Supported processor vendors --->#自行选择处理器平台

(2) Maximum number of CPUs #自行编辑

[\*] SMT (Hyperthreading) scheduler support#超线程支持

[\*] Multi-core scheduler support

High Memory Support (4GB) --->

**5.make**

编译

make -j 100 download #下载包，openwrt是基于源码+patch形式，下载过程比较慢，最好在墙外

make clean

make -j 5 V=99

#编译某个包

make packages/xxx/clean

make packages/xxx/compile

make packages/xxx/install

**6.安装**

* **固件简介**

ls bin/x86

会看到有以下一些文件，与[官方编译](https://downloads.openwrt.org/barrier_breaker/14.07/x86/generic/)的固件名称完全相同，这里大概解释下：

openwrt-x86-generic-combined-ext4.img.gz  
: 包含vmlinuz、rootfs（ext4）、引导信息以及相关分区信息的img，一般是两个分区，可以把它看成是硬盘镜像，直接dd到某个磁盘

openwrt-x86-generic-rootfs-ext4.img.gz  
: rootfs分区镜像，可以直接dd到某个分区，或者mount -o到某个目录

openwrt-x86-generic-rootfs-squashfs.img  
: 同上

openwrt-x86-generic-vmlinuz  
: kernel

openwrt-x86-generic-Generic-rootfs.tar.gz  
: rootfs用gz打包后的文件

可以看出，要让系统启动，需要引导器（x86是使用grub，好比是路由中的uboot，当然uboot管的内容更多）、kernel、rootfs三者。

* **简易部署**

如果你的磁盘（U盘）只用于openwrt系统，可以使用最简单的用combined.img直接dd到目标盘，这样的好处是简单，坏处是如果你的sdb（磁盘/U盘）很大，就带来空间浪费，虽然可以在rootfs中存数据，但是下次升级系统就带来不必要的麻烦，命令行如下：

gunzip bin/x86/openwrt-x86-generic-combined-ext4.img.gz

dd if=bin/x86/openwrt-x86-generic-combined-ext4.img of=/dev/sdb #根据自己情况选择磁盘

* **自定义部署**

适合有一定基础的兄弟，大致步骤：

* + 0.建立分区、文件系统；
  + 1.用dd将rootfs.ext4.img写入到分区；
  + 2.复制vmlinuz到分区；
  + 3.安装引导
    - 如果主板是bios，我习惯用[grub4dos](http://grub4dos.chenall.net/)，参见[无忧启动](http://bbs.wuyou.com/forum.php?mod=forumdisplay&fid=60)；
    - 如果主板是efi，我习惯用[grub2](https://wiki.archlinux.org/index.php/Grub2)，需要efi文件系统（fat文件系统），在分区的时候要记得加入，参照个大linux发行版。
  + 4.编辑menu.lst；
  + 6.启动

**7.关于initramfs && overlayfs**

* **overlayfs：**

由于openwrt x86一般都是安装在U盘/TF卡/硬盘等介质上，默认是ext文件系统，就没有使用overlayfs，如果要使用overlayfs就得用squashfs。由于从路由过度过来，我个人对overlayfs颇有好感，着手在x86的ext4上加入overlayfs支持。稍后，会单独写一篇关于x86下overlayfs的文档。

* **initramfs：**

按照如下编译选项配置，其编译生成的vmlinuz是一个带initramfs的kernel，当然可以在也可以在Use external cpio中选择自定义initramfs目录，如果没选择Use external cpio，openwrt则会将整个rootfs当作initramfs编译进vmlinuz.

Target Images --->

[\*] ramdisk #其实就是initramfs

() Use external cpio#选择外部cpio，作为initramfs

如果用gurb加载它，则整个系统会在ramfs上运行，所有配置在重启后都将不被保留，一般需要在这个上面启动到另一个kernel（kexec）或者switch\_root 到另一个rootfs（真实的磁盘）环境，ramdisk（initramfs）。因此一般是不勾选编译选项中的ramdisk选项，自己着手来做initramfs，加入一些hook（比如加入overlayfs支持、switch\_root到其他rootfs、干脆直接kexec到其他kernel），用gurb的initrd加载，就可以完成系统启动。