

Root Filesystem

단국대학교

컴퓨터학과

2009

백승재

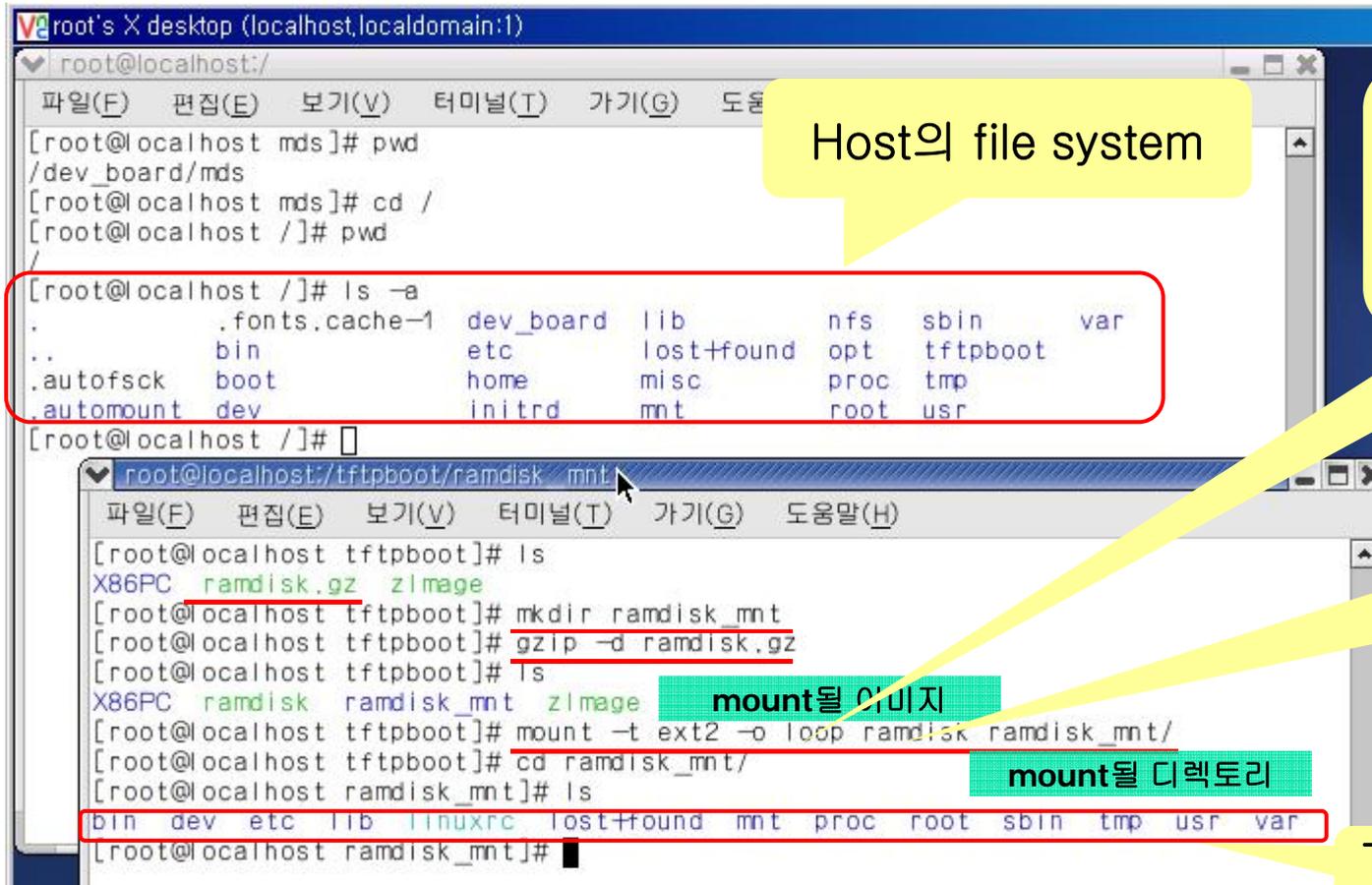
ibanez1383@dankook.ac.kr

<http://embedded.dankook.ac.kr/~ibanez1383>

강의 목표

- Ramdisk에 대한 이해
- Root filesystem의 구조 이해
- Root filesystem 제작 과정 이해
- Ramdisk기반 root filesystem 제작

RAMdisk?



Host의 file system

loopback device 란:
파일하나를 마치 디바이스 처
럼 취급할 수 있게 해준다.
즉 파일 한 개를 마치 하나의
디스크처럼 인식시켜준다.

이 이미지가 마치 하나
의 디바이스처럼 취급
되어 마운트 될 수 있게
해준다.

Target board에서 사
용될 file system

- ✓ HDD가 없는 Embedded System에서 RAM 상에 디스크 처럼 쓸 수 있도록 구축한 공간

RAMDISK 기능과 구조

■ RAMdisk 수정

```

root's X desktop (localhost,localdomain:1)
root@localhost:/
root@localhost:/dev_board/mds
파일(F) 편집(E) 보기(V) 터미널(T) 가기(G) 도움말(H)
[root@localhost root]# cd /dev_board/mds
[root@localhost mds]# ls
Ami-linux.tar.gz          s3c2410_kernel2.4.18_r1.1.tar.bz2
arm-linux-tools.tar.bz2  s3c2410_yivi_r1.0.tar.bz2
bootp_2_4_3_7_i386.rpm   sjf2410_v4.zip
cross_2_4_3_7_i386.rpm
jfl
lin
root@localhost:/tftpboot/ramdisk_mnt/root
파일(F) 편집(E) 보기(V) 터미널(T) 가기(G) 도움말(H)
root@localhost ramdisk_mnt]# ls
root bin dev etc lib linuxrc lost+found mnt proc root sbin tmp
s3c [root@localhost ramdisk_mnt]# cd root
[ro [root@localhost root]# cp /dev_board/mds/test_arm ./
[root@localhost root]# ls -al
합계 17
drwxr-xr-x  2 root  root          1024  9월  11 15:27 .
drwxr-xr-x 14 root  root          1024  6월  24 11:30 ..
-rwxr-xr-x  1 root  root        13408  9월  11 15:27 test_arm
[root@localhost root]# pwd
/tftpboot/ramdisk_mnt/root
[root@localhost root]#
  
```

RAMdisk 에 추가할
임의의 파일을 복사

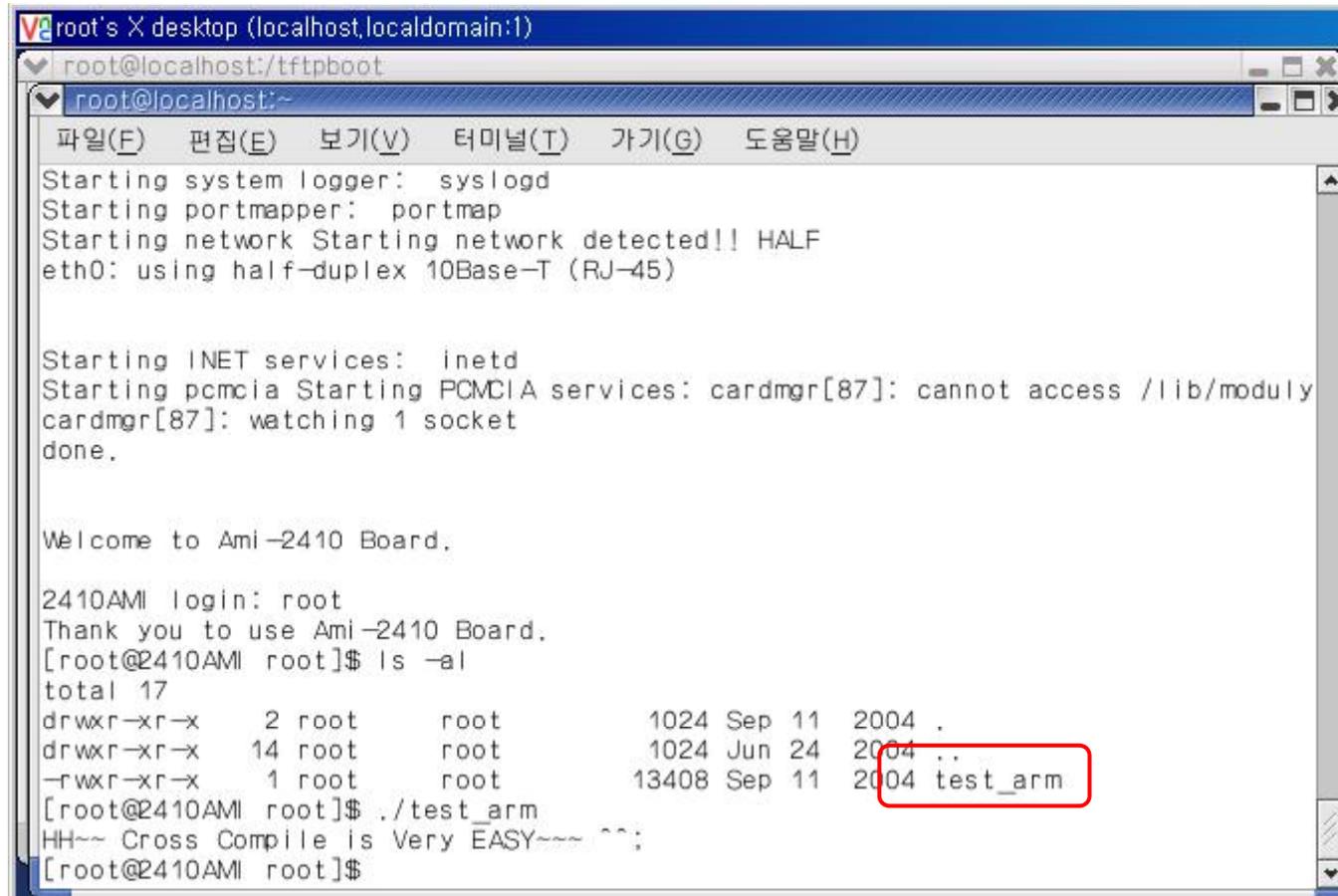
RAMDISK 기능과 구조

```
root's X desktop (localhost,localdomain:1)
root@localhost:/tftpboot
파일(F) 편집(E) 보기(V) 터미널(T) 가기(G) 도움말(H)
[root@localhost tftpboot]# pwd
/tftpboot
[root@localhost tftpboot]# umount ramdisk_mnt/
[root@localhost tftpboot]# gzip ramdisk
[root@localhost tftpboot]# ls -al
합계 3940
drwxr-xr-x  4 root    root      4096  9월 12 08:50 .
drwxr-xr-x 23 root    root      4096  9월 12 06:48 ..
drwxr-xr-x  3 root    root      4096  7월 12 23:15 X86PC
-rw-r--r--  1 root    root    3121035 9월 11 18:16 ramdisk.gz
drwxr-xr-x  2 root    root      4096  9월 12 07:12 ramdisk_mnt
-rw-r--r--  1 root    root    919556  9월 11 18:42 zimage
[root@localhost tftpboot]# █
```

압축 전에 반드시 마운트를 해제

수정된 새로운 RAMdisk확인

- 수정된 RAMdisk로 부팅 확인



```
root's X desktop (localhost,localdomain:1)
root@localhost:/tftpboot
root@localhost:~
파일(F) 편집(E) 보기(V) 터미널(T) 가기(G) 도움말(H)
Starting system logger: syslogd
Starting portmapper: portmap
Starting network Starting network detected!! HALF
eth0: using half-duplex 10Base-T (RJ-45)

Starting INET services: inetd
Starting pcmcia Starting PCMCIA services: cardmgr[87]: cannot access /lib/moduly
cardmgr[87]: watching 1 socket
done.

Welcome to Ami-2410 Board.

2410AMI login: root
Thank you to use Ami-2410 Board.
[root@2410AMI root]$ ls -al
total 17
drwxr-xr-x  2 root  root    1024 Sep 11  2004 .
drwxr-xr-x 14 root  root    1024 Jun 24  2004 ..
-rwxr-xr-x  1 root  root   13408 Sep 11  2004 test_arm
[root@2410AMI root]$ ./test_arm
HH~~ Cross Compile is Very EASY~~~ ^^;
[root@2410AMI root]$
```

RAMDISK 기능과 구조

Root filesystem 제작 순서도 1

image file 생성

```
dd if=/dev/zero of=ramdisk bs=1k count=16384
```



image file에 파일 시스템 생성

```
mke2fs ramdisk
```



image file mount

```
mkdir mnt && mount -t ext2 ramdisk ram_point -o loop
```



기본 디렉토리 생성

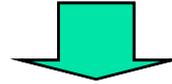
```
mkdir bin dev etc lib mnt proc root sbin tmp usr var
```

- 내부에 file system 구조를 만들어서 loop back device로 mount하여 일반 file system과 같이 사용하는 파일을 말한다.
- Image file 내부에 root filesystem 을 만든다.

Root filesystem 제작 순서도 2



/dev 디렉토리에 device file 구성
복사 할 수 있는 device file은 복사하고 복사 할 수 없는 것은 생성함



/etc 디렉토리에 중요 설정 파일 구성
복사하고 편집함

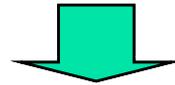


/bin과 /sbin 디렉토리를 구성
busybox를 이용하거나 개개의 실행 파일을 cross 컴파일 하여 구성

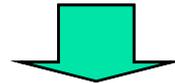


/lib 디렉토리 구성
프로그램의 운영에 필요한 동적 라이브러리들과 module들을 추가함

Root filesystem 제작 순서도 3



/var 디렉토리의 구성
/var/log 디렉토리를 생성



파일 시스템 이미지 압축
umount ram_point 과 gzip -9 ramdisk



kernel과 ramdisk image 보드에 적재
테스트하고 설정을 조정함

■ 램 디스크 제작

```
root's X desktop (localhost,localdomain:1)
root@localhost:/dev_board/mds
[root@localhost mds]# clear
[root@localhost mds]# pwd
/dev_board/mds
[root@localhost mds]# dd if=/dev/zero of=my_ramdisk bs=1024 count=8192
8192+0개의 레코드를 입력
8192+0개의 레코드를 출력
```

디스크에 루트파일 시스템을 만들기 위한 임시파일을 만든다.

블록의 크기만들어질 파일의 크기

0으로 채운다. 출력될 장소

RAMDISK 기능과 구조

```
root's X desktop (localhost,localhost:1)
root@localhost:/dev_board/mds
파일(F) 편집(E) 보기(V) 터미널(T) 가기(G) 도움말(H)
root.cramfs tftp_server_0_17_9_i386.rpm
root_qtopia.cramfs ztelnnet_0.9.1_7mz_i386.rpm
s3c2410_kernel2.4.18_module_mmc.tar.bz2
[root@localhost mds]# clear

[root@localhost mds]# pwd
/dev_board/mds
[root@localhost mds]# dd if=/dev/zero of=my_ramdisk bs=1024 count=8192
8192+0개의 레코드를 입력하였습니다
8192+0개의 레코드를 출력하였습니다
[root@localhost mds]# ls -al | grep my_ramdisk
-rw-r--r-- 1 root root 8388608 9월 11 15:41 my_ramdisk
[root@localhost mds]# mke2fs my_ramdisk
mke2fs 1.32 (09-Nov-2002)
my_ramdisk is not a block special device.
Proceed anyway? (y,n) y
Filesystem label=
OS type: Linux
Block size=1024 (log=0)
Fragment size=1024 (log=0)
2048 inodes, 8192 blocks
409 blocks (4.99%) reserved
First data block=1
1 block group
8192 blocks per group, 8192 fragments per group
2048 inodes per group

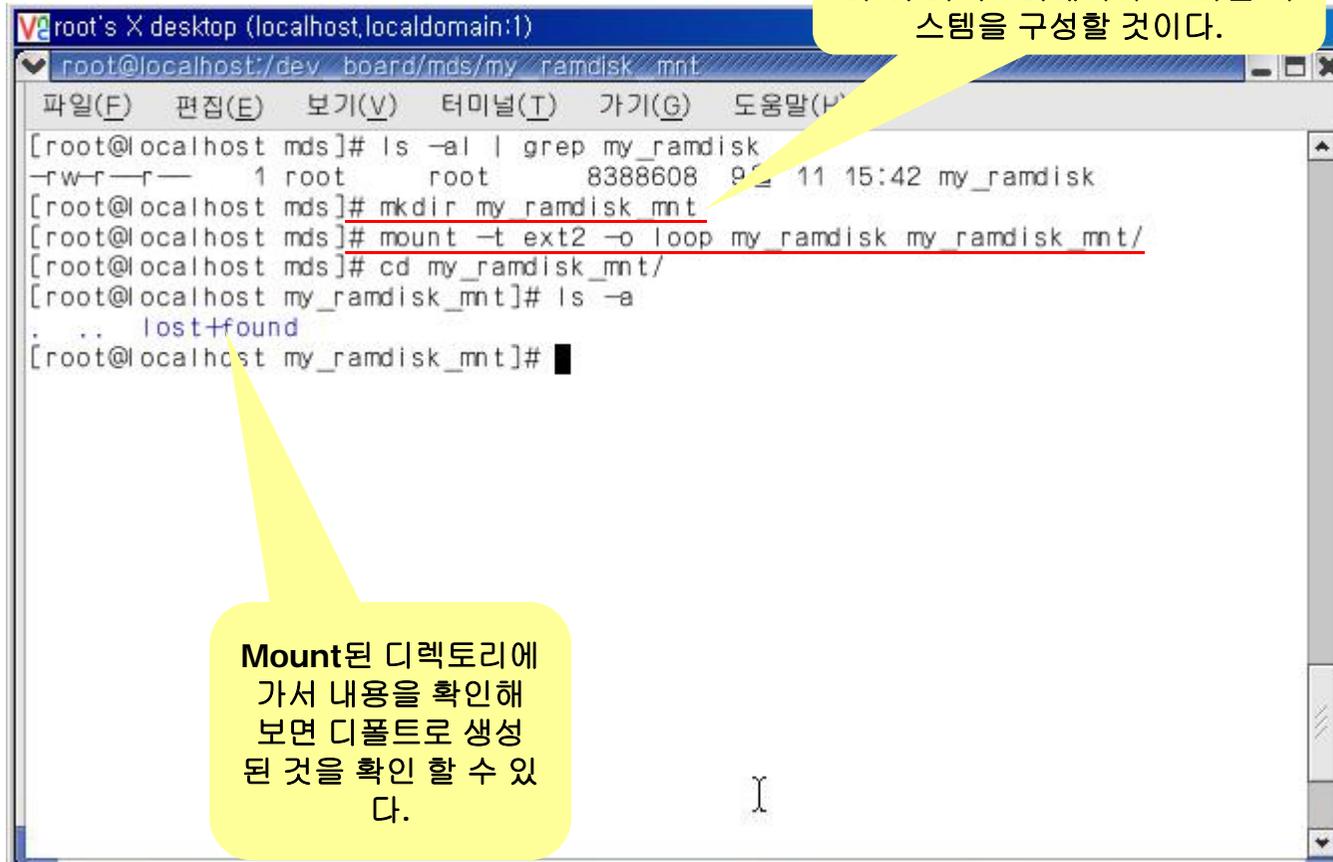
Writing inode tables: done
Writing superblocks and filesystem accounting information: done

This filesystem will be automatically checked every 28 mounts or
180 days, whichever comes first. Use tune2fs -c or -i to override.
[root@localhost mds]#
```

생성된 이미지에 파일시스템을 만들어준다.
즉 ramdisk 에 파일시스템을 만들어준다.

만들어진 이미지를 확인할 수 있다.

RAMDISK 기능과 구조



```
root's X desktop (localhost,localdomain:1)
root@localhost:/dev/board/mds/my_ramdisk_mnt
파일(F) 편집(E) 보기(V) 터미널(T) 가기(G) 도움말(H)
[root@localhost mds]# ls -al | grep my_ramdisk
-rw-r--r-- 1 root root 8388608 9월 11 15:42 my_ramdisk
[root@localhost mds]# mkdir my_ramdisk_mnt
[root@localhost mds]# mount -t ext2 -o loop my_ramdisk my_ramdisk_mnt/
[root@localhost mds]# cd my_ramdisk_mnt/
[root@localhost my_ramdisk_mnt]# ls -a
.  ..  lost+found
[root@localhost my_ramdisk_mnt]#
```

mount 될 디렉토리를 만들어 준다. 이 디렉토리에서 루트 파일 시스템을 구성할 것이다.

Mount된 디렉토리에 가서 내용을 확인해 보면 디폴트로 생성된 것을 확인할 수 있다.

RAMDISK 기능과 구조

```
root's X desktop (localhost,localdomain:1)
root@localhost:/dev_board/mds/my_ramdisk_mnt
파일(F) 편집(E) 보기(V) 터미널(T) 가기(G) 도움말(H)
[root@localhost my_ramdisk_mnt]# mkdir bin dev etc lib mnt proc root sbin tmp usr var
[root@localhost my_ramdisk_mnt]# ls -la
. . bin dev etc lib lost+found mnt proc root sbin tmp usr var
[root@localhost my_ramdisk_mnt]#
```

루트파일시스템의 구성을 만들어준다.

리눅스의 디렉토리의 구조와 유사하게 생성됨을 볼 수 있다.

/dev 디렉토리의 구성 (1/5)

/dev 디렉토리에 사용할 device의 device file을 생성해준다. 기본적인 device file은 host 컴퓨터의 /dev 디렉토리에서 복사하여 사용하면 되며 host 컴퓨터에 없는 device file이라면 mknod 명령어로 만들어 주면 된다. Rebis 보드의 kernel과 device를 기준으로 보면 필요한 device file들은 다음과 같다. (다음 예의 파일명은 디렉토리도 포함됨)

```
console fb3 flh1 mixer null ram3 tty1 tty7 ttyS0
dsp fb4 flh2 mtblock0 ptmx random tty2 tty8 ttyS1
fb fb5 flh3 mtblock1 pts root tty3 ttyP0 ttySA0
fb0 fb6 initctl mtblock2 ram0 ts tty4 ttyP1 ttySA1
fb1 fb7 kmem mtblock3 ram1 tty tty5 ttyP2 ttySA2
fb2 flh0 mem mtblock4 ram2 tty0 tty6 ttyP3 urandom
zero
```

/dev 디렉토리 구성 명령

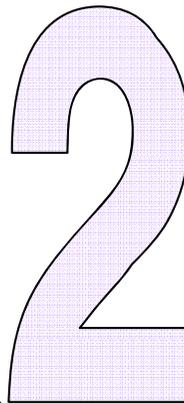
```
cp -dpR /dev/audio ./mnt/dev
cp -dpR /dev/dsp ./mnt/dev
cp -dpR /dev/fb ./mnt/dev
cp -dpR /dev/fb[0-7] ./mnt/dev
cp -dpR /dev/hda ./mnt/dev
cp -dpR /dev/hda[0-8] ./mnt/dev
cp -dpR /dev/initctl ./mnt/dev
cp -dpR /dev/input ./mnt/dev
cp -dpR /dev/ircomm[0-1] ./mnt/dev
cp -dpR /dev/irnet ./mnt/dev
cp -dpR /dev/kmem ./mnt/dev
cp -dpR /dev/log ./mnt/dev
cp -dpR /dev/mem ./mnt/dev
cp -dpR /dev/mixer ./mnt/dev
cp -dpR /dev/mouse ./mnt/dev
cp -dpR /dev/null ./mnt/dev
cp -dpR /dev/pts ./mnt/dev
cp -dpR /dev/ram[0-3] ./mnt/dev
cp -dpR /dev/random ./mnt/dev
cp -dpR /dev/rtc ./mnt/dev
```



cp -dpR 옵션은 파일은 원본
파일의 uid, gid, 권한, 시간
정보가 그대로 복사하고
디렉토리라면 하위 디렉토리
까지 복사하는 것을 의미함

/dev 디렉토리의 구성 (3/5)

```
cp -dpR /dev/tty ./mnt/dev
cp -dpR /dev/tty[0-8] ./mnt/dev
cp -dpR /dev/ttyP[0-3] ./mnt/dev
cp -dpR /dev/ttyS[0-1] ./mnt/dev
cp -dpR /dev/ttySA[0-2] ./mnt/dev
cp -dpR /dev/urandom ./mnt/dev
cp -dpR /dev/video ./mnt/dev
cp -dpR /dev/zero ./mnt/dev
mknod ./mnt/dev/flh0 b 60 0
mknod ./mnt/dev/flh1 b 60 1
mknod ./mnt/dev/flh2 b 60 2
mknod ./mnt/dev/flh3 b 60 3
mknod ./mnt/dev/keypad c 251 0
mknod ./mnt/dev/mmcda b 241 0
mknod ./mnt/dev/mmcda1 b 241 1
mknod ./mnt/dev/mmcda2 b 241 2
mknod ./mnt/dev/mmcda3 b 241 3
mknod ./mnt/dev/mmcda4 b 241 4
mknod ./mnt/dev/mtd3 c 90 6
mknod ./mnt/dev/mtdblock0 b 31 0
mknod ./mnt/dev/mtdblock1 b 31 1
```



/dev 디렉토리의 구성 (4/5)

```
mknod ./mnt/dev/mtdblock2 b 31 2
mknod ./mnt/dev/mtdblock3 b 31 3
mknod ./mnt/dev/mtdblock4 b 31 4
mknod ./mnt/dev/ptmx c 5 2
mknod ./mnt/dev/ts c 11 0
ln -s ram0 ./mnt/dev/root
ln -s ttyS0 ./mnt/dev/console
```

root filesystem이 ram
disk이고 콘솔은 ttyS0
로 심복릭 링크함

3

/dev 디렉토리의 구성 (5/5)

```
★ root@server:/temp/bit computer/rootfs
[root@server rootfs]# ./mkdev.sh
[root@server rootfs]#
[root@server rootfs]#
[root@server rootfs]# ls ./mnt/dev
audio  fb4  hda  hda8  log  mouse  pcmx  rtc  tty5  ttyS0
console fb5  hda1 initctl mem  mtd3  pts  ts  tty6  ttyS1
dsp    fb6  hda2 input  mixer mtdblock0 ram0  tty  tty7  ttySA0
fb     fb7  hda3 ircomm0 nmcda mtdblock1 ram1  tty0  tty8  ttySA1
fb0    flh0 hda4 ircomm1 nmcda1 mtdblock2 ram2  tty1  ttyP0 ttySA2
fb1    flh1 hda5 irnet  nmcda2 mtdblock3 ram3  tty2  ttyP1 urandom
fb2    flh2 hda6 keypad nmcda3 mtdblock4 random tty3  ttyP2 video
fb3    flh3 hda7 kmem   nmcda4 null   root  tty4  ttyP3 zero
[root@server rootfs]#
```

위 명령어들을 shell script로 만든 파일임

[영어] [완성] [두벌식]

/etc 디렉토리의 구성 (1/5)

/etc 디렉토리는 시스템의 중요한 설정 파일들이 들어 있는 디렉토리이다. host 컴퓨터의 /etc 디렉토리에서 필요한 설정 파일들과 script 파일들을 복사하고 파일들을 target board에 맞게 수정해준다. 중요 파일들의 용도와 리스트는 다음과 같다.

/etc 디렉토리의 중요 파일 리스트

fstab	: mount 될 파일 시스템 리스트
inittab	: init 프로세스에 대한 설정
	(앞으로 사용할 busybox의 init에서는 사용하지 않을 것임)
rc.d/*	: system 기동 및 런레벨 변경 script들
passwd	: 사용자의 정보 리스트
shadow	: 사용자들의 패스워드 리스트
group	: system group 리스트
modules	: 부팅시 module loading에 참조하는 파일
기타	: 기타 network 설정에 필요한 파일들이나 shell 환경 설정 파일들

/etc 디렉토리의 구성 (2/5)

/etc 디렉토리 구성 명령

```
TARGETDIR=./mnt/etc
```

```
for FILES in `cat etcfiles.txt`; do cp -rf /etc/$FILES $TARGETDIR; done;
```

```
echo "bin" > $TARGETDIR/ftpusers
```

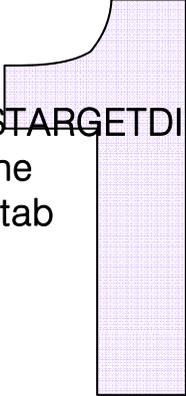
```
echo "" > $TARGETDIR/hosts.equiv
```

```
echo "" > $TARGETDIR/modules
```

```
echo "localnet      127.0.0.1" > $TARGETDIR/networks
```

```
echo "GMT" > $TARGETDIR/timezone
```

```
ln -s /proc/mounts $TARGETDIR/mtab
```

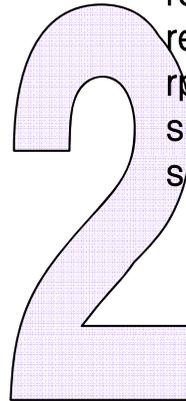


위 명령어는 etcfiles.txt 파일의 내용을 토대로 host 컴퓨터의 /etc 디렉토리의 파일들을 복사

/etc 디렉토리의 구성 (3/5)

etcfiles.txt file의 내용

fstab	passwd
group	pcmcia
host.conf	profile
hosts	protocols
hosts.allow	rc.d
hosts.deny	resolv.conf
inittab	rpc
inputrc	services
ioctl.save	shadow
issue	
issue.net	
ld.so.cache	
localtime	
motd	
nsswitch.conf	



/etc 디렉토리의 구성 (4/5)

```
★ root@server:/temp/bit  computer/rootfs
[root@server rootfs]# ls ./mnt/etc
fstab      hosts.allow  ioctl.save  modules     passwd      resolv.conf
ftusers    hosts.deny   issue       motd        pancia     rpc
group      hosts.equiv  issue.net   mtab        profile     services
host.conf  inittab     ld.so.cache networks     protocols   shadow
hosts      inputrc     localtime   nsswitch.conf rc.d        timezone
[root@server rootfs]#
```

[영어] [완성] [두벌식]

동적 라이브러리들과 kernel module

- 동적 라이브러리는 arm-linux-gcc 컴파일러가 설치된 디렉토리에서 lib 디렉토리의 라이브러리 파일을 사이즈를 줄여서 사용하면 되고 module은 kernel을 컴파일 하여 생성된 module을 사용하면 된다. module은 system의 상황에 따라 틀린 데 현재 system에서 필수요소는 아니기 때문에 없어도 상관없다.

필요한 동적 라이브러리 리스트

ld-2.2.3.so	libdb.so.2	libnsl.so.1	libtermcap.so.2
ld-linux.so.2	libdb.so.3	libnss_dns-2.2.3.so	libtermcap.so.2.0.8
libc-2.2.3.so	libdl-2.2.3.so	libnss_dns.so.2	libutil-2.2.3.so
libc.so.6	libdl.so.2	libnss_files-2.2.3.so	libutil.so.1
libcrypt-2.2.3.so	libm-2.2.3.so	libnss_files.so.2	libcrypt.so.1
libm.so.6	libresolv-2.2.3.so	libdb-2.1.3.so	libnsl-2.2.3.so
libresolv.so.2			

버전 번호는 변경될 수 있음

/lib 디렉토리의 구성 (2/3)

동적 라이브러리 생성 script (mklib.sh)

```
★ root@server:/temp/bit_computer/rootfs
#!/bin/sh
SOURCE_DIR=/usr/local/hybus-arm-linux-R1.1/lib
mkdir /tmp/arm-lib 2> /dev/null

for FILES in `cat ./libfiles.txt`;
do
    cp -dpRf $SOURCE_DIR/$FILES /tmp/arm-lib
    arm-linux-strip /tmp/arm-lib/$FILES 2> /dev/null
    mv -f /tmp/arm-lib/$FILES ./mnt/lib
done

rm -rf /tmp/arm-lib
```

libfiles.txt 파일은 위에서 설명한 필요한 동적 라이브러리 리스트이다.

strip 명령은 binary내에 symbol string을 제거 함으로써 사이즈를 축소함

"mklib.sh" 16L, 301C
[영어] [완성] [두벌식] 16,0-1 모두

/lib 디렉토리의 구성 (3/3)

동적 라이브러리 생성

```
★ root@server:/temp/bit computer/rootfs
[root@server rootfs]# ./mklib.sh
[root@server rootfs]#
[root@server rootfs]#
[root@server rootfs]# ls ./mnt/lib/
ld-2.2.3.so      libdb.so.2      libnsl.so.1      libtermcap.so.2
ld-linux.so.2   libdb.so.3      libnss_dns-2.2.3.so  libtermcap.so.2.0.8
libc-2.2.3.so   libdl-2.2.3.so  libnss_dns.so.2   libutil-2.2.3.so
libc.so.6       libdl.so.2      libnss_files-2.2.3.so  libutil.so.1
libcrypt-2.2.3.so  libm-2.2.3.so  libnss_files.so.2
libcrypt.so.1   libm.so.6       libresolv-2.2.3.so
libdb-2.1.3.so  libnsl-2.2.3.so  libresolv.so.2
[root@server rootfs]#
```

[영어] [완성] [두벌식]

/var 디렉토리의 구성 (1/2)

☞ sub 디렉토리와 log file 생성

- 어떤 login 프로그램이나 init 프로그램의 경우 /var 디렉토리에 log file이 없을때 정상적으로 실행이 안될 수가 있다. 그래서 /var 디렉토리에 필요한 log file과 디렉토리 구조를 만들어 주어야 한다.

명령어

```
mkdir -p ./mnt/var/adm ./mnt/var/lock/subsys ./mnt/var/lib ./mnt/var/log ₩  
./mnt/var/run ./mnt/var/spool/cron/crontabs ./mnt/var/tmp
```

```
touch ./mnt/var/log/dmesg  
touch ./mnt/var/log/lastlog  
touch ./mnt/var/log/messages  
touch ./mnt/var/log/wtmp  
touch ./mnt/var/run/runlevel.dir  
touch ./mnt/var/run/utmp
```

/var 디렉토리의 구성 (2/2)

👉 /var 디렉토리 구성

```
★ root@server:/temp/bit_computer/rootfs
[root@server rootfs]# mkdir -p ./mnt/var/adm ./mnt/var/lock/subsys ./mnt/var/lib
./mnt/var/log \
> ./mnt/var/run ./mnt/var/spool/cron/crontabs ./mnt/var/tmp
[root@server rootfs]#
[root@server rootfs]# touch ./mnt/var/log/dmesg
[root@server rootfs]# touch ./mnt/var/log/lastlog
[root@server rootfs]# touch ./mnt/var/log/messages
[root@server rootfs]# touch ./mnt/var/log/wtmp
[root@server rootfs]# touch ./mnt/var/run/runlevel.dir
[root@server rootfs]# touch ./mnt/var/run/utmp
[root@server rootfs]#
[root@server rootfs]# ls ./mnt/var
adm lib lock log run spool tmp
[root@server rootfs]#
```

[영어] [완성] [두벌식]

RAMDISK 기능과 구조

■ 메인시스템 응용 프로그램

✓ 수천개의 바이너리 명령어를 임베디드 시스템서 어떻게 지원할 것인가?

■ 완전 표준 응용 프로그램

· 응용프로그램을 하나씩 다운받아 컴파일 하여 램디스크에 포함

■ BusyBox

· 대부분의 임베디드 시스템서 사용(대부분의 Arch 지원)
· 작지만 대부분의 명령어를 지원하며 원하는 명령어만 지원가능
· Glibc나 uClibc를 정적 / 동적 링크하여 사용 가능

■ TinyLogin

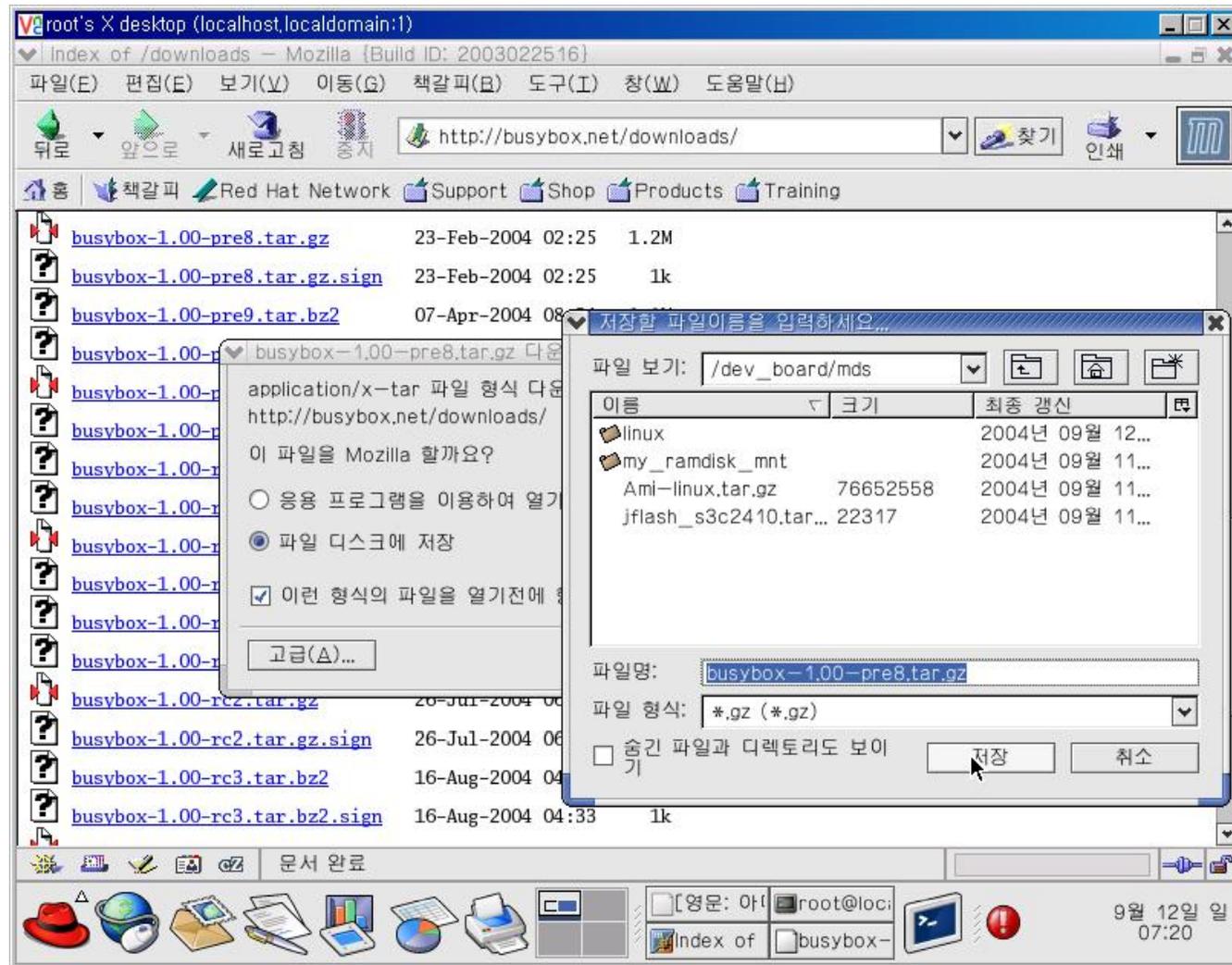
· 여러 로그인 유틸리티를 모아 하나의 바이너리 파일로 만든것
· 보통 BusyBox와 함께 사용

■ Embutils

· 많이 사용되는 명령어 프로그램을 작게 최적화 시킨것
· ARM, i386, PPC, MIPS의 4가지 Arch지원
· 정적 링크만 가능
· BusyBox보다 적은 명령어 지원

RAMDISK 기능과 구조

■ Busybox 다운 받기



■ 압축해제 / 확인

```
root's X desktop (localhost,localdomain:1)
root@localhost:/dev_board/mds
파일(F) 편집(E) 보기(V) 터미널(T) 가기(G) 도움말(H)
[root@localhost mds]# pwd
/dev_board/mds
[root@localhost mds]# ls -al busybox-1.00-pre8.tar.gz
-rw-rw-r-- 1 root root 1295748 9월 12 07:20 busybox-1.00-pre8.tar.gz
[root@localhost mds]# tar zxvf busybox-1.00-pre8.tar.gz

root's X desktop (localhost,localdomain:1)
root@localhost:/dev_board/mds/busybox-1.00-pre8
파일(F) 편집(E) 보기(V) 터미널(T) 가기(G) 도움말(H)
busybox-1.00-pre8/util-linux/getopt.c
busybox-1.00-pre8/util-linux/hexdump.c
busybox-1.00-pre8/util-linux/hwclock.c
busybox-1.00-pre8/util-linux/losetup.c
busybox-1.00-pre8/util-linux/mkfs_minix.c
busybox-1.00-pre8/util-linux/mkswap.c
busybox-1.00-pre8/util-linux/more.c
busybox-1.00-pre8/util-linux/mount.c
busybox-1.00-pre8/util-linux/nfsmount.c
busybox-1.00-pre8/util-linux/nfsmount.h
busybox-1.00-pre8/util-linux/pivot_root.c
busybox-1.00-pre8/util-linux/rdate.c
busybox-1.00-pre8/util-linux/swaponoff.c
busybox-1.00-pre8/util-linux/umount.c
[root@localhost mds]# cd busybox-1.00-pre8
[root@localhost busybox-1.00-pre8]# pwd
/dev_board/mds/busybox-1.00-pre8
[root@localhost busybox-1.00-pre8]# ls
AUTHORS  README  console-tools  editors  libbb  networking  syslogd
Changelog  Rules.mak  coreutils  examples  libpwdgrp  procps  tests
INSTALL  TODO  debian  findutils  loginutils  scripts  testsuite
LICENSE  applets  debianutils  include  miscutils  shell  util-linux
Makefile  archival  docs  init  modutils  sysdeps
[root@localhost busybox-1.00-pre8]#
```

RAMDISK 기능과 구조

■ 파일 수정

```

root's X desktop (localhost.localdomain:1)
root@localhost:/dev_board/mds/busybox-1.00-pre8/init
파일(F) 편집(E) 보기(V) 터미널(T) 가기(G) 도움말(H)
busybox-1.00-pre8/util-linux/mkswap.c
busybox-1.00-pre8/util-linux/more.c
busybox-1.00-pre8/util-linux/mount.c
busybox-1.00-pre8/util-linux/nfsmount.c
busybox-1.00-pre8/util-linux/nfsmount.h
busybox-1.00-pre8/util-linux/pivot_root.c
busybox-1.00-pre8/util-linux/rdate.c
busybox-1.00-pre8/util-linux/swaponoff.c
busybox-1.00-pre8/util-linux/umount.c
[root@localhost mds]# cd busybox-1.00-pre8
[root@localhost busybox-1.00-pre8]# pwd
/dev_board/mds/busybox-1.00-pre8
[root@localhost busybox-1.00-pre8]# ls
AUTHORS  README      console-tools  editors      job          networking  syslogd
Changelog Rules.mak  coreutils      examples     libpwdgrp    procps      tests
INSTALL  TODO       debian         findutils    loginutils   scripts     testsuite
LICENSE  applets   debianutils    include      miscutils    shell       util-linux
Makefile archival  docs          init         modutils     sysdeps
[root@localhost busybox-1.00-pre8]# cd init/
[root@localhost init]# ls
Config.in  halt.c      init_shared.h  msvc.c      reboot.c
Makefile   init.c      msg.c          pidfilehack.c
Makefile.in init_shared.c minit.c       poweroff.c
[root@localhost init]# vi init.c
  
```

Red-hat 설정과는 다른 부분이 있기 때문에 수정 필요

RAMDISK 기능과 구조

```
root's X desktop (localhost,localdomain:1)
root@localhost:/dev/board/mds/busybox-1.00-pre8/init
파일(F) 편집(E) 보기(V) 터미널(T) 가기(G) 도움말(H)
#include <sys/resource.h>
#include <sys/time.h>
#endif

#define KERNEL_VERSION(a,b,c) (((a) << 16) + ((b) << 8) + (c))

#define INITTAB "/etc/inittab" /* inittab file location */
#ifdef INIT_SCRIPT
// #define INIT_SCRIPT "/etc/init.d/rcS" /* Default sysinit script. */
#define INIT_SCRIPT "/etc/rc.d/rc.sysinit" /* modify for RED-HAT */
#endif

#define MAXENV 16 /* Number of env, vars */

#define CONSOLE_BUFF_SIZE 32

/* Allowed init action types */
#define SYSINIT 0x001
#define RESPAWN 0x002
#define ASKFIRST 0x004
#define WAIT 0x008
#define ONCE 0x010
#define CTRLALTDEL 0x020
— 비주열 —
```

데비안의 경우에는 /etc/init.d/rcS에 init에서 처음 실행하는 스크립트가 명시되어있지만 Redhat에서는 /rc.d/rc.sysinit에 명시되어 있다

- config

```
root's X desktop (localhost,localdomain:1)
root@localhost:~/dev_board/mds/busybox-1.00-pre8
파일(F) 편집(E) 보기(V) 터미널(T) 가기(G) 도움말(H)
busybox-1.00-pre8/util-linux/nfsmount.h
busybox-1.00-pre8/util-linux/pivot_root.c
busybox-1.00-pre8/util-linux/rdate.c
busybox-1.00-pre8/util-linux/swaponoff.c
busybox-1.00-pre8/util-linux/umount.c
[root@localhost mds]# cd busybox-1.00-pre8
[root@localhost busybox-1.00-pre8]# pwd
/dev_board/mds/busybox-1.00-pre8
[root@localhost busybox-1.00-pre8]# ls
AUTHORS      README      console-tools  editors      libbb        networking    syslogd
Changelog    Rules.mak   coreutils      examples     libpwdgrp    procps        tests
INSTALL      TODO        debian         findutils    loginutils   scripts       testsuite
LICENSE      applets    debianutils    include      -linux
Makefile     archival   docs           init
[root@localhost busybox-1.00-pre8]# cd init/
[root@localhost init]# ls
Config.in    halt.c      init_shared.h  msvc.c      reboot.c
Makefile     init.c      msg.c          pid.c       stack.c
Makefile.in  init_shared.c  minit.c      swapoff.c
[root@localhost init]# vi init.c
[root@localhost init]# cd ..
[root@localhost busybox-1.00-pre8]# pwd
/dev_board/mds/busybox-1.00-pre8
[root@localhost busybox-1.00-pre8]# make menuconfig
```

BusyBox의 최상위 디렉토리

부팅가능한 BusyBox를 만들기 위해 설정을 해줌

■ Root File system 제작 busybox

설정 menu 항목 1

General Configuration

busybox에 관한 일반적인 설정

Build Options

build에 관한 일반적인 설정 (cross 컴파일 여부 및 공유 라이브러리 사용여부 결정)

Installation Options

install 디렉토리의 설정

Archival Utilities

압축, 패키징 유틸리티의 선택

Coreutils

리눅스 기본 명령어들을 선택 (ex : ls, cp, cat, ...)

Console Utilities

console 관련 명령어들을 선택

Debian Utilities

debian 배포판에서 사용되는 명령어들을 선택

■ Root File system 제작 busybox

설정 menu 항목 2

Editors

editor의 선택 (vi, awk, sed, ...)

Finding Utilities

file을 검색하는 명령어 선택 (find, grep, ...)

Init Utilities

init 파일의 설정

Login/Password Management Utilities

user, group, password 관련 명령어 선택 (adduser, addgroup, passwd, ...)

Miscellaneous Utilities

기타 시스템에서 사용되는 유틸리티 선택 (strings, last, ...)

Linux Module Utilities

linux module 관련 유틸리티 선택 (insmod, lsmod, rmmod, modprobe)

Networking Utilities

network 관련 server daemon 및 유틸리티 선택 (telnet, netstat, ping, httpd, tarceroute, ...)

■ Root File system 제작 busybox

설정 menu 항목 3

Process Utilities

process 관리 유틸리티 선택 (ps, kill, top, ...)

Another Bourne-like Shell

shell 선택 및 설정 (ash, msh, ...)

System Logging Utilities

system log 관련 daemon 및 유틸리티

Linux System Utilities

linux 시스템 설정 유틸리티 선택 (mount, fdisk, dmesg, ...)

Debugging Options

debugging symbol 추가 여부 결정

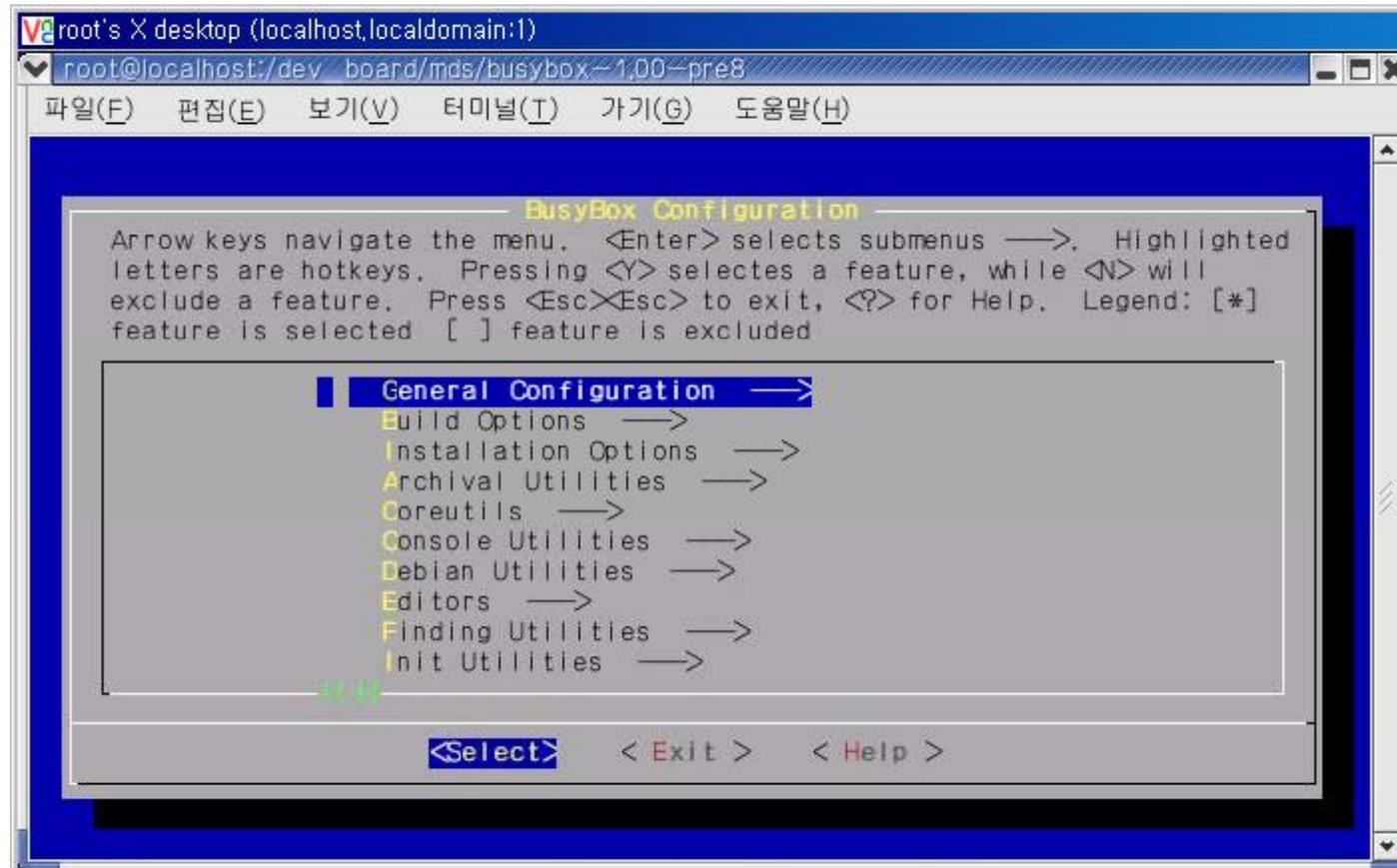
Load an Alternate Configuration File

기존의 설정을 loading

Save Configuration to an Alternate File

현재의 설정을 파일에 저장

■ Configuration



RAMDISK 기능과 구조

root's X desktop (localhost,localhost:1)
root@localhost:/dev_board/mds/busybox-1.00-pre8

파일(F) 편집(E) 보기(V) 터미널(T) 가기(G) 도움말(H)

Build Options
Arrow keys navigate the menu. <Enter> selects submenus —>. Highlighted letters are hotkeys. Pressing <Y> selects a feature, while <N> will exclude a feature. Press <Esc><Esc> to exit, <?> for Help. Legend: [*] feature is selected [] feature is excluded

Cross Compiler prefix
Please enter a string value in the field to the buttons below

arm-linux-

<Select>

Build Options
Arrow keys navigate the menu. <Enter> selects submenus —>. Highlighted letters are hotkeys. Pressing <Y> selects a feature, while <N> will exclude a feature. Press <Esc><Esc> to exit, <?> for Help. Legend: [*] feature is selected [] feature is excluded

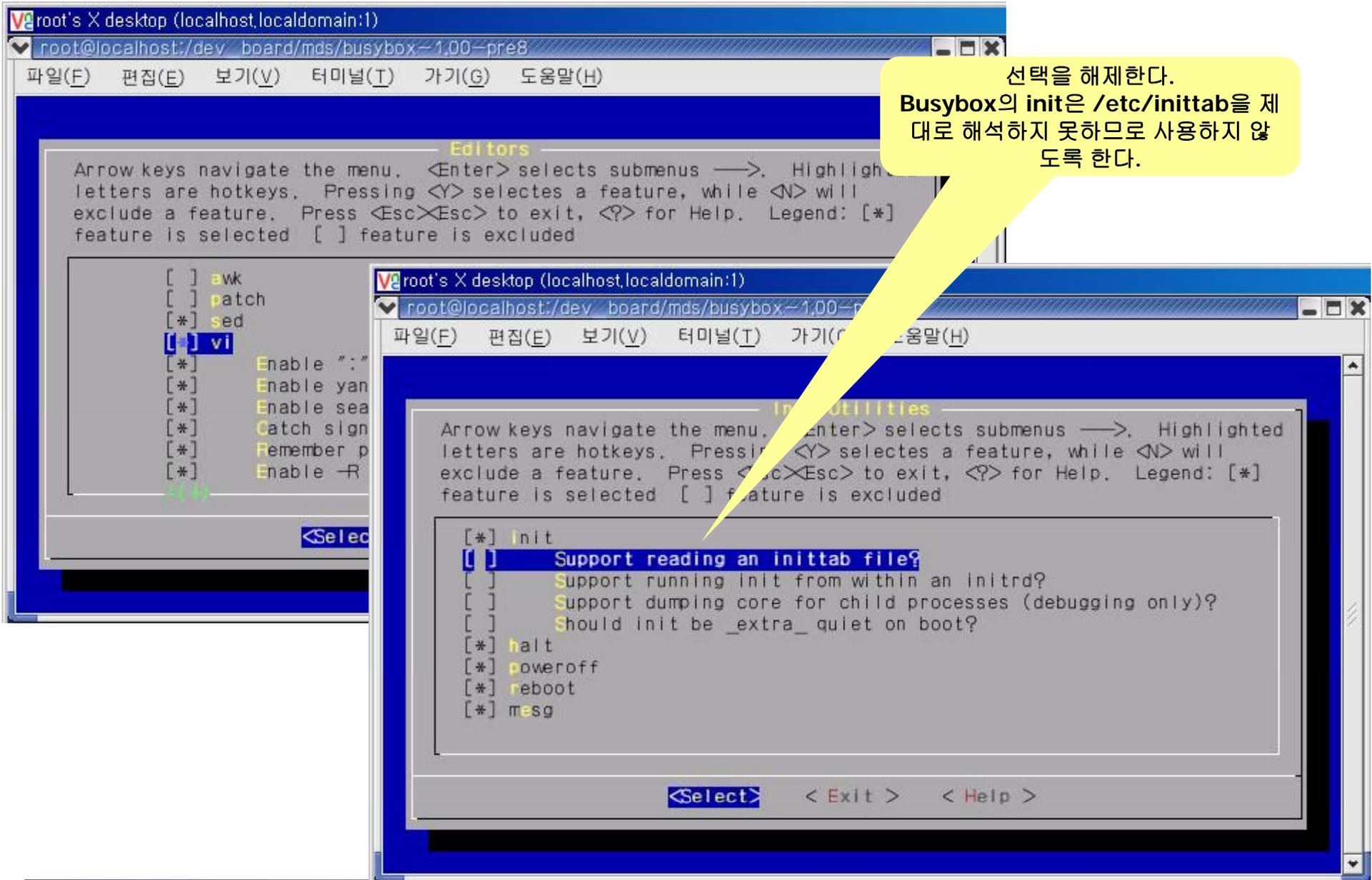
- [*] **Build BusyBox as a static binary (no shared libs)**
- [] Build with Large File Support (for accessing files > 2 GB)
- [*] Do you want to build BusyBox with a Cross Compiler?
(/usr/local/arm/2.9.53/bin/) **C**ross Compiler prefix
- () Any extra CFLAGS options for the compiler?

<Select> <Exit> <Help>

공유라이브러리 사용여부 설정

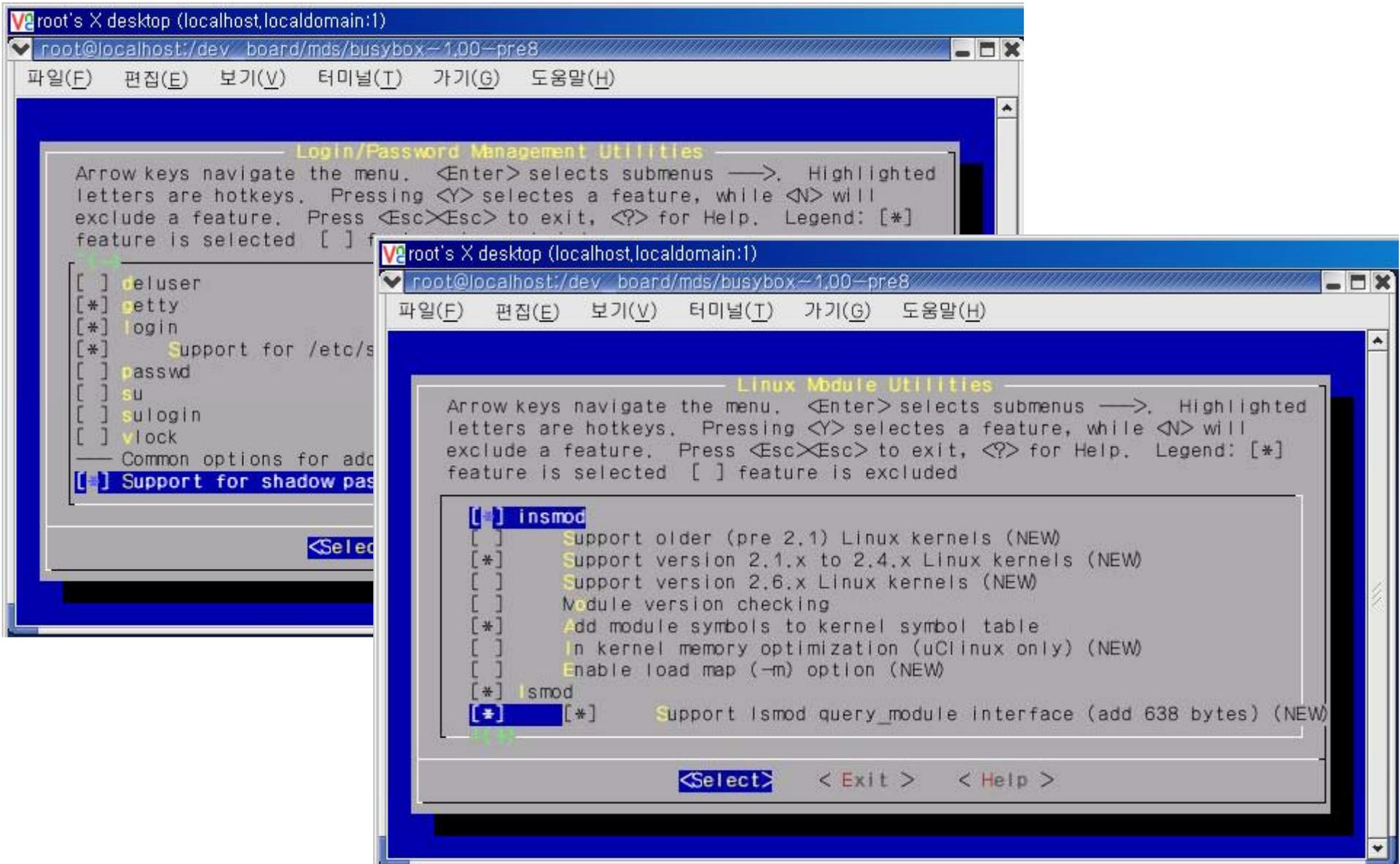
CrossCompiler설정

RAMDISK 기능과 구조

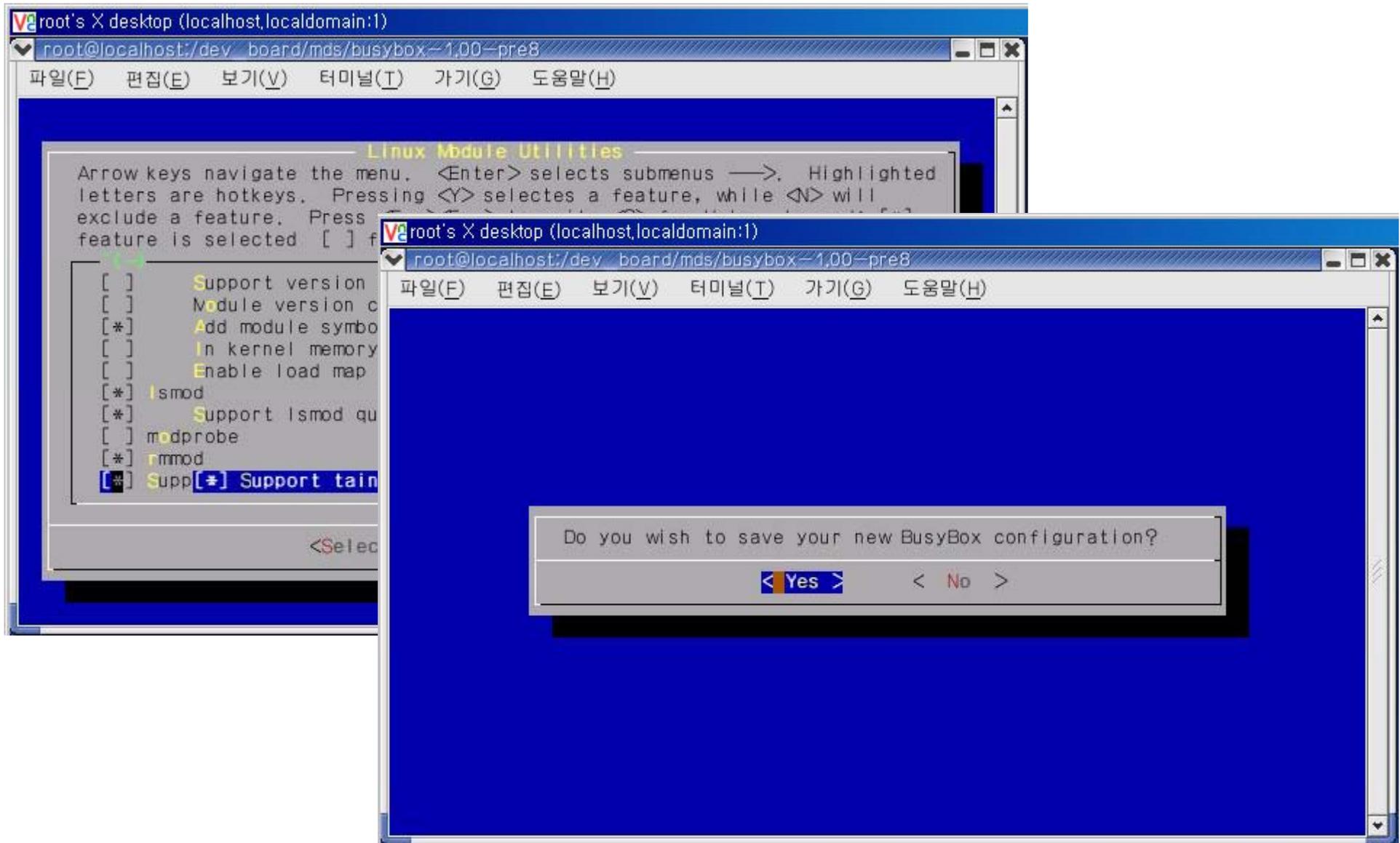


선택을 해제한다.
Busybox의 init은 /etc/inittab을 제대로 해석하지 못하므로 사용하지 않도록 한다.

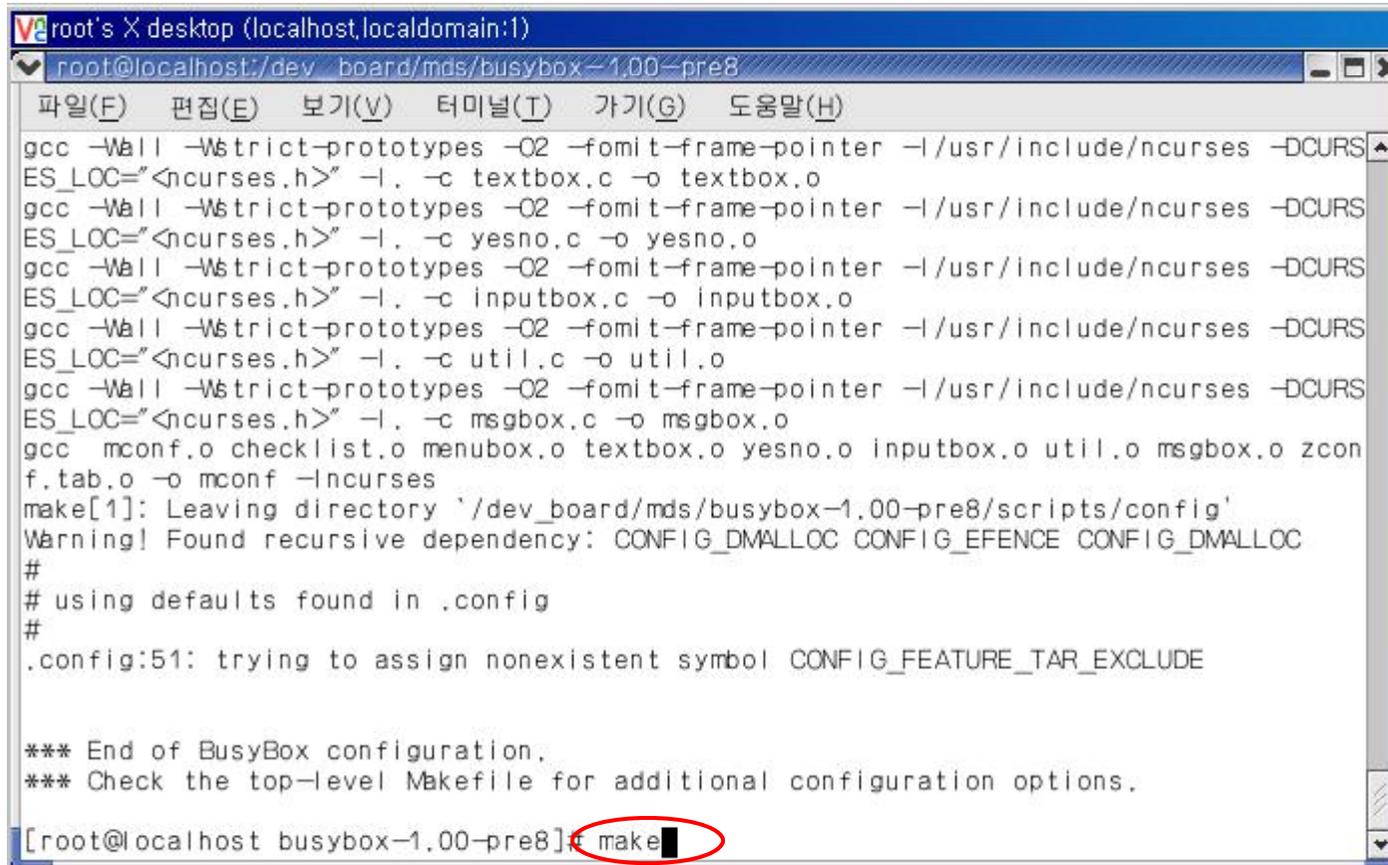
RAMDISK 기능과 구조



RAMDISK 기능과 구조



■ Make 수행

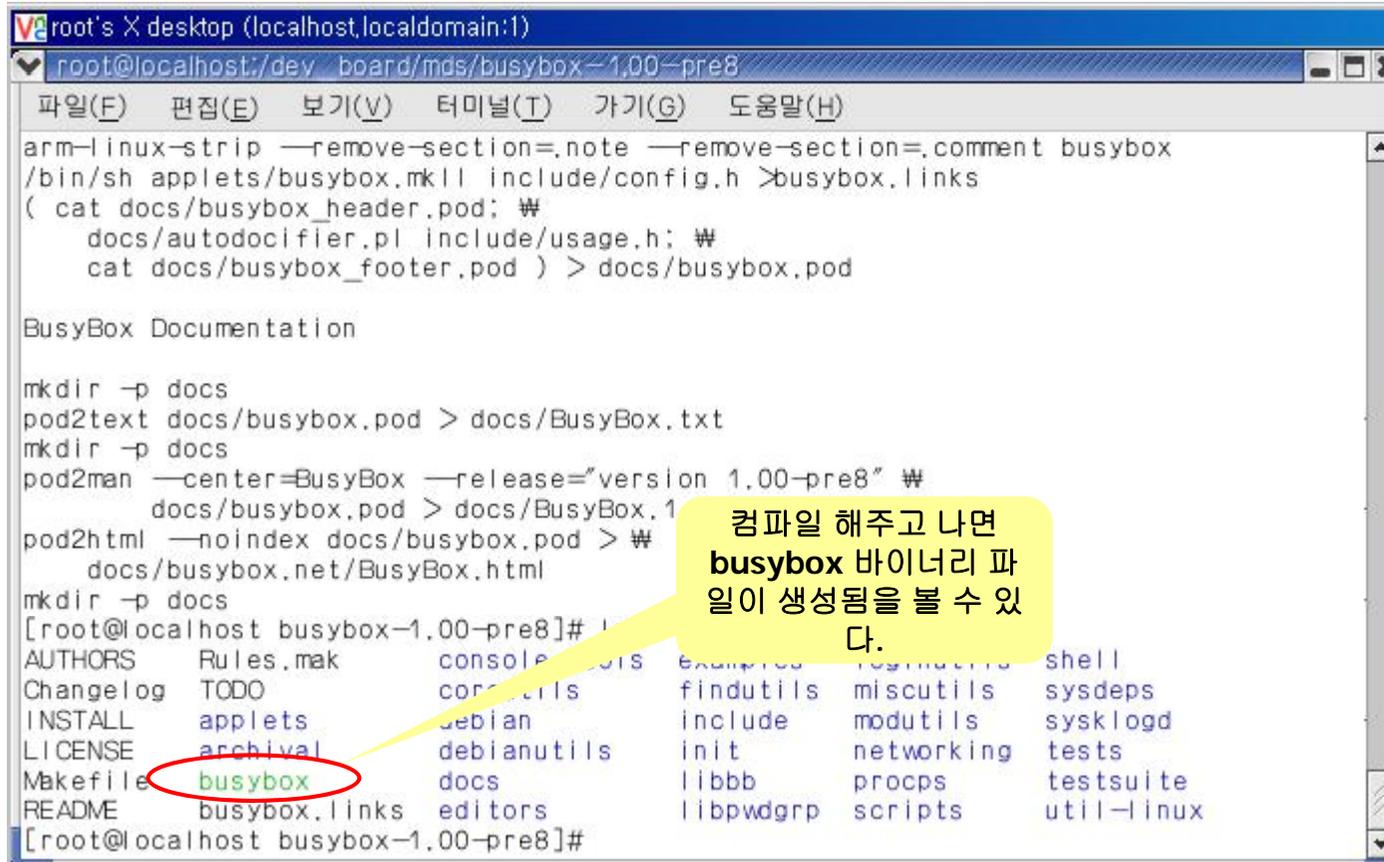


```
root's X desktop (localhost,localdomain:1)
root@localhost:/dev_board/mds/busybox-1.00-pre8
파일(F) 편집(E) 보기(V) 터미널(T) 가기(G) 도움말(H)
gcc -Wall -Wstrict-prototypes -O2 -fomit-frame-pointer -I/usr/include/ncurses -DCURS
ES_LOC="/usr/include/ncurses.h" -I. -c textbox.c -o textbox.o
gcc -Wall -Wstrict-prototypes -O2 -fomit-frame-pointer -I/usr/include/ncurses -DCURS
ES_LOC="/usr/include/ncurses.h" -I. -c yesno.c -o yesno.o
gcc -Wall -Wstrict-prototypes -O2 -fomit-frame-pointer -I/usr/include/ncurses -DCURS
ES_LOC="/usr/include/ncurses.h" -I. -c inputbox.c -o inputbox.o
gcc -Wall -Wstrict-prototypes -O2 -fomit-frame-pointer -I/usr/include/ncurses -DCURS
ES_LOC="/usr/include/ncurses.h" -I. -c util.c -o util.o
gcc -Wall -Wstrict-prototypes -O2 -fomit-frame-pointer -I/usr/include/ncurses -DCURS
ES_LOC="/usr/include/ncurses.h" -I. -c msgbox.c -o msgbox.o
gcc mconf.o checklist.o menubox.o textbox.o yesno.o inputbox.o util.o msgbox.o zcon
f.tab.o -o mconf -lncurses
make[1]: Leaving directory `/dev_board/mds/busybox-1.00-pre8/scripts/config'
Warning! Found recursive dependency: CONFIG_DMALLOC CONFIG_EFENCE CONFIG_DMALLOC
#
# using defaults found in .config
#
.config:51: trying to assign nonexistent symbol CONFIG_FEATURE_TAR_EXCLUDE

*** End of BusyBox configuration.
*** Check the top-level Makefile for additional configuration options.

[root@localhost busybox-1.00-pre8]# make
```

■ 바이너리 파일 확인

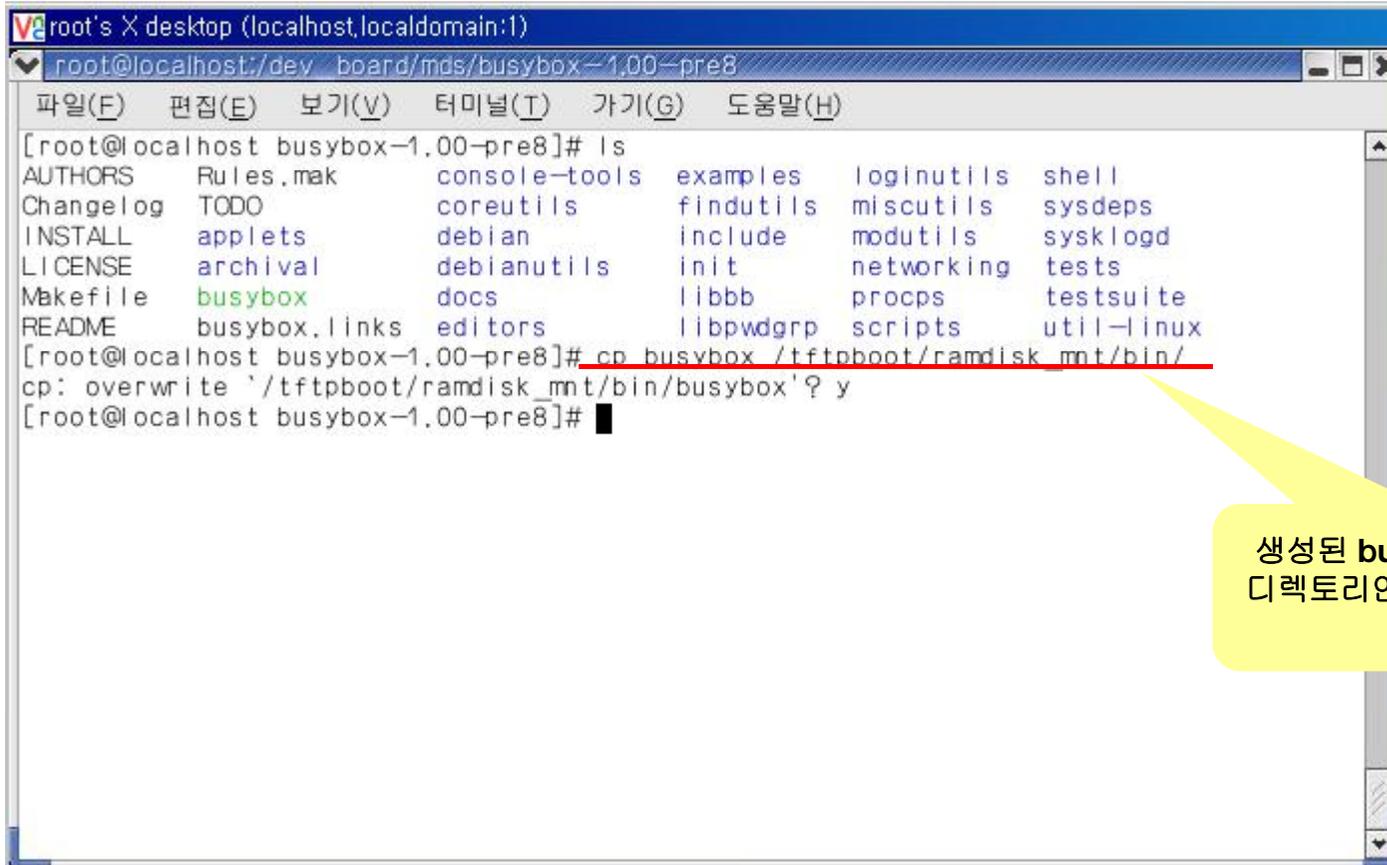


```
root's X desktop (localhost,localdomain:1)
root@localhost:/dev_board/mds/busybox-1.00-pre8
파일(F) 편집(E) 보기(V) 터미널(T) 가기(G) 도움말(H)
arm-linux-strip --remove-section=.note --remove-section=.comment busybox
/bin/sh applets/busybox.mkll include/config.h >busybox.links
( cat docs/busybox_header.pod; #
  docs/autodocifier.pl include/usage.h; #
  cat docs/busybox_footer.pod ) > docs/busybox.pod

BusyBox Documentation

mkdir -p docs
pod2text docs/busybox.pod > docs/BusyBox.txt
mkdir -p docs
pod2man --center=BusyBox --release="version 1.00-pre8" #
  docs/busybox.pod > docs/BusyBox.1
pod2html --noindex docs/busybox.pod > #
  docs/busybox.net/BusyBox.html
mkdir -p docs
[root@localhost busybox-1.00-pre8]#
AUTHORS Rules.mak console utils examples loginutils shell
Changelog TODO coreutils findutils miscutils sysdeps
INSTALL applets debian include modutils syslogd
LICENSE archival debianutils init networking tests
Makefile busybox docs libbb procps testsuite
README busybox.links editors libpwdgrp scripts util-linux
[root@localhost busybox-1.00-pre8]#
```

- BusyBox Binary File copy



```
root's X desktop (localhost,localhost:1)
root@localhost:/dev/board/mds/busybox-1.00-pre8
파일(F) 편집(E) 보기(V) 터미널(T) 가기(G) 도움말(H)
[root@localhost busybox-1.00-pre8]# ls
AUTHORS      Rules.mak      console-tools  examples      loginutils    shell
Changelog    TODO           coreutils      findutils     miscutils     sysdeps
INSTALL      applets        debian          include        modutils      syslogd
LICENSE      archival       debianutils    init           networking    tests
Makefile     busybox        docs            libbb          procps        testsuite
README       busybox.links  editors        libpwdgrp     scripts       util-linux
[root@localhost busybox-1.00-pre8]# cp busybox /tftpboot/ramdisk_mnt/bin/
cp: overwrite '/tftpboot/ramdisk_mnt/bin/busybox'? y
[root@localhost busybox-1.00-pre8]#
```

생성된 busybox를 mount한 디렉토리안의 /bin 디렉토리에 복사한다.

RAMDISK 기능과 구조

■ 확인 / 링크 확인

```

root's X desktop (localhost.localdomain:1)
root@localhost:/tftpboot/ramdisk_mnt/bin
파일(F) 편집(E) 보기(V) 터미널(T) 가기(G) 도움말(H)
[root@localhost ramdisk_mnt]# pwd
/tftpboot/ramdisk_mnt
[root@localhost ramdisk_mnt]# ls
bin dev etc lib linuxrc lost+found mnt proc root sbin tmp usr var
[root@localhost ramdisk_mnt]# cd bin
[root@localhost bin]# ls -al

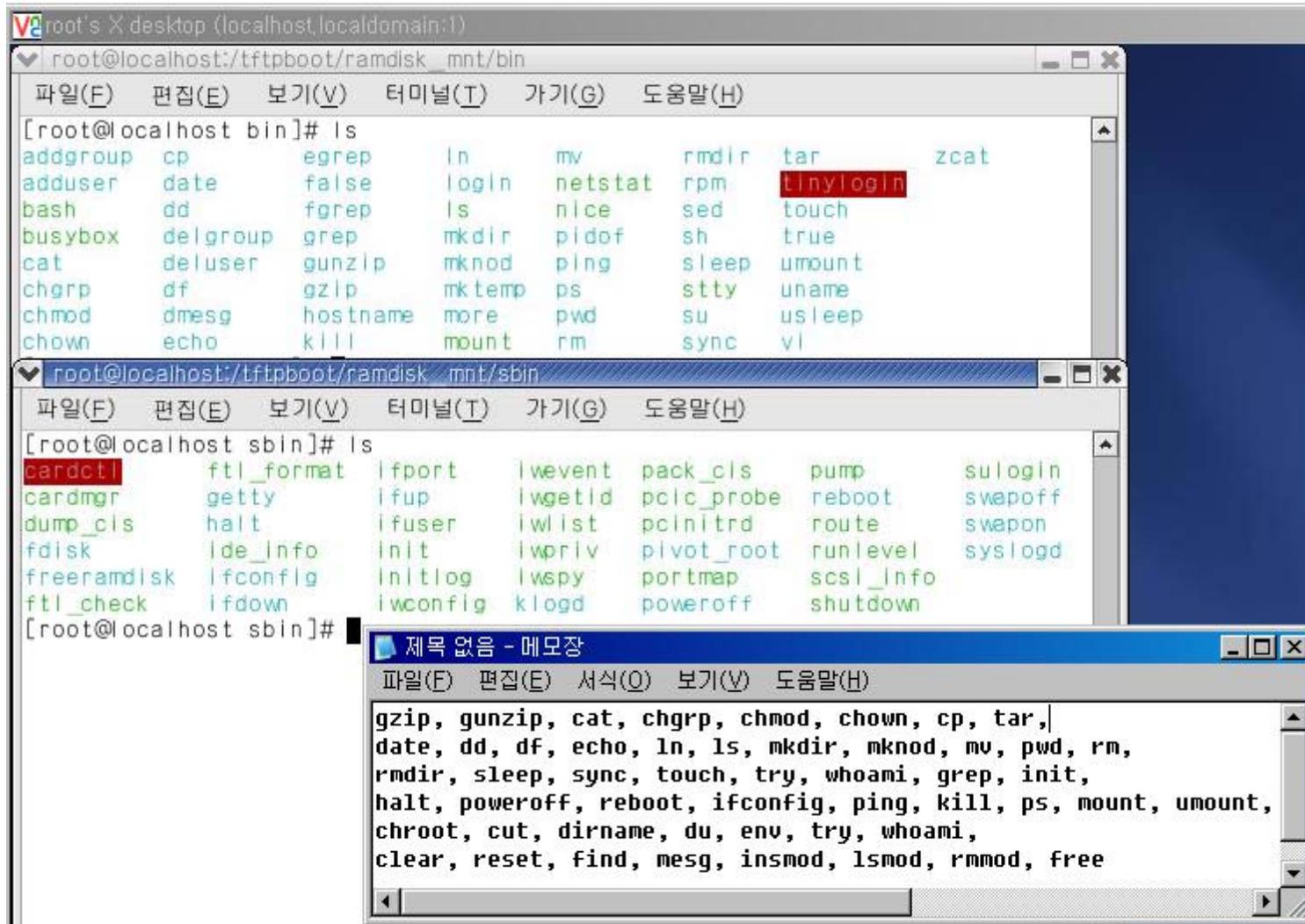
```

```

root's X desktop (localhost.localdomain:1)
root@localhost:/tftpboot/ramdisk_mnt/bin
파일(F) 편집(E) 보기(V) 터미널(T) 가기(G) 도움말(H)
[root@localhost bin]# pwd
/tftpboot/ramdisk_mnt/bin
[root@localhost bin]# ls
addgroup cp egrep ln mv rmdir tar zcat
adduser date false login netstat rpm tinylogin
bash dd fgrep ls nice sed touch
busybox delgroup grep mkdir pidof sh true
cat deluser gunzip mknod ping sleep umount
chgrp df gzip mktemp ps stty uname
chmod dmesg hostname more pwd su usleep
chown echo kill mount rm sync vi
[root@localhost bin]# ls -al busybox
-rwxr-xr-x 1 root root 723848 9월 12 08:29 busybox
[root@localhost bin]# ls -al cp
lrwxrwxrwx 1 root root 7 5월 12 09:42 cp -> busybox
[root@localhost bin]#

```

RAMDISK 기능과 구조

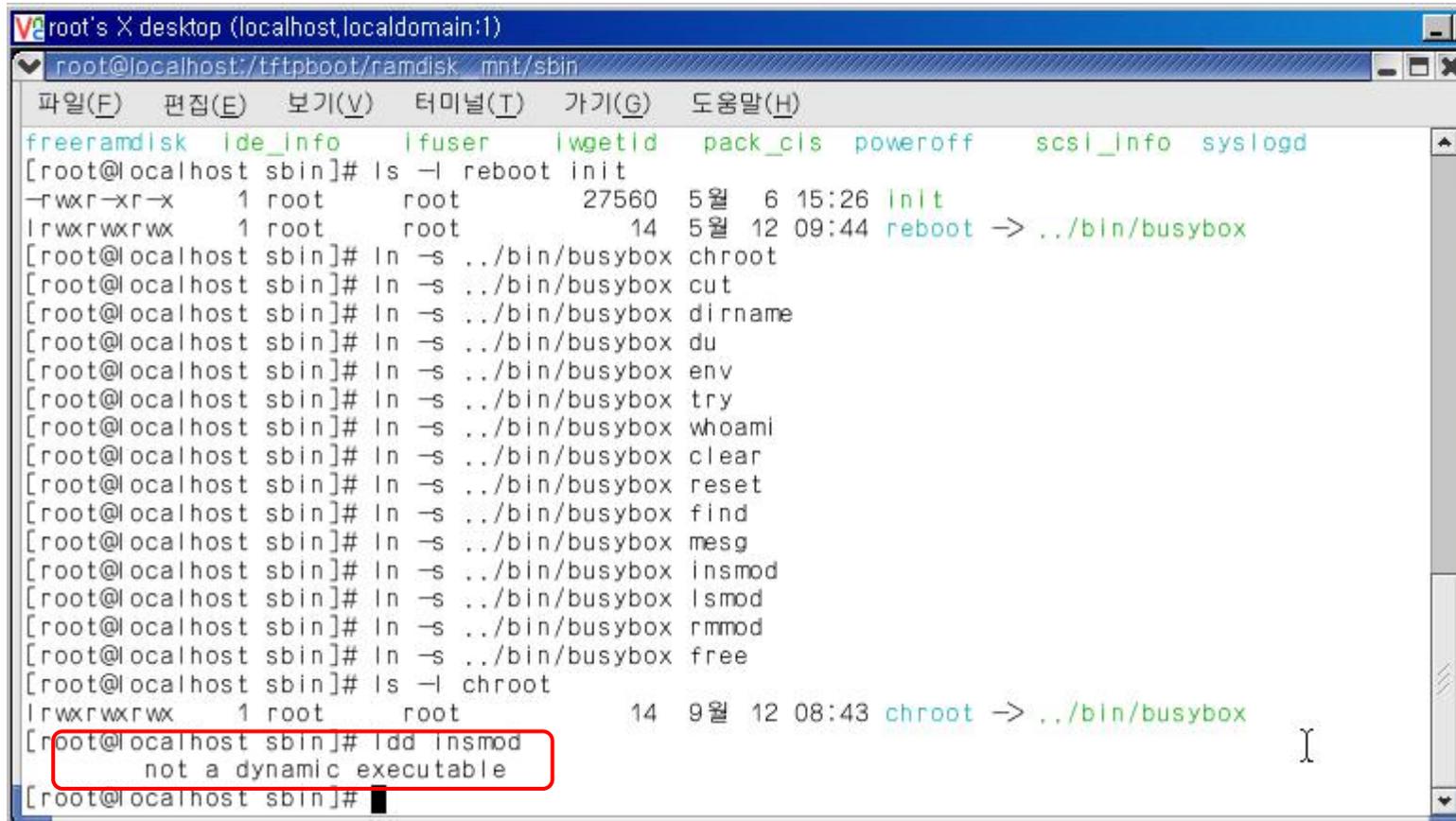


RAMDISK 기능과 구조

```
root's X desktop (localhost,localdomain:1)
root@localhost:/tftpboot/ramdisk_mnt/sbin
파일(F) 편집(E) 보기(V) 터미널(T) 가기(G) 도움말(H)
dump_cis    getty      ifport     lwconfig   lwspy      pivot_root route      swapoff
fdisk       halt      ifup       lwevent    klogd      portmap    runlevel   swapon
freeramdisk ide_info  ifuser     lwgetid    pack_cis   poweroff   scsi_info  syslogd
[root@localhost sbin]# ls -l reboot init
-rwxr-xr-x  1 root    root      27560    5월  6 15:26  init
lrwxrwxrwx  1 root    root      14       5월 12 09:44  reboot -> ../bin/busybox
[root@localhost sbin]# ln -s ../bin/busybox chroot
[root@localhost sbin]# ln -s ../bin/busybox cut
[root@localhost sbin]# ln -s ../bin/busybox dirname
[root@localhost sbin]# ln -s ../bin/busybox du
[root@localhost sbin]# ln -s ../bin/busybox env
[root@localhost sbin]# ln -s ../bin/busybox try
[root@localhost sbin]# ln -s ../bin/busybox whoami
[root@localhost sbin]# ln -s ../bin/busybox clear
[root@localhost sbin]# ln -s ../bin/busybox reset
[root@localhost sbin]# ln -s ../bin/busybox find
[root@localhost sbin]# ln -s ../bin/busybox mesg
[root@localhost sbin]# ln -s ../bin/busybox insmod
[root@localhost sbin]# ln -s ../bin/busybox lsmod
[root@localhost sbin]# ln -s ../bin/busybox rmmod
[root@localhost sbin]# ln -s ../bin/busybox free
[root@localhost sbin]# ls -l chroot
lrwxrwxrwx  1 root    root      14       9월 12 08:43  chroot -> ../bin/busybox
[root@localhost sbin]#
```

실행파일들을 busybox에 심벌릭 링크시켜주면 된다.

RAMDISK 기능과 구조



```
root's X desktop (localhost,localhost:1)
root@localhost:/tftpboot/ramdisk_mnt/sbin
파일(F) 편집(E) 보기(V) 터미널(T) 가기(G) 도움말(H)
freeramdisk ide_info lfuser lwgetid pack_cis poweroff scsi_info syslogd
[root@localhost sbin]# ls -l reboot init
-rwxr-xr-x  1 root    root      27560  5월  6 15:26  init
lrwxrwxrwx  1 root    root         14  5월 12 09:44  reboot -> ../bin/busybox
[root@localhost sbin]# ln -s ../bin/busybox chroot
[root@localhost sbin]# ln -s ../bin/busybox cut
[root@localhost sbin]# ln -s ../bin/busybox dirname
[root@localhost sbin]# ln -s ../bin/busybox du
[root@localhost sbin]# ln -s ../bin/busybox env
[root@localhost sbin]# ln -s ../bin/busybox try
[root@localhost sbin]# ln -s ../bin/busybox whoami
[root@localhost sbin]# ln -s ../bin/busybox clear
[root@localhost sbin]# ln -s ../bin/busybox reset
[root@localhost sbin]# ln -s ../bin/busybox find
[root@localhost sbin]# ln -s ../bin/busybox msg
[root@localhost sbin]# ln -s ../bin/busybox insmod
[root@localhost sbin]# ln -s ../bin/busybox lsmmod
[root@localhost sbin]# ln -s ../bin/busybox rmmmod
[root@localhost sbin]# ln -s ../bin/busybox free
[root@localhost sbin]# ls -l chroot
lrwxrwxrwx  1 root    root         14  9월 12 08:43  chroot -> ../bin/busybox
[root@localhost sbin]# ldd insmod
not a dynamic executable
[root@localhost sbin]#
```

파일 시스템 이미지 압축

umount 후 압축

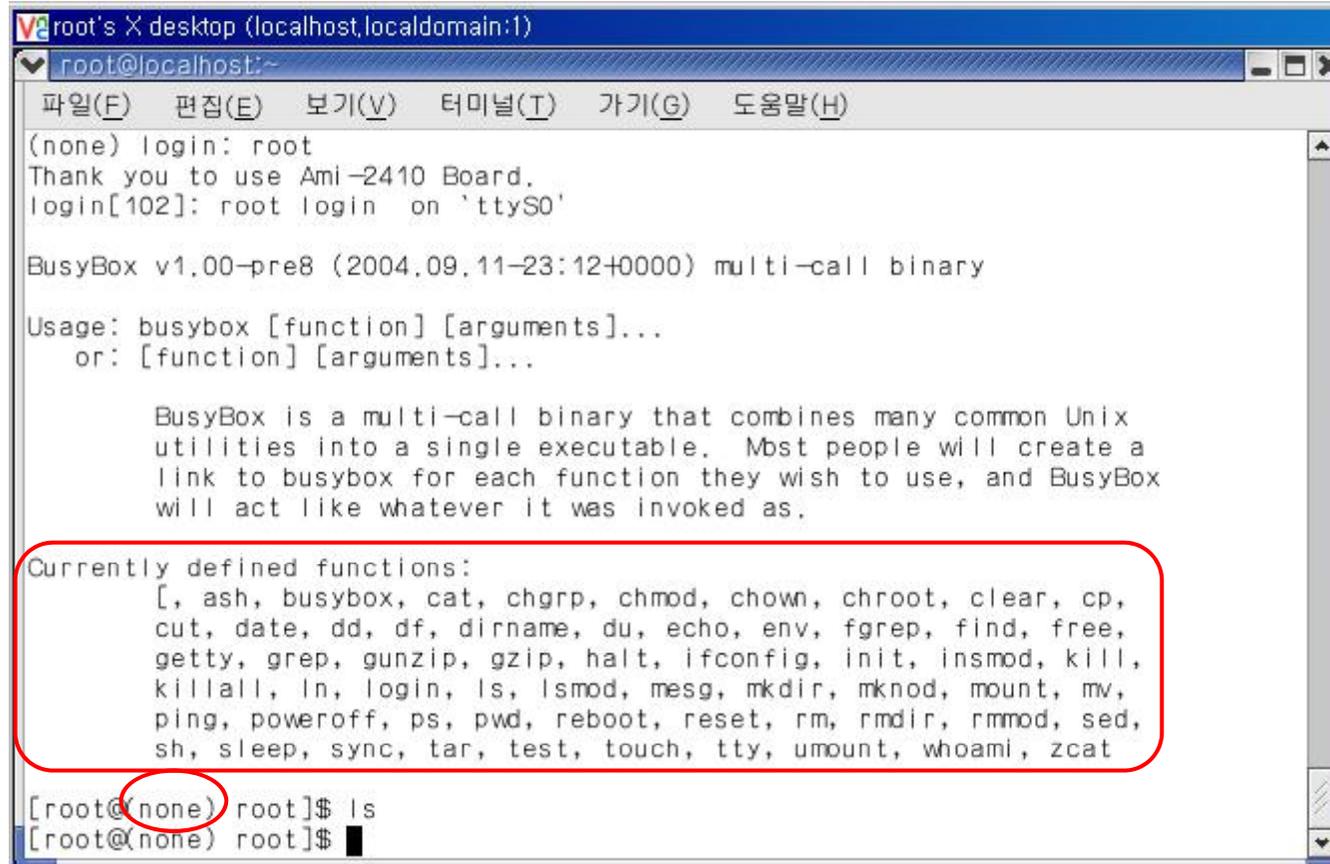
```
★ root@server:/temp/bit_computer/rootfs
[root@server rootfs]# umount ./mnt/
[root@server rootfs]# gzip -c rootfs_img > rootfs_img.gz
[root@server rootfs]#
[root@server rootfs]#
[root@server rootfs]#
[root@server rootfs]# ls -al rootfs_img.gz
-rw-r--r--  1 root  root  1289407 11월  3 21:32 rootfs_img.gz
[root@server rootfs]#
```

최종적으로 완성한 압축된 root filesystem 이미지

[영어] [완성] [두벌식]

RAMDISK 기능과 구조

- Booting using New Kernel & New RAMDisk



```
root's X desktop (localhost,localhost:1)
root@localhost:~
파일(F) 편집(E) 보기(V) 터미널(T) 가기(G) 도움말(H)
(none) login: root
Thank you to use Ami-2410 Board.
login[102]: root login on 'ttyS0'

BusyBox v1.00-pre8 (2004.09.11-23:12+0000) multi-call binary

Usage: busybox [function] [arguments]...
or: [function] [arguments]...

BusyBox is a multi-call binary that combines many common Unix
utilities into a single executable. Most people will create a
link to busybox for each function they wish to use, and BusyBox
will act like whatever it was invoked as.

Currently defined functions:
[, ash, busybox, cat, chgrp, chmod, chown, chroot, clear, cp,
cut, date, dd, df, dirname, du, echo, env, fgrep, find, free,
getty, grep, gunzip, gzip, halt, ifconfig, init, insmod, kill,
killall, ln, login, ls, lsmod, mesg, mkdir, mknod, mount, mv,
ping, poweroff, ps, pwd, reboot, reset, rm, rmdir, rmdir, sed,
sh, sleep, sync, tar, test, touch, tty, umount, whoami, zcat

[root@none] root]$ ls
[root@none] root]$
```

kernel 설정 및 컴파일

- kernel 설정
 - ✓ Ramdisk를 사용하기 위한 kernel 설정

