Root Filesystem

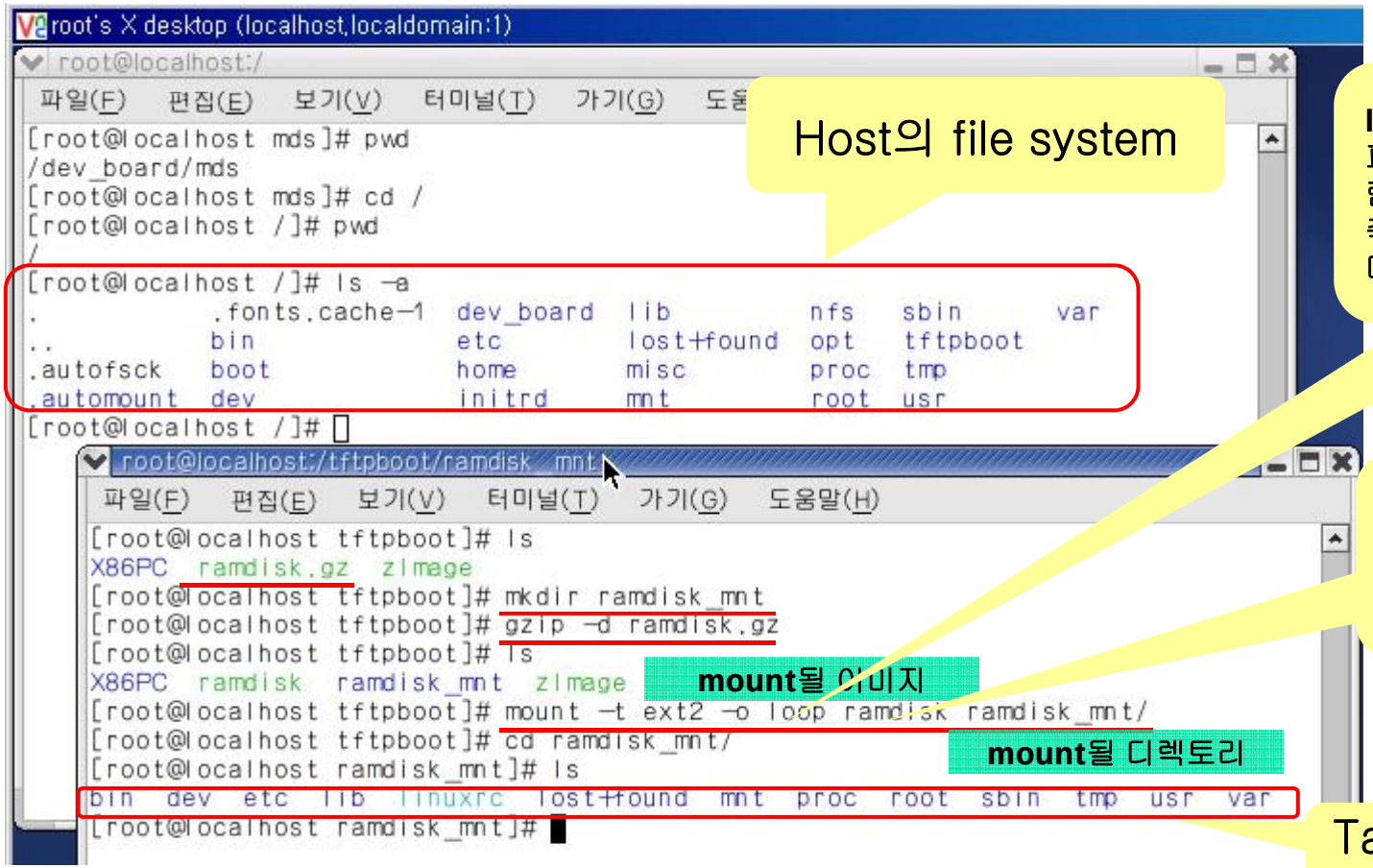
단국대학교
컴퓨터학과
2009
백승재
ibanez1383@dankook.ac.kr
http://embedded.dankook.ac.kr/~ibanez1383

강의 목표

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

RAMDISK 기능과 구조

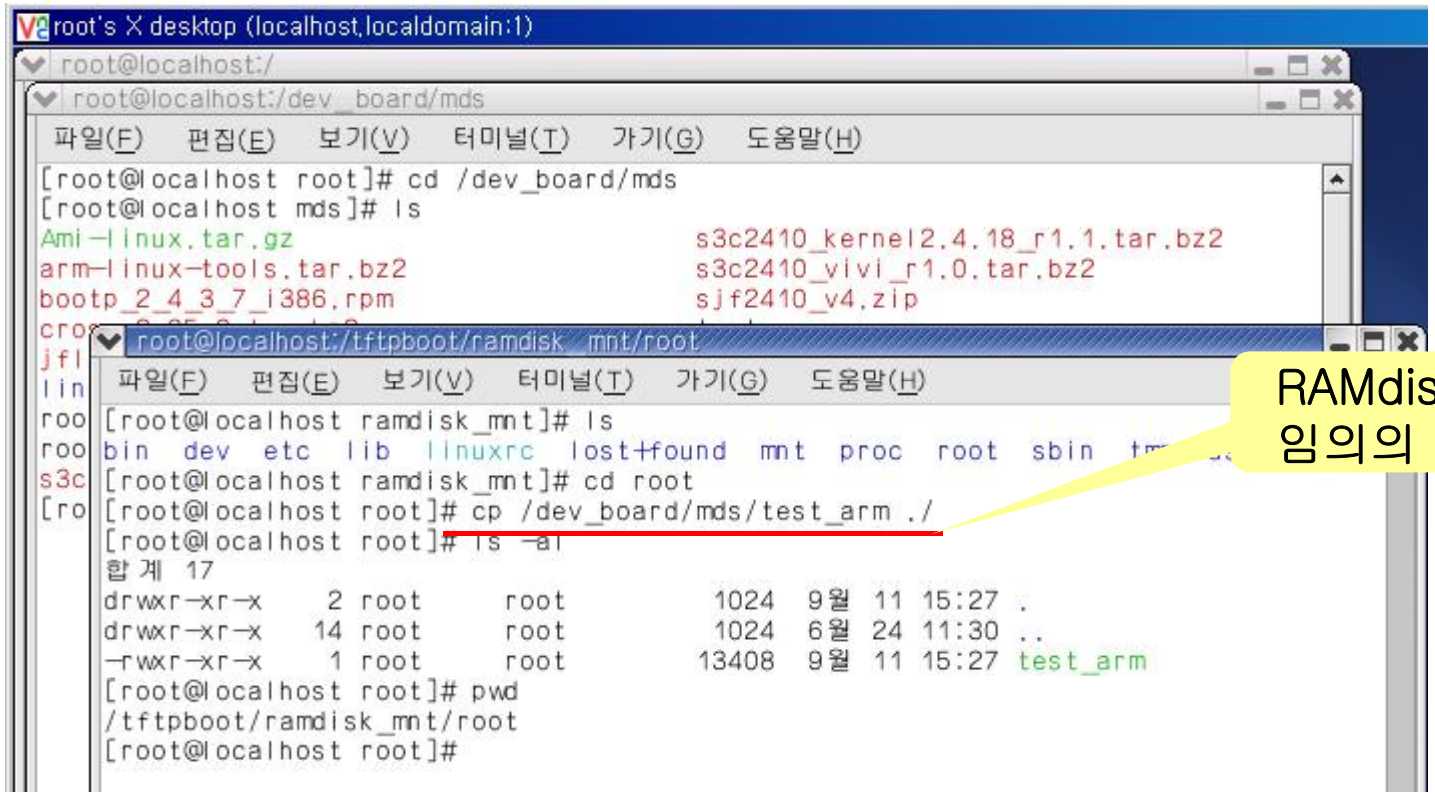
- RAMdisk?



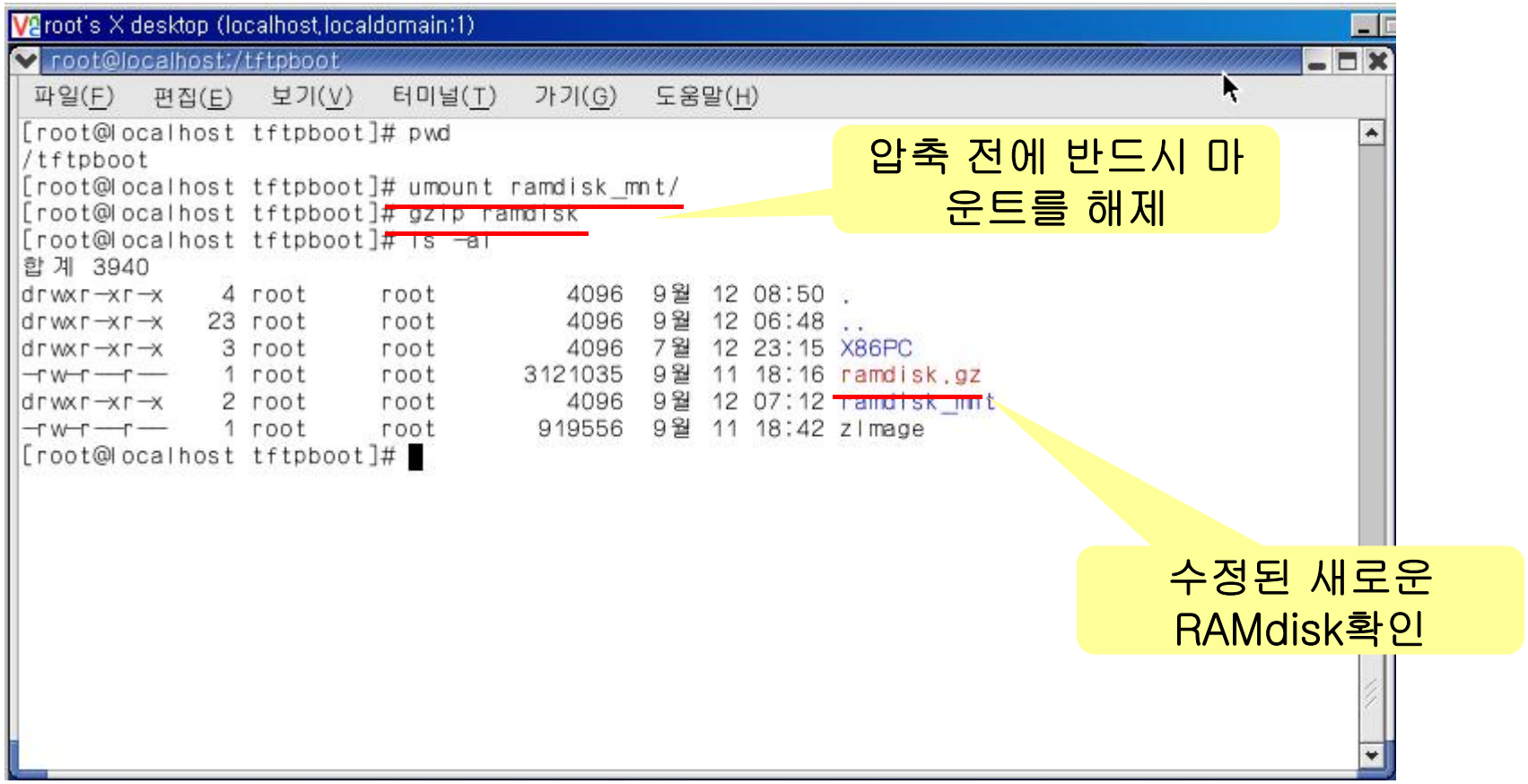
- ✓ **HDD**가 없는 **Embedded System**에서 **RAM** 상에 디스크 처럼 쓸 수 있도록 구축한 공간

RAMDISK 기능과 구조

- RAMdisk 수정

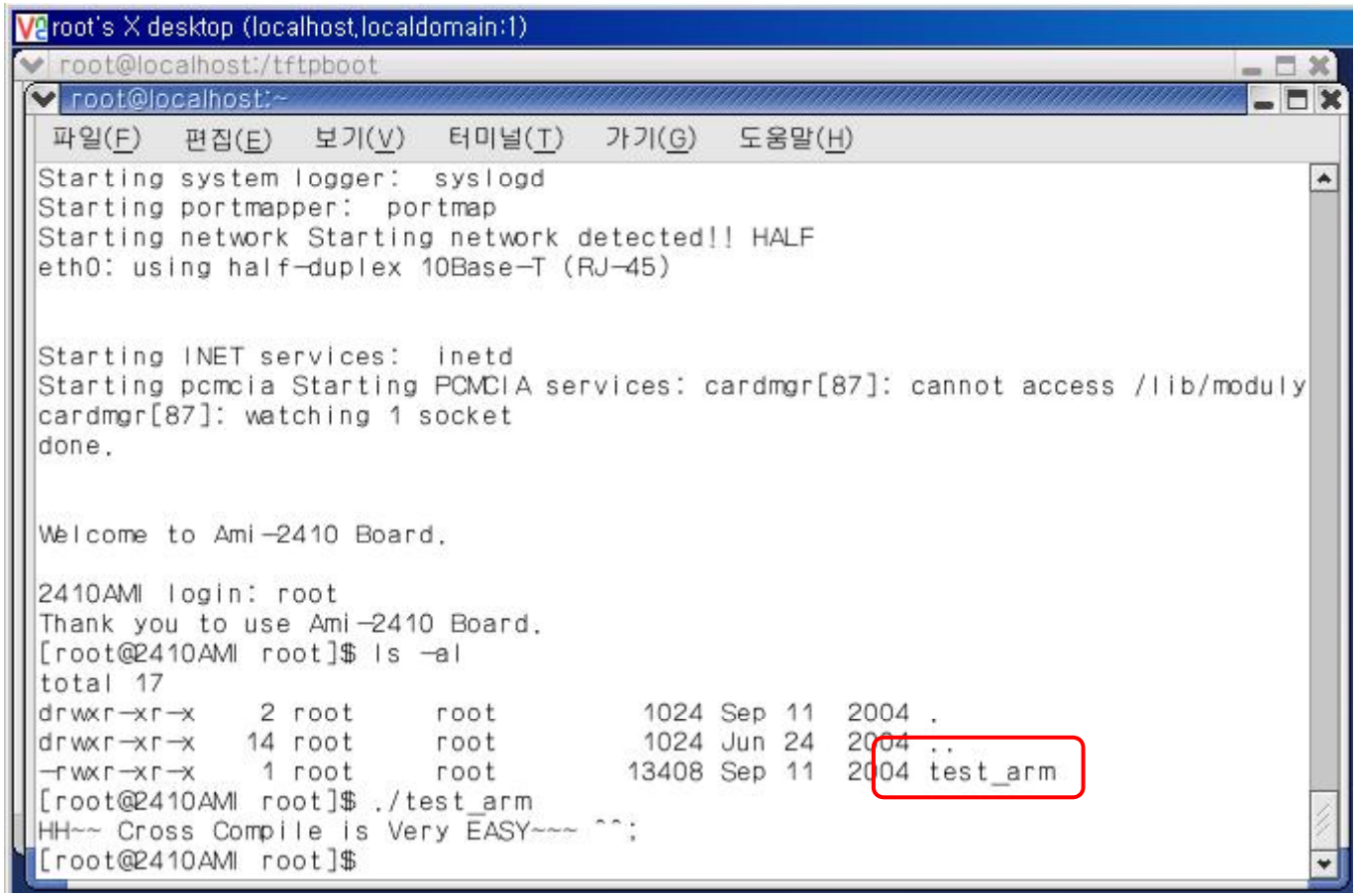


RAMDISK 기능과 구조



RAMDISK 기능과 구조

- 수정된 RAMdisk로 부팅 확인



RAMDISK 기능과 구조

▪내부에 **file system** 구조를 만들어서 **loop back device**로 **mount**하여 일반 **file system**과 같이 사용하는 파일을 말한다.

▪ **Image file** 내부에 **root filesystem** 을 만든다.

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

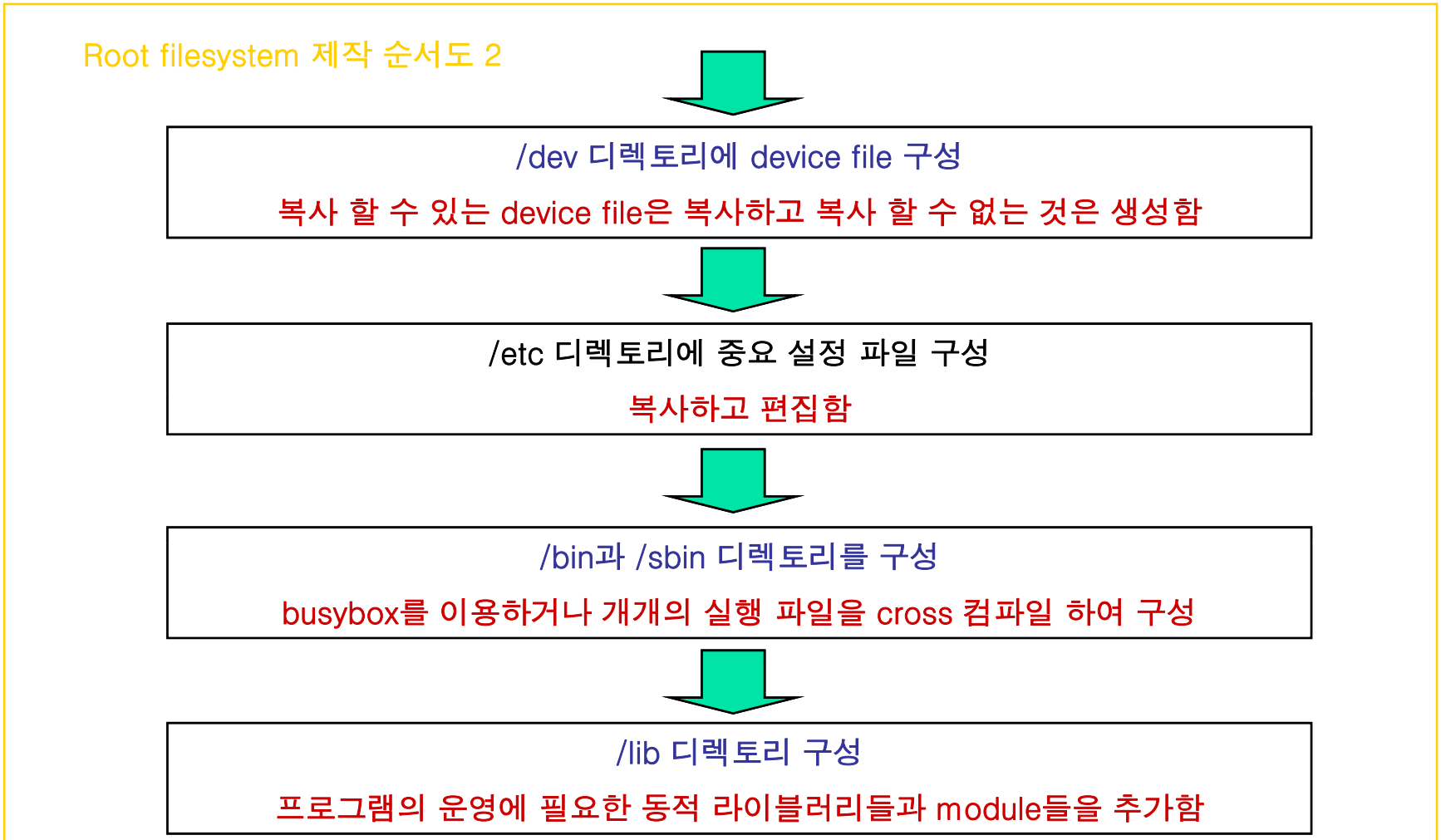
mkdir mnt && mount –t ext2 ramdisk ram_point –o loop

기본 디렉토리 생성

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

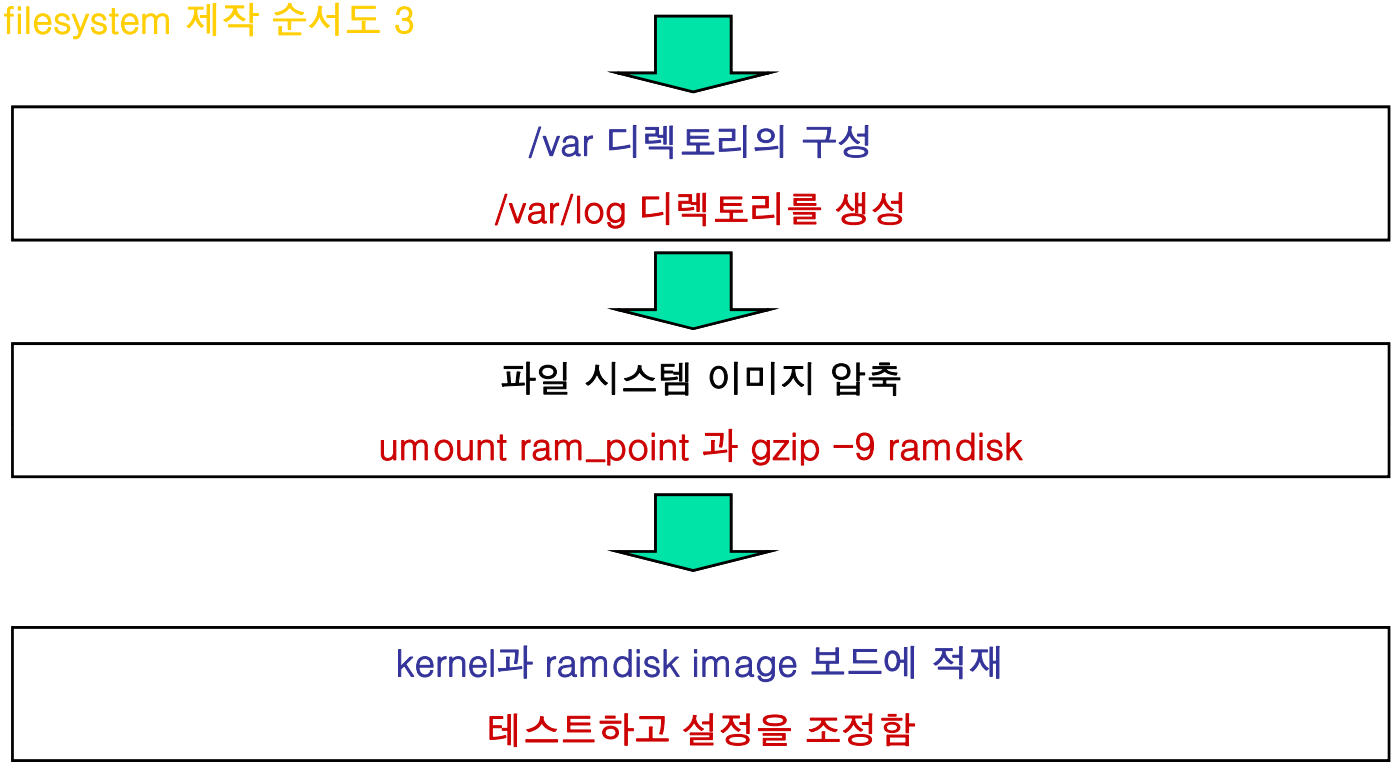
RAMDISK 기능과 구조

Root filesystem 제작 순서도 2



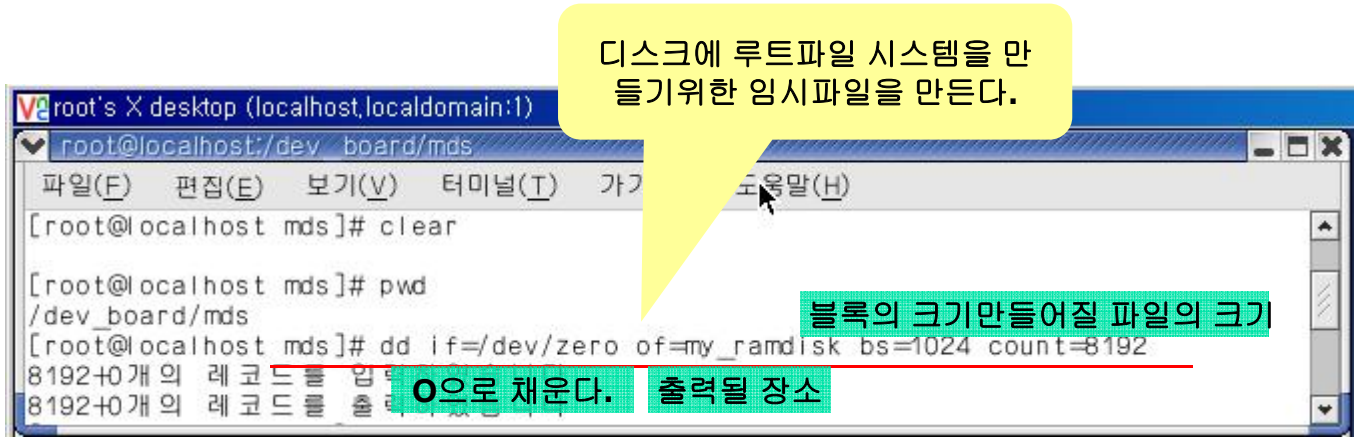
RAMDISK 기능과 구조

Root filesystem 제작 순서도 3

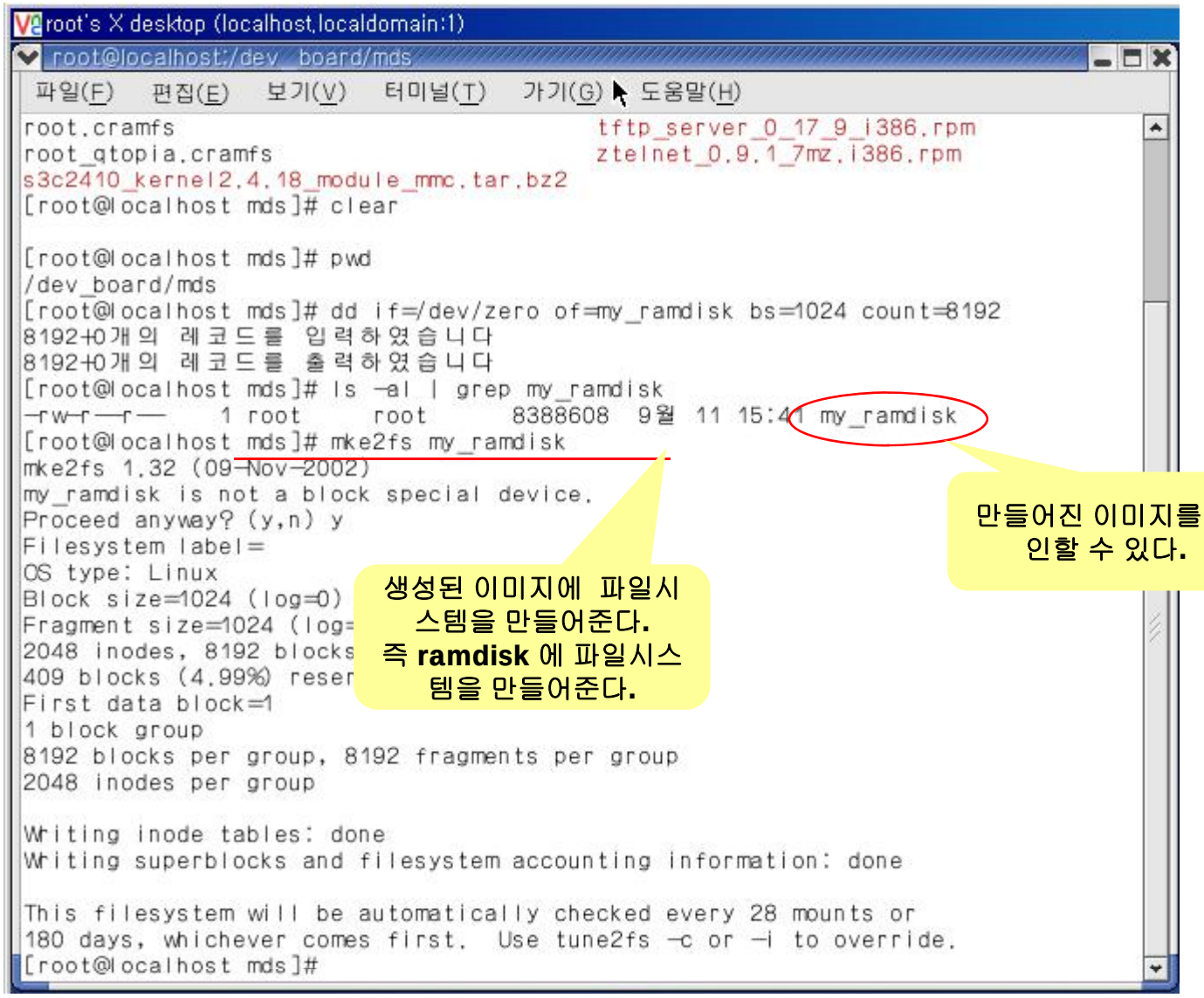


RAMDISK 기능과 구조

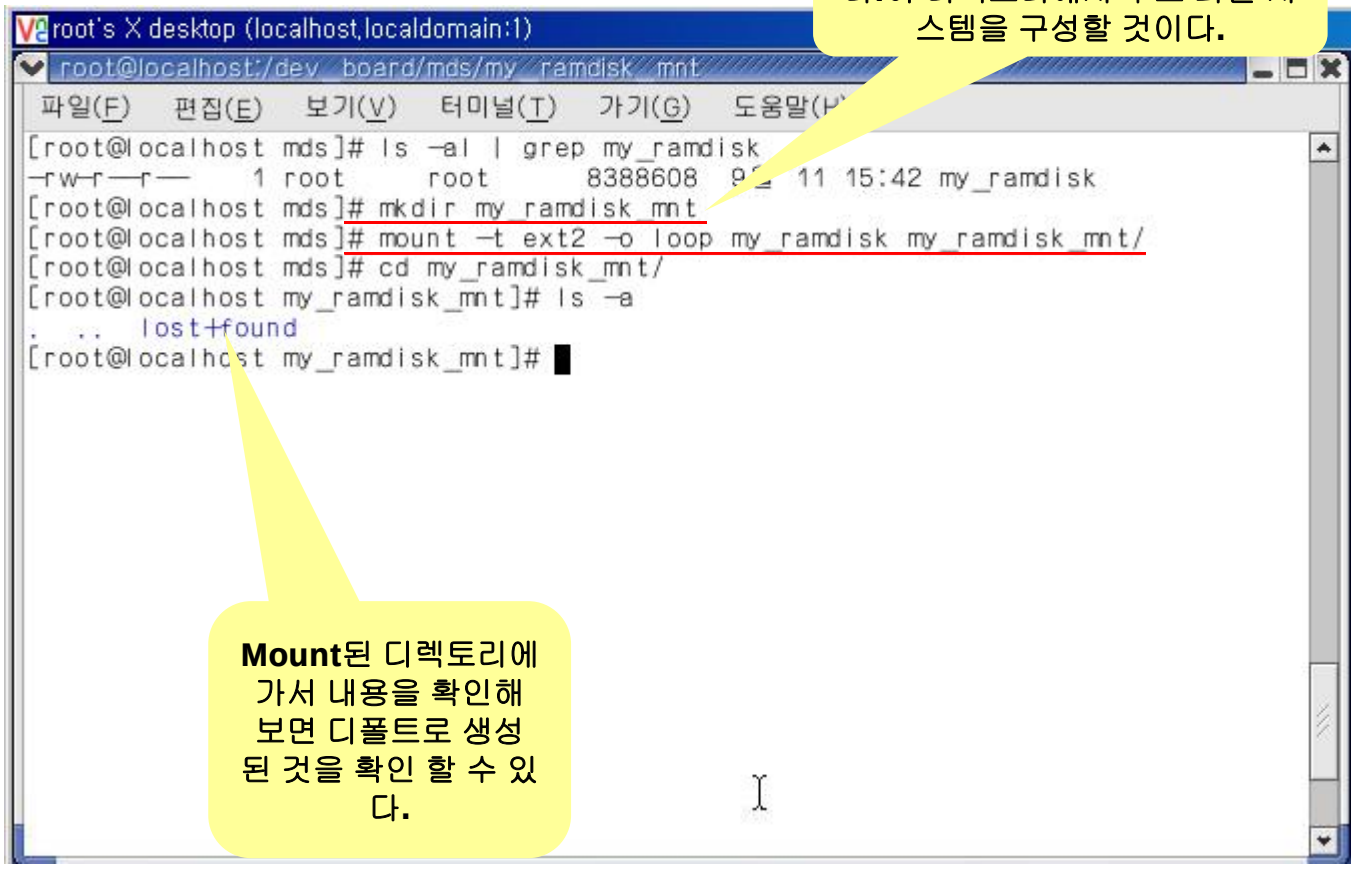
- 램 디스크 제작



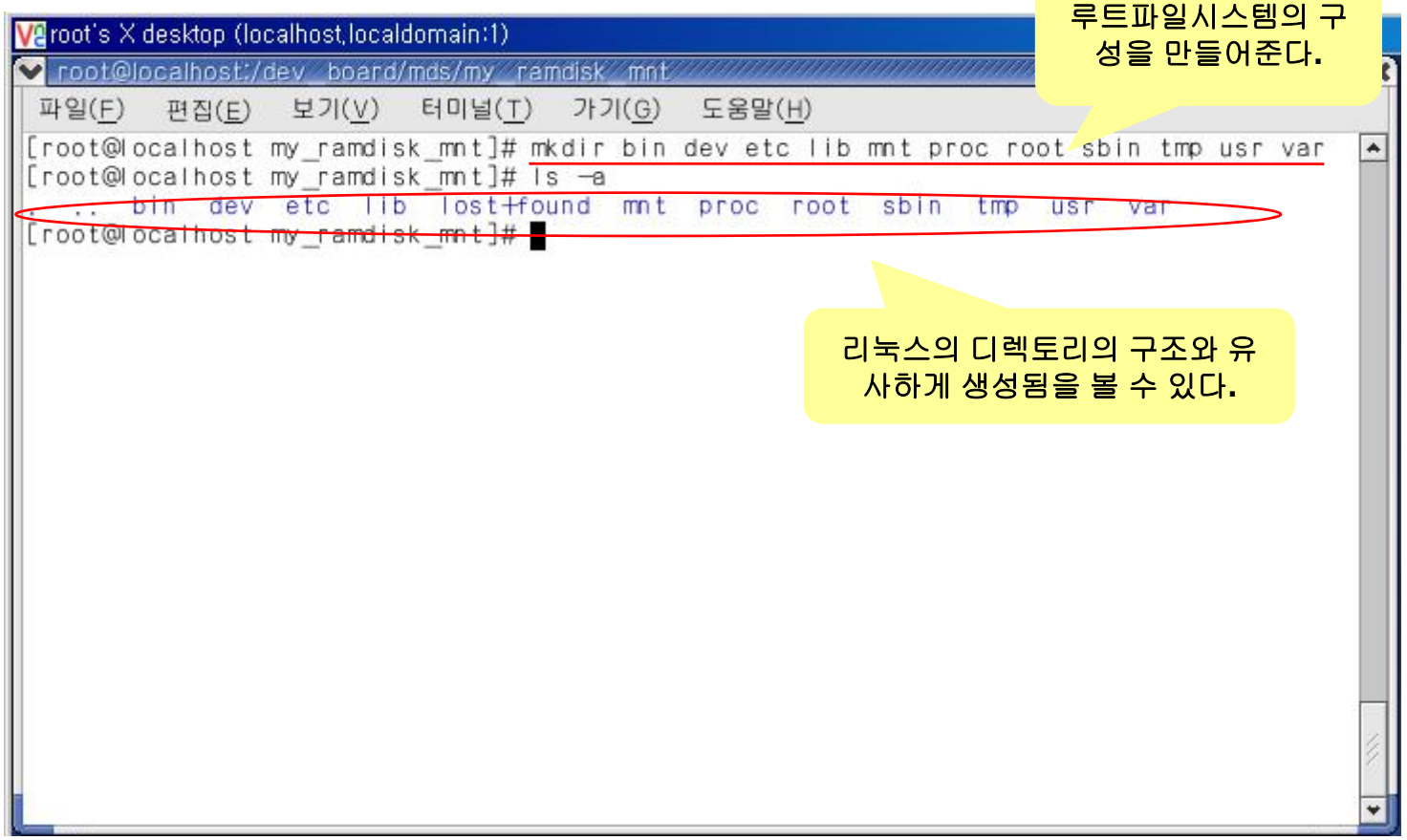
RAMDISK 기능과 구조



RAMDISK 기능과 구조



RAMDISK 기능과 구조



/dev 디렉토리의 구성 (1/5)

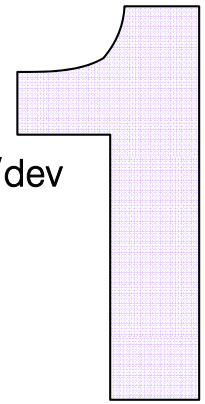
/dev 디렉토리에 사용할 device의 device file을 생성해준다. 기본적인 device file은 host 컴퓨터의 /dev 디렉토리에서 복사하여 사용하면 되며 host 컴퓨터에 없는 device file이라면 mknod 명령어로 만들어 주면 된다. Rebis 보드의 kernel과 device를 기준으로 보면 필요한 device file들은 다음과 같다. (다음 예의 파일명은 디렉토리도 포함됨)

```
console  fb3   flh1     mixer        null   ram3    tty1  tty7   ttyS0
dsp      fb4   flh2     mtdblock0    ptmx   random  tty2  tty8   ttyS1
fb       fb5   flh3     mtdblock1    pts    root    tty3  ttyP0  ttySA0
fb0      fb6   initctl  mtdblock2    ram0   ts      tty4  ttyP1  ttySA1
fb1      fb7   kmem     mtdblock3    ram1   tty     tty5  ttyP2  ttySA2
fb2      flh0  mem      mtdblock4    ram2   tty0    tty6  ttyP3  urandom
zero
```

/dev 디렉토리의 구성 (2/5)

/dev 디렉토리 구성 명령

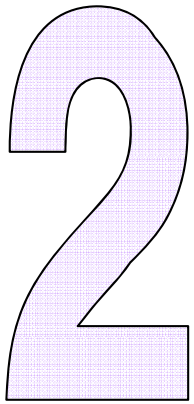
```
cp -dpR /dev/audio ./mnt/dev
cp -dpR /dev/dsp ./mnt/dev
cp -dpR /dev/fb ./mnt/dev
cp -dpR /dev/fb[0-7] ./mnt/dev
cp -dpR /dev/hda ./mnt/dev
cp -dpR /dev/hda[0-8] ./mnt/dev
cp -dpR /dev/initctl ./mnt/dev
cp -dpR /dev/input ./mnt/dev
cp -dpR /dev/ircomm[0-1] ./mnt/dev
cp -dpR /dev/irnet ./mnt/dev
cp -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)

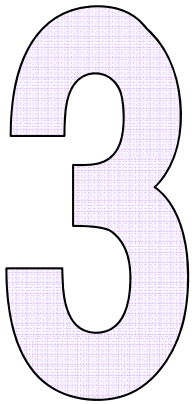
```
cp -dpR /dev/tty ./mnt/dev
cp -dpR /dev/tty[0-8] ./mnt/dev
cp -dpR /dev/ttyP[0-3] ./mnt/dev
cp -dpR /dev/ttyS[0-1] ./mnt/dev
cp -dpR /dev/ttySA[0-2] ./mnt/dev
cp -dpR /dev/urandom ./mnt/dev
cp -dpR /dev/video ./mnt/dev
cp -dpR /dev/zero ./mnt/dev
mknod ./mnt/dev/flh0 b 60 0
mknod ./mnt/dev/flh1 b 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
```



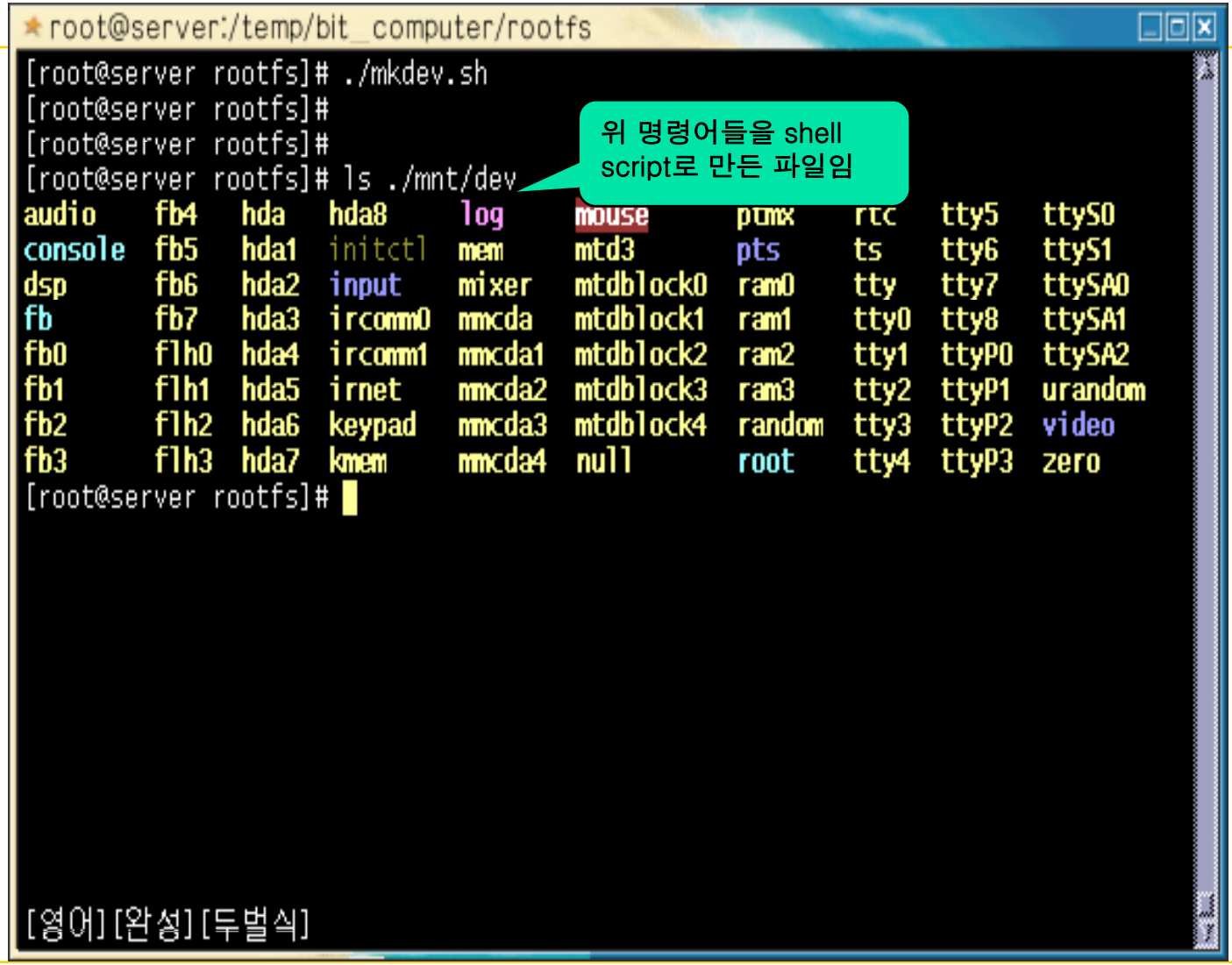
/dev 디렉토리의 구성 (4/5)

```
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



/dev 디렉토리의 구성 (5/5)



/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 loding에 참조하는 파일
기타	: 기타 network 설정에 필요한 파일들이나 shell 환경 설정 파일들

/etc 디렉토리의 구성 (2/5)

/etc 디렉토리 구성 명령

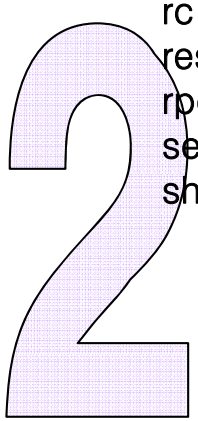
```
TARGETDIR=./mnt/etc

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

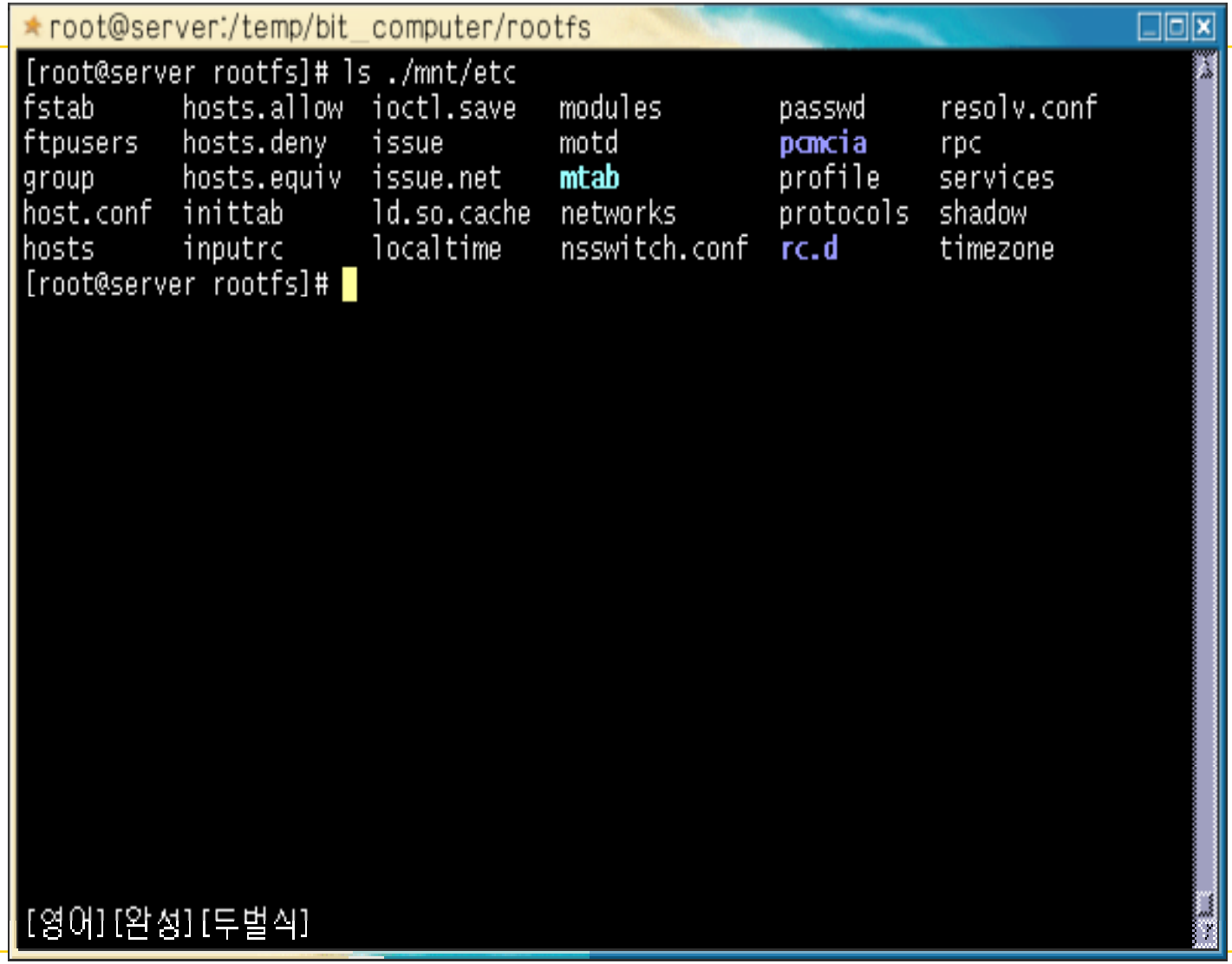
/etc 디렉토리의 구성 (3/5)

etcfiles.txt file의 내용

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



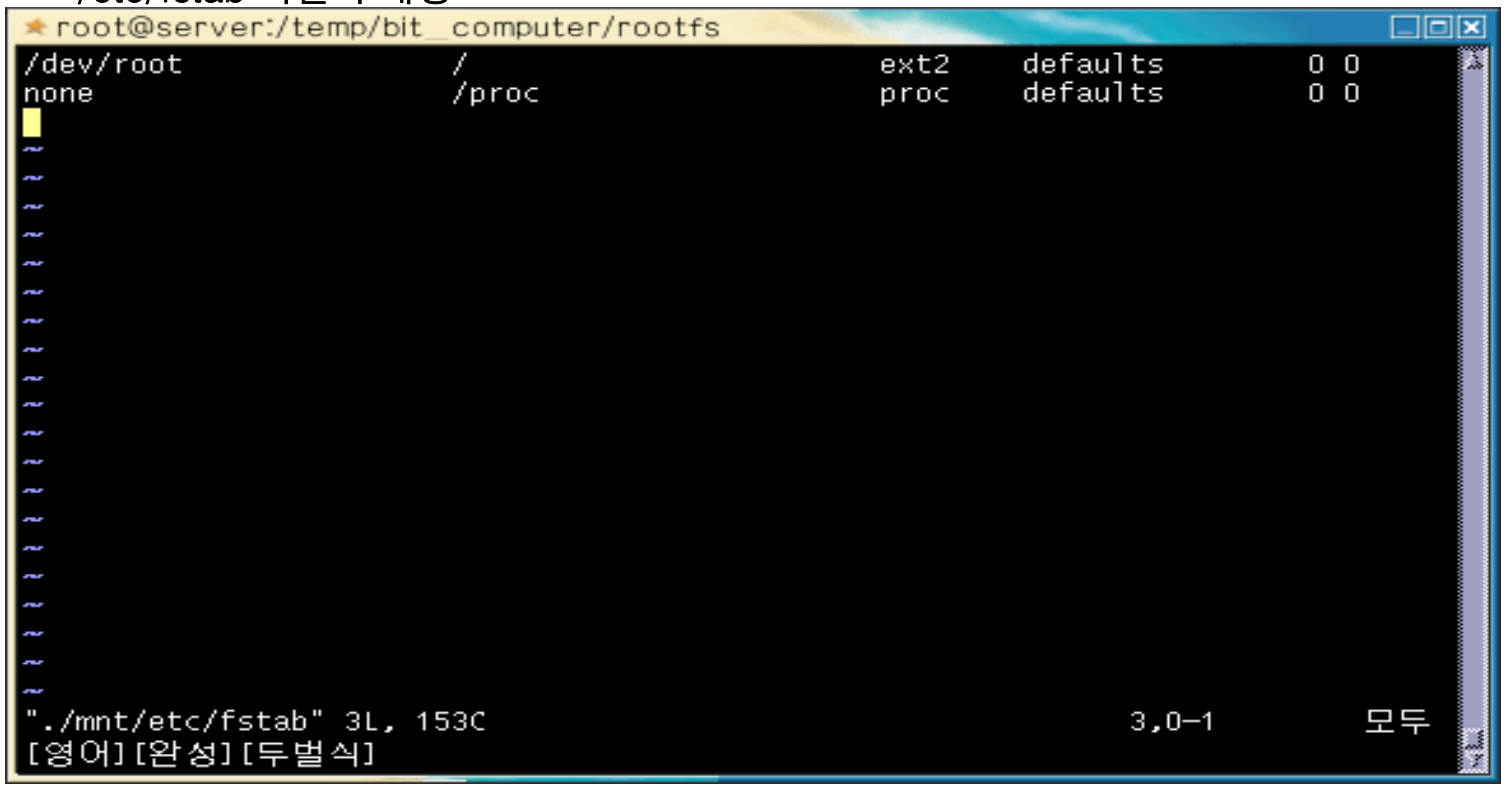
/etc 디렉토리의 구성 (4/5)



/etc 디렉토리의 구성 (5/5)

복사한 파일들을 수정해준다. 기본적으로 fstab파일만 수정해주면 부팅이 가능하나 원칙대로라면 script 파일들과 network관련 설정 파일들을 embedded system의 상황에 맞게 수정해주어야 한다.

/etc/fstab 파일의 내용



/lib 디렉토리의 구성 (1/3)

- 동적 라이블러리들과 kernel module
  - 동적 라이블러리는 arm-linux-gcc컴파일러가 설치된 디렉토리에서 lib 디렉토리의 라이블러리 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 라이블러리 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 라이블러리 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
```

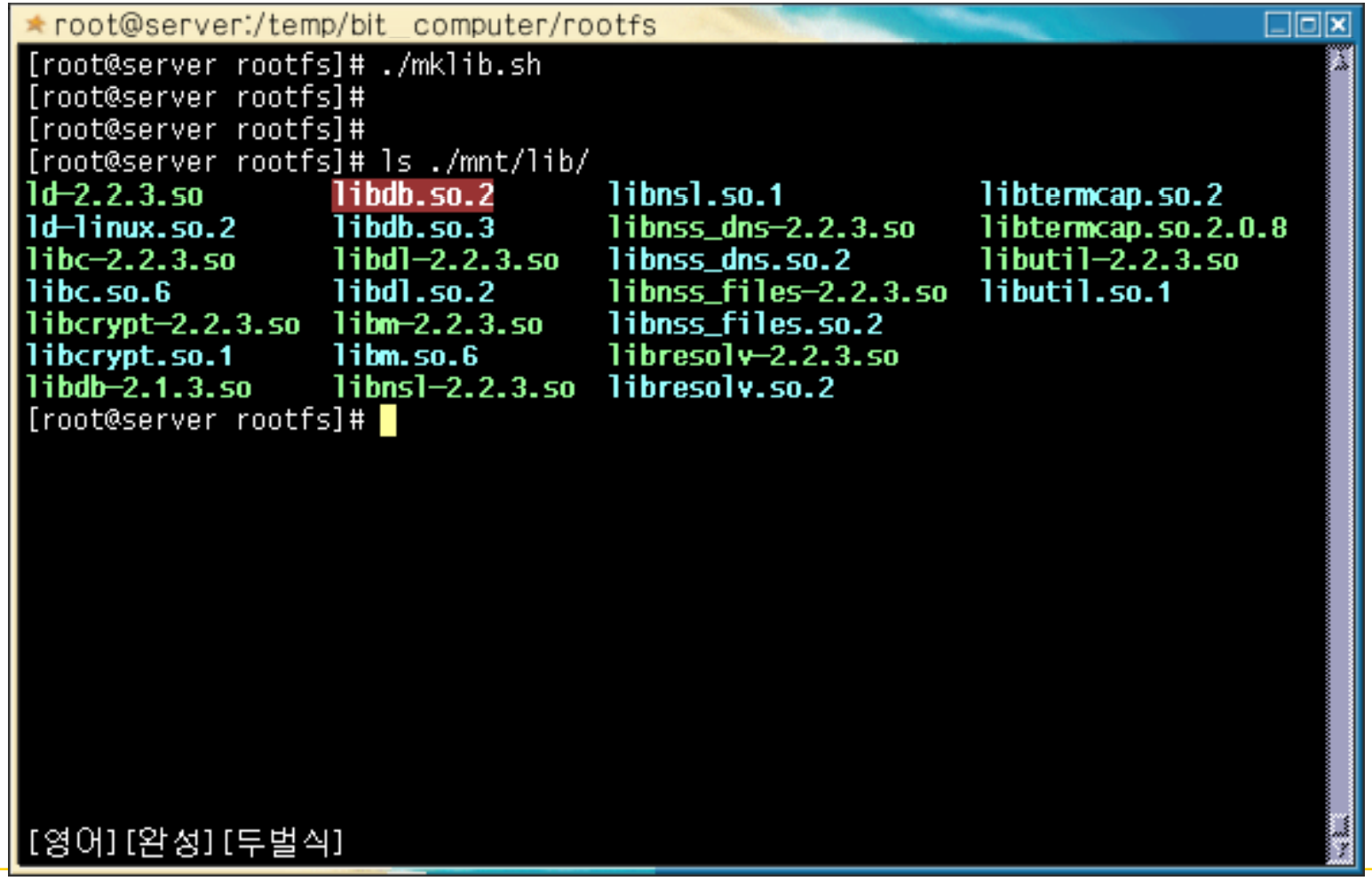
libfiles.txt 파일은 위에서 설명한 필요한 동적 라이블러리 리스트이다.

strip 명령은 binary내에 symbol string을 제거 함으로써 사이즈를 축소함

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

/lib 디렉토리의 구성 (3/3)

☞ 동적 라이브러리 생성



/var 디렉토리의 구성 (1/2)

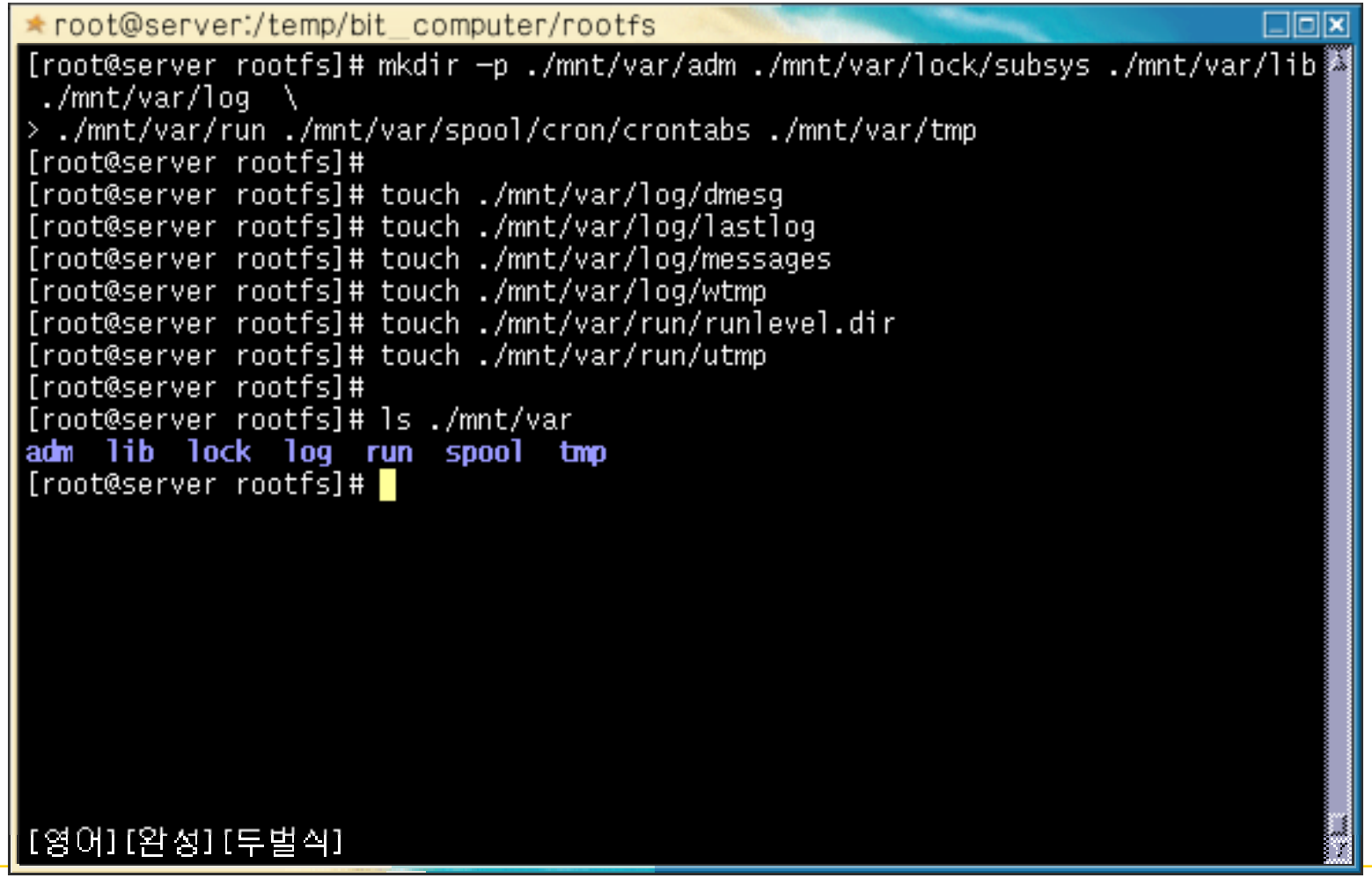
- sub 디렉토리와 log file 생성
  - 어떤 login 프로그램이나 init 프로그램의 경우 /var 디렉토리에 log file이 없을때 정상적으로 실행이 안될 수가 있다. 그래서 /var 디렉토리에 필요한 log file과 디렉토리 구조를 만들어 주어야 한다.

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

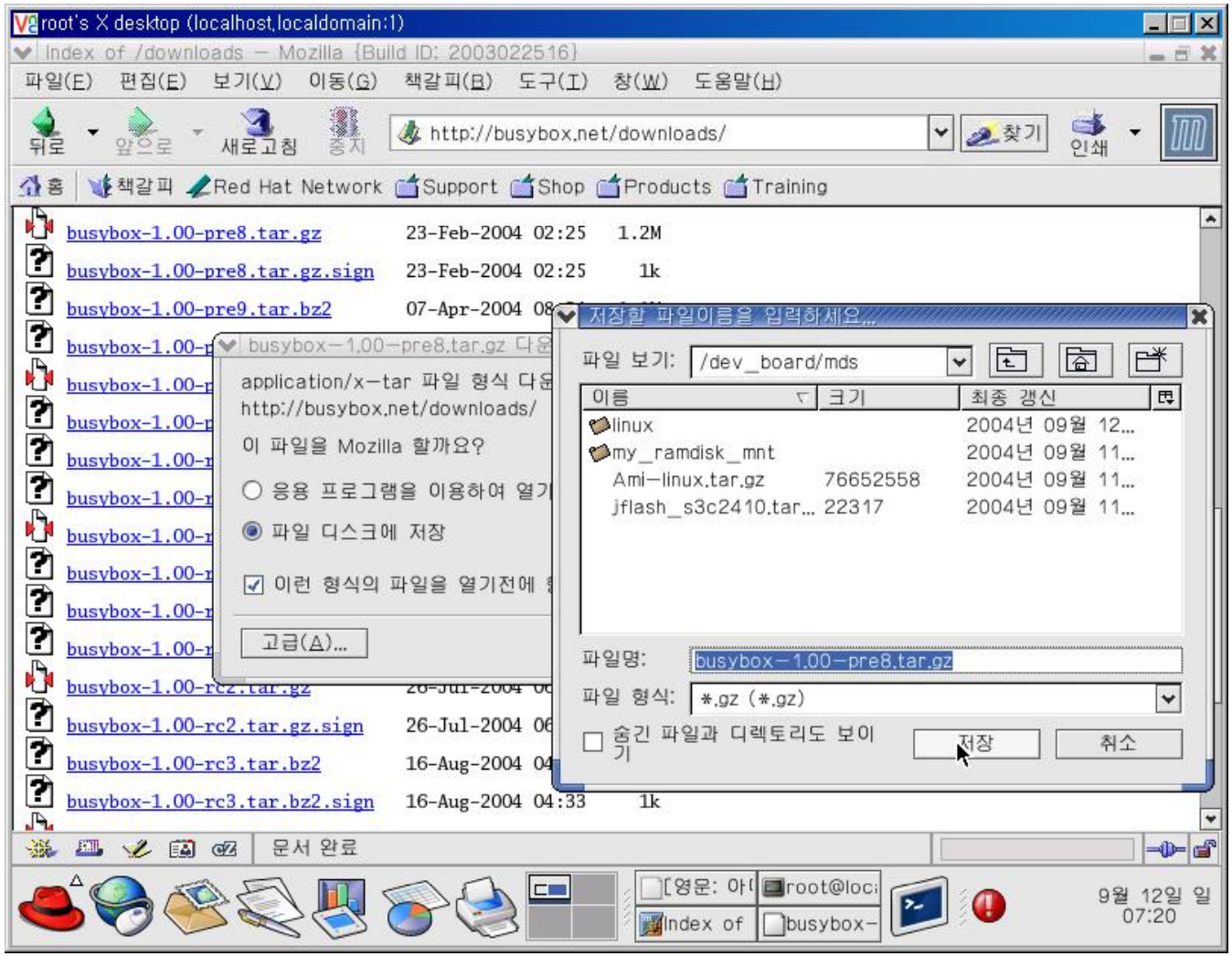


RAMDISK 기능과 구조

- 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

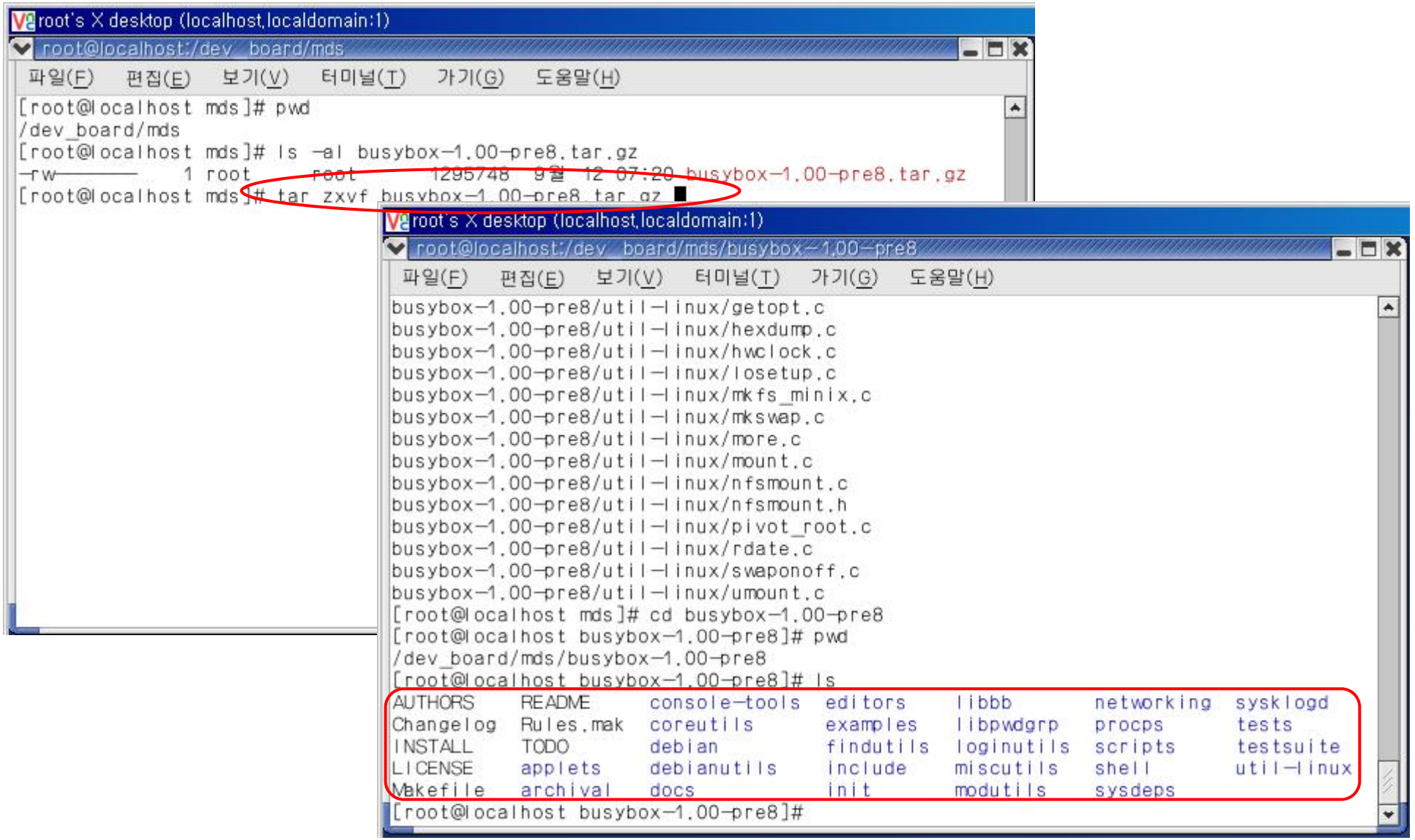
RAMDISK 기능과 구조

- Busybox 다운 받기



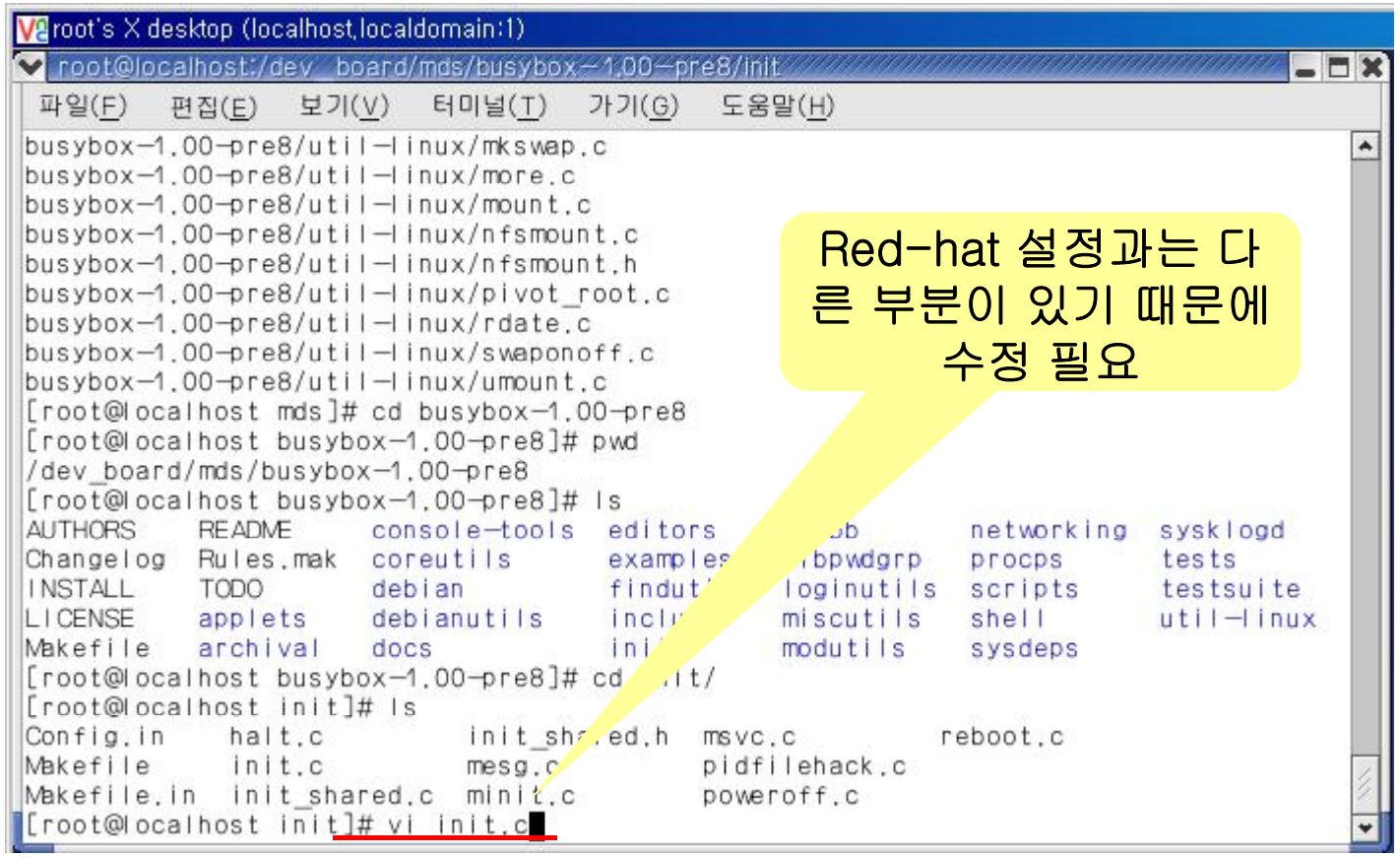
RAMDISK 기능과 구조

- 압축해제 / 확인



RAMDISK 기능과 구조

- 파일 수정



RAMDISK 기능과 구조

root's X desktop (localhost.localdomain:1)

root@localhost:/dev board/mds/busybox-1.00-pre8/init

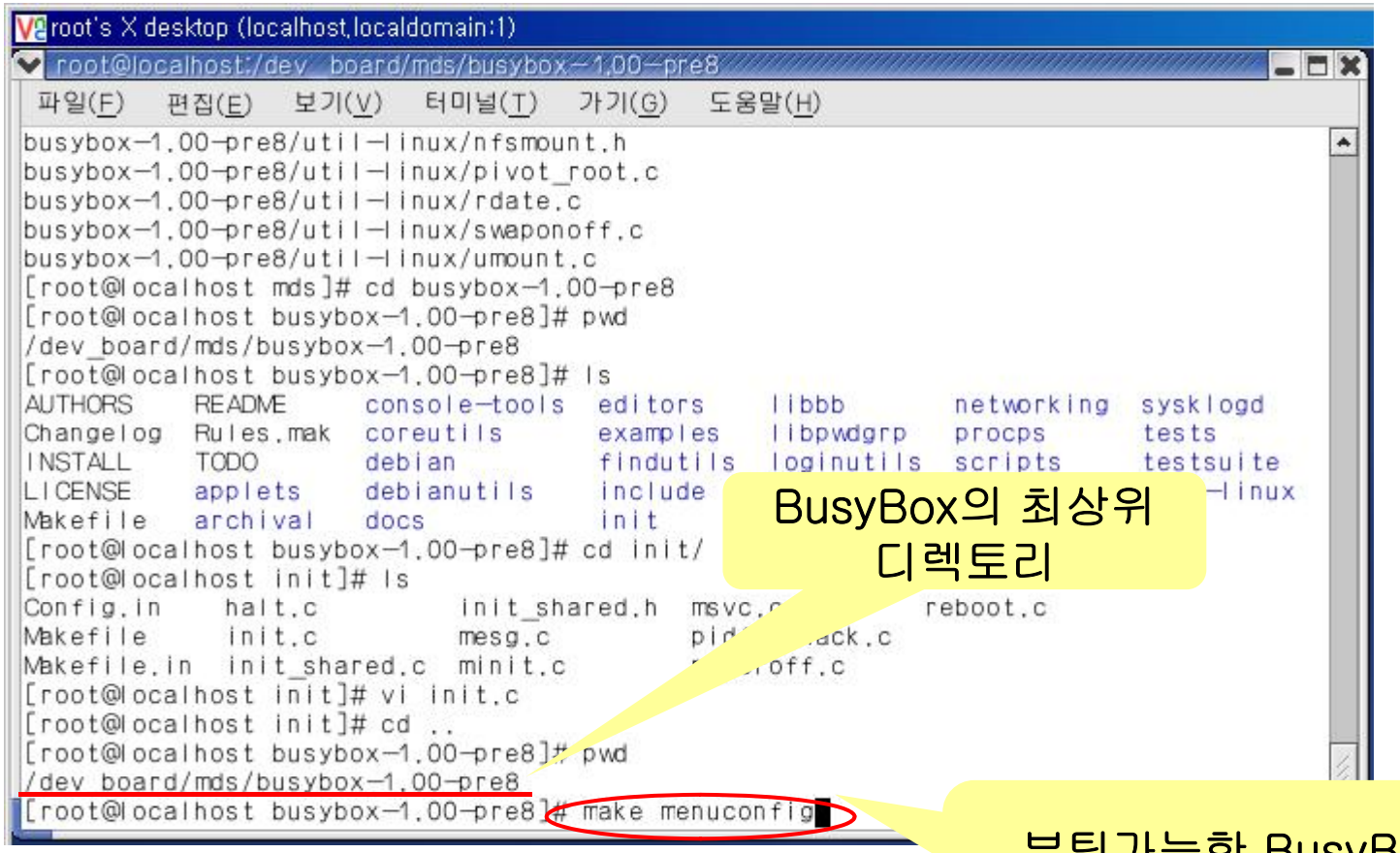
파일(F) 편집(E) 보기(V) 터미널(T) 가기(G) 도움말(H)

```
#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 */
#ifndef 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
-- 비주얼 --
```

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

RAMDISK 기능과 구조

- config



RAMDISK 기능과 구조

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 라이블러리 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

RAMDISK 기능과 구조

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, …)

RAMDISK 기능과 구조

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, …)

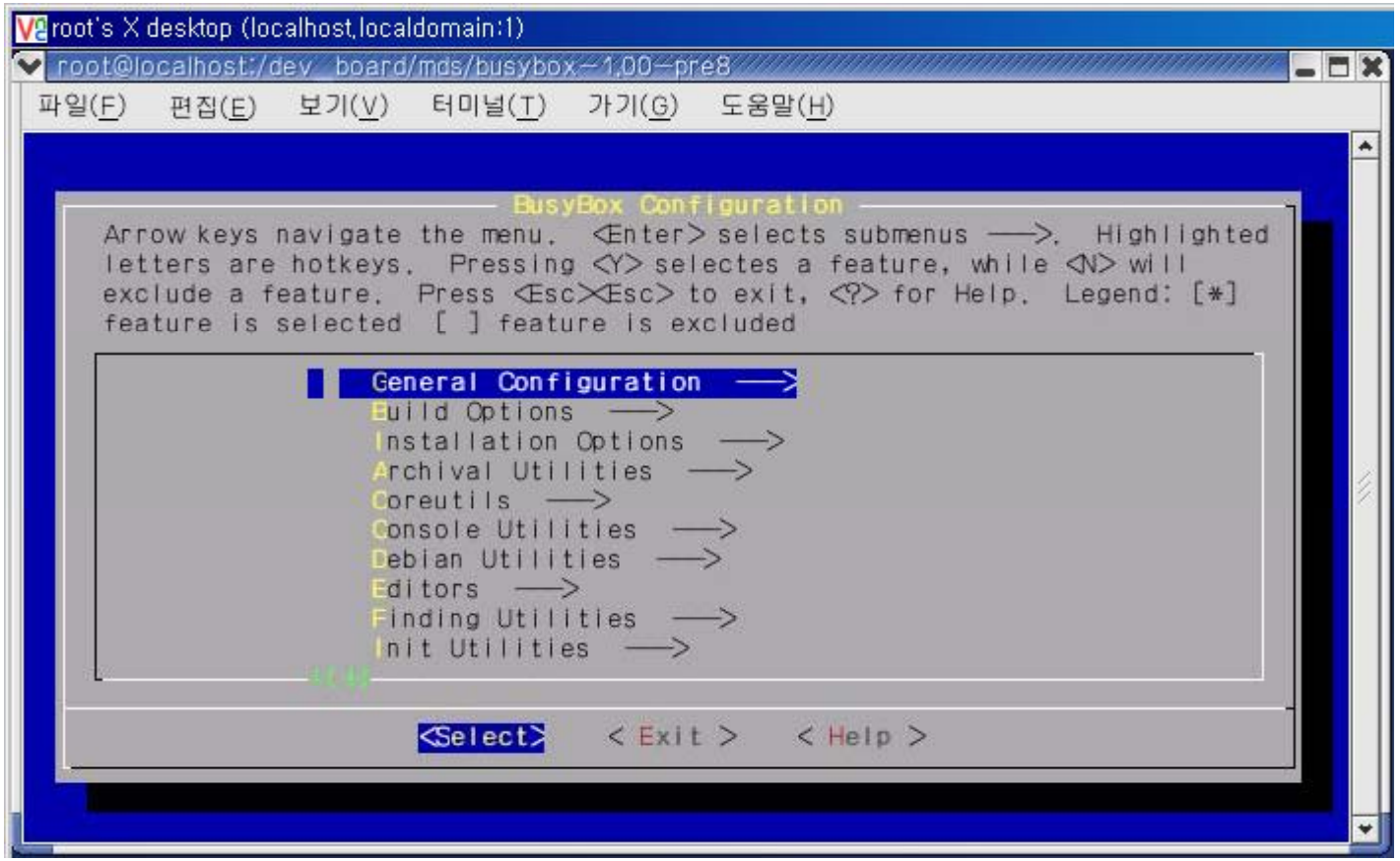
Debugging Options
bebugging symbol 추가 여부 결정

Load an Alternate Configuration File
기존의 설정을 loading

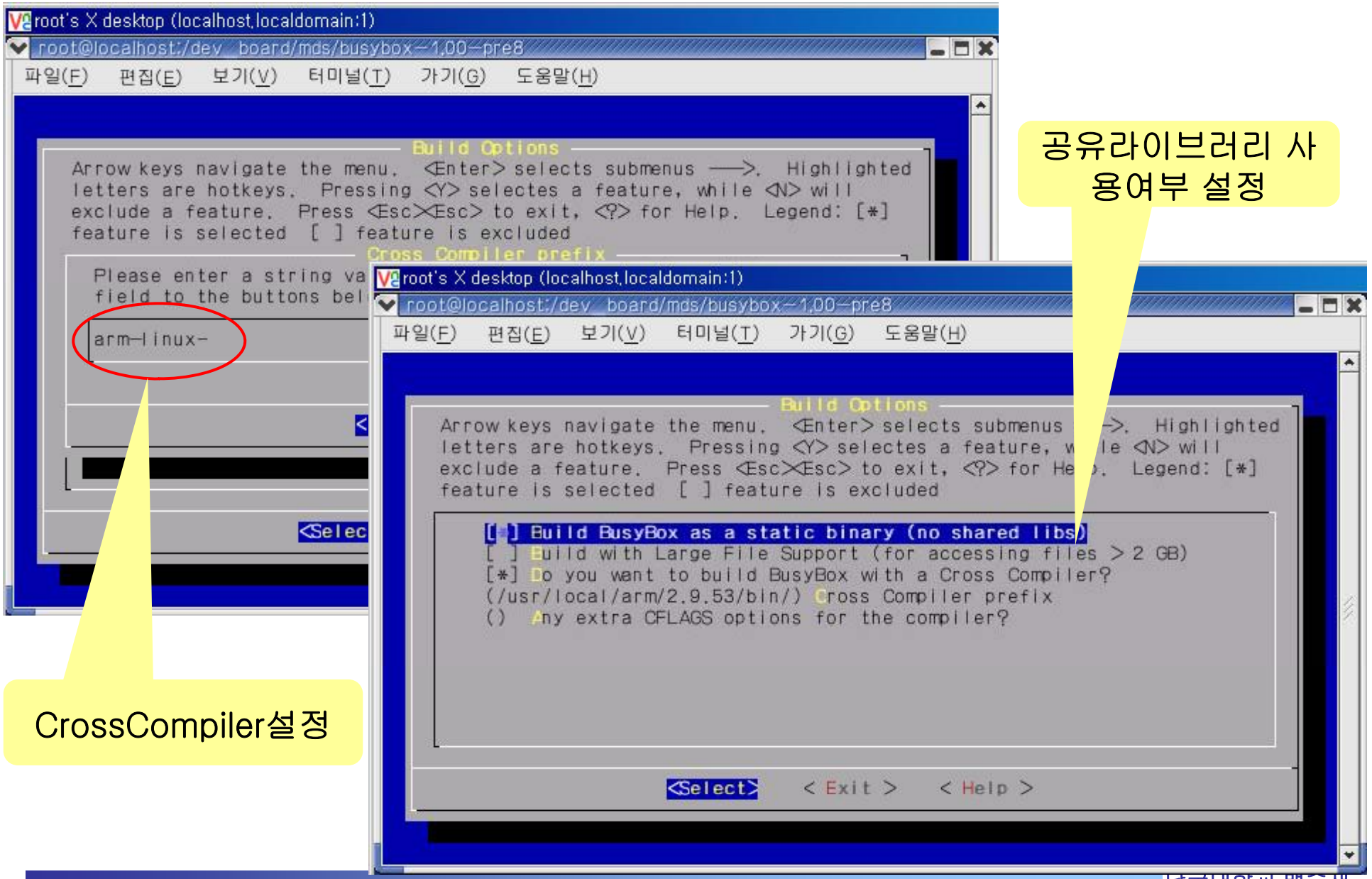
Save Configuration to an Alternate File
현재의 설정을 파일에 저장

RAMDISK 기능과 구조

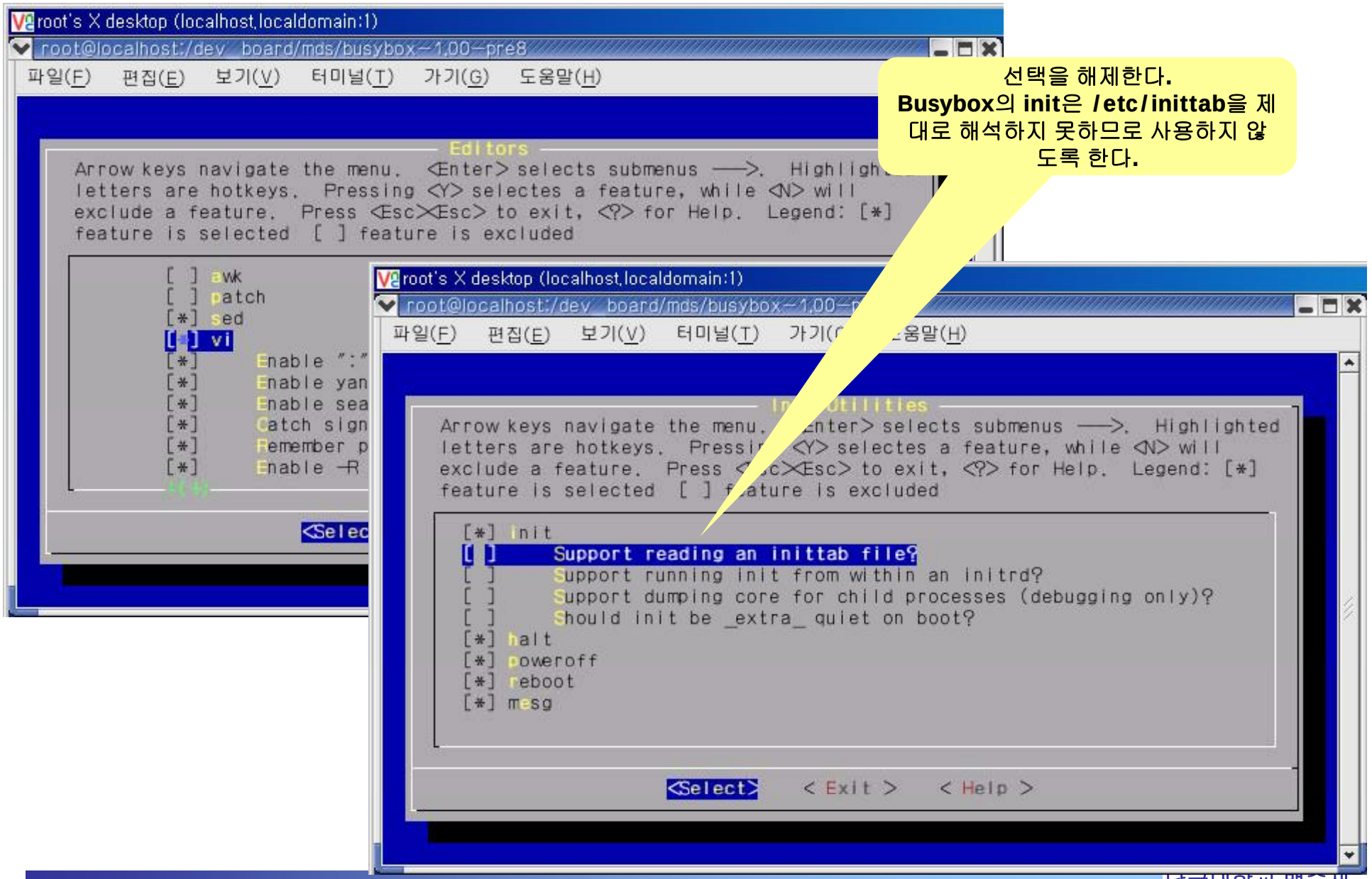
- Configuration



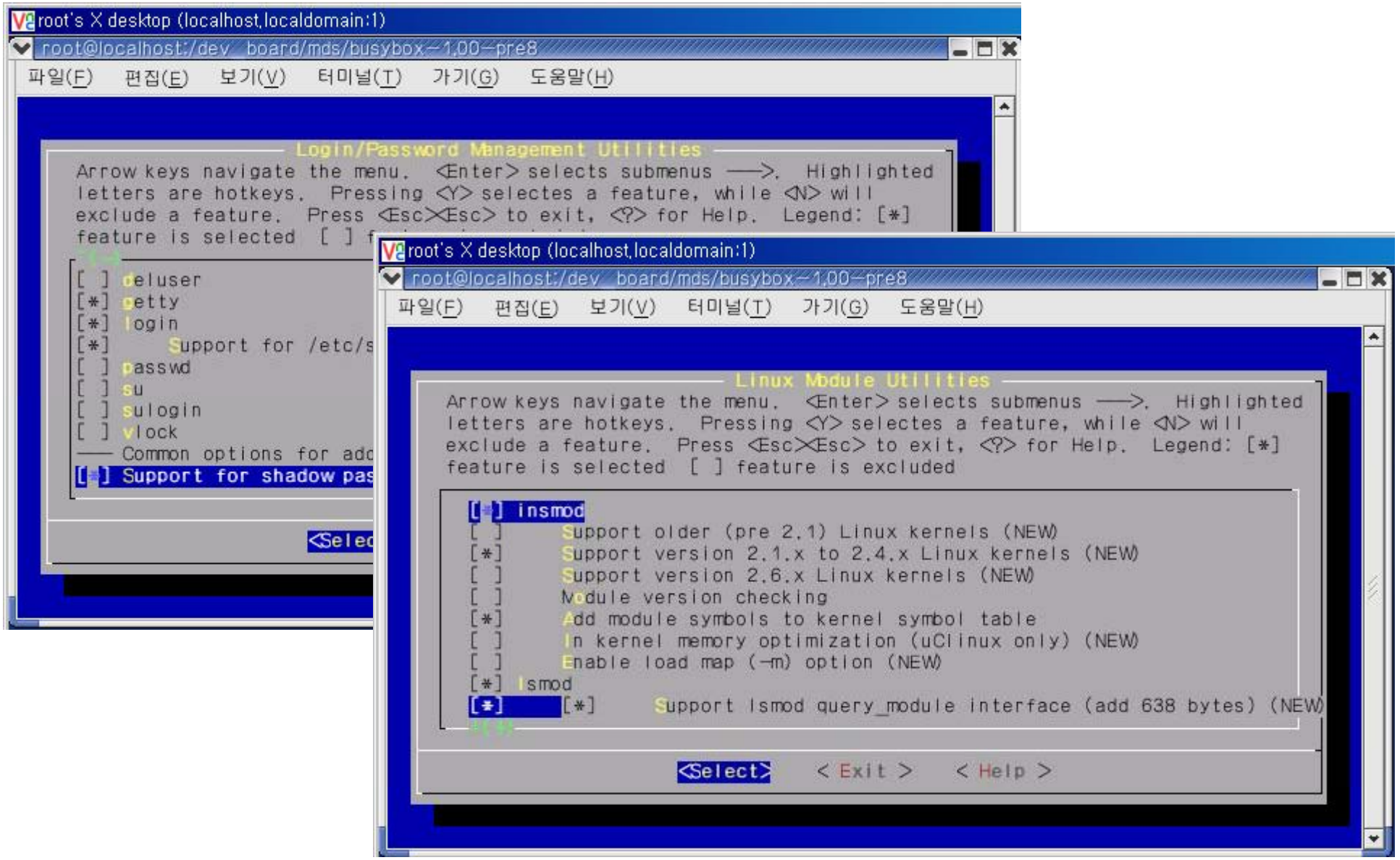
RAMDISK 기능과 구조



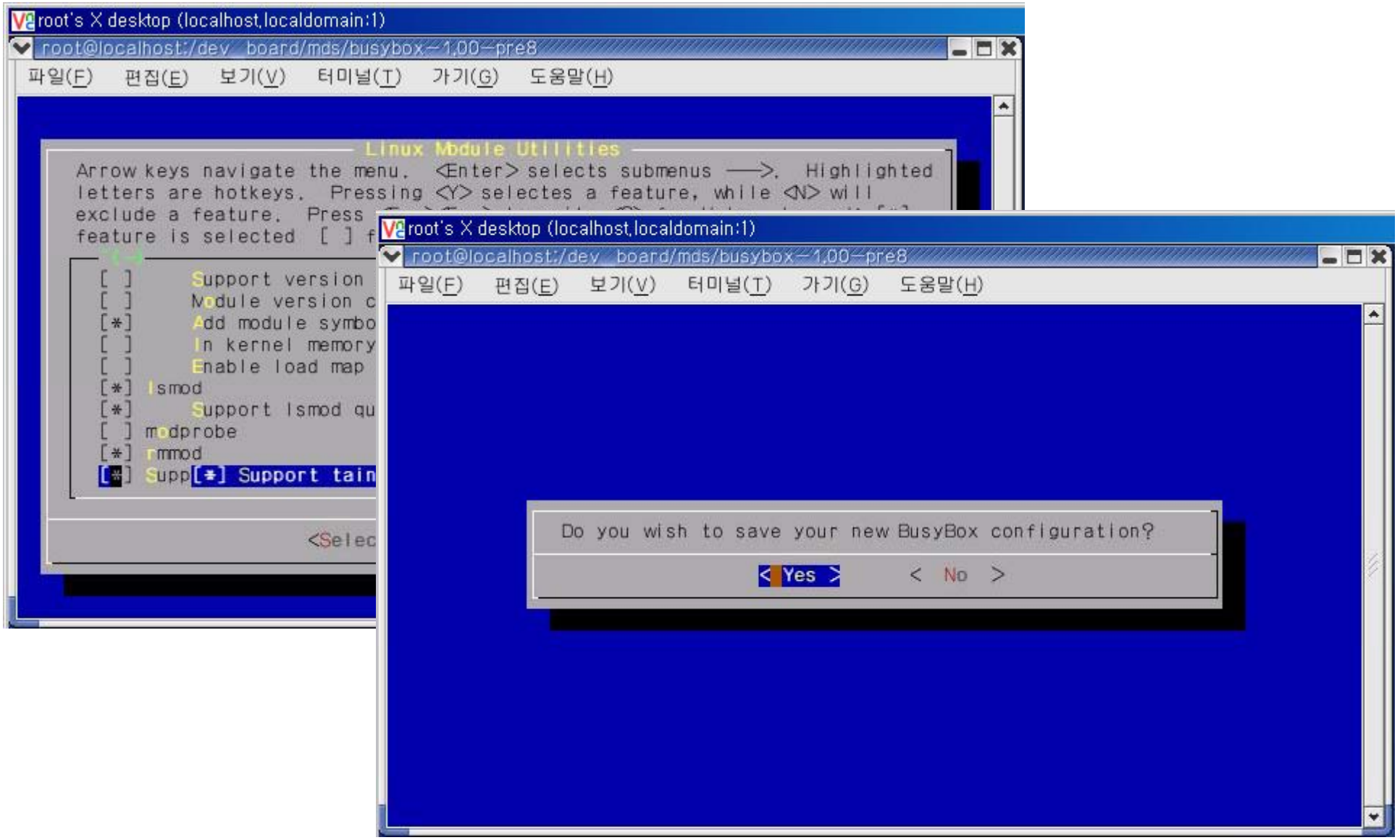
RAMDISK 기능과 구조



RAMDISK 기능과 구조

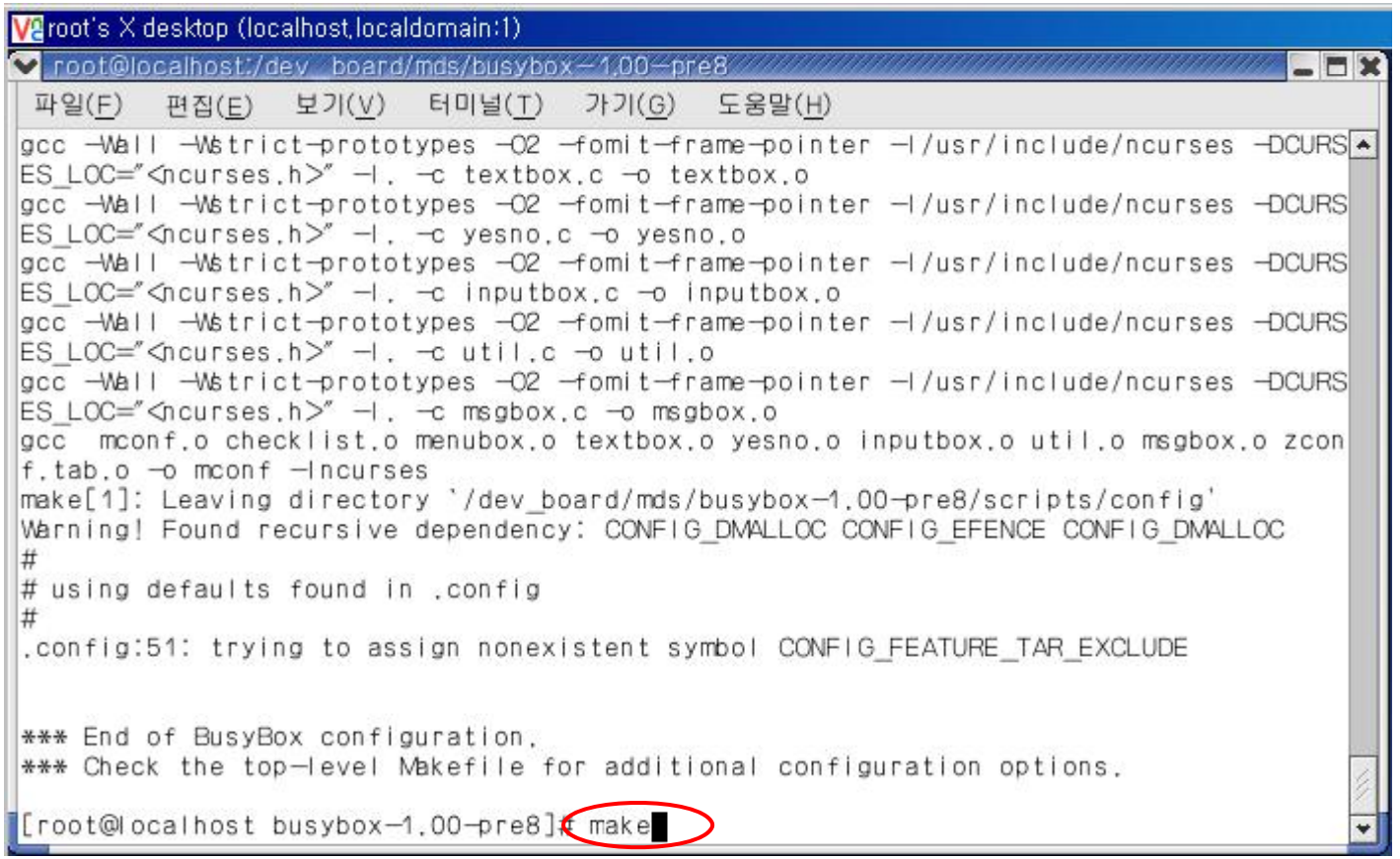


RAMDISK 기능과 구조



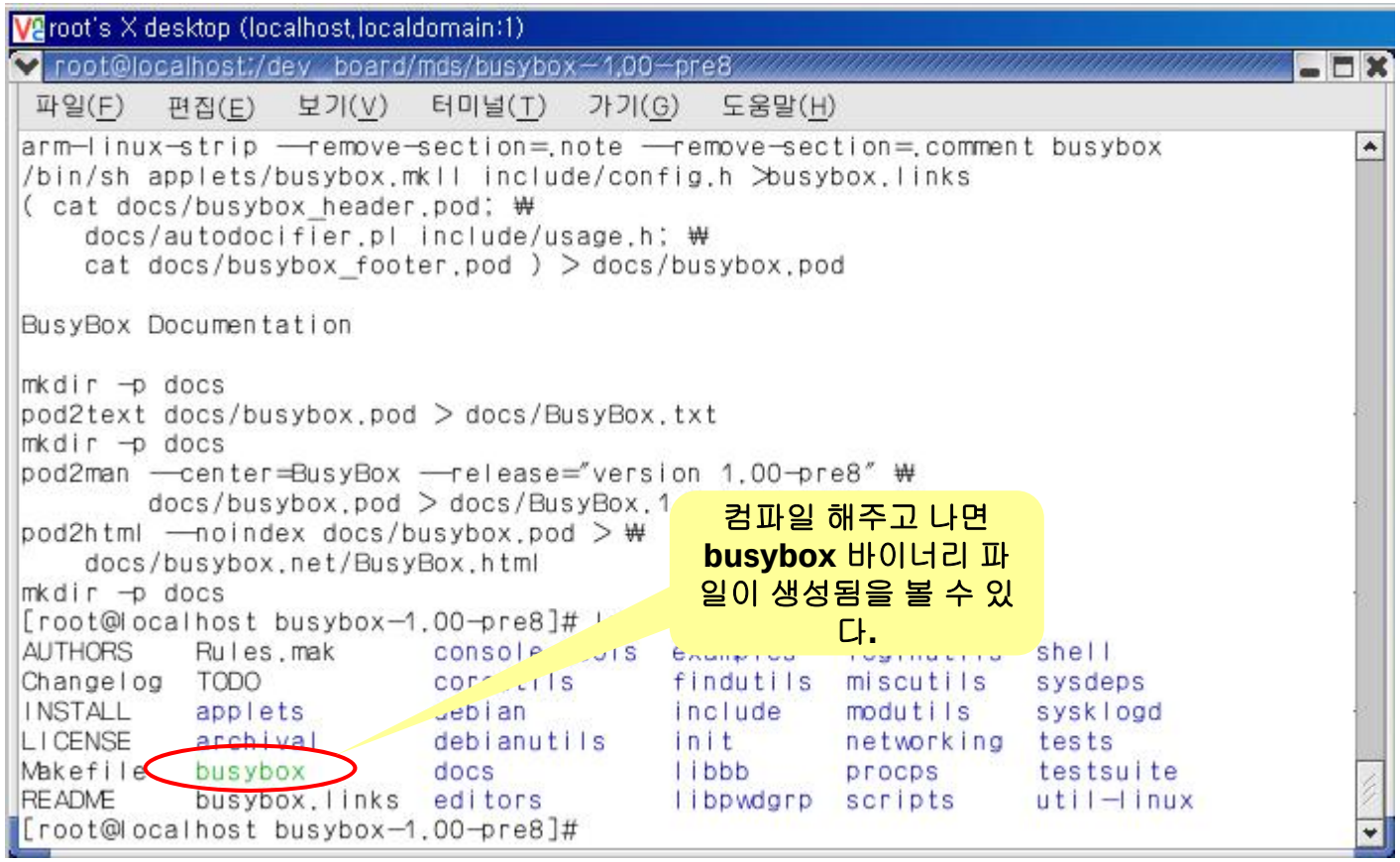
RAMDISK 기능과 구조

- Make 수행



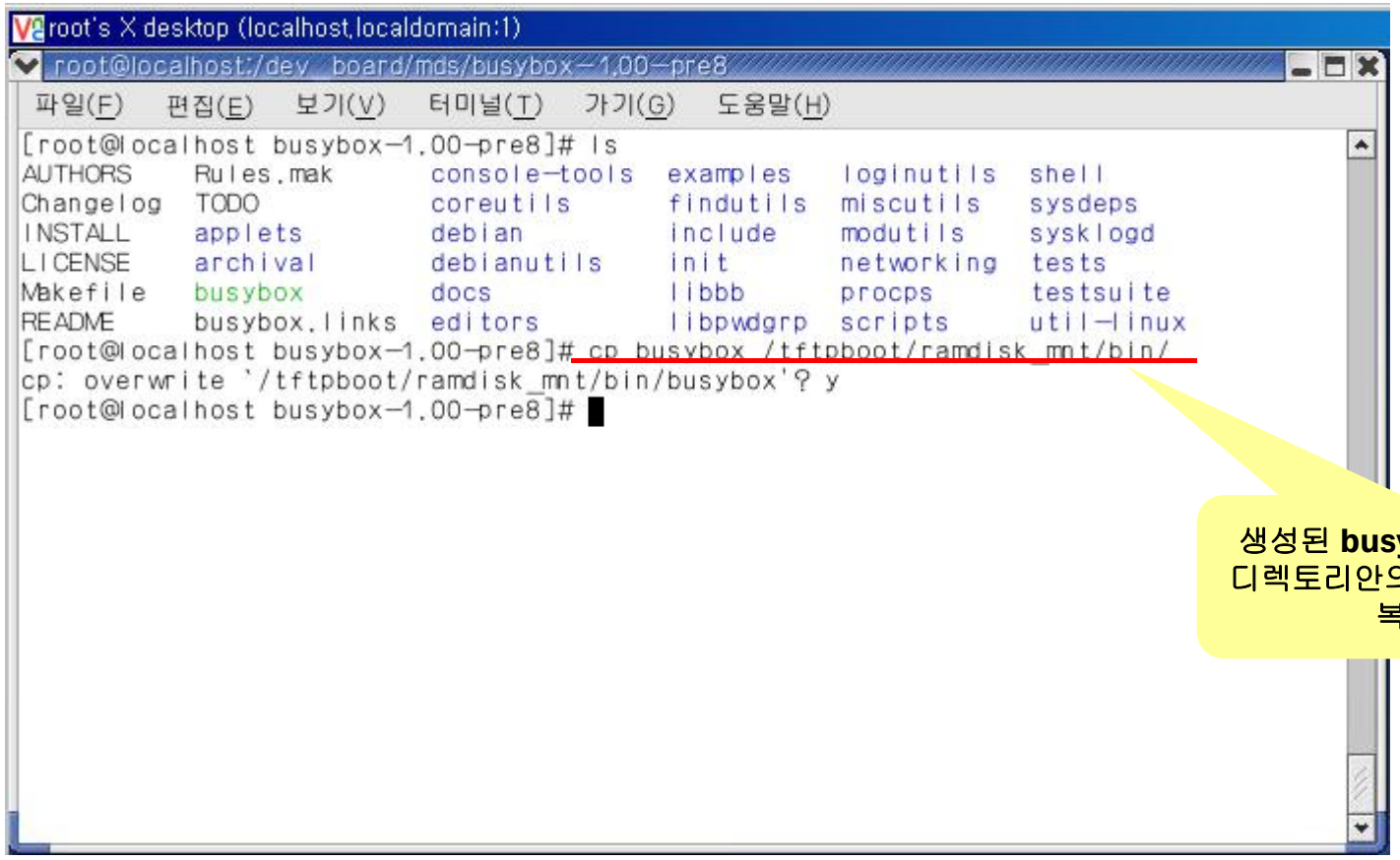
RAMDISK 기능과 구조

- 바이너리 파일 확인



RAMDISK 기능과 구조

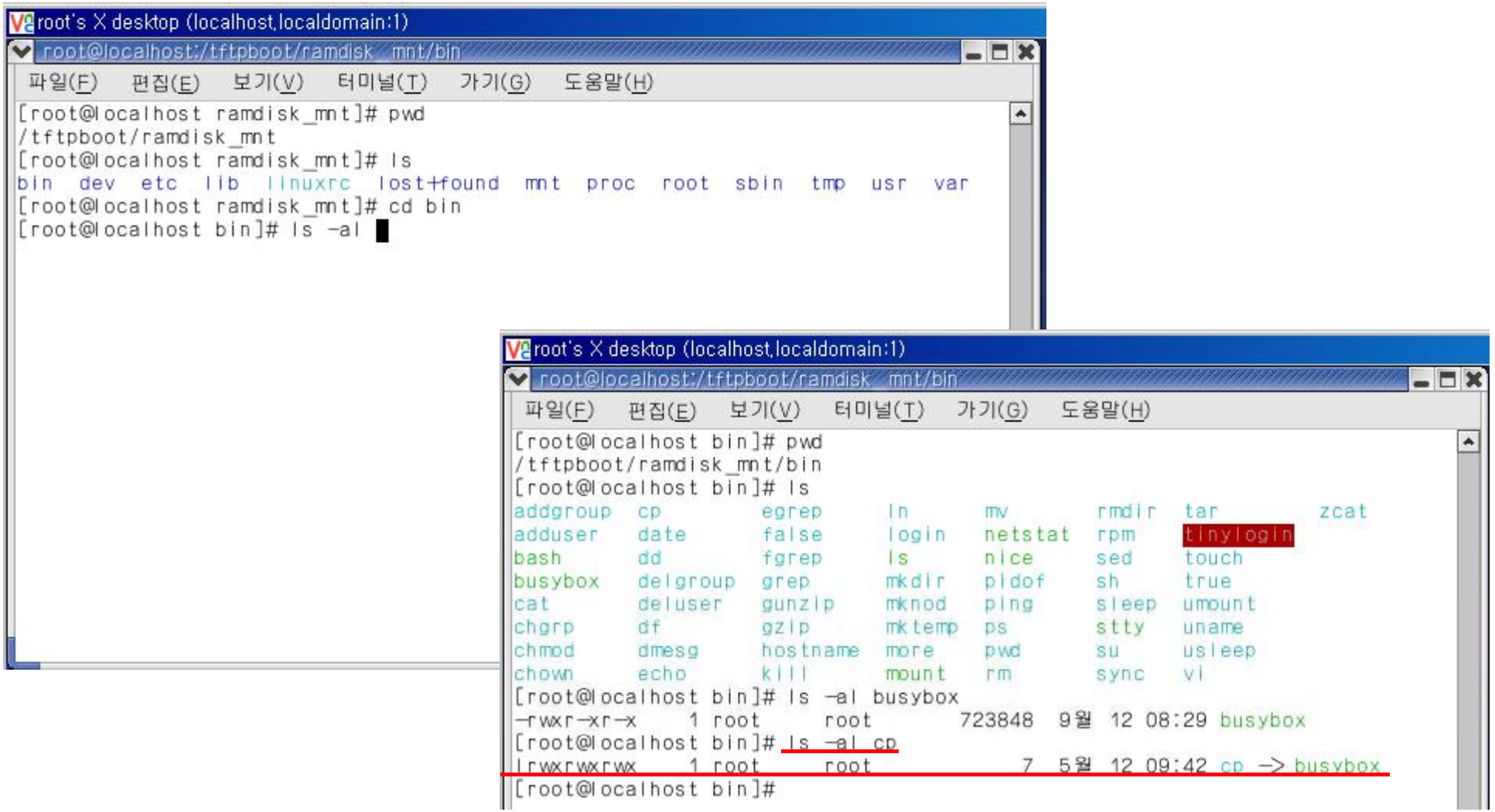
- BusyBox Binary File copy



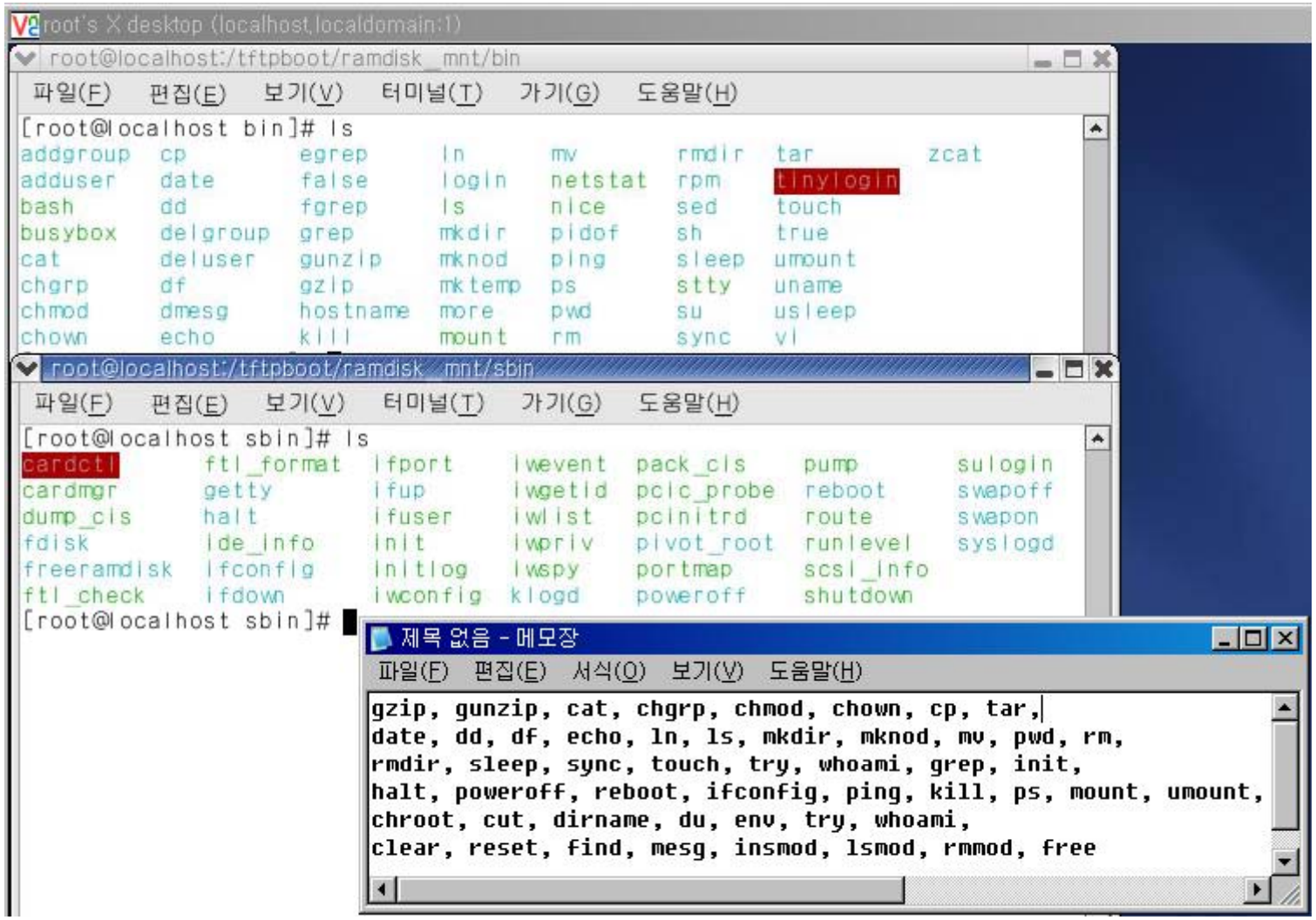
생성된 **busybox**를 **mount**한 디렉토리안의 **/bin** 디렉토리에 복사한다.

RAMDISK 기능과 구조

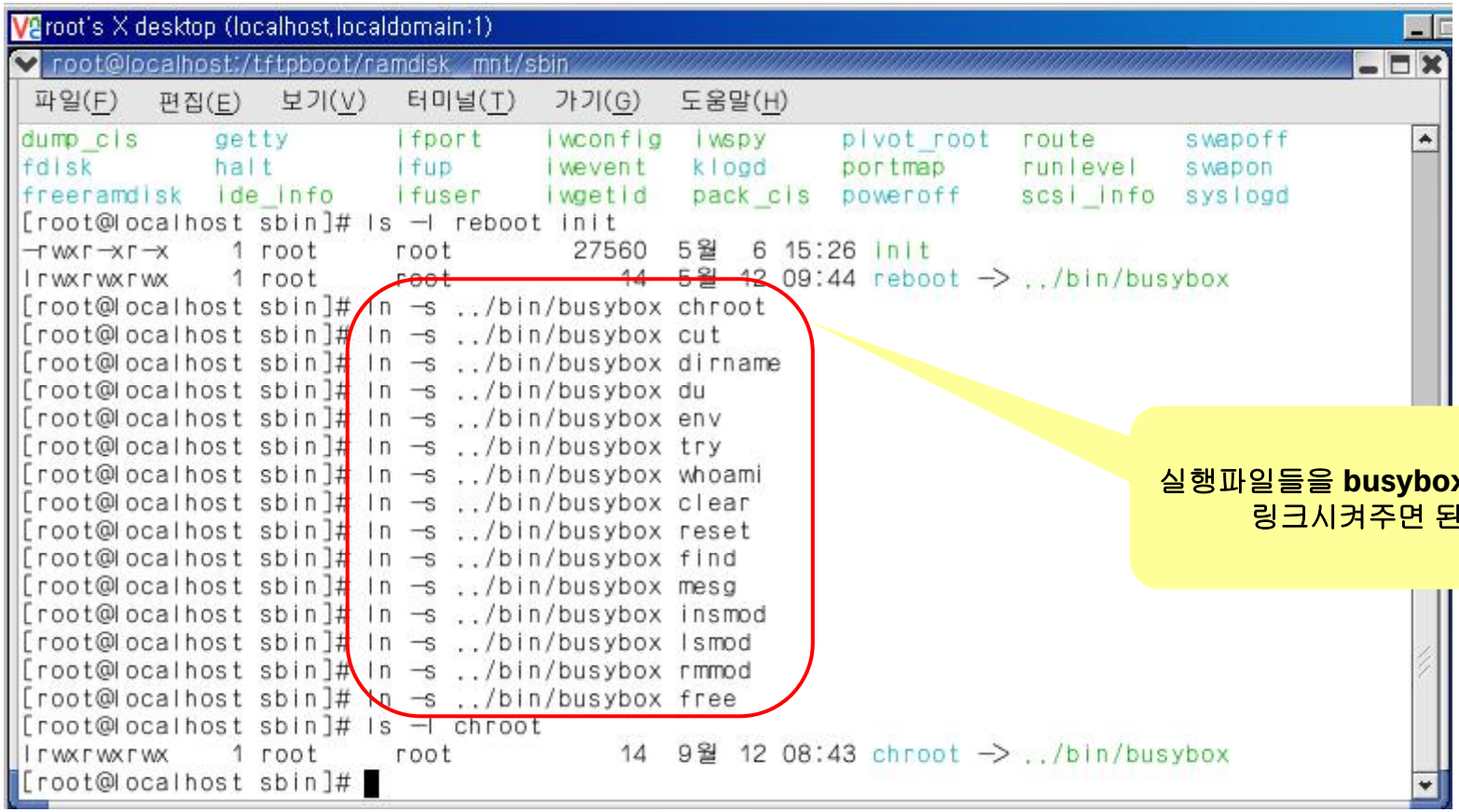
- 확인 / 링크 확인



RAMDISK 기능과 구조

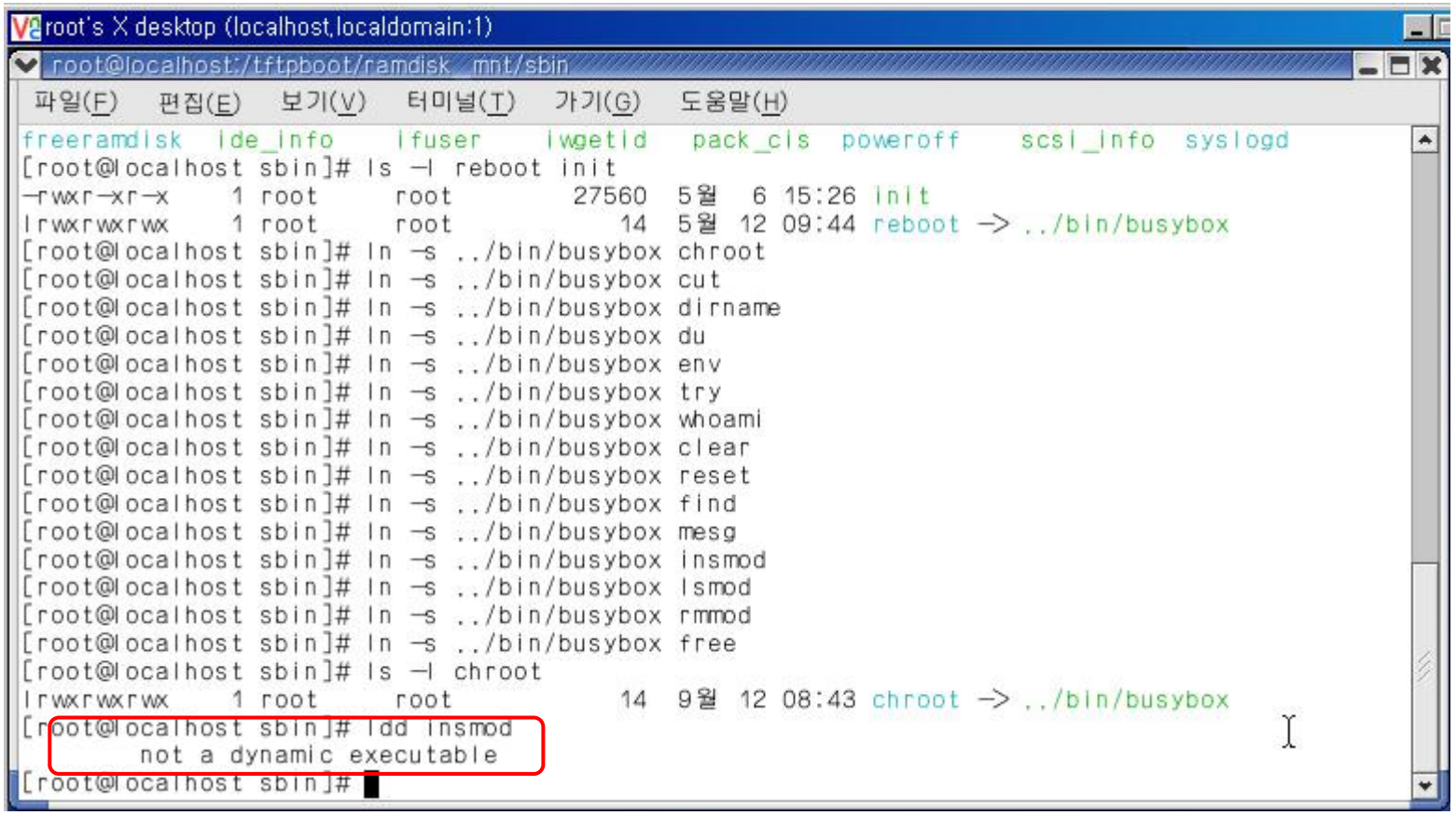


RAMDISK 기능과 구조



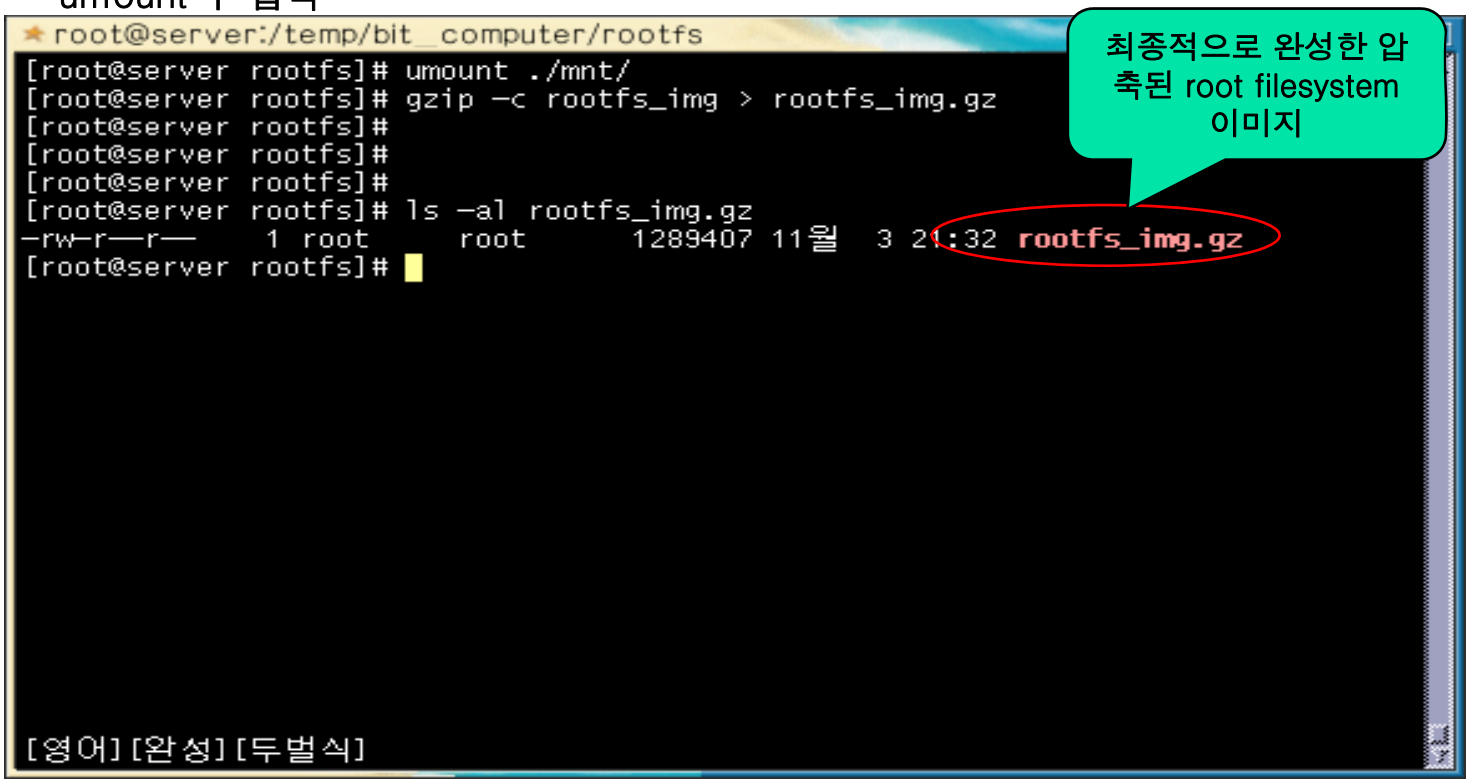
실행파일들을 **busybox**에 심벌릭 링크시켜주면 된다.

RAMDISK 기능과 구조



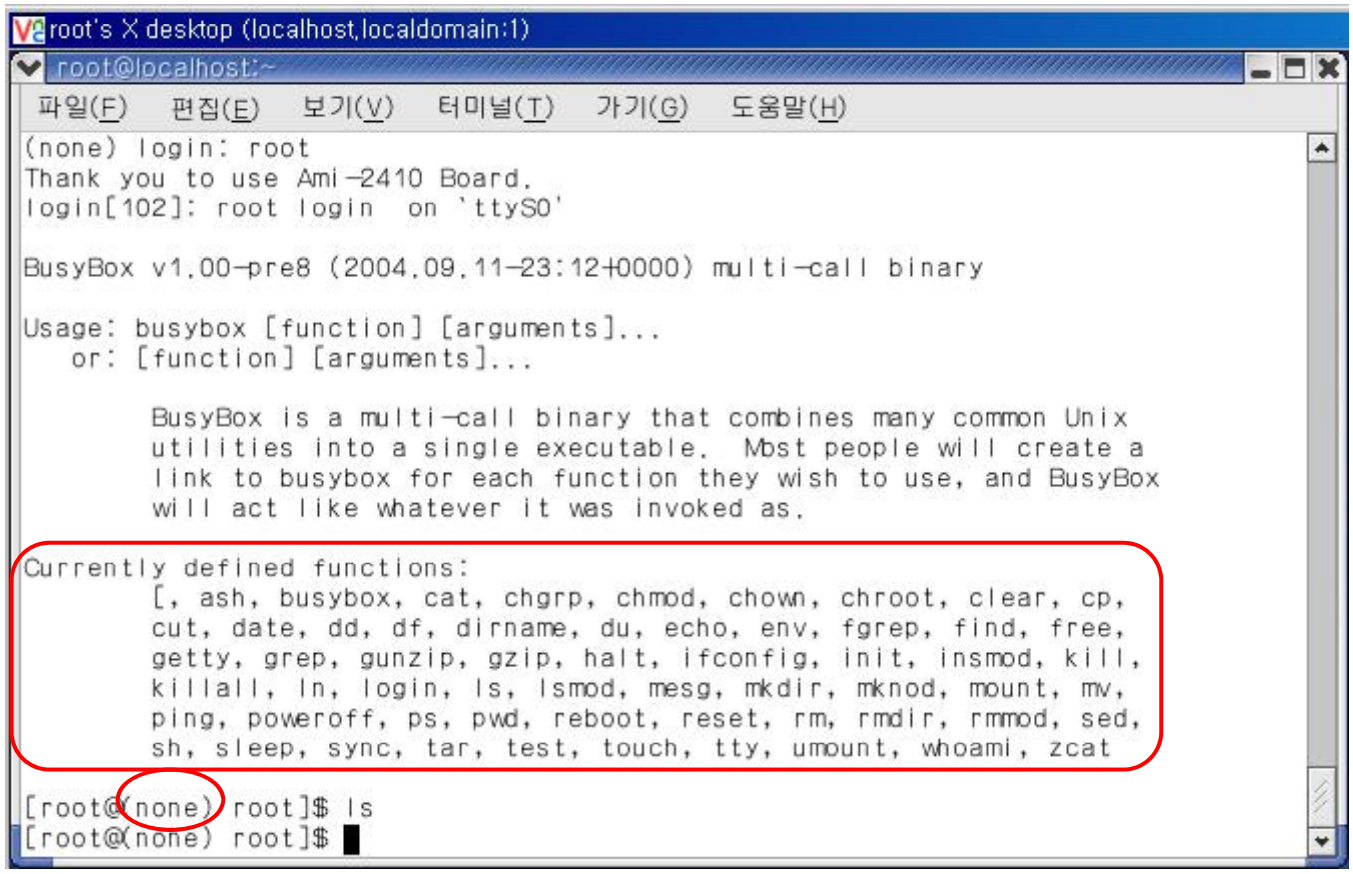
파일 시스템 이미지 압축

umount 후 압축



RAMDISK 기능과 구조

- Booting using New Kernel & New RAMDisk



kernel 설정 및 컴파일

- kernel 설정
  - ✓ Ramdisk를 사용하기 위한 kernel 설정