<http://www.100ask.org/bbs/forum.php?mod=viewthread&tid=17389&highlight=%B4%D3%C1%E3%D6%C6%D7%F7%CE%C4%BC%FE%CF%B5%CD%B3>

ubuntu    ：Ubuntu 14.04 32-Bit

开发板        ：JZ2440 v2(当然V3也可以)

内核版本    ：Linux-3.4.2

交叉编译器：arm-linux-gnueabi-gcc 4.8.2

busybox    ：Busybox-1.24.0.tar.gz（(有答疑助手说韦老师提供的3.4.5编译器编译Busybox 1.21以上版本都有问题，新手可以编1.21)  
  
  
下载busybox-1.24.0.tar.bz2  ，网址： <https://busybox.net/downloads/busybox-1.24.0.tar.bz2>  
解压配置：

tar jxf busybox-1.24.0.tar.bz2  
cd busybox-1.24.0  
make menuconfig

修改配置项

    Busybox Settings  --->  
    Build Options  --->  
        [\*] Build BusyBox as a static binary (no shared libs)  
        (arm-linux-gnueabi-) Cross Compiler prefix  
    Busybox Library Tuning  --->  
        [\*] Username completion

编译安装：

make  
make install

busybox只是生成了各种我们需要的可执行程序，例如（ls,cd,chmod,cp,mv等），但是Linux文件系统若要正常运行，仅仅依靠这些程序还不够.  
下面我们来丰富一下这个文件系统，使之悦己也悦人。  
  
  
1.首先新建一个目录rootfs，用来存放文件系统。

$mkdir rootfs  
$sudo chown root:root rootfs -R

参考Ubuntu 14.04 的目录结构，我们新建以下目录

$cd rootfs  
$sudo mkdir -p root dev etc bin sbin mnt sys proc lib tmp var usr media usr/sbin usr/bin usr/lib usr/local lib/modules etc/sysconfig etc/init.d  var/run var/log  
$chmod 1777 tmp

2.进入dev目录，创建控制台节点 

$cd dev  
$sudo mknod console c 5 1  
$sudo chmod 777 console  
$sudo mknod null c 1 3  
$sudo chmod 777 null  
$cd ..

3.最复杂的一步，也是至关重要的一步，进入etc目录，创建一大堆配置文件

$cd etc  
$sudo touch inittab fstab  profile resolv.conf mdev.conf  inetd.conf init.d/rcS   
$sudo chmod +x init.d/rcS

文件创建好了，开始写入配置信息 。首先是初始化脚本

vi inittab

内容为

::sysinit:/etc/init.d/rcS  
console::askfirst:-/bin/sh  
::ctrlaltdel:-/sbin/reboot  
::shutdown:/bin/umount -a -r  
::restart:/sbin/init

文件系统挂载表

vi fstab

内容为

#<device> <mount point>  <type>  <options> <dump> <pass>  
proc                /proc                 proc        defaults        0            0  
sysfs                /sys                    sysfs        defaults        0            0  
tmpfs              /var                    tmpfs      defaults        0            0  
tmpfs              /tmp                  tmpfs      defaults        0            0  
tmpfs              /dev                   tmpfs      defaults        0            0

启动时自动执行的脚本

vi profile

内容为

ulimit -S -c 0 > /dev/null 2>&1  
USER=`id -un`  
LOGNAME=$USER  
PS1='[\u@\h:\w]\# '  
PATH=$PATH  
export USER LOGNAME PS1 PATH  
export LD\_LIBRARY\_PATH=lib:/usr/lib:$LD\_LIBRARY\_PATH

设置域名服务器 

vi resolv.conf

内容为

nameserver 114.114.114.114

开机时，自动执行的脚本

vi init.d/rcS

内容为

#!/bin/sh  
PATH=/sbin:/bin:/usr/sbin:/usr/bin  
runlevel=S  
prevlevel=N  
umask 022  
export PATH runlevel prevlevel  
mount -a  
mkdir /dev/pts  
mount -t devpts devpts /dev/pts  
echo /sbin/mdev > /proc/sys/kernel/hotplug  
mount -t proc none /proc  
mdev -s  
/bin/hostname JZ2440  
ifconfig eth0 192.168.1.100          #给板子分配一个固定ip

4.etc目录终于搞定了，现在进行最后一步，拷贝busybox的程序和交叉编译器的库文件到这里来

$cd ..  
$cp -rd busybox-1.24.0/\_install/\* ./  
$cp -d /usr/local/arm/4.8.2/arm-linux-gnueabi/lib/\*so\*  lib

说明：上面的操作路径是在rootfs目录下，busybox路径和库文件路径因人而异，不可照搬，切记切记。    
  
5.用NFS启动一下，看看有没有什么问题 

OpenJTAG> set    bootargs    noinitrd      root=/dev/nfs     nfsroot=192.168.1.200:/home/mm/Downloads/jz2440/filesystem/rootfs          ip=192.168.1.100:192.168.1.200:192.168.1.254:255.255.255.0::eth0:off        init=/linuxrc        console=ttySAC0,115200  
OpenJTAG> boot

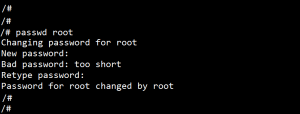
注意：

192.168.1.200是服务器的IP

/home/mm/Downloads/jz2440/filesystem/rootfs   是刚才制作的文件系统的目录

192.168.1.100是开发板IP

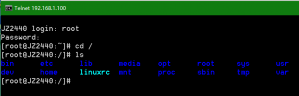
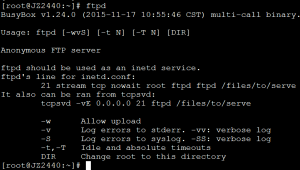
192.168.1.254是网关

文件系统目录/home/mm/Downloads/jz2440/filesystem/rootfs必须在服务器/etc/exports中指明，不然开发板NFS启动时是挂载不了的，NFS启动必然失败  
  
  
6.如果一个用户没有密码，那它在linux系统中就是伪用户，也就是不存在的用户，当然也无法用这个用户登录。现在已经使用NFS启动成功，那么顺便设置一下root用户的密码（本操作在开发板上完成）。  
  
  
7.虽然已经设置了root用户的密码，但是也仅仅是设置了而已，还没有用起来。因为每次开机启动，一个回车就直接进入系统了，这显然是不够完善。那就加一个用户登录验证吧。   
修改 /etc/inittab 

::sysinit:/etc/init.d/rcS  
console::askfirst:-/bin/sh  
::ctrlaltdel:-/sbin/reboot  
::shutdown:/bin/umount -a -r  
::restart:/sbin/init

改为

::sysinit:/etc/init.d/rcS  
::respawn:-/bin/login  
#console::askfirst:-/bin/sh  
::ctrlaltdel:-/sbin/reboot  
::shutdown:/bin/umount -a -r  
::restart:/sbin/init

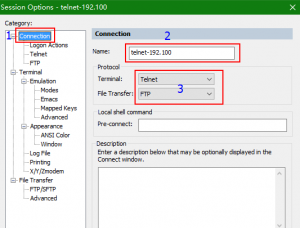
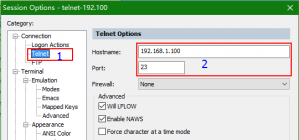
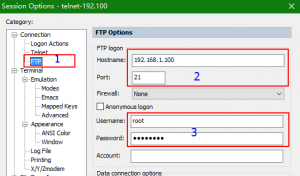
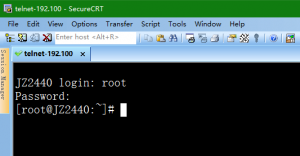
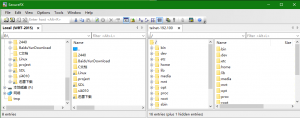
这样，每次启动系统都需要验证身份,并且不会再显示 Please press Enter to activate this console.   
  
8.有很多人想：平时的使用中都是用SSH和FTP登录ubuntu系统来执行各种命令，上传下载各种文件，开发板既然也运行了linux系统，那我应该也可以通过这些方式登录上去才对呀。想法没有错，现在就来实现。由于busybox本身已经提供了这些应用，直接使用就可以了。现在就用telnet、ftp、tftp三种方式来操作开发板。    
  
telnet：这个最简单，直接运行开发板上的telnetd命令，然后你就可以通过telnet工具登录过去了，登录时需要验证身份。没图没真相，我用windows下的cmd登录，截张图看看吧。   
   
    
ftp：同样的道理，需要运行开发板上的ftpd命令。不过这个命令需要参数，或者写入配置文件。一开始我们肯定不知道需要什么参数了，直接执行看看有什么提示吧。    


一下出来这么多，仔细看一下，选择第一种方法吧，将配置信息写入脚本配置文件中，然后启动那个脚本就好了。

网上搜索，发现可以写入这里 ---/etc/inetd.conf

直接一句命令搞定

echo "21 stream tcp nowait root ftpd ftpd -w /" >>/etc/inetd.conf

然后运行命令 inetd 就可以启动ftp服务了。现在你用客户端登录吧。如果你用SecureCRT和SecureFX，那么可参考一下我的设置。   
  
  
  
  
  
  
设置完成后，使用telnet连接，就可看到SecureCRT如图  
  
  
建立ftp连接，如图所示  
  
  
  
  
tftp:这个跟ftp类似，也是把配置信息写入脚本。

echo "69 dgram udp nowait  root  /usr/sbin/tftpd  tftpd -l"  >>/etc/inetd.conf

这三个服务都很有用，干脆开机启动好了。在/etc/init.d/rcS中添加  inetd & 就好了。这个tftp的用法我就不再演示了。

到这里，文件系统也比较完善了，再用NFS启动一下，确保没问题后，使用

mkyaffs2image rootfs 201601214.yaffs2  制作烧录镜像，然后烧写到开发板上  
  
  
还可以完善的地方：  
1.前边将设置开发板ip的命令写在了etc/init.d/rcS中，这其实很不灵活，换个路由器ip就不一定可用了。  
解决办法：打开 udhcpc ,让开发板开机时自动获取ip。  
用法：将  udhcpc 写入etc/init.d/rcS中，开机就会运行  
  
