Locally to build their own root file system directory structure, and to establish the root directory named myrootfs first, then build up the necessary subdirectory in the directory, as follows:

```
#cd /home
#mkdir myrootfs
#mkdir bin dev etc lib proc sbin tmp usr var
#mkdir usr/bin usr/lib usr/sbin
```

Directory is established, it is necessary to give the corresponding directory copy the appropriate files or libraries in the lib directory you want to copy the glibc library (Busybox is statically compiled, you do not need to copy), set up some system configuration etc directoryfile created in the dev directory device file placed bin directory command tool, the following describes how commonly used in embedded systems the Busybox tools to produce command toolset.

Download busybox source package

http://busybox.net/downloads/busybox-1.3.2.tar.bz2

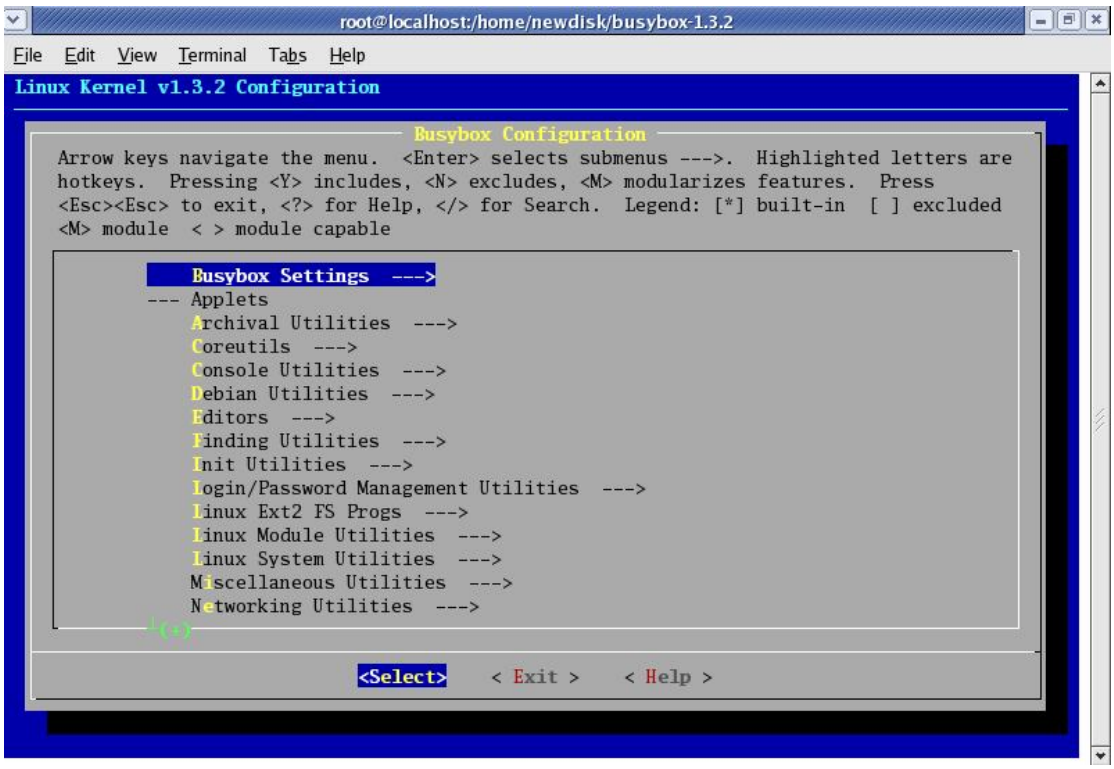
```
#tar xvjf busybox-1.3.2.tar.bz2
#cd busybox-1.3.2
```

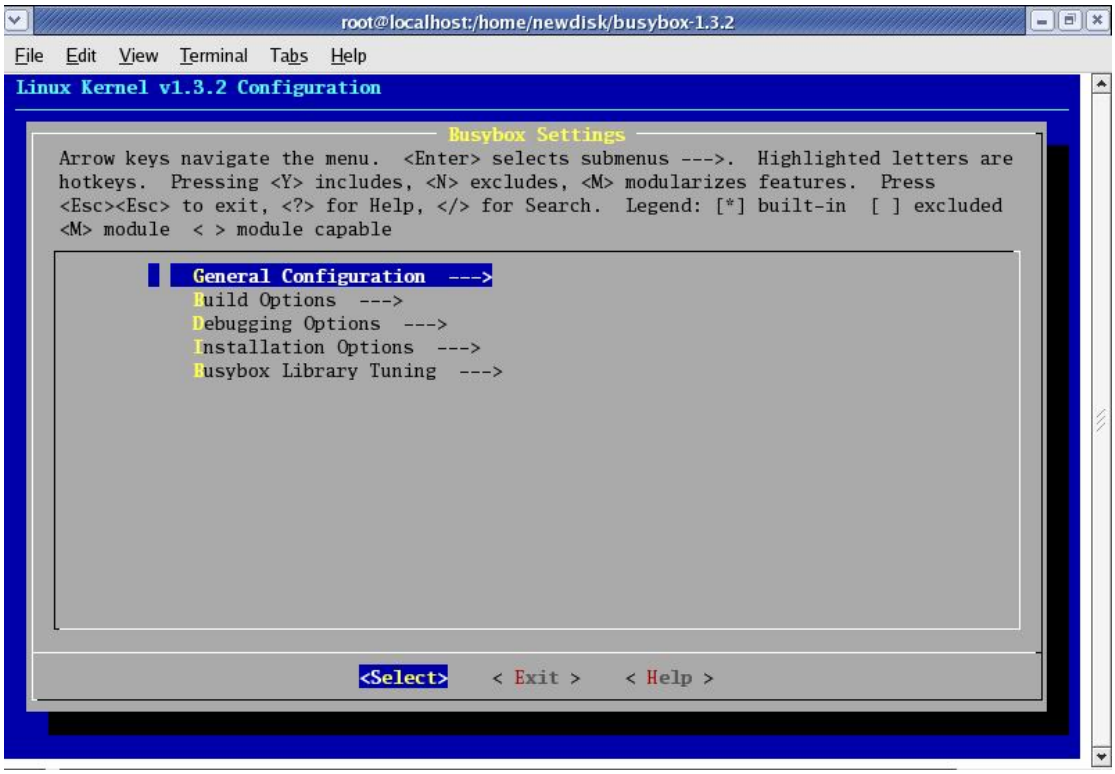
Modify Makefile

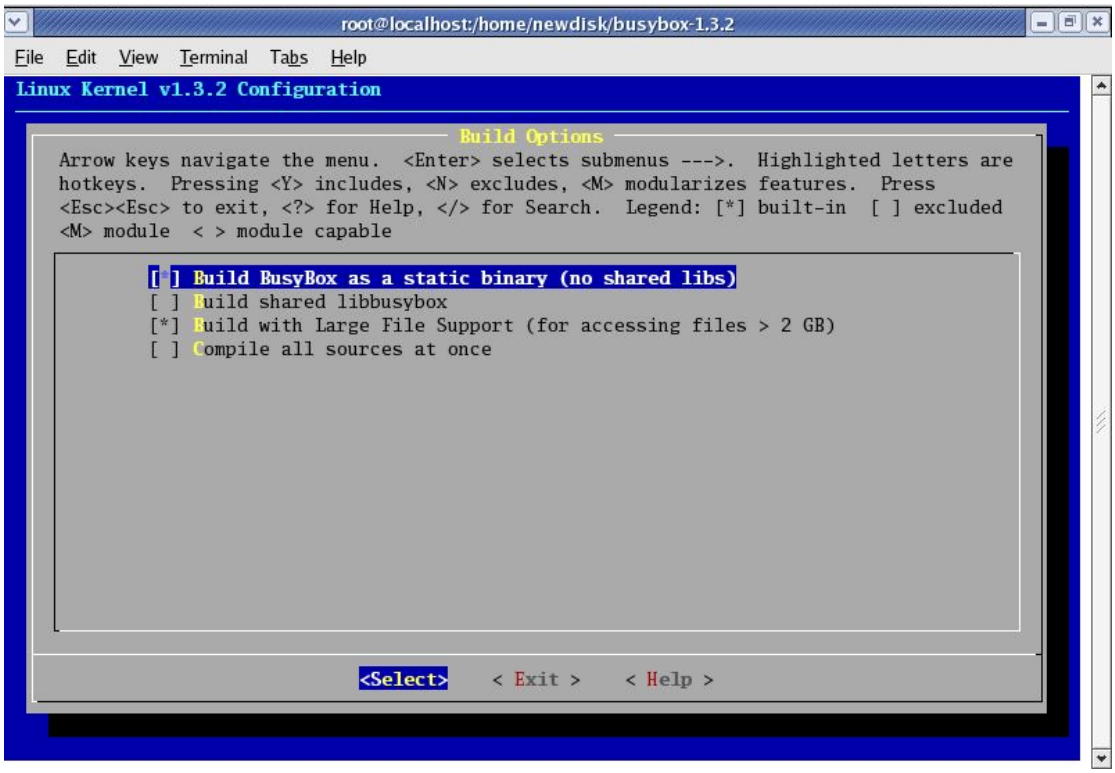
```
ARM = arm
CROSS_COMPILE = /usr/local/arm/3.3.2/bin/arm-linux-
```

Then configure common configuration make menuconfig If you want to select as many configuration items, then you can use make defconfig command, it will be automatically configured to the greatest common configuration options, so that the configuration process to become moresimple and fast.

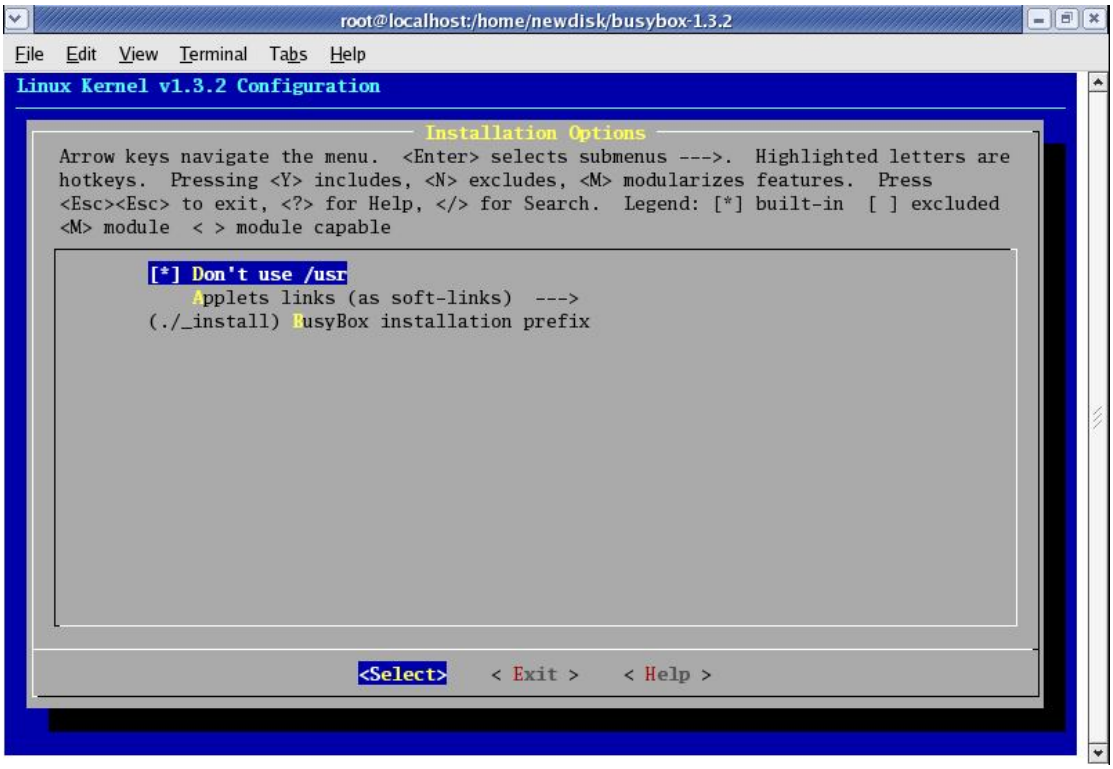
```
#make defconfig
#make menuconfig
```

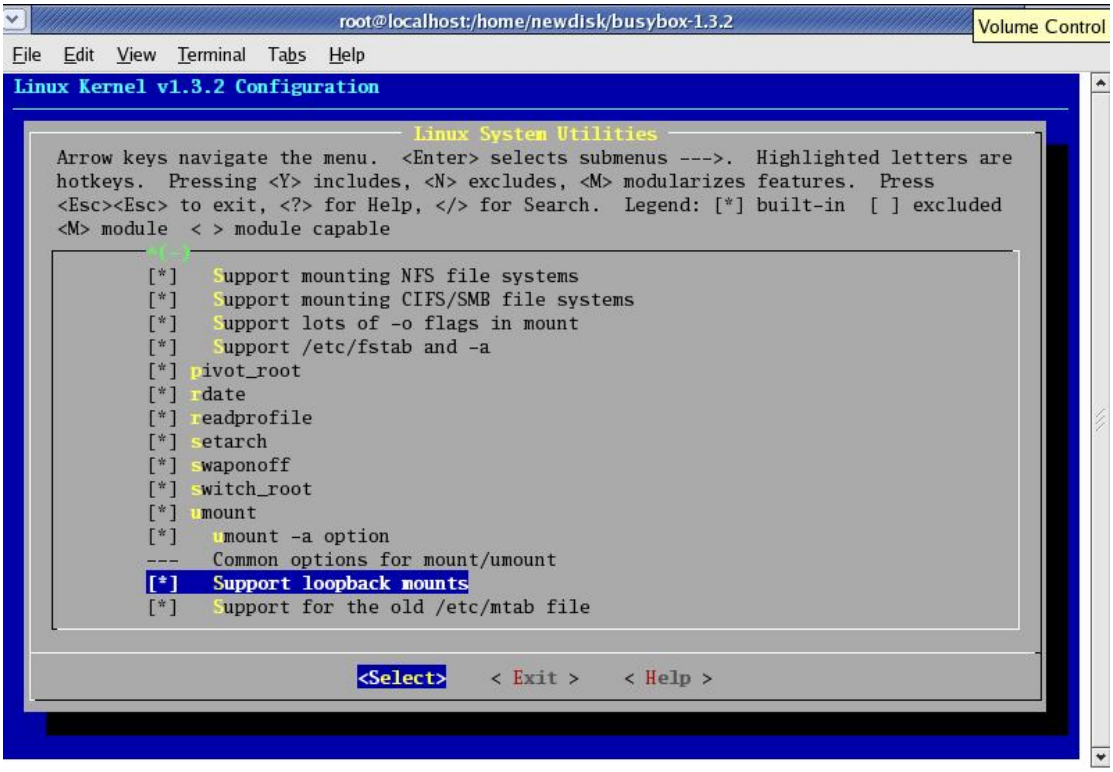




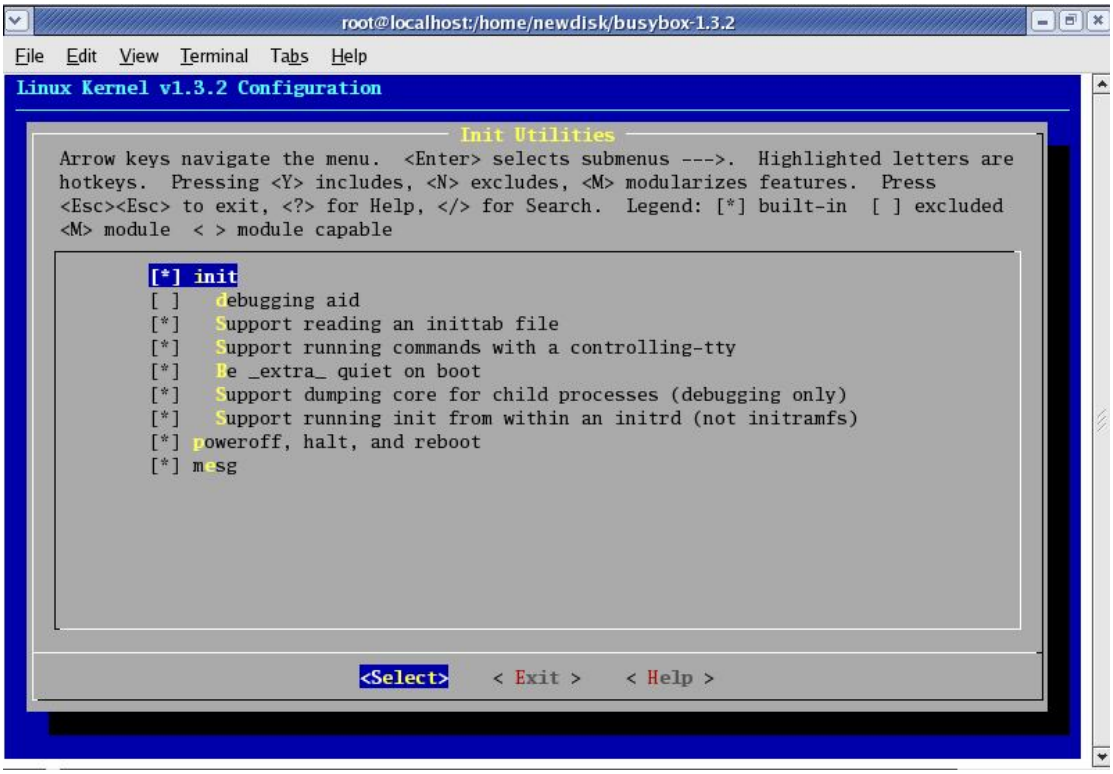


`Build options` Select a static library

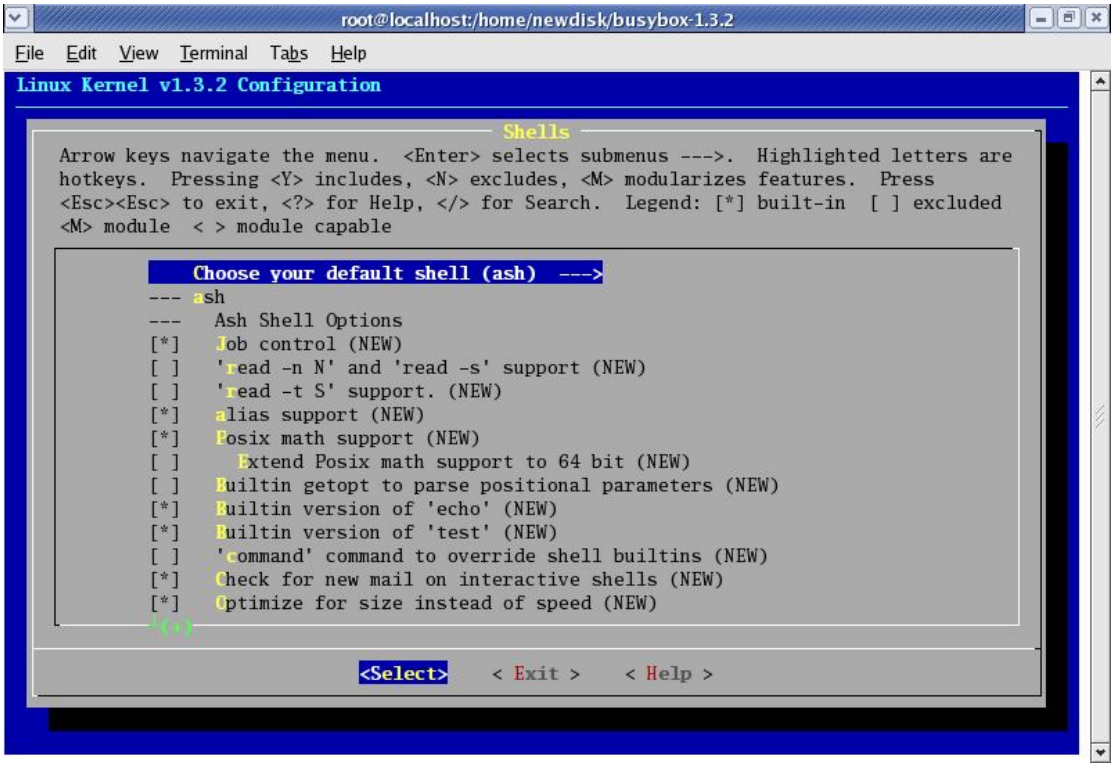




In Linux System Utilities Options，“Support loopback mounts” and “Support for the old /etc/mtab file” 2 options should be selected



`Init Utilities` Options `"Support reading an inittab file"` To select



`In Shells` Options `ash,` Established `rcS` Script to perform `(In etc/init.d/` Directory`)`

Configuration is complete

```
#make
```

Error is as follows:

```
applets/applets.c:22:2: #warning Static linking against glibc
produces buggy executables
applets/applets.c:23:2: #warning (glibc does not cope well with ld
--gc-sections).
applets/applets.c:24:2: #warning See
sources.redhat.com/bugzilla/show_bug.cgi?id=3400
applets/applets.c:25:2: #warning Note that glibc is utterly
unsuitable for static linking anyway.make[1]: *** [applets/applets.o]
Error 1
make: *** [applets] Error 2
```

Modify `busybox-1.3.2/applets/applets.c`

```
/* Apparently uclibc defines __GLIBC__ (compat trick?). Oh well. */
#if ENABLE_STATIC && defined(__GLIBC__) && !defined(__UCLIBC__)
//#warning Static linking against glibc produces buggy executables
//#warning (glibc does not cope well with ld --gc-sections).
//#warning See sources.redhat.com/bugzilla/show_bug.cgi?id=3400
//#warning Note that glibc is utterly unsuitable for static linking
anyway.
#endif
```

The meaning of this warning is to tell you the best with `uclibc` Compiled，Rather than `glibc.` Because `glibc` relatively large， `busybox`Larger and more in the embedded systems Cuntucunjin in the use, so there will be such a requirement.

```
#make
```

In several files compiled the following error is encountered

```
busybox-1.3.2/miscutils/readahead.c
busybox-1.3.2/miscutils/ taskset.c
```

Reconfiguration `make menuconfig,` Remove the corresponding option

```
Miscellaneous Utilities->
                []readahead
                []taskset
#make
```

Compiled successfully

Run the install command

```
#make install
```

`In busybox-1.3.2/_install` Directory to generate `bin sbin` Two directories and `linuxrc file`

```
#cd _install
#ls -l
drwxr-xr-x  2 root root 4096 Jul 15 01:48 bin
lrwxrwxrwx  1 root root   11 Jul 15 01:48 linuxrc -> bin/busybox
```

```
drwxr-xr-x  2 root root 4096 Jul 15 01:48 sbin
```
File modification `busybox-1.3.2/_install/bin/busybox` The properties

```
chmod  4775 _install/bin/busybox
```
Must want to modify the properties, otherwise many commands in busybox restricted
`Will install` directory `bin,sbin` Copy the file to the root file system directory just created directory `myrootfs/bin and myrootfs/sbin.`
Note that using the cp command to copy the-r option to add or copy past files take up a lot of room
Needed to start the four files in the root file system `linuxrc rcS  inittab fstab`
```
(1) linuxrc
#cd myrootfs
#vi linuxrc
```
Reads as follows:
```
#!bin/sh
echo "hello world!"
echo "now exec /sbin/init......"

exec /sbin/init
```
Change the file permissions :`#chmod 775 linuxrc`
`Make linuxrc` has execute permission
```
(2)rcS
#cd myrootfs
#mkdir etc/init.d
#vi etc/init.d/rcS
```
Reads as follows:
```
#!bin/sh
#mount all filesystem defined in "fstab"
echo "mount all......."
/bin/mount –a
```
Change the file permissions:`#chmod 775 etc/init.d/rcS`
`Make rcS` has execute permission
```
(3)inittab
#vi etc/inittab
```
Reads as follows:
```
::sysinit:/etc/init.d/rcS
::respawn:-/bin/sh
::shutdown:/bin/umount -a -r
 (4)fstab
#vi etc/fstab
```
Reads as follows:
```
none      /proc  proc   defaults    0 0
```

In myrootfs/dev nodes created under the directory console,null
With root identity created

```
# mknod -m 600 dev/console c 5 1
# mknod -m 666 dev/null c 1 3
```

Some lib files copied to the myrootfs/lib/ Directory from the development board file system lib directory copy Over, as follows:

```
ld-2.2.2.so           libm.so.6                  libtermcap.so.2
ld-linux.so.2      libnss_dns-2.2.2.so           libtermcap.so.2.0.8
libc-2.2.2.so      libnss_dns.so.2               libutil-2.2.2.so
libcrypt-2.2.2.so  libnss_files-2.2.2.so         libutil.so.1
libcrypt.so.1      libnss_files.so.2             mmc_fix.o
libc.so.6         libpthread-0.9.so              mmcsd_core.o
libdl-2.2.1.so     libpthread.so.0               mmcsd_disk.o
libdl.so.2        libresolv-2.2.2.so             mmcsd_slot.o
libjpeg.so.62      libresolv.so.2                modules
libjpeg.so.62.0.0  libstdc++-3-libc6.1-2-2.10.0.so  yaffs.o
libm-2.2.2.so      libstdc++-libc6.1-2.so.3
```

Manufacture **ramdisk**

Reference Links:

http://blog.csdn.net/wushuan10141/archive/2008/07/25/2709690.aspx

```
#mkdir /mnt/loop
#dd if=/dev/zero of=/home/myrootfsimage bs=1k count=15360
#mke2fs -F -v -m 0 /home/myrootfsimage

#mount -o loop /home/myrootfsimage  /mnt/loop
#cp –fr /home/myrootfs/*   /mnt/loop/
```

Note Be sure to add **-r**

```
#umount /mnt/loop
#gzip  -v9 /home/myrootfsimage
```

I u-boot used bootm command, so you must use the mkimage Tools will myrootfsimage.gz add 0x40 byte header information

```
#mkimage -A ARM -O Linux -T ramdisk -C gzip -a 0x30800000 -e 0x30800040
-n  ramdisk_image -d myrootfsimage.gz  uramdiskimage
```

It should be noted that: #dd if=/dev/zero of=/home/myrootfsimage bs=1k count=15360

Myrootfsimage The size of 15360kb,is 15M,

in the configuration linux Kernel, to set the ramdisk to set the size of greater than 15M

```
Device  Drivers ->
       Block Device->
          <*>RAM disk support
```

```
(16)Default number of RAM disks 16 instead 4
(4096)Default RAM disk size(kbytes) 4096 instead 16384
```

Manufacture **cramfs**

```
Use mkcramfs tool
#mkcramfs myrootfs  te2410rootfs.cramfs
```

`Can mount` Added command to check whether the root file system directory

```
#mount –o loop te2410rootfs.cramfs  /mnt
```

In ramdisk in mount NFS

In doing embedded development, if you want to increase the speed of development is to try to work transferred to the PC to do, which is not only easy to operate, the access speed is not a development board that can be compared. To facilitate debugging and development, will inevitably have to host an NFS file system.

Set nfs server

Setting up the PC `ip:#ifconfig eth0 192.168.1.110`

In the terminal, enter `ntsysv` Startup settings，And then select the `portmap and nfs` Options，Save restart `nfs`

Start the nfs method：`/etc/rc.d/init.d/nfs restart`

Start portmap method：`/etc/rc.d/init.d/portmap  restart`

Set up a shared directory

Modify `/etc/exports` file, add the shared directory, I shared directory is `/home/nfs`

Add the contents are as follows `:/home/nfs *(rw,async)`

By command in the terminal `exportfs -rv` Shared directory is shared out. `(exportfs –uv` To turn off sharing`)`

Turn off the firewall

```
#setup
```

`Select Firewall configuration,` Then select `disable can`

Configuration `linux`Kernel support `NFS`

```
#make menuconfig
 File systems ->
    Network File Systems->
        <*>NFS File system support
        [*]Provide NFSv3 client support
        [*]  Provide client support for the NFSv3 ACL protocol extension
        [*]Provide NFSv4 client support
```

Client (development board) configurations:

```
Setup ip:#ifconfig eth0 192.168.1.105
Mount nfs:#mount –t nfs 192.168.1.110:/home/nfs  /tmp
```

Yaffs

Use Feiling mkyaffsimage Manufacture yaffs File System

```
#mkyaffsimage  myrootfs  myrootfs.yaffs
```

Use uboot of nand write.yaffs command to program yaffs file System

File lines start well uboot after

Will myrootfs.yaffs downloaded to a memory, the memory address is 0x30000000

```
#tftp 0x30000000 myrootfs.yaffs
```

Erase nand flashof usr partition

My nand flash Partition as follows:

```
mtdpart info. (5 partitions)
name            offset       size       flag
-------------------------------------------------
vivi           : 0x00000000    0x00020000     0  128k
param          : 0x00020000    0x00010000     0   64k
kernel         : 0x00030000    0x001c0000     0    1M+768k
rootfs         : 0x00200000    0x02000000     0   32M
user           : 0x02200000    0x01e00000     0   30M(mtdblock4)
#nand erase 0x02200000 0x1E00000
```

Programming yaffs to nand flash in

```
#nand write.yaffs 0x30000000  0x02200000    $(filesize)
```

Embedded file system yaffs transplantation (kernel support yaffs file system)

(1) Online download yaffs file system source code yaffs2.tar.gz

http://www.aleph1.co.uk/cgi-bin/viewcvs.cgi/yaffs2.tar.gz?view=tar

```
#tar xvzf yaffs2.tar.gz
```

(2) To establish yaffs directory, then copy the appropriate files in the the fs directory under the linux source code

```
#cd linux-2.6.14/fs
   #mkdir yaffs
```

Yaffs2 directory and then copy the c source file h header files, Kconfig, Makefile.kernel file to linux-2.6.14/fs/yaffs directory, and Makefile.kernel changed its name to the Makefile.

(3)Modify linux-2.6.14/fs/Kconfig, add source "fs/yaffs/Kconfig"

(4)Modify linux-2.6.14/fs/Makefile, Finally, add

```
   obj-$(CONFIG_YAFFS_FS)      += yaffs/
```

(5)make menuconfig

```
  File systems->
      Miscellaneous filesystems->
         <*>Yaffs file system support
```

To mount yaffs File system

```
mount –t yaffs /dev/mtdblock/4  /usr
```