The Linux BootPrompt HOWTO

Paul Gortmaker, Editor. v1.13, 15 November 1997.

번역자: 황병렬. 번역일: 1998년 5월 20일.

이 문서는 BootPrompt-Howto로서 리눅스 부트시에 커널에 전달되는 모든 가 능한 인자들에 관한 것이다. 또한 모든 커널과 디바이스에 관한 인자들도 포함된다. 커널의 인자들을 분류하고 또한 리눅스 커널 부트에 관하여 사용 되는 인기 있는 프로그램들에 대하여 대략적으로 논의한다. (변역된 문서는 버전이 1.13이다. 98.5월 현재 v1.14 문서가 나와 있음으로 sunsite를 참조하기 바란다.:역주)

목 차

제 1 절 Introduction

커널은 프로그램에 전달되는 커맨드 라인이라는 형태의 방법으로는 리눅스 부트시에 모든 정보를 전달하기에는 제한된 능력을 가지고 있다. 일반적으 로 이는 커널 스스로 결정할 수 없는 하드웨어 파라미들에 대한 정보를 커 널에 공급하는데 사용된다. 단지 커널 이미지를 플로피에 (예를들어 cp zImage /dev/fd0) 복사한다면 커널에게 특별한 인자들을 전달할 기회는 없어지게 된다. 그래서 대부분의 리눅서들은 LILO나 loadlin과 같이 이러한 인자들을 커널에게 넘겨주는 프 로그램을 사용할 것이다. IMPORTANT NOTE TO MODULE USERS: 부트 프롬프트 인자들인 전형적으로 단지 커널에 의하여 이해되는 하드웨어 드라이버에게 공급된다. 이것들은 모듈에 의하여 올려진 드라이버에게는 효과가 없다. 모듈인지 아닌지 의심스럽다면 /etc/conf.modules파일의 내용과 함께 depmod와 modprobe에 대한 매뉴얼 페 이지를 보라. 이 문서는 커널 버전 v2.0.31 과 그 이상에서 유효하며 개발/테스트 버전의 커널에서는 v2.1.6x에서 문서화 되었다.

The BootPrompt-Howto is edited and maintained by: Paul Gortmaker, Paul.*Gortmaker@anu.edu.au* `<mailto:Gortmaker@anu.edu.au>`

[i386 구조가 아니고 현재 문서화 되지 않은 장치(예를들면 Atrai/Amiga)에 는 부트 프롬프트 인자들이 의미가 없음을 주의하라!]

1.1 Disclaimer and Copyright

이 문서는 gospel이 아니다. 이것은 이 문서를 볼 때 쯤에는 아마 문서 자체 가 업데이트 될 수 있음을 의미한다. 당신을 제외하고 어느 누구도 발생한 일에 대하여 책임을 지지 않는다. 당신의 하드웨어에서 김이 모락모락 나도 (거의 발생하지 않겠지만) 나는 책임이 없다. 저자는 이 문서 안에 존재하 는 정보에 기반하여 발생하는 어느 손상에 대하여서도 책임이 없다. This document is Copyright (C) 1995,1996 by Paul Gortmaker. 이 문서는 GNU General Public License, version 2에 따르고 그와 함께 복 사되어야 한다. 좀더 자세한 내용은 리눅스 커널과 함께오는 linux/COPYING 을 보기 바란다. 당신이 만약 이 문서를 출판하려고 한다면 나에게 연락하기 바란다. 나는 당신이 최신 내용이 담긴 문서를 구할 수 있게끔 노력할 것이다. 여지껏 버 전이 낮은 리눅스 하우투 문서들이 출판되었다. 이는 버전업한 문서에서 이 미 대답한 내용에 대한 질문으로 개발자들을 괴롭히게 된다.

1.2 Related Documentation

대부분의 최신 문서들은 항상 커널 소스 그 자체이다. 잠깐! 겁먹지 마라. 소스 파일 안에 있는 주석들을 읽는데는 어느 프로그래밍에 대하여서도 알 필요가 없다. 예를 들면 AHA1542 SCSI 드라이버에게 전달되는 인자들에는 어떤 것들이 있는지를 보고 싶다면 linux/drivers/scsi 디렉토리에 가서 aha1542.c 파일을 보면 된다. 그리고 파일의 처음부터 100번째 라인안에서 당신은 1542 드라이버가 부트시에 허용하는 인자들에 대한 설명을 읽을 수 있다. 물론 영어로된 단순한 text 이다. 다음으로 좋은 문서는 커널을 설치할 때 같이 온 여러 문서들일 것이다. 거 기에는 약간의 문서들이 있고 그들의 대부분은 linux/Documentation 디렉토리나 그 하위 디렉토리에 있다. linux 디렉토리는 보통 /usr/src/에서 찾 을 수 있는데 가끔은 각각의 드라이버에 관련된 문서들은 그 이름의 디렉토 리에 있다.(예를 들면 linux/drivers/XXX/, 여기서 XXX 는 scsi, char 나 net 등이 될 것이다.) 당신이 사용하려는 인자가 무엇인지 생각해 왔고 그리고 당장 커널에서 그 런 정보를 어떻게 얻는지 알고 싶다면 커널을 부트하는데 사용되는 소프트 웨어를 가져올 때 같이 따라온 문서들을 보라(예를들면 LILO 나 loadlin). 아래에는 전체적인 윤곽이 나와 있다. 그러나 이것들은 부트 소프트웨어와 함께온 문서들을 대신하지는 못한다.

1.3 The Linux Newsgroups

커널에 부트 인자들을 전달하는 것에 대하여 질문이 있다면 먼저 당신이 보 고 있는 이 문서를 보아라, 이 문서나 관련된 문서들이 당신의 질문에 대하 여 대답하지 못한다면 Linux newsgroup에 물어 보아라. 물론 질문을 newsgroup에 올리기 전에 다른 사람이 이미 질문을 했던 것이거나 a Frequently Asked Question (a FAQ)에 존재하는 내용은 미리 읽어야 한다. 시스템 환경설정을 어떻게 하는가에 관한 일반적인 질문은 comp.os.linux. setup newsgroup에 직접하라. 우리는 당신이 질문에 있어 그 내용이 일반적인 가이드 라인에 적합하기를 요구한다. 또한 다른 newsgroup에 중복하여 질문을 올리지 마라.

1.4 New Versions of this Document

이 문서의 새버전은 sunsite.unc.edu 사이트의 /pub/Linux/docs/HOWTO 디렉토리에서 anonymous FTP를 통하여 검색할 수 있다. 보통 SunSITE는 네트웍 부하가 많이 걸림으로 당신은 Linux ftp mirror 사이트를 통하여 문서를 얻는 것이 더 좋을 것이다. 업데이트는 새로운 정보나 드라이버가 될 수 있고 각각 또는 따로 발생하기도 한다. 이 문서를 지난 몇달 동안 읽어 왔다면 새로운 버전의 자료가 존재하는지 알아보아라 이 문서는 Linux Howto project에 맞추어 수정된 SGML시스템을 사용하여 만들어졌다. 거기엔 여러 출력물 즉, postscript, dvi, ascii, html 그리고 TeXinfo 형태가 존재한다. 나는 WWW browser를 통하여 html파일로 보거 나 아니면 Postscript/dvi형태로 보기를 권장한다. 이들은 ascii 문서에는 없는 역참조 색인을 가지고 있다. 공식적인 문서를 원한다면 아래의 URL을 보라.

BootPrompt-HOWTO *sunsite.unc.edu/mdw/HOWTO/BootPrompt-HOWTO.html* `<http://sunsite.unc.edu/mdw/HOWTO/BootPrompt-HOWTO.html>`

제 2 절 Overview of Boot Prompt Arguments

이 장은 커널 그 자체에게 부트 인자들을 전달해 주는 소프트웨어의 여러 예들을 보여줄 것이다. 어떻게 인자들이 처리되는가 또한 부트 인자들의 한계는 무엇인가 그리고 어떻게 각각의 원했던 장치로의 접근이 가능한지를 알 수 있다. 부트인자에 스페이스가 사용 안되도록 아주 주의하여야 한다. 단일 인자들 은 그 값들을 나열하며 여러 인자들은 스페이스 없이 콤마로 분리된다. 아래 이어지는 예들을 보라

```
ether = 9,0x300,0xd0000,0xd4000,eth0  root=/dev/hda1          *RIGHT*
ether = 9, 0x300, 0xd0000, 0xd4000, eth0  root = /dev/hda1    *WRONG*
```

2.1 LILO (LInux LOader)

Werner Almesberger가 만든 LILO(LInux LOader)는 매우 일반적으로 사용된 다. 이것은 여러 커널들중에 하나로 부팅할 수 있으며 이런 정보를 보통 텍스트로 저장한다. 대부분의 배포판에서는 기본적인 부트로더로서 LILO가 사용된다. LILO는 DOS, OS/2, Linux, FreeBSD 등을 별 어려움 없이 부팅할 수 있으며 매우 융통성이 있다.

전형적인 환경설정이라면 LILO는 멈추어서 "LILO" 를 출력할 것이다. 간단 히 말하면 당신이 컴퓨터를 켠후에 컴퓨터는 사용자로부터 옵션을 입력 받 기 위하여 몇초 동안 기다린다. 일반적인 시스템은 LILO configuration파일 을 이용하여 "linux", "backup", "msdos"라는 라벨들을 사용한다. 부트 인자 들을 넣으려 한다면 당신이 부트하기 원하는 시스템 라벨을 입력한 후에 아 래의 예처럼 입력하라.

```
        LILO: linux root=/dev/hda1
```

여기서 논의된 부트 인자들에 관해서는 LILO와 함께 오는 문서가 아주 훌룡 하다. LILO 에서의 append=의 명령은 부트시에 인자들을 LILO 설정 파일에 영구적으로 추가하는데 아주 중요하다. 당신은 간단하게

append= "foo_bar" 와 같은 형식으로 /etc/lilo.conf 파일에 추가할 수 있다. 이것은 설정 파 일 처음에 추가되어 모든 섹션에 영향을 주게 할 수도 있고 "image=" 섹션 안에 추가되어 단일 섹션에 영향을 줄 수도 있다. 더 자세한 내용은 LILO문 서를 보라.

2.2 LoadLin

다른 부트 로더로는 자원들이 확실하다는 가정을 하고 도스 프롬프트로부터 리눅스 커널을 작동시키는(부트 인자들 포함) 능력을 가진 LoadLin 이라는 도스 프로그램이 있다. 도스를 사용하며 도스로부터 리눅스를 시작하기를 원하는 사람에게는 좋은 프로그램이다.

만약에 당신이 하드웨어를 알려진 상태로 두는 것을 지원하는 도스 드라이 버에 기반한 하드웨어를 가지고 있다면 이 프로그램은 매우 유용하다. 예를 들면 사블 호환모드로 두기 위해 약간의 레지스터들을 조작하는 도스 드라 이버를 요구하는 'SoundBlaster Compatible' 사운드카드를 가지고 있다면 이 경우에 해당한다. 주어진 드라이버와 함께 도스로 부팅한후에 시스템을 reboot함으로써 카드에 설정된 상황을 읽어버리는 경우 없이 도스 프롬프트 에서 리눅스를 바로 올릴 수 있다. 이렇게 하여 사블 호환 카드는 리눅스 하에서도 그 모드를 유지한다.

리눅스를 부트하는데 사용하는 다른 프로그램도 있지만 완벽한 리스트는 당 신의 지역 Linux ftp 미러의 system/Linux-boot/. 밑을 보라.

2.3 The "rdev" utility

커널 이미지 안에 여러 바이트로서 저장되어 디폴트 값들을 가지고 있는 부 트 인자들은 그리 많지 않다. 이러한 값들이 어디에 있는지 또 이들 값들을 어떻게 바꾸는지 알고 있는 대부분의 시스템에는 rdev 라는 유틸리티가 인스톨 되어 있다. 이는 기본적인 비디오 모드와 같이 커널 부트 인자로는 적합하지 않은 것들을 바꿀 수 있다.

rdev 유틸리티는 대게 swapdev, ramsize, vidmode 그리고 rootflags들로 다 르게 불리기도 한다. 거기엔 rdev가 변경할 수 있는 root device, swap device, RAM 디스크 parameter 디폴트 비디오 모드 그리고 root device의 읽기전용 아니면 읽고/쓰기모드의 5가지가 있다. rdev에 관한 좀더 많은 자료는 rdev -h 을 사용하거나 관련된 맨 페이지 를 보라(man rdev를 사용)

2.4 How the Kernel Sorts the Arguments

대부분의 부트 인자들이 가지는 형태:

```
name[=value_1][,value_2]...[,value_11]
```

name 부분은 커널에 관련되어 사용되는 유일한 키워드이다. 멀티 부트 인자 들은 위의 형태를 가지며 스페이스로 구분된다. 이들이 11개로 제한되어 있 다는 것을 주의 하라. 이들은 키워드당 콤마로 분리된 11개의 parameter들 만을 허용한다. (어떻든 당신은 setup 함수가 그것을 지원한다는 가정하에 매우 복잡한 상황에서 같은 키워드를 가지고 11개의 parameter들을 재 사용 할 수 있다.) 또한 커널은 리스트를 최대 열개의 정수 인자와 이어지는 스트링 으로 나눈다. 그래서 당신은 실제로 드라이버 그 자체에서 11개의 인자들을 문자열에서 정수로 변환시키지 않고는 공급할 수 없다.

대부분 인자들은 linux/init/main.c에 있다. 첫째로, 커널은 인자가 'root=', 'ro', 'rw', or 'debug'와 같은 특별한 인자인가를 검사한다. 이런 특별한 인자들은 이 문서 저 뒤에서 서술한다.

특별한 인자 문자열('foo' 같이)이 setup function(foo_setup())에 연결되 어 있는지를 보기 위하여 이후 장치나 커널의 일부분으로 구분된 setup finctions (bootsetups 배열을 포함하는)들의 목록들로 제어를 옮긴다. 커널에게 foo=3,4,5,6,bar 이라는 라인을 전송했다면 커널은 'foo' 가 등록되 어 있는지를 보기 위하여 bootsetups 배열을 검색한다. 있다면 'foo'(foo_setup())에 연결되어 있는 setup function을 실행 시키고 여기

에 커널 명령행에 주어진 것처럼 정수인자 3, 4, 5그리고 6 을 전달하고 또한 문자열 인자인 bar도 건네 줄 것이다.

2.5 Setting Environment Variables.

위에서 말한 setup function으로 하여금 허용되지 않는 'foo=bar'과 같은 형태는 어느 것이든 값을 가진 환경변수로 변환된다. 'TERM=vt100'와 같은 부트 인자가 그 한 예이다.

2.6 Passing Arguments to the 'init' program

커널에서도 설정되지 않고 환경변수로도 인터프리터 되지 않은 남은 인자들 은 일반적으로 init라고 부르는 process에 전달된다. 대부분의 일반적인 인자들 즉, init process에 전달된 대부분의 일반적인 인자는 단일 사용자 모드로 컴퓨터를 부팅하는 것을 가리키는 하나의 단어이며 이것은 평상시의 모든 데몬을 작동시키지는 않는다. 어떤 인자들이 허용되는 가를 보기 위해서 시스템에 설치된 init버젼의 메뉴얼 페이지를 점검하라.

제 3 절 General Non-Device Specific Boot Args

특정 장치나 주변기기에 정확히 관련되지 않는 부트 인자들이 있다. 이들은 대신에 메모리 핸들링이나 램디스크 핸들링 루트파일시스템의 핸들링 등의 커널 내부 parameter에 관계되어 있다.

3.1 Root Filesystem options

아래의 것들은 어떻게 커널을 선택하고 루트 파일 시스템을 핸들할 것인가에 알맞은 옵션들이다.

3.1.1 The 'root=' Argument

이 인자는 부트하는 동안에 루트파일 시스템으로 사용될 장치가 무엇인가를 커널에게 알려준다. 기본 값은 커널이 설정되어 있는 시스템의 루트 장치의 값이다. 예를 들면 커널이 루트 파티션으로 '/dev/hda1' 장치를 사용하는 시스템에 설치 되어 있다면 기본 루트 장치는 '/dev/hda1'일 것이다. 이 기본 값을 무효로 하고 루트 장치로서 플로피 드라이브를 사용하려 한다면 'root=/dev/fd1'을 사용하면 된다.

```
다음에 정확한 루트 장치들이 있다:
(1) /dev/hdaN to /dev/hddN, ST-506 호환 디스크의 'a 부터 d' 의 파티션 N.
(2) /dev/sdaN to /dev/sdeN, SCSI 호환 디스크의 'a 부터 e' 의 파티션 N.
(3) /dev/xdaN to /dev/xdbN, XT 호환 디스크의 'a 부터 b' 의 파티션 N.
(4) /dev/fdN, 플로피 디스크 드라이버의 번호  N. N 이 0이면 도스의 'A:'
드라이브이고 1이면 도스의 'B:' 드라이브이다.
(5) /dev/nfs, 이것은 실제 장치가 아니며 커널에게 네트워크를 통한 루트
파일 시스템을 구하라고 커널에게 말하여 주는 플래그이다.
```

좀더 다루기 힘들고 이식성이 없는 수치 명세도 major/minor 형식으로 표현 가능한 디스크 장치에 있어서는 가능하다. (예를 들면 /dev/sda3은 major 8, minor 3이다. 따라서 root=0x803으로도 바꿔 사용할 수 있다.) 'root='는 커널 이미지안에서 rdev 유틸리티로 변경할 수 있는 기본적인 몇 몇 부트 인자들중에 한 가지이다.

3.1.2 The 'ro' Argument

커널 부트시에 기본적인 것들을 읽기 위해서는 루트 파일 시스템이 필요하다. 루트 파일시스템은 부트시에 마운트 된다. 어떻든, 루트 파일시스템이 쓰기 가능하게 마운트 되었다면 당신은 반쯤 저장된 파일을 가지고 있는 파일 시스템을 완벽하게 검사할 수 없다. 여기서 'ro' 옵션은 커널에게 루트 파일시스템을 'readonly'(읽기전용)으로 마운트 하게 한다. 그래서 파일시 스템 일관성 검사 프로그램(fsck)이 검사 도중 반쯤 저장된 파일이 없다고 가정하여 안전하게 동작하게 한다. 어떤 프로그램이나 프로세스들도 'remoounted' 될때 까지 파일 시스템안에 어느 파일에도 쓸 수 없다. 'ro'는 커널 이미지안에서 rdev 유틸리티로 변경할 수 있는 기본적인 몇몇 부트 인자들중에 한 가지 이다.

3.1.3 The 'rw' Argument

이 인자는 위의 것과 정확히 반대된다. 커널에게 루트 파일시스템을 읽기/ 쓰기로 부트하게 만든다. 기본적으로 루트파일시스템은 읽기/쓰기로 마운트 된다. 읽기/쓰기로 마운트 되어있는 파일시스템에서는 'fsck' 프로그램을 실행하지 마라.

위에서 언급된 이미지 파일안에 저장된 값으로 rdev를 통하여 접근할 수 있는 parameter의 의미로 사용된다.

3.2 Options Relating to RAM Disk Management

아래의 옵션은 RAM 디스크 드라이브를 어떻게 핸들해야 하는가에 대한 것과 관련되어 있다. 이것은 일반적인 인스톨 과정에서의 bootstrapping machine 들이나 루트 파일 시스템에 인스톨 되어야하는 modular 드라이버를 가진 기 계들을 위하여 사용된다.

3.2.1 The 'ramdisk_start=' Argument

커널 이미지가, 압축된 ramdisk 이미지와 함께 플로피 디스크에 존재하는 것을 허용하기 위하여 'ramdisk_start=<offset>명령이 추가되었다. 커널 은 압축된 ramdisk 파일시스템 이미지에는 포함되지 않는다. 이것은 바이오스 가 bootsector를 올리고 그때 커널이 bootstrap 자신을 실행시키기 위하여 block zero의 처음 위치에서부터 저장되어야 하기 때문이다.

주의: 압축안된 ramdisk 이미지를 사용한다면 커널은 ramdisk로 올려진 파일시스템의 일부분이 될 수 있고 플로피 디스크는 LILO나 아니면 두개로 분 리된 압축이미지를 사용여 부트될 수 있다.

만약 boot/root 이 두개의 셋업 디스크 사용한다면(첫번째 디스크는 커널, 두번째 디스크는 ramdisk 용으로) ramdisk는 block zero에서 시작하고 zero offset이 사용될 것이다. 이런 통상적인 방법으로 당신은 전혀 이 명령을 사용할 필요가 없다.

3.2.2 The 'load_ramdisk=' Argument

이 parameter는 커널에게 ramdisk를 올릴것인지 아닌지를 알려준다. 'load_ramdisk=1'는 커널에게 플로피를 ramdisk로 올리라고 하는 것과 같다. 기본적인 값은 zero 이다. 이는 커널이 ramdisk를 올리려 시도하지 않는다는 뜻이다. 새로운 부트 인자들에 대하여 그리고 그들을 어떻게 사용하는지에 대하여 자세히 알고 싶으면 linux/Documentation/ramdisk.txt 파일을 보라 이 파일은 이러한 인자들을 'rdev'를 이용하여 어떻게 설정하고 커널 이미지에 저장할 수 있는지에 대한 설명 또한 포함한다.

3.2.3 The 'prompt_ramdisk=' Argument

이 parameter는 ramdisk 이지미를 가진 플로피 디스크를 삽입하라는 프로프 트를 보일 것인지 아니면 보이지 않을 것인지를 커널에게 알려준다. ramdisk는 커널이 하나의 플로피로서 loading/booting을 둘다 마치는 경

우에는 프롬프트가 필요없다. 따라서 프롬프트가 필요 없는 단일 플로피 형태에서는 'prompt_ramdisk=0'이 사용될 수 있고 디스크를 바꾸어야 하는 두장의 플로피 환경에서는 'prompt_ramdisk=1'가 사용될 수 있다. 이것은 디폴트임으로 일일이 커널에게 알릴 필요는 없다. (역사적 주의: 어떤 사람은 부트 디스크 로 부터 루트 디스크로 교환하는 기회를 주기 위하여 잠시 멈추기 위한 방 법으로 LILO의 'vga=ask' 옵션을 사용한다.)

새로운 부트 인자들에 대하여 그리고 그들을 어떻게 사용하는지에 대하여 자세히 알고 싶으면 linux/Documentation/ramdisk.txt 파일을 보라 이러한 인자들을 'rdev'를 이용하여 어떻게 설정하고 커널 이미지에 저장할 수 있는지에 대한 설명 또한 포함한다.

3.2.4 The 'ramdisk_size=' Argument

Ramdisk가 요구되는 만큼 동적으로 커지는 동안, 모든 가용한 램을 모두 소 비하지 않게 하기 위한 제한 크기가 있는데 기본적으로 대부분의 요구에 만 족할 만큼 큰 4096이 된다. 이 기본 메모리 사용량을 더 크게도 더 작게도 할 수 있다.

새로운 부트 인자들에 대하여 그리고 그들을 어떻게 사용하는지에 대하여 자세히 알고 싶으면 linux/Documentation/ramdisk.txt 파일을 보라 이러한 인자들을 'rdev'를 이용하여 어떻게 설정하고 커널 이미지에 저장할 수 있는지에 대한 설명 또한 포함한다.

3.2.5 The 'ramdisk=' Argument (obsolete)

(주의: 이 인자는 커널 v1.3.47와 그 이후로는 잘 사용되지 않는다. ramdisk 장치를 위하여 사용된 명령어는 위에서 언급하였다.) 이는 ramdisk 장치의 kB 크기를 명세한다. 예를 들면 램 디스크 장치에 올리기 위한 루트 파일 시스템으로 1.44MB 플로피 장치를 사용하기를 원한다면 이렇게 사용한다.

```
ramdisk=1440
```

커널 이미지에 기본적으로 저장되어 있는 커널 부트 parameter는 드물다. 또한 이것은 rdev 유틸리티로 대체되었다.

3.2.6 The 'noinitrd' (initial RAM disk) Argument

버전 2.x 나 그 이후에 새로운 커널들은 루트 파일 시스템이 처음에는 램 디스크에 위치하는 형태를 가진다. 그리고 커널은 램 이미지에 있는 /linuxrc를 실행 시킨다. 이런 방식은 실제 루트 파일 시스템을 마운트 하는데 필요로 하는 모듈들을 올리는데 종종 사용된다. (예를 들면 램 디스크 이미지안에 있는 SCSI 모듈을 올리고 그리고 나서 SCSI 디스크의 실제 루트 파일 시스템을 마운트 하는 것이다.)

실제 'noinitrd' 인자는 커널이 부트된 후에 initrd data에 무슨일이 발생 할지를 결정한다. 이것이 명시 되었을 경우 램 디스크로 변환하는 것 대신에 /dev/initrd를 통한 접근이 가능하다. 이것은 램 디스크가 시스템으로 돌려졌을 경우에 한번만 읽혀 질 수 있다. 램 디스크를 초기화하는 더 많은 세부사항에 대해서는 linux/Documentation/initrd.txt를 참조하라 덧붙혀서 대부분의 LILO, LOADLIN의 최신 버전들은 유용한 정보를 더 가지고 있다.

3.3 Boot Arguments Related to Memory Handling

아래의 인자는 당신 시스템의 물리적, 가상 메모리를 어떻게 리눅스가 검출하고 다룰 것인가를 선택한다.

3.3.1 The ‘mem=’ Argument

이 인자는 두 가지의 목적이 있다: 원래는 설치된 총 메모리(아니면 리눅스에 사용될 메모리의 총 합을 제한하기를 원할 때 사용) 합을 알려주는 것이 그 목적이었다. 두번째 목적은(잘 사용되지는 않지만) mem=nopentium이라는 명령을 사용하여 리눅스 커널에게 성능이 좋은 4MB 페이지 테이블을 사용하 지 말라고 알리는 것이다.

PC 명세서에 정의 되어 있는 본래의 BIOS call은 단지 64MB 까지만을 보고 하도록 설계되어 있다. (물론, 앞을 준비하지 못하여 디스크에서의 1024 실린더만을 인식하게 해 놓은 것 처럼...휴) 리눅스는 부트시에 얼마나 많은 메모리가 설치되어 있는 가를 결정하기 위하여 이 BIOS call을 사용한다. 설치된 램이 64MB 가 넘는다면 당신은 부트 인자로서 리눅스 커널에게 당신이 가지고 있는 메모리의 양을 알려주어야 한다. 여기에 ‘mem=’ 인자를 사 용하는데 있어 리누스가 한 말을 인용한다.

커널은 어떤 ‘mem=xx’ 인자도 허용할 것이다. 그것이 거짓말로 밝혀진다면 커널은 즉시 아니면 조금 있다가 무지막지하게 멈춰버릴 것이다. ‘mem=0x1000000’는 당신이 16MB 메모리를 가지고 있다는 것을 의미한다.

예를 들면 96MB 기계는 ‘mem=0x6000000’ 인자를 사용하여야 할 것이다.

아주 주의:어떤 기계에서는 BIOS가 케싱하는 메모리의 총합을 사용한다. 그래서 실제 96MB 이상의 주소는 불가능하다. 거꾸로 말하면 어떤 칩셋들은 BIOS가 인식하는 물리적 메모리를 메모리 꼭대기 그 이상으로 mapping 시켜 준다는 뜻이다. 예를 들면 메모리의 가장 윗단은 사실상 96MB + 386kB 이상 이 될 수 있기 때문이다. 리눅스에게 실제 존재하는 메모리모다 많은 양이 존재한다고 알린다면 아마 즉시는 아니더라도 결국에는 안 좋은 일이 꼭 발 생할 것이다.

인자는 16진수로만 이루어 져야만 하는 것은 아니고 꼬리말도 대소문자 상관 없이 ‘k’ and ‘M’ 모두 킬로바이트나 메가 바이트를 나타내기 위해 사용될 수 있다. (여기서 ‘k’는 입력할 바이트 값을 10비트 쉬프트 시킨 것이고 ‘M’은 20비트 쉬프트 시킨것이다.) 위에서 한 경고는 아직 유효하다. mem=97920k 사용하여 96MB 기계를 작동시킨다면 mem=98304k를 사용하든 mem=96M를 사용하든 실패할 것이다.

3.3.2 The ‘swap=’ Argument

이 인자는 디스크를 스왑하는데 관련된 virtural memory(VM)를 사용자가 조 절하게 한다. 다음 8개의 parameter를 허용한다.

```
MAX_PAGE_AGE
PAGE_ADVANCE
PAGE_DECLINE
PAGE_INITIAL_AGE
AGE_CLUSTER_FRACT
AGE_CLUSTER_MIN
PAGEOUT_WEIGHT
BUFFEROUT_WEIGHT
```

관심있는 해커들은 linux/mm/swap.c를 읽고 /proc/sys/vm 의 내용을 참조 하는 것이 도움이 될 것이다.

3.3.3 The ‘buff=’ Argument

swap 인자와 유사하게 ‘buff’ 인자는 사용자가 버퍼 메모리 관리에 관련된 어떤 parameter들을 조율하는 것을 가능케 한다. 아래의 6개의 parameter들을 허용한다.

```
MAX_BUFF_AGE
BUFF_ADVANCE
```

```
BUFF_DECLINE
BUFF_INITIAL_AGE
BUFFEROUT_WEIGHT
BUFFERMEM_GRACE
```

관심있는 해커들은 linux/mm/swap.c를 읽고 /proc/sys/vm의 내용을 참조하는 것이 도움이 될 것이다.

3.4 Boot Arguments for NFS Root Filesystem

리눅스는 루트 파일시스템으로 NFS(Network FileSystem)을 사용하는 디스크 없는 워크스테이션을 지원한다. 이들 인자들은 디스크 없는 워크스테이션 에게 자신의 시스템의 정보를 어디서 것을 얻을 수 있는 가를 알려주는데 사용된다. 또한 root=/dev/nfs라는 인자가 요구된다는 것을 주의하라. NFS rootfs을 사용하는데 대한 자세한 정보는 linux/Documentation/nfsroot.txt 파일 안에 있다. 그 파일을 읽어야 한다. 아래의 것들은 그 파일로 부터 직접 가져온 요약일 뿐이다.

3.4.1 The 'nfsroot=' Argument

이 인자는 어떤 디렉토리로 부터 루트 파일시스템을 읽어 들여야 하는가에 관한 NFS 옵션이다. 인자의 형태는 아래와 같다.

```
nfsroot=[<server-ip>]<root-dir>[<nfs-options>]
```

nfsroot parameter가 명령행에서 주어지지 않는다면 기본적으로 '/tftpboot/%s' 가 사용될 것이다. 다른 옵션들은 아래와 같다.

<server-ip> – NFS 서버의 IP 주소를 결정한다. 이 필드가 안 주어진다면 아래에서 나타날 nfsaddrs 변수에 의하여 기본적인 주소로 결정된다. 이런 parameter가 사용된다면 예를 들어 RARP와 NFS를 서로 다르게 사용할 수 있게 한다. 일반적으로 이 란은 비게 된다.

<root-dir> – Name of the directory on the server to mount as root. If there is a '%s' token in the string, the token will be replaced by the ASCII-representation of the client's IP address.

<root-dir> – 서버에서 루트로 마운트 될 디렉토리의 이름. 문자열로 '%s' 토큰이 사용된다면 이 토큰은 NFS 클라이언트의 IP 주소에서 ASCII 형태의 표현식으로 대치된다.

<nfs-options> – 표준 NFS 옵션들. 모든 옵션들은 콤마로 구분된다. 옵션 이 비게 된다면 아래의 기본적인 값들이 사용될 것이다.

```
port        = as given by server portmap daemon
rsize       = 1024
wsize       = 1024
timeo       = 7
retrans     = 3
acregmin    = 3
acregmax    = 60
acdirmin    = 30
acdirmax    = 60
flags       = hard, nointr, noposix, cto, ac
```

3.4.2 The 'nfsaddrs=' Argument

이 부트 인자는 네트워크간 통신에 요구되어지는 여러가지 네트워크 인터페이스 주소들을 설정한다. 이 인자가 주어지지 않는다면 커널은 그 정보를 알기 위하여 RARP나(와) BOOTP를 사용하려 할 것이다.

```
nfsaddrs=<my-ip><serv-ip><gw-ip><netmask><name><dev><auto>
```

<my-ip> – NFS 클라이언트의 IP 주소. 비게 된다면 주소는 RARP나 BOOTP에 의하여 결정된다. <auto> parameter를 사용하는데 있어서 어떤 프로토콜이 사용될 것인가는 커널 환경 설정중에 인에이블 될 것인가에 달려있다. parameter가 비어 있지 않다면 RARP나 BOOTP 어느 것도 사용되지 않을 것이다.

<serv-ip> – NFS 서버의 IP 주소. 클라이언트 주소를 결정하기 위하여 RARP가 사용되고 <serv-ip> 파리미터가 주어진 서버의 응답으로부터 대답 하여 비어 있지 않다면 서로 다른 RARP와 NFS 서버를 사용하기 위하여 이부분에 RARP 서버를 명시해야 한다. (아니면 비어 있는 그대로 두던지), 그리고 위에서 이야기한 nfsroot parameter를 사용하여 당신의 NFS 서버를 적어야 한다. 이 항목이 비어 있다면 서버의 주소는 RARP나 BOOTP의 요구에 의하여 채워질 것이다.

<gw-ip> – 서로 다른 서브넷에 서버가 있을 경우에 게이트웨이 IP주소. 이 항목이 비어 있어서 어느 게이트웨이도 사용되지 않고 서버가 로컬 네트워크에 있다고 가정한다면 BOOTP에 의하여 값을 받을 것이다. <netmask> – 로컬 네트웨크 인터페이스의 netmask. 이것이 비어 있고 BOOTP에 의하여 값이 결정되지 않는다면 netmask는 클라이언트 IP 주소로 부터 유도된다.

<name> – 클라이언트의 이름. 비어 있다면 클라이언트의 IP 주소가 ASCII 의 형태나 BOOTP에서 받은 값으로 사용된다.

<dev> – 사용할 네트워크 장치 이름. 비어 있다면 모든 장치들은 RARP 요청을 위하여 사용되고 처음에는 BOOTP를 위하여 찾는다. NFS를 장치는 받은 RARP나 BOOTP의 응답으로 사용된다. 당신이 하나의 장치를 사용한다면 이란을 비워두는 것이 안전하다.

<auto> – Autoconfiguration을 위하여 사용되는 방법. 이것이 'rarp'나 'bootp'라면 그 프로토콜이 사용된다. 이 값이 둘다 있거나 비어있다면 양쪽 프로토콜은 커널 환경설정중에 양쪽 프로토콜들이 autoconfiguration을 하지 않겠다는 의미의 'none' 을 사용하여 인에이블된다. 이 경우 당신은 그 전에 모든 필드에 필수적인 값들을 지정하여야만 한다. <auto> parameter 는 nfsaddrs parameter로서 홀로 나타날 수 있다. (<auto> 전에 ':' 문자들이 하나도 없이) 이 경우 autoconfiguration이 사용된다. 어떻든, 이 경우에는 'none' 값은 유용하지 않다.

3.5 Other Misc. Kernel Boot Arguments

이들 인자들은 사용자가 내부 커널의 파리미터들을 확실히 조절할 수 있게 한다.

3.5.1 The 'debug' Argument

커널은 중요한 메세지를 printk() 함수를 통하여 통신한다. 중요하다고 생 각되어진 메세지는 printk()가 현재 콘솔에 보여주고 또한 디스크에 log하기 위하여 klogd()에게 넘겨준다. 디스크에 로깅정보를 남기는 것 뿐만아니라 콘솔에 중요한 메세지를 출력하는 이유는 메세지가 디스크에 저장되지 않고 읽어 버리게 되는 안 좋은 상황 (디스크가 망가지거나 하는 경우)에 경우가 있기 때문이다. (디스크에 에러가 발생한 경우에는 이 메세지를 디스크에 저장(logging)할 수 없기 때문:역주)무엇이 중요하고 안 중요한 것인지는 console_loglevel 변수에 의하여 설정 된다. 기본적으로 DEBUG 보다 더 중요한 모든 것은 콘솔에 나타난다. (이들 레벨들은 kernel.h 파일 안에 정의 되어 있다.) 부트 인자로서 디버그를 명세하는 것은 콘솔 레벨을 0으로 할 것이다. 그래서 모든 커널 메세지들이 콘솔에 나타난다.

콘솔 loglevel은 일반적으로 실행시나 klogd() 프로그램을 사용하는 옵션을 통하여 설정이 가능하다. 이것이 어떻게 작동하는지 알기를 원하면 당신의 시스템에 설치된 버전의 메뉴얼 페이지를 점검하라.

3.5.2 The 'init=' Argument

커널은 부트시에 일반적으로 'init' 프로그램으로 시작한다. 이것은 당신의 컴퓨터를 설정하는 getty같은 프로그램을 실행 시키거나 'rc' 스크립트나 그와 비슷한 것들을 실행함으로서 가능하다. 커널은 처음에 /sbin/init 를 찾는다. 그 다음엔 /etc/init를 찾는다. 그리고 마지막 수단으로 /bin/sh 를 사용하려 할 것이다. (/etc/rc도 가능) 예를 들면 당신의 init 프로그램 이 중지되고 부트할 수 없어졌면 당신은 부트시에 직접 쉘을 가동시키기 위 하여 간단히 init=/bin/sh 인자만을 사용하면 된다. 그리고 잘못되었던 프 로그램을 제대로 되돌리면 된다.

3.5.3 The 'no387' Argument

어떤 I387 코프로세서 칩들은 32 비트 프로텍티드 모드에서 나타나는 버그 가 있다. 예를 들면 초기의 ULSI-387 칩들 일부는 부동소수점을 계산하는 동안 단단히 잠금장치를 하게 되는데 이는 분명히 FRSAV/FRRESTOR 명령어가 가진 버그 때문이었다. 'no387' 부트 인자를 사용하는 것은 리눅스가 수치 연산 코프로세서가 있어도 이것을 무시하도록 한다. 물론 수치연산 에뮬레 이션이 가능하도록 컴파일된 커널을 가지고 있어야 한다. 리눅스는 80287을 사용할 수 없기 때문에 이런 경우는 80287 FPU를 사용할 정도로 정말로 오래 된 386 기계를 가지고 있을 때나 가능한 것이다.

3.5.4 The 'no-hlt' Argument

I386(과 이후의 이어지는) CPU 군들은 외부 장치(키보드, 모뎀, 디스크, 등 등.)에서 CPU에게 일을 하도록 깨우는 일이 발생하기 전까지는 CPU에게 아무일도 하지 말라는 의미를 가진 'hlt' 명령어를 가지고 있다. 이는 외부 장치가 CPU를 깨울 때 까지는 좀비와 같이 'low-power' 모드로 진입하는 것을 가능케 한다. 초기의 일부 i486DX-100 칩은 이 'hlt' 명령어에 문제가 있었는데 이들은 'hlt' 명령어가 사용된 후에 오퍼레이팅 모드로 되돌아올 수가 없었다. 'no-hlt' 명령을 사용하여 되돌아오지 못하는 일이 발생하지 않도록 리눅스가 무한 루프 작동을 하게 한다. 비록 CPU 보증서를 가지고 가능한 교환 점을 찾으라는 조언을 받겠지만 이 'no-hlt' 인자는 문제점이 있는 CPU를 가진 사람들도 리눅스를 사용할 수 있게 하여 준다.

3.5.5 The 'no-scroll' Argument

부트시에 이 인자는 Bralille 터미널 사용을 어렵게 하는 스크롤링 기능을 없엔다.

3.5.6 The 'panic=' Argument

커널 패닉과 같이 잘 일어 나지 않는 사건에 대하여서는(즉, 커널에 의하여 검출되고 커널이 큰 소리로 신음하고 모든 것을 정지 시켜야 할 만큼 충분 히 중요한 내부적 에러) 어떤 사람이 스크린에 패닉 메시지에 대하여 주의 를 주고 기계를 리부트 시켜 줄 때 가지 그 자리에 앉아서 기다리는 것이 대부분일 것이다. 그러나 만약 기계가 동떨어진 장소에서 동작하고 있어서 컴퓨터가 자동적으로 리셋되어 원래대로 되돌아 가기를 바란다면 예를 들어 부트시에 'panic=30' 라는 인자를 주면 커널은 커널이 패닉된후에 30 초후 에 스스로 리부트하게 된다. 이 인자의 0값은 디폴트 설정으로서 커널은 영 원히 기다리게 된다.

이 타임아웃 값은 또한 /proc/sys/kernel/panic sysctl을 통하여 읽기가 가능하다.

3.5.7 The 'profile=' Argument

커널 개발자들은 커널이 최대의 효율과 성능을 발휘하게 하기 위하여 어떻게 또는 어디서 CPU 사이클 시간을 소비하는 지를 알 수 있게 하는 옵션을 가능 하게 설정할 수 있다. 이 옵션은 부트시에 profile shift count를 설정한다. 대부분 이 값은 2 로 되는데 디폴트로 당신의 커널을 profiling을 가능하게 컴파일 할 수 있다. 그리고 어느 경우에나 /proc/profile 출력 값을 사용할 수 있는 readprofile.c 같은 툴이 필요하다.

3.5.8 The 'reboot=' Argument

이 인자는 컴퓨터가 리셋되었을 때 리부트 형을 제어한다. (대부분 Control-Alt-Delete을 다루는 /sbin/init에 의하여) 커널 버전 2.0 이 후에는 디폴트로 'warm' 리부트하는 대신에 'cold' (예를 들면 전체 리셋으로 메모리를 체크하는 등등.) 리부트한다. warm 리부트 요청했을때 리부트가 안되는 값싸고 고장난 하드웨어에서 작업을 하는 이후로 디폴트 리부트의 형태는 cold부트가 되었다. 에전처럼 작동하기 위해서는(예를 들면 warm 리부트) reboot=w 나 w로 시작하는 어느 단어를 사용해도 된다.

왜 이런 것을 가지고 당신을 괴롭히는가 하고 의문을 가질 수 있다. 보드에 캐쉬 메모리를 가지고 있는 어떤 디스크 콘트롤러는 warm 리부트를 감지한다. 그리고 캐쉬한 데이터를 디스크에 저장한다. cold 부트에서는 이 카드는 리셋되고 캐쉬 카드 메모리에 저장된어 있던 write-back 데이타를 모두 잃어버린다. 그리고 메모리 체크하는데 시간이 많이 걸리고 SCSI BIOS들은 콜드 부트시에 더 많은 초기화 시간이 걸린다. 이들이 warm 리부트를 사용 하는 좋은 이유이다. (디폴트는 리부트 형태는 cold인데 위의 경우에는 warm 부트를 사용하는 것이 좋다는 뜻:역자)

(write-back: cpu가 저장을 명령을 내리면 즉시 디스크에 저장하지 않고 콘 트롤러가 우선 보드상에 캐쉬 메모리에 저장하는 방식, 역자주:-)

3.5.9 The 'reserve=' Argument

이 인자는 커널의 검색으로부터 I/O port 영역을 보호하기 위해 사용한다. 명령 형태는 다음과 같다.

```
reserve=iobase,extent[,iobase,extent]...
```

어떤 기계에서는 장치 드라이버가 특별한 영역에 있는 장치들을 검사 (auto-probing)하는 것을 막아야 한다. 이는 잘못 설계 되어서 부팅중에 멈추어 버리는 하드웨어이거나(어떤 이더넷 카드처럼) 제대로 인식되지 않는 하드웨어이거나 초기 검색 상태로 변하는 하드웨어이거나 단지 커널의 초기화를 원하지 않는 하드웨어일 경우에 해당한다.

부트 시에 reserve 인자는 검색하지 않을 I/O port 영역을 주소로서 지정해 준다. 영역은 이미 장치가 그 영역(같은 이름으로 예약)을 발견한 것처럼 커널 포트 예약 테이블에 예약한다. 이런 기법은 대부분의 기계에서 필수적인 것은 아니다. 단지 문제가 발생하거나 특별한 경우에만 해당하는 것이다. 지정된 영역의 I/O 포트들은 I/O 공간 영역에서 검색되기에 우선하여 check_region()을 행하는 디바이스 검색으로부터 보호 받는다. NE2000을 다루는 어떤 드라이버의 경우나 그 자신의 장치로 잘못 인식되었을 경우에 사용된다. 제대로 된 디바이스 드라이버는 다른 부트 인자가 명시적으로 검색하라고 명세하지 않는 한 예약된 영역을 검색하지 않는다. 이 말은 예약은 대부분 다른 부트 인자들과 함께 사용된다는 것을 의미한다. 그러므로 지정한 장치를 보호하기위한 예약 영역을 지정한다면 당신은 보통 그 장치를 위하여 명시적인 검색을 지정하여야 한다. 대부분의 드이버들은 명시적인 주소가 주어지지 않는다면 포트 등록 테이블을 무시한다. 예들 들어 부트 라인의 내용이 아래과 같다면

```
reserve=0x300,32  blah=0x300
```

이것은 'blah' 위한 드라이버를 제외한 모든 장치 드라이버들은 0x300-0x31f 주소를 검색하지 않는다.

보통 부트시간에 지정자들은 11개의 parameter가 한계이다. 그러므로 당신 은 단지 예약된 키워드당 5개의 예약된 영역을 지정할 수 있다. 당신이 특별히 복잡한 요청을 하려한다면 여러개의 예약 지정자들 사용해도 잘 동작 한다.

3.5.10 The 'vga=' Argument

이 인자는 실제적인 부트 인자는 아니다. 다른 부트 인자들 처럼 커널에 의하여서가 아니라 LILO에 의하여 해석되는 인자이다. 그러나 이는 매우 일 반적으로 사용되기 때문에 여기에서 언급할 만하다. 이는 또한

rdev -v를 사용하거나 vmlinuz 파일에 비디오 모드를 이용하여 설정될 수 있다. 이것은 설정 코드가 실제 리눅스 커널로 부팅하기전에 디폴트 디스플레이 모드를 변경하기 위한 비디오 바이오스를 사용하는 것을 허락한다. 전형적으로 80x50 모드, 132x44 모드 등이 있다. 이 옵션을 사용하는 좋은 방법은 커널을 부팅하기 전에 당신의 비디오 어댑터를 이용하여 사용가능한 여러 비디오 모드들의 리스트들을 보여주는 vga=ask를 사용하는 것이다. 당신이 사용하기 원하는 번호를 리스트에서 고른 다음에 후에는 (다시 부팅할 경우:역주) 이 값을 'ask'의 위치에 적어 넣을 수 있다. (lilo.conf 파일안에) 더 많은 정보를 알고 싶다면 최근에 커널을 가져올 때 같이 온 linux/Documentation/svga.txt 파일을 보기 바란다.

근래의 커널(2.1 버젼이나 그 이상)은 옵션으로서 모드를 변경하고 비디오 모드를 선택할 수 있는 코드를 가지고 있다. 그래서 이런 기능을 원한다면 이 옵션을 가능하게 설정해야한다.

제 4 절 Boot Arguments for SCSI Peripherals.

이 장은 SCSI 호스터 어댑터와 SCSI 장치들을 설치할때 커널에게 정보들을 전달하기 위하여 사용하는 부트 인자들에 대한 설명이다.

4.1 Arguments for Mid-level Drivers

중간 레벨 드라이버들은 호스트 어댑터를 지정하지 않는 디스크, CR-ROM들, 그리고 테이프 같은 것들을 조정한다.

4.1.1 Maximum Probed LUNs ('max_scsi_luns=')

```
Each SCSI device can have a number of 'sub-devices' contained within
itself. The most common example is one of the new SCSI CD-ROMs that
handle more than one disk at a time.  Each CD is addressed as a
'Logical Unit Number' (LUN) of that particular device. But most
devices, such as hard disks, tape drives and such are only one device,
and will be assigned to LUN zero.
```

각각의 SCSI 장치는 각각 자신의 'sub-devices' 번호를 가질 수 있다. 한번에 하나 이상의 디스크를 조절하는 새로운 SCSI CD-ROM들 중에 하나가 그 일반적인 예이다. 각각의 CD는 구분된 장치의 'Logical Unit Number' (LUN)로서 주소가 주어진다. 그러나 하드 디스크들이나 테이프 드라이버들 같은 대부분의 장치들은 단지 하나의 장치만을 가지고 LUN zero의 값이 할당될 것이다.

이 문제는 잘못된 펌웨어를 가지고 있는 단일 LUN 장치들에서 발생한다. 불완전하게 설계된 SCSI 장치들은(너무 오래 되었거나 운 나쁘게도 너무 새 것이어서) LUNs이 zero값이 아닌 것들에 대해서는 검색되도록 조정할 수 가 없다. 그들은 SCSI 버스가 lock되고 전체 SCSI 버스가 다운되게 할 가능성에 대해서 책임을 져야 한다. 새로운 커널들은 당신이 검색할 LUN들의 최대 숫자를 설정할 환경 옵션을 가지고 있다. 기본적으로 이는 위에서 언급한 문제점을 피하기 위하여 단지 LUN zero 장치에 대하여서만 검색한다.

```
To specify the number of probed LUNs at boot, one enters
'max_scsi_luns=n' as a boot arg, where n is a number between one and
eight. To avoid problems as described above, one would use n=1 to
avoid upsetting such broken devices
```

부트시에 검색될 LUNs의 숫자를 지정하기 위해서는 부트 인자로 'max_scsi_luns=n'을 입력하라. 여기서 n은 1부터 8 까지이다.위에서 언급 한 문제를 피하기 위하여 n=1을 사용한다면 고장난 장치처럼 오동작하는 것은 피할 수 있다.

4.1.2 Parameters for the SCSI Tape Driver ('st=')

부트시에 SCSI 테이프 드라이버의 환경은 아래의 내용을 사용하여 설정될 수 있다.

```
st=buf_size[,write_threshold[,max_bufs]]
```

처음의 2개의 숫자는 kB단위를 지정한다. 디폴트의 buf_size는 32kB 이고 최대 크기는 터무니 없게도 16384kB도 될 수 있다. write_threshold는 테이프에 약속된 버퍼의 값이다. 디폴트는 30kB이다. 버퍼들의 최대 수는 검출된 드라이브의 수에 의하여 변한다.

```
st=32,30,2
```

완전한 세부사항은 커널 소스 트리의 scsi 디렉토리 README.st 파일에서 찾 을 수 있다.

4.2 Arguments for SCSI Host Adapters

이번 장을 위한 일반적인 표현법

iobase – SCSI 호스트가 차지하는 첫 번째 I/O 포트. 16 진수로 지정되며 0x200부터 0x3ff의 영역이 주로 사용된다.

irq – 카드에 사용하도록 설정된 하드웨어 인터럽트. 제대로 된 값은 카드에 의존적이나 일반적으로 5, 7, 9, 10, 11, 12 그리고 15 의 값이 사용된다. 이외의 다른 값들은 IDE 하드 디스크들, 플로피들, 직렬 포트등 같은 평범한 주변장치에 사용된다.

dma – 카드가 사용하는 DMA(Direct Memory Access) 채널. 전형적으로 버스 마스터링(bus-mastering)에 사용된다. PCI와 VLB카드들은 원래부터

bus-master이며 DMA를 요구하지 않는다. 그리고 ISA DMA 채널이다.

scsi-id – SCSI 버스에서 어뎁터가 자신을 구분하기 위하여 사용하는 ID. 어떤 호스트 어뎁터는 이 값의 변경을 허용한다. 대부분은 내부적으로 변하지 않는 값이 지정되어 있다. 디폴트 값은 7이다. 그러나 씨게이트나 TMC-950보드는 고정된 scsi-id를 사용한다.

parity – SCSI 호스트 어뎁터가 모든 정보를 교환하는데 있어 패리티 값을 연결된 장치에 부가하는지 아닌지 결정. 패리티 검사를 가능케하는 하나의 지정자를 지정하고 0의 값은 패리티 검사를 불가능하게 한다. 모든 어뎁터가 부트 인자로서 패리티 검사를 하는 선택권을 지원하는 것은 아니다.

4.2.1 Adaptec aha151x, aha152x, aic6260, aic6360, SB16-SCSI

('aha152x=') aha 숫자는 카드에 관련되어 있고 aic 숫자는 Soundblaster-16 SCSI를 포함 하는 이런 형의 카드들의 실제적인 SCSI 칩들에 관련되어 있다. 커널은 이런 SCSI 호스트들을 검색하기 위해서 설치된 BIOS를 찾아보고 SCSI 호스트들이 존재하지 않다면 당신의 카드를 찾지 못할 것이다. 이 때 당신은 다음과 같은 형태의 부트 인자를 사용해야 한다.

```
aha152x=iobase[,irq[,scsi-id[,reconnect[,parity]]]]
```

이 드라이버가 디버깅이 가능하지 않게 컴파일 되었다면 6번째의 값은 debug level 값으로 설정될 수 있다.

모든 parameter들은 이 장의 처음 부분에서 설명되었고 reconnect 값은 non zero 값을 가지고 있다면 장치를 disconnect/reconnect 할 것에 주의 하라. 사용예는 다음과 같다.

```
aha152x=0x340,11,7,1
```

parameter들은 순서대로 지정되어야 함에 유의하라, 이는 당신이 parity 설 정을 원한다면 iobase, irq, scsi-id 그리고 reconnect 값까지도 지정해야 함을 의미한다.

4.2.2 Adaptec aha154x (‘aha1542=’)

여기에 aha154x 시리즈 카드들이 있다. aha1542 시리즈 카드들은 aha1540 카드들에는 없는 I82077 플로피 콘트롤러를 보드상에 가지고 있다. 이들은 버스마스터링(busmastering) 카드들이고 다른 장치들과 버스를 공유하는데 사용되는 “fairness”를 설정하는 parameter들을 가지고 있다. 부트 인자들은 아래와 같다.

```
aha1542=iobase[,buson,busoff[,dmaspeed]]
```

유효한 iobase 값들은 데체로 0x130, 0x134, 0x230, 0x234,0x330, 0x334 중 에 하나이다. 원래 카드들은 이외에 값들도 허용한다.

buson, busoff 값들은 ISA 버스를 조절하는 카드의 microseconds 수에 관련 되어 있다. 다른 카드들이(ISA Ethernet card 같은) ISA 버스를 접근할 수 있는 기회를 주기 위하여 디폴트 값은 buson은 11us이며 busoff는 4us이다. dmaspeed 값은 DMA(Direct Memory Access) 전송하는데 있어 그 전송율 (MB/s)에 관련되어 있다. 디폴트 값은 5MB/s 이다. 새롭게 개정된 카드들은 소프트웨어적인 설정으로 이 값을 선택하는 것을 허락하고 예전의 카드들은 점퍼로서 가능하다. 이 값은 당신의 마더보드가 그것을 조절할 수 있다고 가정하고 10MB/s 까지 사용할 수 있다. 5MB/s 이상의 값을 사용하려 한다면 주의하여 실험하라.

4.2.3 Adaptec aha274x, aha284x, aic7xxx (‘aic7xxx=’)

이런 보드들은 다음 형태의 인자를 허락한다.

```
aic7xxx=extended,no_reset
```

extended 값이 zero 가 아니라면 커다란 디스크를 사용가능하게 하는 확장된 해석을 가리킨다. no_reset값이 zero가 아니라면 드라이버에게 부트시 호스트 어뎁터를 설정할 때 SCSI 버스를 재설정하지 않도록 알린다.

4.2.4 AdvanSys SCSI Host Adaptors (‘advansys=’)

AdvanSys 드라이버는 AdvanSys SCSI 카드를 위해 검색되는 주소를 4개까지 허용한다. 이들 값들은(사용되었다면) 어떤 방법으로든 EISA나 PCI를 검색 하는 것에 영향을 주지 못한다. 게다가, 드라이버가 디버깅이 가능하게 컴파일 되었다면 디버깅 출력 레벨은 0xdeb[0-f] 파라미터를 줌으로 인하여 설정이 가능하다. 0-f 의 값은 디버깅 메시지 레벨을 16단계중에 하나로 설 정하게 한다.

4.2.5 Always IN2000 Host Adaptor (‘in2000=’)

다른 SCSI 호스트 부트 인자와 다르게, IN2000 드라이버는 자신의 대부분의 정수 인자앞에 ASCII 문자열을 붙여 사용한다.

ioport:addr – addr는 카드(일반적으로 ROM 보다 아래주소)의 IO 주소.

noreset – No optional args. 부트시에 SCSI 버스가 재설정 되는 것을 막는 다.

nosync:x – x 는 비트 마스크로서 처음 7비트들은 7개의 가능한 SCSI 장치 들과 연관되어 있다.(장치 #0번은 bit 0으로 하는 등). 그 장치가 동시에 작동하는 것을 막기위하여 이 비트를 설정한다.

period:ns – ns는 SCSI 데이터의 전송 주기의 최소 나노세컨트 값이다. 디폴트로 500 이며; 250에서 1000까지의 값이 가능하다.

disconnect:x – 결코 연결을 끊지 않으려면 x = 0으로 항상 연결하려면 x = 2로. x = 1은 'adaptive' 연결 종료를 한다. 이는 디폴트이며 일반적으로 최선의 선택이다. debug:x 'DEBUGGING_ON' 이 정의 되어 있다면 x 는 출력하기 위한 디버그 출력의 여러 가지 형태를 일으키는 비트마스크이다. in2000.h 안에 정의된 DB_xxx를 보라.

proc:x – 'PROC_INTERFACE' 가 정의 되어 있다면, x는 /proc 인터페이스가 어떻게 작동하고 무엇을 할 것인가를 결정하는 비트마스크이다. - in2000.h 안에 정의된 PR_xxx를 보라. 사용 예들이 아래에 열거되었다.

```
in2000=ioport:0x220,noreset
in2000=period:250,disconnect:2,nosync:0x03
in2000=debug:0x1e
in2000=proc:3
```

4.2.6 AMD AM53C974 based hardware ('AM53C974=')

다른 드라이버들과 달리, 이것은 i/o, IRQ나 DMA 채널과 통신하기 위하여 부트 파리미터들을 사용하지 않는다. (AM53C974가 PCI 장치가 되면서부터 그렇게 할 필요가 없어졌다.) 대신에, parameter들은 전송 모드들이나 호스트와 지정된 장치간에 사용되는 전송율을 알리는데 사용되곤 한다. 예제에 잘 나타나 있다.

```
AM53C974=7,2,8,15
```

이 예는 다음과 같이 해석한다: 'SCSI-ID 7을 가지고 있는 콘트롤러와 SCSI-ID 2을 가지고 있는 장치와의 통신을 위하여 전송율은 최대 8MHz 동기 모드로 한다. 15바이트 오프셋은 협상될 것이다. 좀더 세부적인 사항은 linux/drivers/scsi/README.AM53C974 파일 안에 있다.

4.2.7 BusLogic SCSI Hosts with v1.2 kernels ('buslogic=')

예전 커널에서 buslogic 드라이버는 단지 하나의 parameter만을 허용한다. 이는 I/O 베이스 주소이다. 이 주소는 다음의 정확한 값들 중에 하나이어야 한다: 0x130, 0x134, 0x230, 0x234, 0x330, 0x334.

4.2.8 BusLogic SCSI Hosts with v2.x kernels ('BusLogic=')

버전 2.x 커널에서는 BusLogic 드라이버는 많은 parameter들을 허용한다. (위의 경우 BusLogic에서 B 와 L 자가 대문자인 것에 주의하라.) 아래에 커널 버전 2.0에 포함되어 있는 Leonard N. Zubkoff'의 드라이버에 대한 자세한 설명이 있다.

BusLogic 드라이버를 위해서 커널 명령라인의 내용은 "BusLogic=" 이라는 드라이버 구분자들을 포함한다. 이 구분자는 콤마로 구분되는 정수와 문자열들로 이루어진다. 각각의 명령 라인들은 하나의 BusLogic 호스트 어뎁터에 작용한다. 다중 명령 라인들은 여러개의 Buslogic 호스트 어댑터를 가지고 있는 시스템에서도 사용될 것이다.

첫 번째 정수는 호스트 어뎁터가 위치하는 I/O 주소를 지정한다. 지정하지 않는다면 디폴트로 0이 사용되며 이것은 디폴트 검색하는 동안 처음으로 찾는 BusLogic 호스트 어뎁터의 주소가 사용되게 한다. 만약 I/O 주소 parameter가 명령행 라인으로 주어진다면 디폴트 검색은 생략된다.

두 번째 정수는 Tagged Queuing을 지원하는 장치들의 사용하기 위한 Tagged Queue Depth를 지정한다. Queue Depth는 실행시에 동시에 존재할 수 있는 SCSI 명령들의 숫자이다. 지정되지 않으면, 디폴트로 0이 사용되며 이것은 어뎁터의 Total Queue Depth나 원하는 장치들에서 발견한 숫자, 형, 속도, 용량

에 기반하여 자동적으로 결정된 값이 사용된다. ISA Bounce 버퍼들을 요구하는 호스트 어뎁터를 위해서 Tagged Queue Depth는 DMA Bounce Buffer memory 지나친 할당을 피하기 위하여 자동적으로 BusLogic_TaggedQueueDepth_BB를 설정한다. Tagged Queue Depth를 지원하지 않는 지정된 장치들은 BusLogic_UntaggedQueueDepth의 Queue Depth를 사용한다.

세 번째 정수는 초 단위로 버스 점유 시간을 지정한다. 이는 SCSI버스 재설 정을 가리키는 호스트 어뎁터의 하드 리셋와 어떤 SCSI 명령들을 넘겨주는 동안에 기다리는 시간의 총 합계이다. 지정되지 않으면 디폴트로 0이 사용 되고 이는 BusLogic_DefaultBusSettleTime의 값을 사용한다는 의미이다. 네 번째 정수는 Local Options을 지정한다. 지정되지 않으면 디폴트로 0이 된다. Local Options은 단지 지정된 호스트 어뎁터에만 적용된다는 점에 주의 하라.

다섯 번째 정수는 Global Options을 지정한다. 지정되지 않으면 디폴트로 0 이 된다. Global Options은 모든 호스트 어뎁터에 걸쳐 적용된다는 점에 주의 하라. 문자열 옵션들은 Tagged Queuing과 에러 정정 호스트 어뎁터 검색을 조절하 는데 사용된다. Tagged Queuing 지정자는 "TQ" 로 시작하고 지정된 장치들이 그것을 지원 하는지를 허가하는지 안 하는지를 명시적으로 지정하는 것을 허용한다.

다음의 지정 옵션들이 사용 가능하다: TQ:Default – Tagged Queuing은 BusLogic 호스트 어뎁터의 펌웨어 버전이나 다중 명령을 queuing 하는 것을 가능하게하는 Tagged Queue Depth에 기반하여 제한된다.

TQ:Enable – Tagged Queuing은 호스트 어뎁터 펌웨어 버전에 기반하도록 강요되는 어느 제한도 무시하는 호스트 어뎁터에서 모든 대상 장치들에 대하여 작동가능하게 설정될 것이다.

TQ:Disable – Tagged Queuing은 이 호스트 어뎁터에 연결된 모든 장치들에 대하여 작동하지 않는다.

TQ:<Per-Target-Spec> – 각각의 연결된 장치에 대하여 서로 다르게 Tagged Queuing이 조절될 것이다. <Per-Target-Spec>는 "Y", "N", 그리고 "X" 중에 하나로 이루어 진다. "Y"는 Tagged Queuing을 가능하게 하고 "N"는 이를 불가능하게 하고 "X"는 펌웨어 버전에 기반한 디폴트를 작동을 허용한다. 첫 번째 문자는 연결된 장치 0에 관련되어 있고 두 번째 는 장치1에, 등등; "Y", "N", 그리고 "X"의 나열이 모든 연결된 장치들에게 미치지는 않는다면 지정되지 않는 문자들은 "X" 로 가정된다.

명시적인 Tagged Queuing은 문제를 발생 시킨다는 것에 주의하라; 위의 방법은 Tagged Queuing이 제대로 구현되지 않은 장치를 디스에이블 시키는데 우선 사용된다.

에러 복구 방법의 지정은 "ER:"로 시작하며 명시적으로 제대로 성공하지 못한 SCSI 명령에 의하여 Reset-Command가 불리었을 때 해야할 에러 복구 행동을 지정한다.

아래의 옵션들이 가능하다:

ER:Default – 에러 복구를 Hard Reset과 SCSI 서브시스템의 권장사항 옵션 인 Bus Device Reset 사이에서 선택할 것이다.

ER:HardReset – 에러 복구 SCSI 버스 리셋을 유발시키는 호스트 어뎁터의 Hard Reset를 선택할 것이다.

ER:BusDeviceReset – 에러 복구는 에러를 유발시키는 개개의 지정 장치에 Bus Device Reset 메시지를 전송한다.

ER:None – 에러 복구는 금지될 것이다. 이 옵션은 SCSI 버스 Reset이나 버스 장치 reset이 대상 장치를 완전히 복구 불가능하게 할 경우에만 선택되어 진다.

ER:<Per-Target-Spec> – 에러 북구는 각각의 장치마다 개별적으로 제어될 것이다. <Per-Target-Spec>는 "D","H","B", 그리고 "N" 문자들로 이루어 진 다. "D"는 디폴트를 선택하고. "H"는 Hard Reset을 "B"는 버스 장치 리셋을 "N"은 None를 선택한다. 첫 번째 문자는 0번 대상 장치와 관련되고 두 번째 문자는 1번 대상 장치를 ......; "D", "H", "B", 그리고 "N" 문자들의 나열 이 모든 가능한 대상 장치를 커버할 수 없다면 열거하지 문자들은 "D" 로 가정된다. 호스트 어뎁터 검색은는 아래의 문자열들로 구성된다.

NoProbe – 어떤 종류의 검색도 하지 않는다. 그러므로 어느 BusLogic 호스 트 어뎁터도 발견되지 않을 것이다.

NoProbeISA – 표준 ISA I/O 주소들은 검색되지 않는다. 그러므로 단지 PCI 호스트 어뎁터들이 검출될 것이다.

NoSortPCI – PCI 스캐닝 순서를 위해 버스나 장치의 번호를 사용하는 AutoSCSI의 어떤 설정도 무시하며 PCI 바이오스에 제공되는 순서로 PCI 호스트 어뎁터들이 열거될 것이다.

4.2.9 EATA SCSI Cards ('eata=')

최근의 버전 2.0 커널들에 있어서, EATA 드라이버들은 검색되는 i/o 주소들 을 지정하는 부트 인자를 허용한다. 다음은 이것의 형태다:

```
eata=iobase1[,iobase2][,iobase3]...[,iobaseN]
```

드라이버는 여기에 열거되는 순서로 주소들을 검색할 것이다.

4.2.10 Future Domain TMC-8xx, TMC-950 ('tmc8xx=')

이런 SCSI 호스트을 위한 검색 코드는 우선 설치된 BIOS를 찾아보고 없다면 당신의 카드를 찾지 못할 것이다. 아니면 당신의 BIOS의 기호 문자열을 찾을 수 없어 인식되지 않는다면 어느 경우에도 당신은 다음 형식의 부트 인자를 사용해야만 할 것이다.

```
tmc8xx=mem_base,irq
```

mem_base 값은 카드가 사용하는 I/O 영역에 사상되는 메모리의 값이다. 이는 일반적으로 아래의 값들 중에 하나이다: 0xc8000, 0xca000, 0xcc000, 0xce000, 0xdc000, 0xde000.

4.2.11 Future Domain TMC-16xx, TMC-3260, AHA-2920 ('fdomain=')

드라이버는 이런 카드들을 알려진 롬 바이오스 기호 리스트에 따라서 검출한다. 알려진 바이오스의 전체 리스트에 대해서는 linux/drivers/scsi/fdomain.c를 보기 바란다. 이 파일의 앞 부분에는 많은 정보가 있다. 당신의 BIOS가 드라이버를 모른다면 아래의 형식의 오버라이 드를 사용할 수 있다.

```
fdomain=iobase,irq[,scsi_id]
```

4.2.12 IOMEGA Parallel Port / ZIP drive ('ppa=')

이 드라이버는 IOMEGA ZIP 드라이버에 들어있는 IOMEGA 병렬 포트 SCSI 어뎁터를 위한 것이다. 이건 오리지날 IOMEGA PPA3 장치와도 동작한다. 이 드라이버를 위한 부트 인자의 형식이다.

```
ppa=iobase,speed_high,speed_low,nybble
```

부가적으로 지정된 값들인 iobase를 제외한 모든 값들에 대하여 당신이 3개의 옵션 parameter들을 가지고 변경하려 한다면 그들이 무엇을 제어하는지에 대한 세세한 사항을 linux/drivers/scsi/README.ppa에서 읽으라는 충고를 받을 것이다.

4.2.13 NCR5380 based controllers ('ncr5380=')

당신 보드에 따라서, 5380 은 i/o 사상하거나 메모리 사상을 할 것이다. (일반적으로 0x400 아래의 주소는 i/o 사상을 의미한다. 그러나 PCI나 EISA 하드웨어는 0x3ff 이상의 i/o 주소들을 사용한다.) 어느 경우든, 당신은 주소, IRQ 값, DMA 채널 값을 지정한다. i/o 사상 카드의 한 예는 다음이 될 것이다.: ncr5380=0x350,5,3. 카드가 인터럽트를 사용하지 않는다면 255의 IRQ 값은 인터럽트를 불가능하게 한다. 254의 IRQ 값은 자동 검색을 의미한 다. 더 자세한 것은 linux/drivers/scsi/README.g_NCR5380 파일에서 볼 수 있다.

4.2.14 NCR53c400 based controllers (‘ncr53c400=’)

일반적인 53c400 지원은 위에서 언급된 일반적인 5380의 지원과 같은 드라 이버를 사용한다.부트 인자는 53c400에서 사용되지 않는 DMA 채널을 제외하 면 동일하다.

4.2.15 NCR53c406a based controllers (‘ncr53c406a=’)

이 드라이버는 다음 형태의 부트 인자를 사용한다:

```
ncr53c406a=PORTBASE,IRQ,FASTPIO
```

IRQ와 FASTPIO parameter들은 옵션이다. 0의 인터럽트 값은 인터럽트 사용을 금지 시킨다. FASTPIO 마라미터를 위해 사용하는 1의 값은 single-byte inb나 outb 명령어 대신에 insl이나 outsl 명령어의 사용을 가능하게 한다. 드라이버는 컴파일 시의 옵션으로 DMA를 사용할 수 있다.

4.2.16 Pro Audio Spectrum (‘pas16=’)

PAS16은 NCR5380 SCSI 칩을 사용하고, 새로운 모델들은 점퍼 없는 환경설정 을 지원한다. 부트인자는 다음의 형식이다:

```
pas16=iobase,irq
```

단지 차이점은 255의 IRQ값을 지정할 수 있는 것이고 이는 드라이버에게 비 록 성능은 떨어지더라도 인터럽트를 사용하지 말라고 말할 것이다. iobase는 일반적으로 0x388이다.

4.2.17 Seagate ST-0x (‘st0x=’)

이런 SCSI 호스트을 위한 검색 코드는 우선 설치된 BIOS를 찾고 없다면 당신의 카드를 찾지 못할 것이다. 아니면 당신의 BIOS의 기호 문자열이 찾을 수 없어 인식되지 않는다면 어느 경우에도 당신은 다음 형식의 부트 인자를 사용해야만 할 것이다.

```
st0x=mem_base,irq
```

mem_base 값은 카드가 사용하는 I/O 영역에 사상되는 메모리의 값이다. 이는 일반적으로 아래의 값들 중에 하나이다: 0xc8000, 0xca000, 0xcc000, 0xce000, 0xdc000, 0xde000.

4.2.18 Trantor T128 (‘t128=’)

이런 카드들은 또한 NCR5380 칩에 기반한고 아래의 옵션들을 허용한다.

```
t128=mem_base,irq
```

mem_base의 정확한 값들은 다음과 같다: 0xcc000, 0xc8000, 0xdc000, 0xd8000.

4.2.19 Ultrastor SCSI cards ('u14-34f=')

이 카드에는 두 개의 비 의존적인 드라이버가 있는데 즉, CONFIG_SCSI_U14_34F 는 u14-34f.c를 사용하고 CONFIG_SCSI_ULTRASTOR는 ultrastor.c를 사용한다. u14-34f는 다음 형태의 부트 인자들을 허용한다 (최근 커널 버전 2.0에서).

```
u14-34f=iobase1[,iobase2][,iobase3]...[,iobaseN]
```

드라이버는 나열되는 순서대로 주소들을 검색할 것이다.

4.2.20 Western Digital WD7000 cards ('wd7000=')

이 드라이버는 wd7000을 위하여 알려진 BIOS ROM 문자열을 찾고 약간 떨어지는 표준 환경 설정에 대하여 알고 있다. 당신의 카드에 정확한 값이 없거나 당신이 BIOS 버전을 알지 못한다면 당신은 아래의 형태로 부트 인자를 사용할 수 있다:

```
wd7000=irq,dma,iobase
```

4.3 SCSI Host Adapters that don't Accept Boot Args

현재 아래의 SCSI 카드들은 부트시에 어느 parameter들도 사용이 불가능하다. 요구된다면 드라이버 자체를 편집하여 직접적인 값을 줄 수 있다.

```
Adaptec aha1740 (EISA probing),
NCR53c7xx,8xx (PCI, both drivers)
Qlogic Fast (0x230, 0x330)
Qligic ISP (PCI)
```

제 5 절 Hard Disks

이 절은 표준 MFM/RLL, ST-506, XT 그리고 IDE 디스크 드라이버 장치들에 관련된 모든 부트 인자들을 나열한다. IDE나 generic ST-506 HD 드라이버는 모두 'hd=' 옵션을 허용한다.

5.1 IDE Disk/CD-ROM Driver Parameters

IDE 드라이버는 많은 parameter들을 허용한다. 이들은 디스크 geometry specifications로부터 규정되며 좀 더 이것은 향상되거나 고장난 controller 칩을 지원하기 위함이다. 아래의 것들은 모든 가능한 부트 인자들의 모음이다. 좀더 자세한 내용은 당신이 실제로 linux/Documentation 디렉토리에 있는 ide.txt 파일을 참고 하여라. 이로부터 아래의 요약문이 발췌되었다.

```
 "hdx=" 는 "hdc" 와 같은 예처럼 모든 "x"  는 "a" 로부터 "h" 로 인식된다.
"idex=" 는 "ide1"과 같은 예처럼 모든 "x"  는 "0" 로부터 "3" 로 인식된다.
 "hdx=noprobe"          : 드라이버가 존재하면 그것들을 검색하지  않는다.
 "hdx=none"             : 드라이버가 없으면 cmos를 무시하고 검색하지
                          않는다.
 "hdx=nowerr"           : 이 드라이버에 WRERR_STAT 비트를 무시한다.
 "hdx=cdrom"            : 드라이버가 있고 그것이 cdrom 드라이브이다.
```

```
"hdx=cyl,head,sect"    : 드라이버가 있고 지정값들을 가지고 있다.
"hdx=autotune"         : 이 드라이버에서만 가능하다면 빠른 PIO mode
                         지원을 위하여 인터페이스 속도를 조절하기 위한
                         시도를 할 것이다. 모든 칩형태에서 지원 되지는
                         않으며 옛날의 IDE 드라이버나 이상한 IDE
                         드라이버들에서 상당한 문제를 발생할 수도 있다.
"idex=noprobe"         : 인터페이스를 검색하거나 사용하기 위한 시도를
                         하지 않는다.
"idex=base"            : 일반적으로 "base"가 0x1f0 이나 0x170의 값을
                         가지며 "ctl" 이 "base"+0x206 으로 가정되는 주소에서
                         인터페이스를 검색한다.
"idex=base,ctl"        : base와 ctl을 지정
"idex=base,ctl,irq"    : base, ctl 그리고 irq 번호를 지정
"idex=autotune"        : 이 드라이버에서만 가능하다면  빠른 PIO mode
                         지원을 위하여 인터페이스 속도를 조절하기 위한
                         시도를 할 것이다. 모든 칩 형태가 지원 되지는
                         않으며 옛날의 IDE 드라이버나 이상한 IDE
                         드라이버들에서 상당한 문제를 발생할 수도 있다.
"idex=noautotune"      : 드라이버는 인터페이스 속도를 조절하려하지
                         않을 것이다. 이는 cmd640을 제외한 대부분의
                         칩셋에서 디폴트이다.
"idex=serialize"       : idex와 ide(x^1)에서 연산을 중복하지 않음
```

아래의 것들은 base,ctl ports들이 대체되지 않은 디폴트 상황인 ide0 에서만 적합하다.

```
"ide0=dtc2278"         : DTC2278 인터페이스를 검색하고 지원함
"ide0=ht6560b"         : HT6560B 인터페이스를 검색하고 지원함
"ide0=cmd640_vlb"      : CMD640 칩을 가지고 있는 VLB 카드들에서
                         요구된다. (자동적으로 선택되는 PCI를 위한
                         것이 아님)
"ide0=qd6580"          : qd6580 인터페이스를 검색하고 지원함
"ide0=ali14xx"         : ali14xx 칩셋들을  검색하고 지원함
                         (ALIM1439/M1445)
"ide0=umc8672"         : umc8672 칩셋들을 검색하고 지원함
```

모든 예외는 ”BAD OPTION” 메시지와 함께 거절된다.

5.2 Standard ST-506 Disk Driver Options (‘hd=’)

표준 디스크 드라이버는 IDE 드라이버와 비슷한 디스크들을 위한 지리적인 인자들을 허용할 수 있다. 이들은 단지 세가지의 값들(C/H/S)을 기대한다. 더 많거나 더 적다면 무시된다. 인자로서 단지 ‘hd=’ 만을 허용하는데 예를 들면 ‘hda=’, ‘hdb=’ 등등이며 여기서는 정확한 값은 아니다. 그 형태는 아 래에 있다:

```
hd=cyls,heads,sects
```

두 개의 디스크가 설치되었다면 위의 문장이 두 번째 디스크의 지리적 parameter와 함께 반복된다.

5.3 XT Disk Driver Options ('xd=')

당신이 운이 안좋게도 옛날의 8bit 카드들 중에 하나를 사용하고 있다면 이들 카드를 검색하는 코드들은 우선 설치된 BIOS를 찾아보고 없다면 당신의 카드를 찾지 못할 것이다. 그리고 당신의 BIOS의 기호 문자열을 찾을 수 없어 인식되지 않는다면 어느 경우에도 당신은 다음 형식의 부트 인자를 사용해야만 할 것이다.

```
xd=type,irq,iobase,dma_chan
```

type은 카드의 개개의 제조업자를 지정하는데 아래와 같다: 0=generic; 1=DTC; 2,3,4=Western Digital, 5,6,7=Seagate; 8=OMTI. 같은 제조업자로 부터의 여러 type 간의 차이점은 단지 발견시에 사용되는 BIOS 문자열인데 이는 type이 지정된다면 사용되지 않는다. xd_setup() 함수는 값들을 검사하지 않고는, 당신이 모든 값을 입력하였을 것이라 가정한다. 그러므로 꼭 입력하라. 여기에 'default' XT 콘트로러 parameter를 가지는 BIOS에서 사용불가능하거나 옮겨진 WD1002 콘트롤러의 한 예가 있다.

```
xd=2,5,0x320,3
```

제 6 절 CD-ROMs (Non-SCSI/ATAPI/IDE)

이절은 CD-ROM 장치에 관계한 모든 가능한 부트 인자들을 보인다. 이는 SCSI와 IDE/ATAPI CD-ROM들을 포함하지 않는다. CD-ROM 들의 대부분은 당신이 읽어야할 문서 파일들을 가지고 있고 그들은 모두 가까운 곳 linux/Documentation/cdrom에 있음을 주의하라.

6.1 The Aztech Interface ('aztcd=')

이 카드를 위한 문법:

```
aztcd=iobase[,magic_number]
```

당신에 magic_number를 0x79로 설정하였다면 드라이버는 알려지지 않는 펌 웨어 버전에서 어떻게 해서든 실행하려고 노력할 것이다. 모든 다른 값들은 무시된다.

6.2 The CDU-31A and CDU-33A Sony Interface ('cdu31a=')

이 CD-ROM 인터페이스는 Pro Audio Spectrum 사운트 카드 일부에서 발견되 고 이와 다르게 Sony는 인터페이스 카드가 공급한다. 형태는 다음과 같다:

```
cdu31a=iobase,[irq[,is_pas_card]]
```

IRQ 값을 0으로 지정하여 드라이버에게 하드웨어 인터럽트가 지원되지 않는다는 것을 알린다(어떤 PAS 카드들에서 처럼). 당신이 인터럽트를 지원한다면 드라이버의 CPU 사용량을 줄여 사용하여야 한다. 'is_pas_card' 는 Pro Audio Spectrum 카드를 사용한다면 'PAS'를 입력하고 그렇지 않다면 절대 지정해서는 안된다.

6.3 The CDU-535 Sony Interface ('sonycd535=')

CD-ROM 인터페이스는 다음과 같다:

```
sonycd535=iobase[,irq]
```

하나의 IRQ 값을 지정하기를 바란다면 'placeholder' 값에 기반한 I/O base 값으로 0이 사용된다.

6.4 The GoldStar Interface ('gscd=')

CD-ROM 인터페이스는 다음과 같다:

```
gscd=iobase
```

6.5 The ISP16 Interface ('isp16=')

CD-ROM 인터페이스는 다음과 같다:

```
isp16=[port[,irq[,dma]]][[,]drive_type]
```

irq나 dma 값에 0을 사용하는 것은 이를 사용치 않겠다는 것을 의미한다. drive_type을 위하여 허가된 값들은 noisp16, Sanyo, Panasonic, Sony, 그리고 Mitsumi이다. noisp16을 사용하는 것은 드라이버 전체를 사용치 못하게 만든다.

6.6 The Mitsumi Standard Interface ('mcd=')

CD-ROM 인터페이스는 다음과 같다:

```
mcd=iobase,[irq[,wait_value]]
```

wait_value는 드라이버에 문제가 있는 사람들을 위한 내부 타이머 값으로 사용된다. 컴파일 시에 DEFINE의 의존하여 구현되기도 하고 안되기도 한다.

6.7 The Mitsumi XA/MultiSession Interface ('mcdx=')

최근에 이런 '실험적인' 드라이버는 설정 함수를 가지고 있다, 그러나 어떤 parameter도 아직 구현되지 않았다(커널 버전 1.3.15 현재). 이는 위와 같 은 하드웨에 있어서 그렇다. 그러나 드라이버는 확장 되어져 왔다.

6.8 The Optics Storage Interface ('optcd=')

이런 형의 카드를 위한 문법은:

```
optcd=iobase
```

제 7 절 The Phillips CM206 Interface ('cm206=')

이런 형의 카드를 위한 문법은:

```
cm206=[iobase][,irq]
```

드라이버는 IRQ 값이 3에서 11사이일 것이고 I/O ports 값은 0x300에서 0x370사이일 것이라고 가정한다. 그래서 당신은 어떤 순서든 하나나 양쪽 숫자를 모두 지정할 수 있다. 또한 자동검색을 하기 위하여 'cm206=auto'도 허용한다.

7.1 The Sanyo Interface ('sjcd=')

이런 형의 카드를 위한 문법은:

```
sjcd=iobase[,irq[,dma_channel]]
```

7.2 The SoundBlaster Pro Interface ('sbpcd=')

이런 형의 카드를 위한 문법은:

```
sbpcd=iobase,type
```

type의 자리에는 아래의 문자열(대소문자 구분)중에 하나가 온다: 'SoundBlaster', 'LaserMate', 이거나 'SPEA'. I/O base는 사운드 부분이 아닌 CD-ROM 인터페이스의 것이다.

제 8 절 Other Hardware Devices

위의 구분에 적합하지 않는 다른 어느 장치들은 여기에서 하나로 총괄하여 말한다.

8.1 Ethernet Devices ('ether=')

서로 다른 드라이버는 서로 다른 parameter들을 사용한다. 그러나 최소한 IRQ, I/O port base값, 이름은 공유한다. 이들의 가장 일반적인 형태를 아래에 보인다:

```
ether=irq,iobase[,param_1[,param_2,...param_8]]],name
```

첫 번째 비수치 인자는 이름으로 분리된다. param_n 값들은(적절하다면) 일반적으로 각각의 서로 다른 카드나 드라이버에 서로 다른 의미를 가진다. 전형적인 param_n 값들은 공유 메모리 주소나 인터페이스의 선택, DMA 채널 같은 것을 지정하는데 종종 사용된다. 이 parameter의 대부분의 일반적인 사용은 디폴트로 ethercard 하나는 검색 한 후에, 강제로 두 번째 ethercard를 검색하는데 있다. 이는 다음과 같이 간단히 이루어질 수 있다.

```
ether=0,0,eth1
```

위의 예제에서 IRQ나 I/O base의 값이 0이면 드라이버에게 자동 검색하라는 것을 알리는 것임을 주의하라. IMPORTANT NOTE TO MODULE USERS: 당신이 실행시 loadable modules를 사용 하지 않는다면 위의 예에서 두 번째 카드를 위한 검색은 강요되지 않을 것이다 (대신에 컴파일때에 강요할 수 있다). 대부분의 리눅스 배포판들은 modular 드라이버들의 큰 선택과 결합된 기본적인 골격의 커널을 사용한다. ether= 는 단지 커널에게 직접 컴파일된 드라이버들에게만 적용된다.(이들 인자는 ethernet 드라이버들과 함께 컴파일된 커널에게만 효력이 있다:역주) Ethernet-Howto는 여러 카드들을 사용하는데 관련되거나 카드나 드라이버가 param_n 값들이 어디서 사용되는지를 구현하는 것을, 지정하는데에 관련된 완벽하고 보다 많은 문서를 가지고 있다. 흥미로운 독자들은 좀더 완벽한 정보를 얻기 위하여 관련된 절들을 참조하여라 Ethernet-HowTo *sunsite.unc.edu/mdw/HOWTO/Ethernet-HOWTO.html* `<<http://sunsite.unc.edu/mdw/HOWTO/Ethernet-HOWTO.html>`

8.2 The Floppy Disk Driver ('floppy=')

많은 플로피 드라이버 옵션들이 있고 이들은 모두 linux/drivers/block안에 있는 README.fd 안에 나열되어 있다. 이 정보는 그 파일에서 직접 뽑아낸 것이다.

floppy=mask,allowed_drive_mask 허가된 드라이브를 마스크 하기 위하여 bitmask를 설정한다. 디폴트로 각각 플로피 콘트롤러의 장치 0번과 1번만이 허용된다. 이는 장치 2나 3을 접근 하여 키보드를 엉망으로 만드는 비표준 하드웨어(ASUS PCI motherboards)에서 확실히 작동한다. 이 옵션은 cmos 옵션에 의하여 잘 사용되지 않는다.

floppy=mask,allowed_drive_mask floppy=all_drives 모든 드라이브로 허가하기 위해 bitmask를 설정한다. 플로피 콘트롤러에 연결된 두 개 이상의 드라이브들을 가지고 있다면 이를 사용하라.

floppy=mask,allowed_drive_mask floppy=asus_pci 장치 0과 1만을 가능케 하기 위해 bitmask를 설정한다. (기본 설정)

floppy=mask,allowed_drive_mask floppy=daring floppy=mask,allowed_drive_mask 플로피 드라이버에게 잘 동작하는 콘트롤러를 가지고 있다고 알린다. 이는 좀더 효율적이고 부드럽게 작동하도록한다. 그러나 어떤 콘트롤러에서는 실 패할 것이다. 이것은 어떤 동작의 속도를 빠르게 하여준다.

floppy=0,daring 플로피 드라이버에게 플로피 콘트롤러는 조심스럽게 사용된다고 알린다.

floppy=one_fdc 플로피 드라이버에게 단지 하나의 플로피 콘트롤러만을 가지고 있다고 알린 다(기본 설정).

floppy=two_fdc or floppy=address,two_fdc 플로피 드라이버에게 두 개의 플로피 콘트롤러들을 가지고 있다고 알린다. 두 번째 플로피 콘트롤러는 address 위치에 있다고 가정되는데 address가 주어지지 않는다면 0x370번지로 가정된다.

floppy=thinkpad 플로피 드라이버에게 Thinkpad를 가지고 있다고 알린다. Thinkpad들은 디스크 변환 라인을 위하여 역 관례를 사용한다.

floppy=0,thinkpad 플로피 드라이버에게 Thinkpad를 안 가지고 있다고 알린다.

floppy=drive,type,cmos 드라이브의 cmos 형을 type으로 설정한다. 게다가, 이 드라이브는 비트 마스크를 허락한다. 이는 당신이 두 개 이상의 플로피 드라이브를 가지고 있거나(물리적 cmos 에서는 단지 두 개 만이 서술된다.) BIOS가 비표준 CMOS 형을 가지고 있을 경우에 유용하다. 처음의 두 드라이브들을 CMOS에서 0으로 만드는 것은 플로피 드라이버가 이들 드라이브들을 위해 물리적인 cmos를 읽도록 만든다.

floppy=unexpected_interrupts 예상치 않은 인터럽트를 받았을 경우 경고 메시지를 프린트 한다.

floppy=no_unexpected_interrupts or floppy=L40SX 예상치 않은 인터럽트를 받았을 경우 경고 메시지를 프린트 하지 않는다. 이는 어떤 비디오 모드에 IBM L40SX 랩탑 컴퓨터에서 필요하다. 여기엔 비디오와 플로피 사이에 내부적인 상호 작동이 있는 것으로 보인다. 예상치 않는 인터럽트는 단지 성능에 영향을 주며 무시해도 안전하다.)

8.3 The Sound Driver ('sound=')

사운드 드라이버는 제대로 된 값으로 컴파일된 것을 오버라이드하기 위하여 부트 인자들을 허용한다. 이것은 오히려 복잡함으로 추천 하지 않는다. 이는 linux/drivers/sound에 Readme.Linux 파일에 서술되어 있다. 부트인자 형태는 다음과 같다:

```
sound=device1[,device2[,device3...[,device11]]]
```

각각의 deviceN 값들은 아래의 0xTaaaId 이런 형태이며 다음과 같은 바이트 들을 사용한다.

```
T - 장치의 형태 : 1=FM, 2=SB, 3=PAS, 4=GUS, 5=MPU401, 6=SB16,
                  7=SB16-MPU401
aaa - 16 진수의 I/O 주소들
I - 16진수의 인터럽트 라인(예를 들면 10=a, 11=b, ...)
d - DMA 채널.
```

이것을 매우 지저분하다고 생각할 수 있음으로 자신의 개인적인 값들로 컴파일 하는 것이 추천된다. 'sound=0' 부트 인자를 사용하는 것은 사운드 드라이버 전체를 불능으로 만든다.

8.4 The Bus Mouse Driver ('bmouse=')

버스 마우스 드라이버는 단지 하나의 parameter만을 취한다. 이는 사용되는 하드웨어 IRQ 값이다.

8.5 The MS Bus Mouse Driver ('msmouse=')

MS 마우스 드라이버는 단지 하나의 parameter만을 취한다. 이는 사용되는 하드웨어 IRQ값이다.

8.6 The Printer Driver ('lp=')

1.3.75 이상의 커널에 있어 당신은 프린터 드라이버에게 사용할 포트와 사 용하지 않을 포트를 알려줄 수 있다. 후자의 경우 프린터 드라이버가 모든 프린터 포트를 요구하는 것을 원하지 않는다면 다른 드라이버들은 (예를 들 면 PLIP, PPA) 들은 그들 대신에 사용할 수 있다.

인자의 형식은 여러 i/o와 IRQ 쌍이다. 예를 들면 lp=0x3bc,0,0x378,7는 0x3bv포트에 IRQ없는(polling)모드를 사용하고 0x378포트에 IRQ 7번을 사용한다. 0x278포트는 검색되지 않을 것이다.

자동 검색은 단지 '인자가 없을 시에만 수행된다. 모든 프린터 드라이버를 불능으로 만들기 위해서 단지 lp=0 이것 하나만 사용해도 된다.

8.7 The ICN ISDN driver ('icn=')

ISDN드라이버는 아래 형식의 부트 인자들을 원한다:

```
icn=iobase,membase,icn_id1,icn_id2
```

iobase는 카드의 i/o 포트의 주소이고, membase는 카드의 공유메모리의 베이스 주소이고, 두 개의 icn_id는 유일한 ASCII 문자열 구분자들이다.

8.8 The PCBIT ISDN driver ('pcbit=')

이 부트 인자들은 아래 형식의 쌍으로 된 정수 인자들을 가진다:

```
pcbit=membase1,irq1[,membase2,irq2]
```

membaseN는 N 번째 카드의 공유 메모리 베이스 주소이고 irqN 는 N 번째 카드의 인터럽트 설정이다. 디폴트로는 IRQ 5, membase가 0xD0000을 사용한다.

8.9 The Teles ISDN driver ('teles=')

ISDN 드라이버는 아래 형식의 부트 인자들을 원한다:

```
teles=iobase,irq,membase,protocol,teles_id
```

iobase는 카드의 i/o 포트의 주소이고, membase는 카드의 공유메모리의 베이스 주소이고, irq는 카드가 사용하는 인터럽트 채널이다. 그리고 teles_id는 유일한 ASCII 문자열 구분자들이다.

8.10 The DigiBoard Driver ('digi=')

DigiBoard 드라이버는 여섯 개의 콤마로 구분된 구분자들이나 정수들을 허용한다. 순서를 가지는 6개의 값들은 아래와 같다:

```
이 카드를 사용가능/불능
카드의 형: PC/Xi(0), PC/Xe(1), PC/Xeve(2), PC/Xem(3)
대체 핀 배열을 사용가능/불능
이 카드의 포트 수
카드에 설정된 I/O 포트(문자열을 사용한다면 16진수)
메모리 윈도우 베이스(문자열을 사용한다면 16진수)
```

정확한 부트 프롬프트 인자의 예(구분자나 정수형 모두):

```
digi=E,PC/Xi,D,16,200,D0000
digi=1,0,0,16,512,851968
```

digi= 부트 프롬프트가 없을 시에는 디폴트로 0x200의 i/o, 0xD0000의 공유 메모리 주소가 사용됨을 주의하라. 여기에는 자동검색은 없다. 좀더 자세한 내용은 linux/Documentation/digiboard.txt에서 찾을 수 있다.

8.11 The RISCom/8 Multiport Serial Driver ('riscom8=')

4개까지의 보드는 각각의 보드 설정에 유일한 i/o 포트 값들을 제공함으로써 지원될 수 있다. 다른 세부적인 사항은 linux/Documentation/riscom8.txt에서 찾을 수 있다.

8.12 The Baycom Serial/Parallel Radio Modem ('baycom=')

이들 장치들의 부트 인자의 형식은:

```
baycom=modem,io,irq,options[,modem,io,irq,options]
```

modem=1을 사용하는 것은 ser12장치를 가지고 있음을 의미하고 modem=2는 당신이 par96 장치를 가지고 있음을 의미한다. options=0을 사용하면 하드웨어 DCD를 사용하는 것이고 options=1는 소프트웨어적인 DCD를 사용하는 것이다. io와 irq는 평상시 i/o 포트 베이스 그리고 인터럽트 설정들이다. /linux/drivers/char/ 디레토리에 README.baycom 파일에 자세한 내용이 있다.

제 9 절 Closing

이 문서에서 오자나 뒤떨어지는 정보를 발견했다면, 나에게 알려달라. It is easy to overlook stuff. Thanks, Paul Gortmaker, *gpg109@rsphy1.anu.edu.au* `<mailto:gpg109@rsphy1.anu.edu.au>`