Linux Ethernet-Howto

by Paul Gortmaker v2.7, 5 May 1999

맹지찬, jcmaeng@kldp.org 2000년 7월 22일

이 문서는 리눅스에서 사용될수 있는 이더넷 장치들과 그들을 어떻게 설치하는지 에 대해서 기록한 Ethernet-Howto이다. 알아둘 것은 이 하우투가 이더넷 카드의 하드웨어와 로우 레벨 드라이버에 관해서 다루며, `ifconfig`나 `route`와 같은 소프트웨어적인 면은 다루지 않는다는 점이다. 그러한 것들은 Network Howto 를 보기 바란다.

목 차

제 1 절 소 개

Ethernet-Howto는 여러분이 어떤 카드를 사거나 사지 말아야 하고 또, 어떻게 그 들을 설치하는지, 어떻게 하나이상 작동하게 하는지와 다른 일반적인 문제와 답에 대해 다루고 있다. 이 문서는 현재 지원되는 사용 가능한 대부분의 일반적인 이더 넷 카드들에 대한 상세한 정보를 담고 있다.

여기서는 소프트웨어 면에 대해서는 *다루지 않으며*, 그러한 것은 NET-3 Howto에서 다루고 있다. 또한 여기서는 일반적인 리눅스 외적인 일반적인 이더넷에 관한 질문들에 답하지 않는다는 점에 주의하라. 그런 종류의 질문들은 *comp.dcom.lans.ethernet*FAQ에 수많은 양의 정보가 있으니 보길 바란다. 이 문서들은 다른 모든 뉴스그룹 FAQ들 처럼 `rtmf.mit.edu`에서 FTP를 통해 얻을수 있다.

현재 개정판은 배포판 커널을 2.2.7까지 다룬다.

Ethernet-Howto는

Paul Gortmaker, `p_gortmaker@yahoo.com`

가 작성했다.

초기 ASCII 버전만 있는 Ethernet-Howto에 대한 정보의 원래 자료는

Donald J. Becker, `becker@cesdis.gsfc.nasa.gov`

에게 있었다. 그는 현재 리눅스에서 사용할수 있는 거의 주요한 이더넷 카드의 드라이버를 만든 사람이다. 그는 또한 오리지날 NFS 서버의 작성자이기도 하다. 고맙습니다, Donald!

1.1 이 문서의 새 버전들

이 문서의 새 버전들은 다음의 장소에서 가져올수 있다.

Ethernet-HOWTO `<http://metalab.unc.edu/mdw/HOWTO/Ethernet-HOWTO.html>`

또>는 FTP를 사용하거나 HTML 이외의 포맷을 얻으려 한다면 다음에서 얻을수 있다.

Sunsite HOWTO Archive `<ftp://metalab.unc.edu/pub/Linux/docs/HOWTO/>`

이것>은 '공식' 장소이다 - 즉, 다른 수많은 리눅스 WWW/ftp 미러 사이트에서도 찾 을수 있다. 사용할 수 있는 드라이버들에 대한 새 정보들로 갱신될 것이다. 만일 여러분이 보고있는 것이 6개월 이상된 것이라면 새로 갱신된 문서가 있는지 확인해 보기 바란다.

이 문서는 다양한 포맷으로 이용할 수 있다.(postscript, dvi, ASCII, HTML 등등). 나는 HTML(WWW 브라우저를 통해서)이나 Postscript/dvi 포맷으로 보기를 권한다. 이 두가지는 일반 텍스트 ASCII 포맷이 가지지 못한 상호 참조를 포함하고 있기 때문이다.

1.2 Ethernet-Howto의 사용

이 안내문서의 양이 점점 많아지기 때문에, 전부 다 읽기 위해 여러분의 시간을 낭비하고 싶지 않을 것이다. 그리고 이 문서를 전부 다 읽을 필요도 없다. HTML 이나 Postscript/dvi 판은 여러분에게 필요한 것을 더 빨리 찾을 수 있도록 도와 주는 목차를 포함하고 있다.

여러분이 이 문서를 읽는 것은 무언가가 잘 안되거나 무엇을 해야할지 알지 못할 경우이다. 다음 장(**??** (도와주세요 - 잘안되요!)) 은 리눅스를 처음 접하는 사람들에게 올바른 방향을 제시해 주기 위한 것이다.

일반적으로 같은 문제와 질문들이 여러 사람들에게서 *반복해서* 제기된다. 여러분의 문제나 질문이 자주 제기되는 질문들 중의 하나일 경우, 이 문서의 FAQ 부분에 그 대답이 있다. (3 (FAQ 장)). 도움을 요청하기 전에 누구든지 이 장을 보기 바란다.

여러분이 이더넷 카드를 가지고 있지 않다면, 그리고 카드를 선택하는 것부터 시 작하고 싶다면. (2 (무슨 카드를 사야하나...))

만일 이미 이더넷 카드를 가지고 있지만, 그것이 리눅스에서 사용할 수 있을지 확신할수 없다면, 각각의 제조업체와 그들의 카드에 대한 정보를 담고 있는 장을 읽어보면 된다. (5 (특정 업체...))

리눅스 드라이버의 기술적인 면에 좀 관심이 있다면, 그에 관한 부분을 볼 수도 있다. (8 (기술 정보))

1.3 도와주세요 - 잘안되요!

자, 당황하지 말라. 이 글은 비록 여러분이 이더넷 하드웨어나 리눅스에 관한 특 별한 배경지식이 없어도, 그 과정을 이끌어 줄 것이다.

우선 여러분이 할 일은 여러분의 카드가 어떤 모델인지 알아내서 리눅스에 그 카 드의 드라이버가 있는지 찾아내는 것이다. 다른 종류의 카드는 호스트에 의해 서 로 다른 방법으로 제어가 된다. 리눅스 드라이버는 리눅스에서 이 카드를 사용할 수 있도록 일정한 형식으로 그 제어 정보를 담고 있다. 만약 여러분이 어떤 매뉴 얼이나 다른 어떤 종류의 카드 모델에 대한 정보도 없다면, 알지 못하는 카드를 찾아낼수 있도록 시도할 수 있다. (참고 부분: 5.44 (알지 못하는 카드 알아내기)).

이제 여러분이 가진 카드가 어떤 것인지 알았으면, 특정 카드 부분 (참고 부분: 5 (특정 업체...)) 에서 여러분의 카드에 대한 자세한 내용을 읽어보기 바란다. 여기에는 알파벳 순서로, 카드 제조업체들, 각 모델번호와 그들의 리눅스 드라이 버가 있는지 없는지가 담겨있다. 만일 이 목록에 '지원되지 않음'이라고 나오면 여기서 포기해라. 만일 여러분의 카드를 이 목록에서 찾지 못한다면, 카드 매뉴 얼에 알려진 다른 카드 종류의 '호환'이라고 적혀있지 않은지 확인하라. 예를 들 어 수백의, 하지만 수천은 아닌 수많은 카드들이 오리지날 노벨 NE2000 모델과 호환제품으로 만들어진 것들이다.

여러분의 카드에 대한 리눅스 드라이버가 있다는 것을 확인했다고 가정하고, 이 제 여러분은 그것을 찾아서 사용해야 한다. 여러분의 카드에 대한 리눅스 드라이 버가 있다는 것이 모든 커널에 적재되어 있다는 뜻은 *아니기 때문이다. (커널은 부트시에 가장 먼저 적재되는 운영체제 핵심으로, 수많은 종류의 하드웨어에 대한 드라이버와 다른 여러가지를 포함하고 있다.) 여러분이 사용하고 있는 리눅스 배포판을 만든 이에 따라서, 그안에는 단지 약간의 이미 만들어진 커널과 더욱 작게 분리된 모듈로 된 모든 드라이버들이 있거나, 수많은 종류의 드라이버가 안 에 적재된 큰 커널이 들어있을수도 있다.*

현재 대부분의 리눅스 배포판들은 다양한 드라이버들을 작은 모듈로 해서 담고 있다. 필요한 모듈들은 부팅 과정중에 늦게 적재되거나, 아니면 특정한 장치에 접근할 필요가 있을 때 적재하게 된다. 여러분은 부팅이 된 후에 커널에 모듈을 적재해야 할 것이다. 여러분의 배포판에 모듈을 설치하고 사용하는 것에 대한 정보는 이 문서의 모듈 부분을 보기 바란다. (10.2 (이더넷 드라이버를 모듈로 사용하기))

여러분이 커널에 미리 만들어져서 들어있는 커널이나 모듈로된 드라이버를 찾지 못했다면, 여러분은 일반적이지 않은 카드를 가지고 있는 것이고, 그러므로 이제 그 카드의 드라이버를 포함한 여러분 자신만의 커널을 만들어야 할 것이다. 리눅 스를 인스톨했었다면, 사용자 커널을 만드는 일은 그리 어려운 일이 아니다. 여 러분이 커널에 넣고자 하는 것에 대해서, 예 또는 아니오로 확실히 답해주고, 그 대로 만들라고 말해주기만 하면 된다. 이 부분에서는 Kernel-HowTo에서 도움을 받을 수 있을 것이다.

이제 여러분은 드라이버가 커널에 함께 들어있거나 또는 모듈로 적재된 커널로 부팅을 했을 것이다. 사람들이 가진 문제들중 절반가량은 어떤식으로든 적재된 드라이버를 가지지 못하는 것인데, 이제 여러분은 올바른 것을 찾았을 것이다.

만일 아직도 작동하지 않는다면, 여러분은 커널이 카드를 제대로 찾아내는지 확 인해봐야 한다. 이것을 확인하기 위해서는, 시스템이 부팅해서 모든 모듈들이 적 재된 후에 로그인해서 `dmesg | more` 명령을 쳐보면 된다. 이 명령은 커널의 부팅 과정중에 스크린을 지나가는 부트 메세지들을 다시 보게 해주는 것이다. 카드가 제대로 발견되면, 메세지 가운데 어딘가에서 `eth0`으로 시작하고, 드라이버의 이름과 카드가 설정되어있는 하드웨어 인자들(인터럽트 설정, 입출력 포트 주소, 등등)이 있는 것을 볼수 있을 것이다. (알아둘것: 부트시에, 리눅스는 시스템에 있는 모든 PCI카드들의 목록을 보여주는데, 어떤 드라이버가 사용될수 있는 것인지는 신경쓰지 말라 - 드라이버 찾기는 나중에 나오므로 이것을 보고 실수하지 말라!)

여러분이 이와 같은 드라이버 확인 메세지를 보지 못했다면, 드라이버가 여러분 의 카드를 찾지 못해서 제대로 작동하지 않은 것이다. 카드가 발견되지 않을 경 우 어떻게 해야 하는지는 FAQ (3 ()FAQ장)을 보기 바란다. 여러분이 NE2000 호환카드를 가지고 있다면, 그것 또한 몇몇 NE2000만의 특정한 팁들로 카드를 찾아내는 방법이 FAQ 장에 잘 나와있다.

여러분의 카드가 발견되었지만, 발견 메세지가 몇몇 에러를 담고 있다면, 자원충 돌처럼, 드라이버가 카드를 제대로 초기화하지 못해서 아직 사용할수 없는 것이 다. 이런 종류의 대부분의 일반적인 메세지도, 그 해결법과 함께, FAQ 장에 나와 있다.

발견 메세지가 제대로 된것 같으면, 드라이버가 보여주는 카드의 자원과 실제 카 드의 물리적인 설정(카드 상의 조그마한 검은색 점퍼들이나 카드 제조사가 주는 소프트웨어 유틸리티에 의한것 모두)이 같은지를 확인해라. 이들이 정확히 같아 야만 한다. 예를 들어, 여러분의 카드가 IRQ 15로 되어있는데 드라이버는 부트 메세지에 IRQ 10 으로 보여준다면, 이것은 제대로 작동하지 않는다. FAQ장은 수 많은 카드들의 설정 정보를 제대로 찾지 못하는 드라이버에 관한 대부분의 경우 에 대하여 다루고 있다.

이제, 여러분은 모두 제대로 된 인자들을 가지고 찾아진, 모든것이 제대로 작동 하는 카드를 가지고 있다. 만일 아니라면, 소프트웨어나 하드웨어 설정 에러가 있는 것이다. 소프트웨어 설정 에러는 `ifconfig`와 `route` 명령으로 올바른 네트워크 주소를 설정해주지 않아서이다. 이에 관한 내용은 Network HowTo와 '네트워크 관리자 가이드'에 자세하게 기술되어 있다. 두 문서 모두 여러분이 설치했던 CD-ROM에 들어있을 것이다.

하드웨어 설정 에러는 자원의 충돌이나 잘못된 설정 (드라이버가 부트시에 찾아 내지 못하는)이 카드가 제대로 작동하지 못하고 멈추게 하는 것을 말한다. 이것 은 보통 몇가지 다른 면으로 보여질 수 있다. (1) 여러분이 `ifconfig` 를 사용해 장치를 사용하기 위해 열고자 할때 "SIOCSFFLAGS: 다시 시도하시오"같은 에러 메세지가 뜬다. (2) 드라이버가 `eth0` 에러 메세지들(`dmesg | more`로 볼수 있는)을 내보내거나 데이타를 보내거나 받을려 할때마다 잘되지 않는다. (3) `cat /proc/net/dev` 라고 치면 `eth0`의 errs, drop, fifo, frame 또는 carrier 열중 에서 영이아닌 숫자들이 보인다. (4) `cat /proc/interrupts` 라고 치면 카드의 인터럽트 카운트가 0으로 나온다. 대부분의 일반적인 하드웨어 설정 에러들도 FAQ 장에서 다루고 있다.

자, 아직까지도 제대로 되지 않는다면, 이 문서의 FAQ 장을 읽고, 여러분의 카 드에 관한 특정 업체의 부분을 읽어라. *그리고나서도 안되면*, 도움을 요청할수 있는 적절한 뉴스그룹에 게시해보길 바란다. 게시할때 관련있는 모든 정보들, 즉, 카드 상표, 커널 버전, 드라이버 부트 메세지, `cat /proc/net/dev` 출력결과, 문제에 대한 명확한 설명, 그리고 여러분이 시도했던 방법들의 과정같은 정보를 자세하게 적어야 한다.

여러분은 얼마나 많은 사람들이 "누구 도와줄 사람 없나요? 이더넷이 되질 않 아요."같은 쓸모없는 글을 게시하는지 알면 놀랄것이다. 뉴스그룹을 읽는 사람 들은 그러한 게시물들은 무시하려하므로, 자세하고 쓸모있는 정보들이 있는 문 제설명들만이 '리눅스 도사'들의 주목을 받을수 있다.

제 2 절 리눅스에서 쓰려면 무슨 카드를 사야하나?

이 질문에 대한 대답은 전적으로 여러분이 네트워크 연결로 하려는 일과 그것이 얼마나 많은 전송량을 갖는지에 달렸다.

여러분이 단지 잠깐씩 ftp와 WWW 연결을 하려는 단일 사용자라면, 낡은 8 비트 ISA 카드만으로도 만족할수 있을 것이다.

만일 여러분이 서버를 설치하고자 한다면, 주고 받는 네트워크 데이타로 인한 CPU 부하를 최소로 유지하는것이 필요하므로, DEC tulip (21xxx) 칩이나 AMD PCnet-PCI 칩처럼 버스 마스터링 기능이 있는 PCI 카드중에 하나를 고르면 된다.

여러분이 위의 중간정도에 해당된다면, 안정된 드라이버가 있는 저가의 PCI 나 16 비트 ISA 카드중에 하나를 고르면 될 것이다.

2.1 어떤 드라이버가 안정된거지?

16비트 ISA 카드들 중에서는, 다음의 드라이버들이 매우 안정되어서, 여러분이 이 드라이버들을 사용하는 카드를 산다면 어떠한 문제도 발생하지 않을것이다.

SMC-Ultra/EtherEZ, SMC-Elite (WD80x3), 3c509, Lance, NE2000.

여기서 말하는 것이 다른 드라이버들이 불안정하다는 말이 아니다. 단지 위의 것들이 모든 리눅스 드라이버들 중에 가장 오래되고 가장 많이 사용되어서, 가 장 안전한 선택이 된것이다.

주의할 점은 몇몇 값싼 마더보드들은 ISA Lance 카드가 사용하는 버스마스터링 과 문제가 있을 수 있고, 몇몇 저가의 NE2000 호환품들은 부팅할때 발견하기에 어려움이 있을수 있다.

리눅스에서 가장 일반적으로 사용되는 PCI 드라이버는 아마도 3Com Vortex/Boomerang (3c59x/3c9xx), DEC tulip (21xxx), 그리고 Intel EtherExpressPro 100. 수많은 PCI-NE2000 호환카드들도 아주 많이 쓰이지만, PCI-NE2000 호환 카드를 사는 것은 낮은 가격이 현재의 고성능 디자인 카드보 다 중요하지 않다면 별로 권장하지 않는다.

2.2 8 bit 대 16 bit 카드

여러분은 아마도 새 8 bit ISA 이더넷 카드를 사려고 하지 않을 것이다. 그러 나 여러분은 다음 몇년동안 컴퓨터를 바꿀때까지 매우 낮은 가격의 그들을 수 없이 찾아볼수 있을 것이다. 이것은 그들을 "홍-이더넷" 시

스템의 인기제품 으로 만들것이다. 지금은 PCI 카드들이 일반적이기 때문에, 위의 내용은 지금 의 16 bit ISA 카드들에게도 그렇다.

몇몇 8 비트 카드들은 낮은 가격에 적당한 성능을 보이는데, wd803, 3c503 그 리고 ne1000이 그렇다. 3c501은 낮은 성능을 제공하며, 이러한 12년 이상된 XT시절의 것들은 피해야 한다. (그들을 Alan에게 보내주길, 그는 수집한다..)

8 비트 wd8003 카드(fast ISA 버스 상에서)로 빠른 호스트에서 대략 500에서 800kB/s의 다운로드 속도를 얻을수 있으므로, 8 비트 데이타 패스는 성능에 해를 끼치진 않는다. 그리고 만일 여러분의 실제 전송량의 대부분이 원격 사 이트로 가는 것이라면, 어딘가에 병목현상이 있을 것이다. 그렇다면 여러분 이 신경쓸 속도의 차이는 여러분 로컬 서브넷 상에서 실제 활동하는 동안일 것이다.

2.3 32 Bit (VLB/EISA/PCI) 이더넷 카드들

10Mbs 네트워크는 일반적으로 반드시 32 비트 인터페이스를 필요로 하지 않는 다는 것에 주의하라. 8MHz ISA 버스상의 10Mbps 이더넷 카드가 왜 정말로 병 목지점이 아닌지는 8.1 (Programmed I/O vs. ...)를 보아라. 빠른 버스상에 이더넷 카드가 있어도 그것이 더 빠른 전송을 의미하는 것은 아니고, 일반적으로 다중 사용자 시스템에 좋은, CPU 부하의 감소를 의미한다.

물론 지금은 일반적인 100Mbps 네트워크에서는 32 비트 인터페이스가 전체 대 역폭을 사용할수 있도록 해준다. AMD는 32 비트 PCnet-VLB 와 PCnet-PCI 칩들 을 가지고 있다. LANCE / PCnet-ISA 의 32 비트 버전에 관한 정보는 5.4.2 (AMD PCnet-32)부분을 보라.

DEC 21xxx PCI 'tulip' 칩은 파워유저들의 또다른 선택이다.(5.16.4 (DEC 21040)을 보라) 수많은 제조업체들이 이 칩을 사용해서 카드를 생산하고, 그러한 이름 없는 카드들의 가격은 보통 아주 저렴하다.

3Com의 'Vortex' 와 'Boomeran' PCI 카드또한 다른 선택이며, 그것을 사용하 는 기간에 비교해볼때 가격느 매우 저렴하다. (5.1.14 (3c590/3c595)를 보라)

Intel의 EtherExpress Pro 10/100 PCI 카드 또한 리눅스에서 잘 돌아간다고 알려져 있다. (5.22.4 (EtherExpress)를 보라)

수많은 호환품 제조업체들은 RealTek 이나 Winbond 칩 기반의 PCI NE2000 호 환품을 만들기 시작했다. 이 카드들도 v2.0.31 이상의 커널에 있는 리눅스의 ne2000 드라이버가 잘 지원한다. 그러나 카드가 시대가 지난 ne2000 드라이버 인터페이스를 사용하기 때문에, 더 빠른 버스 인터페이스로 인한 것 이외에는 이점이 없다. v2.0.34 (그 이상)에서는 이 카드들을 위해 독립된 PCI특화된 드라이버인 `ne2k-pci.c`도 사용할수 있는데, 이것은 ISA `ne.c` 드라이버보다 아주 약간의 성능 향상이 있다.

2.4 사용할수 있는 100Mbs 카드와 드라이버들

현재 지원되는 100Mbs 하드웨어의 목록은 다음과 같다. DEC 21140 칩 기반의 카드들; 3c595/3c90x Vortex 카드들; EtherExpressPro10/100B; PCnet-FAST; SMC 83c170 (epic100) 과 HP 100VG ANY-LAN.

각각에 대한 정보는 이 문서의 특정 업체에 관한 정보를 보기 바란다. 그리고 다음의 곳도 확인해보라.

Linux and 100Mbs Ethernet `<http://cesdis.gsfc.nasa.gov/linux/misc/100mbs.html>`

Donald's 100VG Page `<http://cesdis.gsfc.nasa.gov/linux/drivers/100vg.html>`

Dan Kegel's Fast Ethernet Page `<http://alumni.caltech.edu/dank/fe/>`

2.5 100VG 대 100BaseT

100BaseT가 100VG보다 훨씬 더 많이 알려져 있고, `comp.os.linux`에 게시한 Donald의 오래된 정보안의 다음 내용은 그 상황을 매우 잘 요약하고 있다.

"사정을 잘 모르는 이들을 위해 말하면, 100Mbs 이더넷 표준은 두가지가 경쟁하 고 있다. 그것은 100VG (100baseVG 와 100VG-AnyLAN) 과 100baseT (100baseTx, 100baseT4 그리고 100baseFx 케이블 타입)이다.

100VG가 먼저 시장에 선보였고, 내가 느끼기에도 그것이 100baseTx보다 더 잘 만들어진것 같다. 나는 그가 이기길 바라지만, 확실히 그렇게 되지 않을 것이다. HP et al. 은 몇가지 안좋은 선택을 했다.

1) 표준을 지연시키고 IBM과 협력하여 토큰 링 프레임을 지원한다. 이것은 토큰 링 샵을 업그레이드함으로써 관리자들이 그들이 적합치 않은 기술에 의존함으로써 아주 값비싼 대가를 치룰 필요가 없었기 때문에 '그 당시에는 좋은 생각같아 보였다' 그러나 네트워크 상에 두가지 프레임 타입이 공존할수 없었기 때문에 아무것도 얻을 수 없었고, 토큰링은 너무 복잡하므로, IBM은 100BaseT를 선택했다.

2) 단지 ISA 와 EISA 카드들만 만들어졌다. (PCI 모델은 최근에야 발표되었다.) ISA 버스는 100mbs에는 너무 느리고, EISA 머신들은 매우 적은 수만 존재한다. 그 당시 에는 VLB가 PCI와 함께 일반적이고, 빠르며, 저렴한 선택이었다. 그러나 "구시대" 지식인은 서버는 더 고가의 EISA 버스를 가지고 있어야 한다고 생각했다.

3) 나에게 데이타북을 보내지 않았다. 그렇다, 이것이 100VG가 추락한 진짜 이유다 :-). 나는 모든 프로그래밍 정보를 요청했지만, 내가 받은것이라고는 AT&T에서 보낸 Regatta 칩셋이 얼마나 훌륭한지에 대해 설명해놓은 몇페이지짜리 그럴듯한 컬러 팜플렛 뿐이었다.

2.6 여러분의 카드가 지원하는 케이블의 종류

만일 여러분이 소규모의 "개인적인" 네트워크를 구성하려 한다면, thinnet 즉 thin 이더넷 케이블을 사용하려 할 것이다. 이것은 표준 BNC 커넥터의 형태로 된 것이다. BNC (금속제의 누르고 돌려서 잠그는) 커넥터를 가진 thinnet, 즉 thin 이더넷 케이블링, (RG-58 동축 케이블) 은 기술용어로 10Base2라고 불린다.

대부분의 이더넷카드들은 단지 $10-$20만 더하면 '콤보' 버전을 구할수 있다. 이 카드들은 꼬임선과 thinnet 단자를 내장하고있어서, 여러분이 나중에 마음을 바꾼다해도 괜찮다.

RJ-45 (대형 전화 잭)을 가진 꼬임선 케이블 커넥터는 기술용어로 10BaseT라고 불린다. 이것은 또 UTP (Unshielded Twisted Pair)라고도 불리운다.

예전의 설치물에서만 찾아볼수 있는 구형의 thick 이더넷 (10mm 동축케이블)은 10Base5라고 불린다. 몇몇 카드들에서 찾아볼수 있는 15핀 D형 플러그 (AUI 커넥터)는 thick 이se2 대신에 10BaseT가 사용된다.

10Base2는 100Base계열로의 어떠한 업그레이드도 제공하지 않는다. 더넷과 외부 단자에 연결하기 위해 사용하는 것이다.

다른 형태의 이더넷 케이블에 관해서는 6 (케이블, 동축...) 를 보기 바란다.

제 3 절 Frequently Asked Questions

이곳에는 이더넷에 연결된 리눅스를 사용하는 것에 대한 자주 물어보는 질문들(FAQ)이 있다. 몇몇 특정 질문들은 '제조업체별'에 정리되어 있다. 여러분이 답변을 원하는 질문들은 이미 다른 누군가가 질문한 것(그리고 답변이 되었다!)이고, 만일 여기서 원하는 답변을 찾지 못할 경우,적어도 아래같이 여러분이 원하는 뉴스 아카이브가 있는 곳을 찾을수 있을 것이다. *Dejanews* `<http://www.dejanews.com>`.

3.1 Alpha Drivers – 구하고 사용하기

내가 듣기로는 내 카드에 사용할 수 있는 갱신되거나 시험버전의 알파 드라이버가 있다 고 하는데 어디서 구할수 있을까?

'새' 드라이버들의 가장 최신버전은 Donald의 ftp 사이트 `cesdis.gsfc.nasa.gov` 안의 `/pub/linux/` 에서 구할수 있다. 여기있는 것들은 자주 바뀌므로, 찾을수 있을 것이다. 또한, WWW 브라우저를 사용해서

Don's Linux Home Page <http://cesdis.gsfc.nasa.gov/linux/>

에 가면 여러분이 찾고자 하는 드라이버를 더욱 쉽게 찾을수 있다. (WWW 브라우저로 찾으면 소스의 탭을 스페이스나 기타등등으로 바꾸어버린다 - ftp를 사용하거나 확실하지 않다면 적어도 다운받을 FTP URL 은 알아둬라.)

자, 만일 그것이 정말로 알파 드라이버, 또는 알파 이전의 드라이버라면, 그 이름처럼 다루기 바란다. 다른 말로 하면, 여러분이 그것으로 무엇을 하는지 이해하지 못할지라도 불평하지 말라는 것이다. 만일 여러분이 어떻게 설치하는지 이해할수 없다면, 아마도 시험 해보지 못할 것이다. 또, 그것 때문에 여러분의 머신이 다운되더라도 불평하지 마라. 대신 잘 작성된 버그 리포트를 보내거나, 패치를 보내준다면 더 좋다!

표준 커널 소스 트리에 포함되어 있는 몇몇 '사용가능한' 실험적인/알파 드라이버들에 대해 알아두어야 할 것들이다. `make config`를 했을때 가장 먼저 물어보는 것은 "개발중인 또는 완전하지 않은 코드/ 드라이버들에 대해 표시(Prompt for development and/or incomplete code/drivers)" 할것인지 아닌지 이다. 알파/실험적인 드라이버들을 포함할 것인지에 관한 질문들을 받으려면 여기에 'Y'라고 답해야 한다.

3.2 머신당 하나이상의 이더넷 카드 사용하기

리눅스에서 두개의 이더넷 카드를 사용하려면 무엇이 필요하나요?

이 질문에 대한 답은 드라이버가 적재가능한 모듈로 사용되고 있는지 커널에 직접 컴파 일되어 들어가 있는 것인지에 따라 달라진다. 지금의 대부분의 리눅스 배포본들은 모듈러 드라이버를 사용한다. 이들은 배포되고 있는 수많은 커널들과 각각의 다른 드라이버들을 만들어 저장한다. 단일 기본 커널이 사용되는 대신에 특정 사용자의 시스템에 필요한 각각 의 드라이버들이 시스템이 부팅될 때 드라이버 모듈 파일들에 충분히 접근할수 있도록 한번 로드된다. (일반적으로 `/lib/modules/`에 저장된다.)

드라이버를 모듈로: PCI 드라이버들의 경우, 보통 설치된 모든 카드의 브랜드 모델을 자동적으로 찾아낼 것이다. 그러나, ISA 카드들의 경우, 카드를 찾아내는 작업이 안전하지 않기 때문에, 보통 모듈이 어디서 카드를 찾을수 있는지 I/O 주소를 가르쳐 줄 필요가 있다. 이 정보는 `/etc/conf.modules`에 저장되어 있다.

예를 들어, 사용자가 두개의 ISA NE2000 카드를 가지고 있고, 하나는 `0x300`에 그리고 다른것은 `0x240`에 있다. 이들에 대한 `/etc/conf.modules`의 내용을 보면,

```
alias eth0 ne
alias eth1 ne
options ne io=0x240,0x300
```

이것이 하는 것은 이렇다. 이것은 관리자 (혹은 커널)이 `modprobe eth0` 혹은 `modprobe eth1`라고 하면, ne.o 드라이버가 eth0나 eth1를 위한 드라이버를 로드할 것이다. 그리고 ne.o 모듈이 적재될때, io=0x240,0x300 라는 옵션을 가지고 로드되어, 드라이버가 어디에서 카드를 찾을지 알려주게 된다. 0x는 중요하다 - DOS 세상에서 일반적으로 사용하던 300h같은 것들은 통하지 않는다. 0x240 와 0x300의 순서를 바꾸는 것은 eth0 와 eth1의 물리적 카드 순서를 바꾸는 것이 된다. 대부분의 ISA 모듈 드라이버들은 이 예와 같이 여러개의 카드를 다루기 위해 콤마로 구분된 여러개의 IO 값을 받을수 있다. 그러나, 3c501.o 모듈과 같은 몇몇 (구형의?) 드라이버들은 모듈을 로드할 때마다 단지 하나의 카드만을 다룰수 있다. 이 경우에 두 장의 카드를 모두 찾기 위해서 여러분은 모듈을 두번 로드할 수 있다. 이 경우에 `/etc/conf.modules` 화일은 다음과 같다.

```
alias eth0 3c501
alias eth1 3c501
options eth0 -o 3c501-0 io=0x280 irq=5
options eth1 -o 3c501-1 io=0x300 irq=7
```

이 예에서 `-o` 옵션은 여러분이 같은 이름으로 두 모듈을 로드할 수 없기 때문에 각 모듈 객체마다 유일한 이름을 부여하기 위해서 사용된다. `irq=` 옵션도 또한 카드의 하드웨어 IRQ 설정을 정해주기 위해서 사용

된다. (이 방법은 콤마로 구분된 I/O 값들을 받아들이는 모듈들을 사용할 때에도 쓸수 있다. 그러나 이것은 그것이 정말 필요하지 않을 때에도 모듈이 두번씩 로드되기 때문에 덜 효율적이다.)

마지막 예로, `0x350`에 있는 3c509 카드와 `0x280`에 있는 SMC Elite16 (WD8013) 카드를 가진 유저가 있다. 그 설정은 다음과 같을 것이다.

```
alias eth0 wd
alias eth1 3c503
options wd io=0x280
options 3c503 io=0x350
```

PCI 카드들의 경우, PCI 카드의 I/O 주소는 안전하게 찾아낼수 있기 때문에 여러분은 보통 적절한 드라이버 이름과 같이 `ethN` 인터페이스와 연관된 `alias` 줄만이 필요하다.

사용가능한 모듈들은 보통 `/lib/modules/`uname -r`/net`에 저장되어 있다. 여기서 `uname -r` 명령은 커널 버전 (예: 2.0.34)을 돌려준다. 여러분은 거기서 여러분의 카드에 맞는 것을 찾을수 있다. 여러분이 `conf.modules` 화일에 한번 제대로 설정을 했다면, 다음과 같이 해서 시험해 볼수 있다.

```
modprobe ethN
dmesg | tail
```

'N'은 여러분이 시험해 보고자하는 이더넷 인터페이스의 숫자이다.

커널 안에 컴파일되어 들어있는 드라이버로: 만일 여러분이 커널에 컴파일되어 들어있는 드라이버를 가지고 있다면, 여러개의 이더넷 카드를 사용하기 위한 모든 것이 그 안에 있다. 그러나, 기본적으로 *하나의* 이더넷 카드만이 자동으로 찾아진다는 것에 주의하라. 이것은 예민한 카드들을 찾을때 발생할수 있는 부팅시의 에러를 피하도록 해준다.

(알아둘것: 2.1.x 후반대의 커널에서는, 부트 검색이 안전과 불안전으로 나누어져 있고, 그래서 모든 안전 (예: PCI와 EISA) 검색은 모든 관련된 카드들을 자동적으로 찾아주게 된 다. (여러개의 이더넷 카드를 가진 시스템에서 적어도 하나의 ISA 카드를 가지고 있는 경우 에는 여전히 다음의 과정중 하나를 해야만 한다.)

두번째 (그리고 새번째, 그리고...) 카드를 자동으로 검색하는데는 두가지 방법이 있다. 가장 쉬운 방법은 보통 LILO를 통해 하는것처럼 부팅시에 커널로 인수를 전달하는 것이다. 두번째 카드를 찾는 것은 부팅시에 `ether=0,0,eth1`처럼 간단한 인수를 사용하 면 된다. 이 경우에 `eth0`와 `eth1`는 부팅시에 찾아지는 순서대로 정해지게 된다. 만일 카드가 `eth0`는 `0x300`에, 그리고 `eth1`는 `0x280`에 있다면, 다음과 같이 하면 된다.

```
LILO: linux ether=5,0x300,eth0 ether=15,0x280,eth1
```

`ether=` 명령은 위에서 보여지는 바와 같이 IRQ + I/O + 이름을 받아들이게 된다. 전체 문법과 특정 카드 인자들, 그리고 LILO 팁들을 보려면 다음을 보면 된다. 10.1 (이더넷 인수 전달하기...)

이 부팅시의 인수들은 영구적이기 때문에 여러분은 매번 다시 쳐넣을 필요가 없다. LILO 설정 옵션중 '`append`'는 LILO 매뉴얼을 보기 바란다.

두번째 방법은 (권장하지 않는다) `Space.c` 를 편집해서 I/O 주소 항목의 `0xffe0` 부분을 영으로 바꿔주는 것이다. `0xffe0` 부분은 이 장치에 대해서 검색을 하지 않는다는 것을 말해준다 – 이것을 영으로 바꾼다는 것은 장치에 대한 자동검색을 할수 있게 하는 것이다.

여기서 알아둘 점은 여러분이 만약 리눅스를 두 네트워크 사이의 게이트웨이로 사용하려고 한다면, 커널을 IP 포워딩 가능으로 해서 재컴파일 해야만 한다는 것이다. 보통 구식 AT/286에 'kbridge'같은 소프트웨어를 사용하는 것이 더 좋은 해결방법이다.

여러분이 이것을 *넷 서핑* 도중에 보고 있다면, Donald가 그의 WWW 사이트에 갖고 있는 미니 하우투를 볼수도 있을 것이다. 다음을 확인해 보라. *Multiple Ethercards* `<http://cesdis.gsfc.nasa.gov/linux/misc/multicard.html>`.

3.3 ether= 명령이 아무것도 하지 않는다. 왜지?

위에서 설명한 것처럼, `ether=` 명령은 *단지* 커널안에 컴파일되어 들어있는 드라이버들에 대해서만 작동한다. 요즘 대부분의 배포판들은 모듈 형식으로된 드라이버들을 사용하므로 `ether=` 명령은 더이상 거의 사용되지 않는다.(몇몇 오래된 문서들은 이 변화를 반영하여 갱신되고 있다.) 만일 여러분이 이더넷 드라이버 모듈에 옵션들을 적용하려 한다면, *반드시* `/etc/conf.modules` 화일을 고쳐야만 한다.

만일 여러분이 *지금* 컴파일된 드라이버를 사용하고 있고, 여러분의 LILO 설정화일 에 `ether=`를 추가했다면, 바뀐 설정 화일로 실행되도록 `lilo`를 재실행하기 전까지는 효과가 없다는 것을 명심해라.

3.4 3.4 NE1000 / NE2000 카드(그리고 호환제품)들의 문제

Problem: PCI NE2000 호환카드가 v2.0.x로 부팅시 찾질 못한다.

Reason: v2.0.30 이하에서의 `ne.c` 드라이버는 단지 RealTek 8029 기반 호환카드들의 PCI ID 넘버만을 알고있기 때문이다. 그러므로 PCI NE2000 호환카드로 나온, 다른 PCI ID 넘버를 가진 카드들을 드라이버가 찾아 내지 못하는 것이다.

Solution: 가장 쉬운 해결책은 리눅스 커널버전 v2.0.31 (또는 그 이상)으로 업그레이드하는 것이다. 이들은 다섯가지 다른 NE2000-PCI 칩들에 대한 ID 넘버를 알고 있기 때문에, 부팅시에나 모듈이 적재되는 시간에 자동으로 그들을 찾아낼 것이다. 만일 여러분이 2.0.34 (또는 그 이상)으로 업그레이드 하면, 거기에는 오리지날 ISA/PCI 드라이버보다 약간 더 작고 보다 효율적인 PCI만의 특정 NE2000 드라이버가 있다.

Problem: PCI NE2000 호환 카드가 v2.0.x에서 부팅시나 ne.o 모듈을 적재할때 ne1000 (8비트 카드!) 라고 나오고, 그리고나서는 작동하지 않는다.

Reason: 몇몇 PCI 호환제품들은 바이트 폭의 접근을 구현하지 않는다.(그리고 진짜 100% NE2000 호환이 아니다). 이 때문에 NE1000 카드로 생각하고 찾아내는 결과가 나타나게 된다.

Solution: 위에서 설명했던 것처럼 v2.0.31 (또는 그 이상)으로 업그레이드 해야만 한다. 그 드라이버(들)은 현재 이 하드웨어 버그를 검사한다.

Problem: PCI NE2000 카드가 성능 팁 부분에 설명된대로 윈도우 사이즈를 줄일때에도 정말 최악의 성능을 나타낸다.

Reason: 개발해서 판매된지 십년도 더 된 오리지날 8390 칩의 스펙 표를 보면, 최상의 안정성을 위해 각 쓰기 작업하기 전에 칩이 느린 읽기를 요청한다고 알려져 있다. 그 드라이버는 v1.2 커널 때부터 기본적으로 그런 기능을 사용할수 없게 되어 있다. 한 사용자가 말하기로는 그 '잘못 된 기능'을 다시 사용가능하게 하면 값싼 PCI NE2000 호환 카드의 성능에 도움이 된다고 한다.

Solution: 이 문제의 해결책은 단지 한 사람한테서만 나왔기 때문에, 그렇게 희망적이지는 않다. 쓰기 전에 읽기를 다시 가능하게 고치는 것은 `linux/drivers/net/`안의 드라이버 화일 을 간단하게 편집하면 된다. `NE_RW_BUGFIX`를 포함하고 있는 줄의 주석을 제거하 고 커널이나 모듈을 적절하게 재컴파일해주면 된다. 만일 이것이 여러분에게 도움이 된다면, 성능의 차이와 카드/칩셋 종류를 기술하여 우리에게 e-mail을 보내주기 바란다. (`ne2k-pci.c` 드라이버에 대해서도 동일하게 할 수 있다.)

Problem: `ne2k-pci.c` 드라이버가 PCI NE2000 카드에서 `timeout waiting for Tx RDC`와 같은 에러 메세지를 보내고 제대로 작동하지 않는다.

Reason: 여러분의 카드와/또는 카드에서 PCI 버스로의 연결이 이 드라이버에서 사용된 long word I/O optimization을 다룰수 없는 것이다.

Solution: 우선, BIOS/CMOS 설정에서 안정적인 작동을 방해하는 PCI 버스 타이밍에 관련된 어떠한 설정이라도 확인해 보라. 그렇지 않다면 ISA/PCI `ne.c` 드라이버를 사용해서 (아니면 `ne2k-pci.c`에서 `#define USE_LONGIO`부분을 없애고) 카드를 사용하도록 해야한다.

Probem: ISA Plug and Play NE2000 (RealTek 8019같은)이 잡히지 않는다.

Reason: 원래의 NE2000 사양에는 (그리고 리눅스 NE2000 드라이버도) 플러그 앤 플레이에 대한 지원은 없다.

Solution: PnP를 사용할수 없게 하기 위해서 카드와 함께 따라오는 DOS 설정 디스크를 사용해서, 카드에 특정 I/O 주소와 IRQ를 설정한다. 그리고 `/etc/conf.modules`에 `options ne io=0xNNN`와 같은 라인을 추가한다. 여기서 `0xNNN`는 여러분이 카드에 설정한 16진수 I/O 주소이다. (여기서는 여러분이 모듈 드라이버를 사용한 다고 가정한다. 만일 아니라면 부트시에 `ether=0,0xNNN,eth0` 인수를 사용한다). 여러분은 또한 BIOS/CMOS 설정에 들어가서 PnP 대신에 Legacy-ISA용 IRQ에 표시해야 한다. 여러분이 만약 몇몇 다른 운영체제와의 호환성을 위해서 PnP를 가능한 상태로 남겨둬야 한다면 *isapnptools* 패키지를 찾아보라. `man isapnp`를 쳐서 이것이 여러분의 시스템에 이미 설치되어 있는지 확인해보라. 아니면, 다음의 URL을 찾아보라.

ISA PNP Tools `<http://www.roestock.demon.co.uk/isapnptools/>`

Problem: NE*000 드라이버가 부트 검색시에 'not found (no reset ack)'라는 메세지를 출력한다.

Reason: 이것은 위의 변화와 관계가 있다. 초기 확인작업 후에 8390은 검색된 I/O 주소에 있게되고 리셋이 이루어진다. 카드가 완전하게 리셋이 될때, 리셋이 끝났다고 알리게 된다. 여러분의 카드가 그렇지 않다면, 드라이버는 현재 어떠한 NE 카드도 없다고 가정하게 된다.

Solution: 드라이버에게 부팅시에 `0xbad`의 사용하지 않는 `mem_end` 16진수 값을 사용해 여러분이 안좋은 카드를 가지고 있다는 것을 알려줄수 있다. 여러분은 `0xbad`의 재정의를 사용할 때 카드에 영이 아닌 I/O 주소를 제공해야만 한다. 예를 들어, 리셋되지 않는 카드가 `0x340`에 있다면 다음과 같이 쓰면 될것이다.

```
LILO: linux ether=0,0x340,0,0xbad,eth0
```

이것은 여러분의 카드가 리셋을 받아들이지 않더라도 카드 탐색을 계속하도록 해준다. 만일 여러분이 드라이버를 모듈로 사용하고 있다면, I/O 주소를 준것처럼 `bad=0xbad` 옵션을 넣어줄수 있다.

Problem: 처음 네트워크에 접속할 때 NE*000 카드가 머신을 정지시킨다.

Reason: 이 문제는 1.1.57정도의 오래된 커널에서 현재에까지 보고되고 있다. 이것은 몇몇 소프트웨어 로 설정가능한 호환 카드들에서만 나타난다. 그들은 어떤 특별한 방법으로 초기화를 해주어야 한다.

Solution: 몇몇 사람들은 리눅스에서 카드를 작동시키기위해 웜부팅 (즉, loadlin 이나 '세손가락인사 - ctrl+alt+del:역자주')하기 전에 제공되는 DOS 소프트웨어 설정 프로그램이나 DOS 드라이버 를 실행할때에 나타난다고 보고했다. 이것은 이 카드들이 현재 리눅스 드라이버들이 하는 것 과는 약간 다르게, 특정한 방식으로 초기화되어야 한다는 것을 나타낸다.

Problem: `0x360`에서 NE*000 이더넷 카드가 잡히질 않는다.

Reason: NE2000 카드는 `0x20`의 I/O 폭을 갖는데, 이때문에 패러렐 포트의 주소인 `0x378`를 침범하게 된다. 그자리에는 두번째 플로피 콘트롤러(만일 있다면)가 `0x370`에 그리고 두번째 IDE 콘트롤러가 `0x376--0x377`가 있을수 있다. 만일 그 포트(들)가 이미 다른 드라이버에 의해 등록이 되어 있다면, 커널은 탐색을 하지 않을 것이다.

Solution: 여러분의 카드 주소를 `0x280, 0x340, 0x320`같은 주소로 옮기거나 패러렐 프린터 지원 없이 컴파일하라.

Problem: 뭔가를 프린트하기만 하면 네트워크가 '죽어버린다' (NE2000)

Reason: 위와 같은 문제이지만, I/O영역을 확인하지 않는 더 오래된 커널을 사용중이다. 위에서 처럼 해결하면 되고, 여러분이 쓰는 것보다 새 커널을 구하라.

Problem: NE*000 ethercard probe at 0xNNN: 00 00 C5 ... not found. (invalid signature yy zz)

Reason: 우선 먼저, 0xNNN 주소에 NE1000 또는 NE2000 카드가 있습니까? 그리고 만일 있다면, 하드웨어 주소가 제대로된 것처럼 나오는가? 그렇다면, 여러분은 형편없는 NE*000 호환카드를 갖고 있는 것이다. 모든 NE*000 호환제품들은 카드의 SA PROM의 14 와 15 번째 바이트에 `0x57`값을 가지고 있다. 여러분이 가진것은 그렇지 않다 – 대신 'yy zz'를 가지 고 있다.

Solution: 여기에는 두가지 방법이 있다. 가장 쉬운 방법은 위의 'no reset ack' 문제에서 설명한 것처럼 `0xbad` mem_end 값을 사용하는 것이다. 이렇게 하면 서명 확인을 하지않고 지 나갈 것이고, 영이 아닌 I/O 주소값도 주어질 것이다. 이 방법은 커널을 재컴파일할 필요가 없다.

두번째 방법은(해커들에게 해당되겠지만) 드라이버를 바꾸고, 여러분의 커널(또는 모듈)을 재컴파일하는 것이다. 그 드라이버(/usr/src/linux/drivers/net/ne.c)는 약 42번 라인정도에 "Hall of Shame(부끄러움의 전당)" 목록이 있다. 이 목록은 잘못된 호환품들을 찾아내는데 사용된다. 예를 들어, DFI 카드들은 14와 15 바이트에 `0x57`를 사용하는 대신, PROM의 처음 3바이트에 'DFI'를 사용한다.

Problem: 머신이 부팅중에 '8390...' 이나 'WD....' 메세지 바로 다음에서 멈춰버린다.

Solution: 여러분의 NE2000 주소를 `0x340` 같은 것으로 바꾸어라. 아니면, "ether=" 인수와 함께 "reserve=" 인수를 사용함으로써 다른 장치 드라이버의 검색에서 카드를 보호할수 있다.

Reason: 여러분의 NE2000 호환제품은 충분히 좋은 호환품이 아니다. 작동하는 NE2000은 어떠한 드라이버의 자동검색에도 걸린다. NE2000을 다른 자동검색에서 벗어나도록 덜 쓰이는 주소로 바꾸면, 여러분의 머신은 부팅될 것이다.

Problem: 부팅시에 SCSI 탐색도중 멈춰버린다.

Reason: 이것도 위의 문제와 같으므로, 이더넷 카드의 주소를 바꾸거나, 아니면 reserve/ether 부팅 인수들을 사용하면된다.

Problem: 부팅시에 사운드 카드를 찾는도중에 머신이 멈추어 버린다.

Reason: 아니다, 그것이 실제로는 조용한 SCSI 탐색도중이므로, 위의 문제와 같다.

Problem: NE2000 이 부팅시에 찾아지지 않는다 - 부트 메세지가 전혀 없다.

Solution: 그것이 찾아지지 않는데는 수많은 원인이 있을수 있기 때문에 '마법의 해결책'은 없다. 다음의 내용들은 여러분이 가능한 문제들을 해결하는데 도움을 줄 것이다.

1) 여러분이 필요로 하는 장치 드라이버들만 가지고 새 커널을 만든다. 여러분이 정말로 새 커널로 부팅하고 있는 것인지 확인하라. lilo를 실행하는 것을 까먹지 않았는지 등등..으로 인해 이전의 것으로 부팅될수 있다. (부팅시에 나오는 만든 시간/날짜를 자세히 보라.) 우리는 이전에 모든것을 다 했다.`System.map` 화일안의 `ne_probe`와 같은 이름들을 확인해서, 새 커널에 정말로 그 드라이버가 포함되어 있는지 확인하라.

2) 부트 메세지들을 주의해서 살펴보라. 그곳에 'NE*000 probe at 0xNNN: not found (어쩌구 저쩌구)' 같은 ne2k 검색에 관한 어떠한 언급이 있는지, 아니면 조용하게 실패하는지 말이다. 거기에는 큰 차이가 있다. 로그인한 뒤에 부트 메세지를 다시보려면 `dmesg|more`를 사용하거나, 부팅된 후 로그인 프롬프트가 나온뒤에 Shift-PgUp을 눌러서 화면을 위로 스크롤해 가며 보면된다.

3) 부팅한 후에, `cat /proc/ioports`를 치고 카드가 필요로하는 입출력공간 전부가 비어있는지 확인하라. 여러분의 카드가 `0x300`에 있다면 ne2k 드라이버는 `0x300-0x31f` 를 요구할 것이다. 만일 어떤 다른 장치 드라이버가 그 범위내에 한 포트라도 등록했다면, 그 주소의 검색은 되지 않고 다음 검색 주소로 넘어가 계속하게 될 것이다. 일반적인 경우에 lp 드라이버가 `0x378`를 갖거나 두번째 IDE 채널이 `0x376`를 가지므로 ne 드라이버가 `0x360-0x380`를 검색하지 못하게 한다.

4) `cat /proc/interrupts`에 대해서도 위와 같이 해보라. 여러분의 이더넷 카드가 설정되어 있는 인터럽트에 혹시 다른 장치가 등록되어 있는지 확인하라. 이 경우에는, 검색은 이루어지지만, 이더넷 드라이버는 원하는 IRQ 라인을 얻을수 없다며 부팅시해 크게 불평할 것이다.

5) 만일 여러분이 아직도 드라이버의 말없는 실패에 당황해하고 있다면, 드라이버를 편집해서 검색을 위한 몇줄의 printk()를 추가하라. 예를 들어, ne2k에서는 `linux/drivers/net/ne.c`를 다음과 같이 몇몇줄에 추가/삭제('+' 나 '-' 로 표기) 할수 있다.

```
    int reg0 = inb_p(ioaddr);

+     printk("NE2k probe - now checking %x\n",ioaddr);
-     if (reg0 == 0xFF)
```

```
+    if (reg0 == 0xFF) {
+       printk("NE2k probe - got 0xFF (vacant I/O port)\n");
        return ENODEV;
+    }
```

그렇게 하고나면 각각의 포트 주소에 대한 확인 메세지를 출력하게 되고, 여러분은 여러분의 카드 주소가 검색되는지 안되는지 볼수 있을 것이다.

6) 여러분은 또한 Don의 ftp 사이트(하우투내에 잘 설명되어 있다)에서 ne2k 점검 도구를 가져와서 여러분이 리눅스로 부팅한 후에 카드를 찾을수 있는지 없는지 볼수 있다. '`-p 0xNNN`' 옵션을 사용해서 카드를 찾을 곳이 어디인지 말해줄수 있다. (기본적으로 `0x300`가 설정되어 있지만 부팅시의 검색과는 달리 다른 주소에 대한 검색은 이루어 지지 않는다.) 카드를 찾았을 경우에 대한 결과 출력은 다음과 같다:

```
Checking the ethercard at 0x300.
  Register 0x0d (0x30d) is 00
  Passed initial NE2000 probe, value 00.
8390 registers: 0a 00 00 00 63 00 00 00 01 00 30 01 00 00 00 00
SA PROM  0: 00 00 00 00 c0 c0 b0 b0 05 05 65 65 05 05 20 20
SA PROM 0x10: 00 00 07 07 0d 0d 01 01 14 14 02 02 57 57 57 57

        NE2000 found at 0x300, using start page 0x40 and end page 0x80.
```

여러분의 리지스터 값과 PROM 값들은 아마 서로 다를 것이다. 알아둘 것은 16비트 카드의 경우 모든 PROM 값들은 두배이며, 이더넷 주소 (00:00:c0:b0:05:65)는 처음 행에, 그리고 두개의 `0x57` 사인은 PROM의 마지막에 나타난다.

`0x300`에 설치된 카드가 없을때의 결과 출력은 다음과 같다:

```
Checking the ethercard at 0x300.
  Register 0x0d (0x30d) is ff
  Failed initial NE2000 probe, value ff.
8390 registers: ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
SA PROM        0: ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
SA PROM 0x10: ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff

 Invalid signature found, wordlength 2.
```

`0xff` 값들은 비어있는 I/O 포트를 읽을때 반환되는 값이다. 만일 검색된 영역 안에 어떤 다른 하드웨어가 있다면, `0xff` 아닌 값들을 보게 될 것이다.

7) 제공되는 DOS 드라이버나 설정 프로그램을 실행하고 난 후에 DOS 부트 플로피에서 (loadlin으로) 리눅스로 웜부팅을 해보라. 이것은 카드를 초기화하는 좀 다른(즉 비표준적인) "마법"이다.

8) Russ Nelson의 ne2000.com 패킷 드라이버로 여러분의 카드를 볼수 있는지 해보라 – 만일 아니면, 상황은 별로 좋지 않다. 예는 다음과 같다.

```
A:> ne2000 0x60 10 0x300
```

인수들은 소프트웨어 인터럽트 벡터, 하드웨어 IRQ, 그리고 I/O 주소이다. 여러분은 어떠한 msdos 아카이브에서나 pktdrv11.zip을 얻을수 있다 – 현재 버전은 아마 11이상일꺼다.

3.5 SMC Ultra/EtherEZ 와 WD80*3 카드들의 문제

Problem: 다음과 같은 메세지가 나타난다:

```
eth0: bogus packet size: 65531, status=0xff, nxpg=0xff
```

Reason: 이것은 공유 메모리 문제이다.

Solution: 이문제의 가장 일반적인 원인은 ISA 메모리 장치들안에 매핑이 설정되지 않은 PCI 머신들 때문이다. 그러므로 받은 패킷들의 데이타를 가지고 있는 카드의 RAM 대신에 PC의 RAM(모두 `0xff` 값)을 끝까지 읽는다.

쉽게 고칠수 있는 다른 전통적인 문제들은 보드 충돌, 캐쉬를 가지거나 그 영역에 대해 'shadow ROM'을 가능할게 하는것, 아니면 여러분의 ISA 버스가 8Mhz보다 빨르게 작동하는 경우이다. 이들도 또한 이더넷 카드상의 메모리 실패 숫자가 많은데, 만일 여러분의 카드중에 그러한 것이 있다면 점검 프로그램을 실행해 보라.

Problem: SMC EtherEZ 가 비공유 메모리 (PIO) 모드에서 작동하지 않는다.

Reason: Ultra 드라이버의 오래된 버전들은 카드가 공유 메모리 모드에서만 작동하도록 되어 있다.

Solution: 커널 버전 2.0 이상에 포함된 드라이버는 programmed I/O모드에서의 작동도 지원한다. v2.0 이나 그 이상으로 업그레이드하라.

Problem: 구형 wd8003 과/또는 점퍼설정이 가능한 wd8013가 항상 IRQ를 잘못 얻는다.

Reason: 구형 wd8003 카드들과 점퍼설정이 가능한 wd8013 호환제품들은 드라이버가 IRQ 설정을 읽어들일 EEPROM을 가지고 있지 않다. 드라이버가 IRQ를 읽어들일수 없으면, 그다음에는 자동으로 IRQ를 찾는다. 그리고 만일 자동 IRQ가 영을 반환하면, 드라이버는 8비트 카드에게는 IRQ 5를, 16비트 카드에게는 IRQ 10을 할당한다.

Solution: 자동 IRQ 검색 코드를 피하려면, 여러분의 모듈 설정 화일에(또는 커널내의 드라이버일 경우 부팅시에 인수를 이용해서) 여러분의 카드 점퍼가 설정되어 있는 IRQ가 무엇인지 적어서 커널에 알려야 한다.

Problem: SMC Ultra 카드가 wd8013로 잡히는데, IRQ와 공유 메모리 주소가 잘못되게 잡힌다.

Reason: Ultra 카드는 wd8013과 매우 비슷해 보여서, 만일 Ultra 드라이버가 커널내에 존재하지 않으면, wd 드라이버는 ultra를 wd8013으로 잘못 인식하게 된다. ultra의 검색은 wd의 검색보다 먼저하게된다. 그래서 이런일은 보통 일어나지 않는다. ultra는 wd8013과 달리 IRQ와 메모리 주소를 EEPROM에 저장하므로, 가짜 값들이 보고된다.

Solution: 여러분에게 필요한 드라이버들만을 커널내에 담아 재컴파일을 한다. 만일 여러분이 한 머신에 wd 와 ultra 카드를 모두 가지고 있고, 모듈을 사용한다면 ultra 모듈을 먼저 로드하라.

3.6 3Com 카드들의 문제

Problem: 3c503이 IRQ N을 고르는데, 다른 장치들도 IRQ N을 필요로한다. (eg. CD ROM 드라이버, 모뎀, 등등.) 커널안에 컴파일하지 않고 이것을 고칠수는 없을까?

Solution: 3c503 드라이버는 {5, 9/2, 3, 4}의 순서대로 비어있는 IRQ를 탐색한다. 그리고 사용하고 있지 않은것을 고른다.드라이버는 카드가 `ifconfig`되어지고 있을때 고르게 된다.

만일 여러분이 모듈 드라이버를 사용하고 있다면, 여러분은 IRQ 값을 포함해서 많은 것들을 설정하기위해 모듈 인자들을 사용할수 있다.

아래에서는 IRQ9를 선택하고, 주소는 `0x300`,<무시된 값>, 그리고 if_port #1(외부 단자:external transceiver)로 정하고 있다.

```
io=0x300 irq=9 xcvr=1
```

또한, 만일 드라이버가 커널내에 컴파일되어 있으면, 부팅시에 LILO에 인자들을 넘겨줌으로해서 같은 값을 설정할수 있다.

```
LILO: linux ether=9,0x300,0,1,eth0
```

다음에서는 IRQ3을 선택하고, 기반 주소를 탐색하며, <무시된 값>, 그리고 기본 if_port #0 (내부 단자:internal transceiver)로 정한다.

```
LILO: linux ether=3,0,0,0,eth0
```

Problem: 3c503: 설정된 인터럽트 X 가 잘못된 것이므로, 자동 IRQ를 사용할 것임.

Reason: 3c503 카드는 단지 IRQ{5, 2/9, 3, 4}중에 하나만 사용한다. (이들은 카드에 연결되어있는 선일뿐이다.) 만일 여러분이 위에 설정된 값이 아닌 IRQ 값을 넘겨주면, 여러분은 위와같은 메세지를 받게될 것이다. 보통, 3c503에는 특정 인터럽트 값을 정해줄 필요가 없다. 3c503은 ifconfg시에 자동IRQ를 사용해서, {5, 2/9, 3, 4}중의 하나를 IRQ값으로 갖게 된다.

Solution: 위에 나열된 올바른 IRQ값들중에 하나를 사용하거나, IRQ 행을 전혀 사용하지 말고 자동IRQ를 쓸수있게 하라.

Problem: 제공되는 3c503 드라이버들은 AUI (thicknet) 포트를 사용하지 않는다. 어떻게 기본 thinnet 포트를 통해 사용할수 있을까?

Solution: 3c503 AUI 포트는 커널내장 드라이버의 경우에는 부팅시에, 모듈의 경우는 모듈 삽입시에 선택할수 있다. 그 선택은 현재 사용되지 않는 dev->rmem_start 변수의 낮은 비트에 오버로드되어진다. 그러므로 커널에 내장된 드라이버에 사용하기 위한 부트시의 인자는 다음과 같다.

```
LILO: linux ether=0,0,0,1,eth0
```

모듈로 적재할때 AUI 포트를 정해주려면, 단지 `xcvr=1`를 여러분의 I/O 와 IRQ 값과 함께 모듈 옵션 행에 추가하면 된다. To specify the AUI port when loading as a module, just append `xcvr=1` to the module options line along with your I/O and IRQ values.

3.7 어떤 카드에도 특화되지 않은 FAQ들.

3.7.1 리눅스와 ISA Plug and Play 이더넷 카드들

최선의 결과를 얻기 위해서는(그리고 악영향을 최소한으로) 여러분의 카드에 딸려오는 (보통 DOS)프로그램을 사용해서 PnP 메카니즘을 사용하지 못하게 하고, I/O 주소와 IRQ를 정하면 된다. 여러분이 사용하는 I/O 주소가 부트시에 드라이버에 의해 탐색되는지 확인하고, 만일 모듈을 사용한다면 `/etc/conf.modules`안에 `io=`에 주소를 적어준다. 여러분은 또한 BIOS/CMOS 설정에 들어가서 IRQ를 PnP 대신에 Legacy-ISA로 설정해야 한다(여러분의 컴퓨터가 이 옵션을 가지고 있다면 말이다).

여러분은 보통 DOS기반의 설정 프로그램을 실행하기 위해 DOS를 설치할 필요는 없다. 제공되는 플로피 디스크에서 바로 실행하기 위해 DOS 부팅 디스크를 사용해도 되고, 공짜인 OpenDOS 와 FreeDOS를 다운받아 쓸수도 있다.

만일 여러분이 다른 운영체제와의 호환을 위해 PnP가 필요하다면 부팅때마다 리눅스가 카드를 설정하도록 하기 위해서 isapnptools 패키지를 사용해야 한다. 이 경우에도 여전히 드라이버가 탐색할수 있도록 카드의 I/O 주소를 정해주거나 `io=` 옵션을 주고 확인해야 한다.

3.7.2 이더넷 카드가 부팅시에 잡히지 않는다.

이 경우의 일반적인 원인은 사람들이 그들의 특정 카드에 대한 드라이버를 내장하지 않은 커널을 사용하기 때문이다. 모듈식 커널의 경우에는, 모듈의 적재 요청이 없다거나, 아니면 모듈 옵션으로 특정한 I/O 주소를 정해주지 않았기 때문이다.

만일 여러분이 대부분의 리눅스 배포판에서 설치하는 것처럼 모듈 기반의 커널을 사용하고 있다면, 그 배포판이 제공하는 설정 유틸리티를 사용해서 카드의 모듈을 선택하길 바란다. ISA 카드들의 경우에는, 만일 설정 유틸리티가 옵션에 관해 물어본다면, I/O 주소를 정해서 옵션으로(예를 들어 `io=0x340`) 추가하면 된다. 만일 설정 유틸리티가 없다면, 여러분은 올바른 모듈 이름(그리고 옵션들)을 `/etc/conf.modules`에 추가해야 한다 – 더 자세한 내용은 `man modprobe`를 보기 바란다.

만일 여러분이 배포판에서 제공되는 미리 컴파일된 커널을 사용하고 있다면, 여러분이 설치한 커널의 문서를 보고, 여러분의 특정 카드에 대한 지원이 들어있는지를 확인하라. 만약 들어있지 않다면, 여러분의 카드에 대한 지원이 있는 커널을 구하거나, 여러분 자신의 것을 만들면 된다.

만일 여러분이 자신에게 필요한 드라이버들만을 가지고 자신의 커널을 만들수 있다면, 커널의 크기도 줄고 (응용 프로그램들을 위해 여러분의 중요한 RAM을 절약!) 민감한 하드웨어를 망가트릴수도 있는 수많은 장치에 대한 탐색을 줄일수 있다. 커널을 만드는 것은 들리는 것처럼 그리 복잡하지 않다. 여러분이 필요한 드라이버가 무엇인지에 대한 질문에 네 또는 아니오로 대답만 해주면, 나머지는 알아서 한다.

그 다음 주된 원인은 여러분의 카드가 필요로하는 I/O 공간의 일부분을 다른 장치가 사용하고 있는 경우이다. 대부분의 카드들은 16 또는 32 바이트의 I/O 공간을 필요로한다. 만일 여러분의 카드가 `0x300` 에 32 바이트의 공간으로 설정되어 있다면, 드라이버는 `0x300-0x31f`를 요구하게 된다. 만일 어떤 다른 장치 드라이버가 그 범위내에 어디라도 등록이 되어 있으면, 그 주소에 대한 탐색은 이루어지지 않으며, 드라이버는 아무말없이 다음 탐색 주소로 넘어가서 탐색을 계속하게 된다. 그러므로, 부팅 후에, `cat /proc/ioports`를 쳐서 카드가 필요로 하는 I/O 주소 공간 모두가 비어있는지를 확인해 보아야 한다.

또다른 문제는 여러분의 카드가 점퍼로 설정된 I/O 주소가 기본적으로 탐색이 되지 않는 것이다. 각 드라이버의 탐색하는 주소 목록은 드라이버 소스내의 주석문 다음에서 쉽게 발견할 수 있다. 비록 여러분의 카드가 설정된 I/O 주소가 목록에 없더라도, 부팅시에(커널에 내장된 드라이버의 경우) `ether=`명령을 통해 주소를 넘겨줄수 있다. 이 명령은 다음 장소에 설명 되어 있다. 10.1 (이더넷 인수들 넘겨주기...) 모듈 드라이버들의 경우에는 `/etc/conf.modules`내에 `io=` 옵션을 사용해서 기본적으로 탐색되지 않는 주소를 정해줄수 있다.

3.7.3 `ifconfig`가 카드에 대해 잘못된 I/O 주소를 보여준다.

그렇지 않다. 여러분이 그 내용을 잘못 해석한 것일뿐이다. 이것은 버그가 *아니다*. 그리고 보여주는 숫자들은 올바른 것이다. 이러한 현상은 처음 정해진 I/O 포트와 상충되는 자리에 실제 8390칩을 가진 몇몇 8390 기반의 카드들(wd80x3, smc-ultra, 등등)한테서 일어난다. 이것은 `dev->base_addr`에 있는 값으로, `ifconfig`로 볼수 있다. 만일 여러분이 여러분의 카드가 사용하는 포트의 전체 범위를 보려면, `cat /proc/ioports`를 해보면, 여러분이 기대했던 숫자들을 볼수 있을 것이다.

3.7.4 PCI 머신은 카드를 찾아내지만 드라이버는 탐색에 실패한다.

몇몇 PCI BIOS들은 전원을 켰을때 모든 PCI 카드들을 사용할수 있게 하지는 않는데, 특히 BIOS 의 옵션이 'PNP OS'가 사용가능하게 되어있다면 그렇다. 이 잘못된 부분은 아직도 여전히 몇몇 리얼모드 드라이버들을 사용하고 있는 현재의 윈도우 계열을 지원하기 위한 것이다. 이 옵션을 disable로 하거나, 사용할수 없게 설정된 카드를 사용할수 있게하는 코드를 가진 새로운 드라 이버로 업그레이드하면 된다.

3.7.5 PCI 머신내의 공유 메모리 ISA 카드들이 작동하지 않는다 (`0xffff`)

이것은 보통 수많은 `0xffff` 값들을 읽어들이기 때문에 일어난다. PCI 머신내에서는 공유 메모리 카드들은

만일 여러분이 PCI ROM BIOS/CMOS SETUP 설정을 제대로 하지 않는다면 작동하지 않는다. 여러분은 여러분의 카드가 사용하려고 하는 메모리 영역에 대해서 ISA 버스에서 공유 메모리 접근을 허용하도록 설정해야 한다. 만일 여러분이 어떻게 설정해야 하는지 이해하지 못하겠다면, 여러분을 도와주는 사람이나 지역의 고수들에게 물어보라. AMI BIOS의 경우에는, 보통 "Plug and Play"부분이 있고 그안에 "ISA Shared Memory Size" 와 "ISA Shared Memory Base" 설정이 있다. wd8013 이나 SMC Ultra와 같은 카드들의 경우에는 기본적으로 'Disabled'라고 되어있는 것을 16kB로 크기를 바꾸어주고, 여러분 카드의 공유 메모리 주소를 바꾸어주면 된다.

3.7.6 카드가 데이타를 보내는것 같은데 아무것도 받지를 못한다.

`cat /proc/interrupts` 해보라. 여러분의 카드가 생성한 실행중인 인터럽트 이벤트의 총숫자가 목록에 있을 것이다. 만일 그것이 여러분이 카드를 사용하려고 할때에도 0이거나 더이상 증가하지 않는다면 컴퓨터 내에 물리적으로 인터럽트가 충돌하는 다른 장치가 있는 것이다(다른 장치의 드라이버가 설치/사용 가능한가는 아닌가는 신경쓸 필요도 없다). 두 장치중에 하나의 IRQ를 비어있는 것으로 바꾸어라.

3.7.7 비동기 전송 모드 (ATM) 지원

Werner Almesberger 는 리눅스에서의 ATM 지원을 작업하고 있다. 그는 Efficient Networks ENI155p 보드(*Efficient Networks* `<http://www.efficient.com/>`) 와 Zeitnet ZN1221 보드 (*Zeitnet* `<http://www.zeitnet.com/>`) 를 사용해서 작업중이다.

Werner 는 ENI155p용 드라이버가 좀더 안정적이고, ZN1221용 드라이버는 현재 완료되지 않았다고 말한다.

최신의/갱신된 자료는 다음의 URL을 확인해보기 바란다.

리눅스 ATM 지원 `<http://lrcwww.epfl.ch/linux-atm/>`

3.7.8 기가바이트 이더넷 지원

리눅스에 기가바이트 이더넷에 대한 지원이 있나?

있다, 현재 적어도 두개가 있다. Packet Engines G-NIC PCI 기가비트 이더넷 아답터용 드라이버는 v2.0과 v2.2 커널에서 사용할수 있다. 더 자세한 내용과 지원, 그리고 드라이버 업데이트는 다음을 보라.

`http://cesdis.gsfc.nasa.gov/linux/drivers/yellowfin.html`

v2.2 커널에서 사용할수 있는 `acenic.c` 드라이버는 Alteon AceNIC 기가비트 이더넷 카드와 3Com 3c985 같은 다른 Tigon 기반 카드들에서 사용할수 있다. 그 드라이버는 NetGear GA620 에서도 작동해야 하지만, 이것은 아직 확인되지 않았다.

3.7.9 FDDI 지원

리눅스에 FDDI 지원이 있나?

있다. Larry Stefani은 Digital's DEFEA (FDDI EISA)와 DEFPA (FDDI PCI) 카드들로 v2.0용 드라이버를 만들었다. 이것은 v2.0.24 커널에 포함되어 있다. 현재 다른 카드들에 대한 지원은 없다.

3.7.10 Full Duplex 지원

Full Duplex 가 20MBps를 내는가? 리눅스가 그것을 지원하는가?

Cameron Spitzer는 full duplex 10Base-T 카드들에 대하여 다음과 같이 썼다: "만일 여러분이 full duplex 스위치 허브에 연결한다면, 여러분의 시스템은 충분히 빠르겠지만 아주 월등히는 아니다. 이것은 양방향 연

결이 계속되도록 할 뿐이다. full duplex 10BASE-2 나 10BASE-5 같은 것(thin 과 thick coax)은 없다. Full Duplex는 아답터의 충돌 검출을 사용할수 없게 만듦으로 써 작동한다. 이것이 동축 케이블로 할수없는 이유이다; LAN은 그길로는 가지 않을 것이다. 10BASE-T (RJ45 단자)는 보내고 받기위해 분리된 선들을 사용한다. 그래서 동시에 양방향으로 가는 것이 가능하다. 스위칭 허브는 충돌 문제를 살핀다. 전송률은 10Mbps이다.

그러므로 여러분이 볼수 있는 것과 같이, 여전히 단지 10Mbps로 보내고 받을수 있고, 두배의 성능 향상은 기대하지 않는것이 좋다. 그것이 지원되는지 안되는지변믹 카드와 사용가능한 드라이 버에 달려있다. 몇몇 카드들은 자동처리를 하기도 하고, 몇몇은 드라이버 지원을 필요로하며, 또 어떤것들은 카드의 EEPROM 설정에서 사용자들이 옵션을 설정해줄 필요가 있다. 단지 심각한/ 걱정많은(serious/heavy) 사용자들만이 두 모드간의 차이에 신경쓸 뿐이다.

3.7.11 SMP 머신상의 리눅스를 위한 이더넷 카드들

만일 여러분이 여분의 돈을 다중 프로세서(MP) 컴퓨터에 쓰려고 한다면, 그만큼 좋은 이더넷 카드를 사야한다. v2.0 커널에서는 정말 이야기의 대상이 되지도 않았지만, v2.2에서는 되고 있다. 대부분의 똑똑하지 못한 구형의 카드들(예를 들어 ISA 버스 PIO 와 공유 메모리 디자인) 은 MP 머신상에서 사용에 대해 조금도 고려하지않고 만들어졌다. 결론적으로 말하자면, 똑똑한 현대적 디자인의 카드를 사고 MP 작업을 다룰 수 있게 작성된 (또는 갱신된) 드라이버가 있는지 확인하라. (핵심은 '현대적 디자인'이다 - PCI NE2000은 현대적인 버스상의 적어도 10년이상된 구 디자인이다.) 드라이버 소스내의 `spin_lock` 를 찾으면, 이는 그 드라이버가 MP 작업을 다룰수 있게 작성된 것이라는 것을 알려주는 것이다. 왜 여러분이 MP를 사용하기 위해서 좋은 카드를 사야 하는지에 대해서는 (그리고 그렇지 않을 경우에는 어떠한 일이 일어나 는지) 다음을 보기 바란다.

커널 v2.0 에서는, '커널내에'(즉, 커널 데이타를 바꾸고, 또는 장치 드라이버들을 실행하는데) 언제나 단지 하나의 프로세서만이 허용되었다. 그러므로 카드의 관점에서는 (그리고 연관된 드 라이버에서도) 단일 프로세서 (UP) 작업과는 작동이 계속된다는 것 말고는 아무런 차이가 없었 다. (이것이 동작하는 리눅스의 MP버전을 구하는 가장 손쉬운 방법이다 - 일정 시간에 단지 하 나의 프로세서만이 전체 커널에 큰 락을 걸도록 하는 것이다. 이 방법은 여러분도 알다시피 동 시에 같은 것을 두 프로세서가 변경하지 못하게 하는 것이다

특정 시간에 커널내에 하나의 프로세서만이 허용되는 상황하에서, 여러분은 실행되는 프로그램 이 자신을 포함하고 의도적인 계산을 할 경우에만 MP 성능을 얻을수 있다. 만일 프로그램이 디스크나 네트워크를 통하여 데이타를 읽고 쓰는 일 같은 입/출력(I/O)을 많이 한다면, 커널내 에 실행중인 하나의 프로세서가 장치 드라이버들의 입출력 요청을 실행하기 위해 시도하는 동안에 다른 모든 프로세서들은 그들의 입출력 요청이 끝나기만을 기다려야 한다. 커널이 병목 이 되어 단지 하나의 프로세서만이 실행중에 있게 되므로, single-lock 인, 입출력이 많은 MP 머신의 성능은 급속도로 단일 프로세서 머신에 가깝게 떨어지게 된다.

이것은 생각했던 것보다 확실히 떨어지기 때문에 (특히, 파일/WWW 서버, 라우터, 등등) v2.2 커널에서는 더 좋은 grained locking을 가지게 되었다 - 이것은 동시에 하나이상의 프로세서가 커널내에 존재할수 있다 는 것을 뜻한다. 전체 커널에 대한 하나의 big lock 대신에, 하나 이상 의 프로세서가 동시에 데이타를 복제하는 것을 방지하기 위해서 다수의 작은 락들이 존재하게 되었다. - 즉 하나의 프로세서가 네트워크 카드의 드라이버를 실행하는 동안에, 다른 프로세서 는 동시에 디스크 드라이브에 대한 드라이버를 실행할수 있다.

Okay, with that all in mind here are the snags: The finer locking means that you can have one processor trying to send data out through an ethernet driver while another processor tries to access the same driver/card to do something else (such as get the card statistics for `cat /proc/net/dev`). Oops - your card stats just got sent out over the wire, while you got data for your stats instead. Yes, the card got confused by being asked to do two (or more!) things at once, and chances are it crashed your machine in the process.

그래서, UP에서 작동하는 드라이버들은 더이상 충분치 않다 - 이들은 설정 데이타의 받고, 전송하고, 복사하는 세가지 작업들을 카드가 안정된 작동을 할정도로 직렬화된 카드의 접근제어 락들을 갱신해야만 한다. The scary part here is that a driver not yet updated with locks for stable MP operation will probably appear to be working in a MP machine under light network load, but will crash the machine or at least exhibit strange behaviour when two (or more!) processors try to do more than one of these three tasks at the same time.

The updated MP aware ethernet driver will (at a minimum) require a lock around the driver that limits access at the entry points from the kernel into the driver to 'one at a time please'. With this in place, things will be serialized so that the underlying hardware should be treated just as if it was being used in a UP machine, and so it should be stable. The downside is that the one lock around the whole ethernet driver has the same negative performance implications that having one big lock around the whole kernel had (but on a smaller scale) - i.e. you can only have one processor dealing with the card at a time. [Technical Note: The performance impact may also include increased interrupt latencies if the locks that need to be added are of the `irqsave` type and they are held for a long time.]

Possible improvements on this situation can be made in two ways. You can try to minimize the time between when the lock is taken and when it is released, and/or you can implement finer grained locking within the driver (e.g. a lock around the whole driver would be overkill if a lock or two protecting against simultaneous access to a couple of sensitive registers/settings on the card would suffice).

However, for older non-intelligent cards that were never designed with MP use in mind, neither of these improvements may be feasible. Worse yet is that the non-intelligent cards typically require the processor to move the data between the card and the computer memory, so in a worst case scenario the lock will be held the whole time that it takes to move each 1.5kB data packet over an ISA bus.

The more modern intelligent cards typically move network data directly to and from the computer memory without any help from a processor. This is a big win, since the lock is then only held for the short time it takes the processor to tell the card where in memory to get/store the next network data packet. More modern card designs are less apt to require a single big lock around the whole driver as well.

3.7.12 Alpha/AXP PCI 보드들 상의 리눅스를 위한 이더넷 카드들

v2.0에서는, 단지 3c509, depca, de4x5, pcnet32, 그리고 모든 8390 드라이버들(wd, smc-ultra, ne, 3c503, 등등.)만이 DEC Alpha CPU 기반 시스템들상에서 작동할수 있을 정도로 '아키텍처 독립적'으로 만들어졌다. Donald의 WWW 페이지에서도 아키텍처 독립적으로 만들어진 다른 업데이트된 PCI 드라이버들을 찾을수 있을 것이다.

드라이버를 아키텍처 독립적으로 바꾸는 것은 복잡하지 않다. 여러분은 단지 다음을 따라 하면 된다.

-multiply all `jiffies` related values by HZ/100 to account for the different HZ value that the Alpha uses. (i.e `timeout=2;` becomes `timeout=2*HZ/100;`)

-replace any I/O memory (640k to 1MB) pointer dereferences with the appropriate readb() writeb() readl() writel() calls, as shown in this example.

```
-       int *mem_base = (int *)dev->mem_start;
-       mem_base[0] = 0xba5eba5e;
+       unsigned long mem_base = dev->mem_start;
+       writel(0xba5eba5e, mem_base);
```

-replace all memcpy() calls that have I/O memory as source or target destinations with the appropriate one of `memcpy_fromio()` or `memcpy_toio()`.

Details on handling memory accesses in an architecture independent fashion are documented in the file `linux/Documentation/IO-mapping.txt` that comes with recent kernels.

3.7.13 SUN/Sparc 하드웨어 상의 리눅스를 위한 이더넷

스팍 상의 모든 최신의 정보는 다음의 사이트에서 볼수 있다.

Linux Sparc <http://www.geog.ubc.ca/sparc>

알아둬야 할 것은 몇몇 스팍 이더넷 하드웨어는 호스트 컴퓨터로부터 MAC 주소를 가져오므로, 여러분은 여러개의 인터페이스를 모두 같은 동일한 MAC 주소로 해줄수 있다. 만일 여러분이 동일한 네트워크 상에 하나 이상의 인터페이스를 놓아야 한다면, `ifconfig`에 유일한 MAC 주소를 할당하기 위해 `hw` 옵션을 사용하라.

PCI 드라이버들을 스팍 플랫폼에 포팅하는데의 문제는 위에서 언급한 AXP 플랫폼의 경우와 같다. 또한 여기에는 스팍이 빅 엔디안을, 그리고 AXP와 ix86이 리틀 엔디안이기 때문에, 이 엔디안에 관한 문제도 있다.

3.7.14 다른 하드웨어상의 리눅스를 위한 이더넷

여기에는 리눅스가 실행될수 있는, Atari/Amiga (m68k) 같은 몇몇 다른 하드웨어 플랫폼이 있다. Sparc의 경우에는 각 리눅스 포트의 홈 사이트에 가는것이 그 플랫폼에서 현재 지원되는 것을 볼수 있는 가장 좋은 방법이다. (그런 사이트들이라면 링크를 환영합니다 - 여기로 보내주 세요!)

3.7.15 허브없이 10 또는 100 BaseT 연결하기

허브없이 10/100BaseT (RJ45) 기반 시스템들을 함께 연결할수 있는가?

여러분은 2 머신은 쉽게 이을수 있지만, 그 이상은 별도의 장비들/기즈모들(영화 '그렘린'에 나오는 동물이름인것 같네요:역자주)이 필요하다. 다음을 보라. 6.2 (Twisted Pair) – 이 글은 어떻게 해야 하는지에 대해 설명하고 있다. 그리고 몇개의 선과 장비들을 교차해가며 함께 허브에 물릴수는 없다. 허브에서 복제됨 없이 충돌 신호를 보정하는것은 불가능하다.

3.7.16 SIOCSIFxxx: No such device

나는 부팅시에 'SIOCSIFxxx: No such device' 메세지를 받았다. 'SIOCADDRT: Network is unreachable'이라는 메세지에 이어서 말이다. 뭐가 잘못된건가?

여러분의 이더넷 장치가 부트/모듈 삽입시에 탐색되지 않고, `ifconfig`와 `route`를 실행하면, 작동시킬 장치가 없다고 한다. `dmesg | more`를 사용해서 부트 메세지를 살펴보고 이더넷카드 탐색에 대한 어떤 메세지가 없는지 보라.

3.7.17 SIOCSFFLAGS: Try again

'ifconfig'를 실행하자 'SIOCSFFLAGS: Try again' 이라는 메세지를 받았다 – 헛?

여러분의 이더넷 카드가 사용하려고 하는 IRQ를 어떤 다른 장치가 가져서, 이더넷 카드가 그 IRQ를 사용하지 못하는 것이다. 몇몇 장치들은 그들이 IRQ가 필요할때 잡았다가 다시 작업이 끝나면 풀어주므로, 이것을 할당하기 위해 리부팅할 필요가 없다. 예를들면 몇몇 사운드 카드, 시리얼 포트, 플로피 디스크 드라이버, 등등이 있다. 여러분은 `cat /proc/interrupts` 라고 쳐서 어느 인터럽트가 현재 *사용중인*가를 볼수 있다. 대부분의 리눅스 이더넷 카드 드라이버들은 'ifconfig'를 통해 사용하려고 열렸을 경우에만 IRQ를 차지한다. 만일 여러분이 필요한 IRQ 라인을 다른 장치가 '놓고 가게'할수 있다면, 여러분은 ifconfig로 '다시 시도할'수 있을 것이다.

3.7.18 'ifconfig'를 사용해서 00:00:00:00:00:00 값의 HW-addr로 UNSPEC 연결

아무런 인수없이 ifconfig를 하면, LINK가 UNSPEC (10Mbs 이더넷 대신)이고 내 하드웨어 주소는 모두 영이다.

이것은 사람들이 그들의 커널 버전보다 높은 새 버전의 'ifconfig' 프로그램을 실행하기 때문에 일어난다. 이 새버전의 ifconfig는 구형 커널과 함께 사용될 때 이러한 속성들을 보고하지 못한다. 여러분은 커널도 업그레이드 하거나, ifconfig를 '다운그레이드'하거나, 아니면 간단히 무시할수 있다. 커널은 여러분의 하드웨어 주소를 알고 있으므로, ifconfig가 그것을 읽지 못한다고 해서 정말로 무슨 일이 일어나는 것은 아니다.

만일 여러분이 사용하는 `ifconfig` 프로그램이 여러분이 사용하고 있는 커널보다 아주 많이 구형일 경우에는 엉뚱한 정보를 얻어낼수도 있다.

3.7.19 엄청난 양의 RX 와 TX 에러들

아무런 인수들없이 ifconfig를 실행하면, 보내고 받은 패킷 모두에 엄청난 양의 에러숫자가 있는것을 본다. 모두 제대로 작동하는 것 같은데 – 무엇이 잘못된 것인가?

다시 잘 보라. 이것은 `RX packets` *big number* **PAUSE** `errors 0` **PAUSE** `dropped 0` **PAUSE** `overrun 0` 이다. `TX` 열의 경우도 마찬가지다. 그러므로 여러분이 본 큰 숫자들은 여러분의 머신이 주고 받은 패킷의 총 숫자이다. 아직도 혼란스럽다면, 대신에 `cat /proc/net/dev`라고 쳐보라.

3.7.20 `/dev/` 내의 이더넷 카드들을 위한 내용물들

나는 /dev/eth0가 /dev/xxx에 링크되어 있다. 이것이 옳은 것인가?

여러분이 들은것과는 달리, /dev/* 내의 파일들은 사용되지 않는다. 여러분은 `/dev/wd0, /dev/ne0`와 같은 어떠한 비슷한 내용들도 지울수 있다.

3.7.21 리눅스와 "trailers"

'ifconfig"를 내 이더넷 카드에 사용할때 트레일러를 사용할수 없게 할수는 없나?

여러분은 여러분이 원하지 않더라도, 트레일러를 사용할수 없게 할수 없다. '트레일러'는 네트워킹 레이어에서의 데이타 복사를 피하기 위해 만들어진 것이다. 이 아이디어는 'H' 크기의 작은 고정 크기 헤더를 사용하기 위한 것으로, 패킷의 끝에 다양한 크기의 헤더 정보를 넣고, 페이지가 시작하기 전에 모든 패킷의 'H' 바이트를 할당한다. 이것은 좋은 생각이지만, 실제로는 잘 동작하지 않는것으로 드러났다. 만일 누군가가 '-트레일러'의 사용을 제안한다면, 이것은 수염소의 피의 희생과 같은 것임을 알아두라. 이것은 문제를 해결하는데 아무런 도움도 주지 못하나, 만일 그 스스로 문제가 고쳐진다면 그 누군가는 깊은 마법같은 지식을 알릴수 있을 것이다.

3.7.22 저수준 이더넷 장치에 접근하기

리눅스에서 TCP/IP나 그러한 것들을 통하지 않고 저수준 이더넷 장치에 접근하려면 어떻게 해야 하나?

```
int s=socket(AF_INET,SOCK_PACKET,htons(ETH_P_ALL));
```

이것은 여러분이 모든 프로토콜 타입을 받는 소켓을 제공한다. `recvfrom()`를 호출하면 sa_family내의 장치 타입과 sa_data 배열내의 장치이름으로 sockaddr를 채울 것이다. 나는 누가 리눅스용 SOCK_PACKET을 처음 개발했는지 모르지만 정말 대단한 것이다. 여러분은 `sendto()`를 호출해서 가공하지 않은 것들도 보낼 수 있다. 물론 루트권한을 가지고 있어야만 한다.

제 4 절 성능 팁들

만일 여러분이 낮은 이더넷 출력이나 좀더 빠른 ftp 전송속도를 얻고자한다면, 여기에 몇가지 팁들이 있다.

`ttcp.c` 프로그램은 전송속도를 측정하는데 좋다. 또다른 일반적인 방법은 `ftp> get large_file /dev/null`를 치고 전송하는 머신의 버퍼 캐쉬를 보는 것이다. 여기서 `large_file`는 1MB보다 큰 파일을 말한다. ('get'은 적어도 두번은 해야 한다. 첫번째 시도때 보내는 머신의 버퍼캐쉬에 적재가 되기 때문이다.) 여러분이 측정할 때 디스크 에서 파일로 접근하는 속도에는 관심이 없으므로 버퍼 캐쉬 안에 파일이 있어야 한다. 또한 이때문에 들어오는 데이타를 `/dev/null`로 보내는 것이다.

4.1 일반적인 개념들

8 비트 카드조차도 아무런 문제없이 패킷을 back-to-back으로 보낼수 있다. 문제는 카드가 더 이상 패킷이 들어올수 있게 충분한 공간을 빠르게 만들수 없기 때문에 받은 패킷들을 컴퓨터가 받아들이지 못하는데 있다. 만일 컴퓨터가 카드의 메모리내에 이미 받은 패킷들을 빨리 비우지 않으면, 카드는 새 패킷을 받아들일 공간이 없게 된다.

이러한 경우에 카드는 새 패킷을 버리거나, 이전에 받은 패킷의 맨 위에서 덮어쓰게 된다. 어느 것이든 재전송을 야기/요구함으로써 부드러운 전송 흐름을 방해하게 되고, 심각한 성능 저하를 나타낼수 있다!

더 많은 메모리를 내장한 카드들은 더 많은 패킷들을 "저장"할수 있고, 패킷을 버리지 않고 더 많은 양의 back-to-back 패킷 흐름을 제어할수 있다. 다시말하면, 카드가 패킷들을 버리지 않기위해 버퍼내의 패킷들을 밀어내도록 요청하지 않는다는 뜻이다.

대부분의 8비트 카드들은 8kB의 버퍼를 가지며, 대부분의 16비트 카드들은 16kB의 버퍼를 갖는 다. 대부분의 리눅스 드라이버들은 버퍼로 3kB를 남겨두며(두개의 전송 버퍼), 8비트 카드를 위해서 단지 5kB를 남겨놓는다. 이 공간은 단지 풀 사이즈의 이더넷 패킷 세개(1500 bytes) 정도의 공간일 뿐이다.

4.2 ISA 카드들과 ISA 버스 속도

위에서 언급했다시피, 패킷들이 카드에서 충분히 빠르게 없어지면, 수신 패킷 버퍼 메모리가 작아도 패킷을 버리거나 넘쳐버리는 일을 일어나지 않는다. 카드에서 컴퓨터의 메모리로 패킷을 이동하는 비율은 두개가 만나는 부분 - 이것이 ISA 버스 속도이다 - 의 데이타 속도에 달려 있다. (만일 CPU가 386sx-16이라도, 이것은 적용된다)

권장하는 ISA 버스 클럭은 약 8MHz이지만, 많은 메인보드와 주변기기들은 더 높은 주파수로 동작할수 있다. ISA 버스 클럭 주파수는 보통 메인보드/CPU 클럭 주파수를 나눈 것을 선택하는 것으로, CMOS 설정에서 정할수 있다. 몇몇 ISA 와 PCI/ISA 메인보드는 이 옵션을 가지고 있지 않아서 출하시 기본값으로만 사용할수 있다.

예를 들어, 8비트 WD8003EP 카드가 꼽힌, 40MHz 486에서 TTCP 프로그램으로 측정한 서로다른 ISA 버스 속도가 아래에 있다.

```
ISA Bus Speed (MHz)      Rx TTCP (kB/s)
-------------------      --------------
6.7                      740
13.4                     970
20.0                     1030
26.7                     1075
```

여러분은 *어떠한* 10Mb/s 이더넷 카드로도, TCP/IP를 사용해서 1075kB/s이상의 결과를 얻지 못할것이다. 그러나, 모든 시스템에서 빠른 ISA 버스 속도가 나올거라고는 기대하지 말라. 대부분의 시스템에서는

13MHz이상의 속도에서 제대로 돌아가지 않는다. (또한, 몇몇 PCI 시스템 에서는 ISA 버스 속도를 8MHz로 고정시켜 놓아서, 사용자들이 속도를 올리지 못하게 하고 있다.)

또한 더욱 빠른 전송 속도를 얻으려면, CPU 점유율을 낮추어서 메모리와 I/O 사이클에 좀더 짧은 지속시간을 갖게 하면된다. (ISA 버스상의 하드 디스크와 비디오 카드도 또한 향상된 ISA 버스 속도 덕분에 성능이 올라가는 것을 경험할수 있다.)

8MHz가 넘는 ISA 버스 속도를 시험하기에 앞서서 여러분의 데이타를 백업해 두었는지 확인하고, 속도를 올린 후에는 모든 ISA 주변기기들이 제대로 작동하는지 시험해보기 바란다.

4.3 TCP 수신 윈도우의 설정

다시한번 말하자면, 적은 양의 RAM을 가진 카드와 느린 데이타 전송로를 가진 카드와 컴퓨터 메모리 사이에는 문제가 발생한다. TCP 수신 윈도우 설정의 기본값은 32kB인데, 이것은 여러분의 컴퓨터상의 데이타를 잘 받아졌는지 아닌지 보기위해 멈추는것 없이 32k의 데이타를 덤프할수 있는것같이 동일 서브넷의 빠른 컴퓨터에서도 할수 있다는 뜻이다.

최신 버전의 `route` 명령으로 이 윈도우의 크기를 실시간으로 설정할수 있다. 보통 라우터나 게이트웨이들 뒤에 있는 컴퓨터들은 문제가 없도록 '버퍼링을 하기' 때문에, 로컬 네트워크에서 이 윈도우의 크기는 작아져야 한다. 사용예는 다음과 같다.

```
        route add <whatever> ... window <win_size>
```

여기서 `win_size` 는 여러분이 사용하길 원하는 윈도우의 크기이다.(바이트로) 8MHz나 그 이하의 속도로 작동하는 ISA 버스상의 8비트 3c503 카드는 4kB의 윈도우 크기에도 잘 돌아간다. 너무 큰 윈도우는 패킷 손실이나 넘침을 야기시키고, 이더넷 전송량의 급격한 감소를 가져온다. 여러분의 `cat /proc/net/dev`라고 쳐서 운영체제의 상태를 확인해볼수 있다. 이 명령은 손실되거나 넘친 상태가 발생했는지 보여준다.

4.4 NFS 성능 올리기

몇몇 사람들은 NFS 클라이언트에서 8비트 카드를 사용하면서 8kB (Sun 고유의) NFS 패킷 크기를 이용할때, 기대했던 성능보다 더 나쁜것을 발견했다.

이 경우에 가능한 원인은 8비트와 16비트 카드에 장착된 버퍼 크기의 차이이다. 이더넷 패킷의 최대 크기는 1500 바이트 이다. 8kB의 NFS 패킷은 대략 최대크기의 이더넷 패킷 6개와 같다. 8비트와 16비트 카드가 패킷을 수신할 때는 아무런 문제가 없다. 문제는 머신이 카드의 버퍼에 서 패킷을 제시간에 제거하지 못해서, 버퍼가 넘칠때 일어난다. 사실 전송량당 정량 이상의 ISA 버스 사이클을 가진 8비트 카드들은 어느것에도 도움이 되지 않는다. 여러분이 할수 있는것은 8비트 카드를 가진 경우 NFS 전송 패킷 크기를 2kB(또는 1kB)로 설정하거나, 카드가 버퍼를 더욱 빨리 비우게 하기 위해서 ISA 버스 속도를 올리는 것이다. 나는 8MHz에서 구형 WD8003E 카드가 (다른 시스템 부하 없이) 2kB NFS 크기를 유지할수 있었지만, 4kB는 세가지 사실때문에 성능이 낮아짐으로 유지할수 없었다.

다른 말로 하면, 기본 크기 옵션을 1kB 크기로 사용하고, 여러분이 적어도 16비트 ISA 카드를 가지고 있다면, 여러분은 4kB (또는 8kB까지) 증가하는 것을 발견할수 있을 것이다.

제 5 절 특정 판매자/제조업체/모델별 정보

다음의 목록에서는 판매자 이름과 제품명을 알파벳 순서로 수많은 카드들을 나열하고 있다. 각 제품 ID 옆에는 '지원', '부분지원', '지원안됨'을 볼수 있을 것이다.

지원된다는 의미는 그 카드의 드라이버가 존재하고, 많은 사람들이 문제없이 사용하고 있으며 매우 안정적이라는 뜻이다.

부분지원이라는 뜻은 드라이버가 존재하지만 다음의 설명가운데 적어도 한가지 이상 해당된다는 뜻이다. (1) 드라이버 그리고/또는 하드웨어에 버그가 있어서 낮은 성능이나, 접속 실패 또는 크래쉬도 발생할수 있다. (2) 드라이버가 새로운 것이거나 카드가 일반적이지 않아서 드라이버에 대한 사용/테스트가 매우 적어 드라이버 제작자가 피드백을 별로 받지 못했다. 명확하게 (2)인 경우가 (1)인 경우보다 낫고, 그 카드/드라이버에 대한 개별적인 설명에서 어느 것에 해당하는지 확인해야 한다. 두가지 경우 모두, 여러분은 `make config`시에 "Prompt for development and/or incomplete code/drivers?"부분에서 'Y'라고 답해야 한다.

지원안된다는 것은 현재 그 카드에 사용할수 있는 드라이버가 존재하지 않는다는 뜻이다. 이것은 구하기 힘들거나 평범하지 않은 하드웨어에 대한 관심이 부족하거나, 아니면 판매자들이 드라이버를 만드는데 필요한 하드웨어 문서를 배포하지 않기 때문이다.

'지원'과 '부분지원'의 차이는 다소 주관적이며, 뉴스 그룹 게시판과 메일링 리스트 메세지들을 통한 사용자 피드백을 조사한것에 기반한 것임을 알아두라. (한사람이 각 커널 버전에서 모든 카드들을 가지고 모든 드라이버를 테스트한다는것은 절대로 불가능하다!!!) 그러므로 카드 리스트에 부분지원이라고 되어있는 것이 여러분의 경우에 완벽하게 동작한다거나(정말 좋군), 또는 지원이라고 적힌 카드가 문제가 발생하는 것을 (그리 좋지 못하군) 발견할수도 있다는 것에 주의하길 바란다.

status 다음에는, 리눅스 커널내에 주어진 드라이버 이름이 나열되어 있다. 이것은 또한 `/etc/conf.modules` 모듈 설정 화일내의 `alias eth0 driver_name` 행으로 사용되는 드라이버 모듈의 이름이기도 하다.

5.1 3Com

만일 여러분이 가지고 있는 카드가 무엇인지 잘 모르지만, 아마 3Com 카드일꺼라고 생각하고 있다면, 어셈블리 넘버로부터 무엇인지 알아낼수 있을 것이다. 3Com에는 가장 명확하게 알수 있는 '어셈블리 넘버로 3Com 아답터 알아내기(Identifying 3Com Adapters By Assembly Number)' (ref 24500002)라는 문서가 있다. 3Com 문서를 어떻게 얻는지는 다음을 보라. 8.4 (Technical Information from 3Com)

또한 3Com은 다양한 제품들에 대한 FTP 사이트를 가지고 있다. `ftp.3Com.com`에서 여러분이 원하는 것을 찾을수 있을 것이다.

이문서를 WWW 브라우저로 보고 있는 사람들은, 3Com의 WWW 사이트도 볼수 있다.

5.1.1 3c501

Status: 부분지원, 드라이버 이름: 3c501

이 구석기 시대의 8비트 카드는 사용하려면 너무 머리가 아플정도다. 전염병을 피하는것처럼 피하라. 농담으로라도 이 카드를 구입하지 말라. 이것의 성능은 끔찍하고, 많은 경우에 문제가 발생한다.

For those not yet convinced, the 3c501 can only do one thing at a time – while you are removing one packet from the single-packet buffer it cannot receive another packet, nor can it receive a packet while loading a transmit packet. This was fine for a network between two 8088-based computers where processing each packet and replying took 10's of msecs, but modern networks send back-to-back packets for almost every transaction.

AutoIRQ works, DMA isn't used, the autoprobe only looks at `0x280` and `0x300`, and the debug level is set with the third boot-time argument.

Once again, the use of a 3c501 is *strongly discouraged*! Even more so with a IP multicast kernel, as you will grind to a halt while listening to *all* multicast packets. See the comments at the top of the source code for more details.

5.1.2 EtherLink II, 3c503, 3c503/16

Status: Supported, Driver Name: 3c503 (+8390)

The 3c503 does not have "EEPROM setup", so a diagnostic/setup program isn't needed before running the card with Linux. The shared memory address of the 3c503 is set using jumpers that are shared with the boot PROM address. This is confusing to people familiar with other ISA cards, where you always leave the jumper set to "disable" unless you have a boot PROM.

These cards should be about the same speed as the same bus width WD80x3, but turn out to be actually a bit slower. These shared-memory ethercards also have a programmed I/O mode that doesn't use the 8390 facilities (their engineers found too many bugs!) The Linux 3c503 driver can also work with the 3c503 in programmed-I/O mode, but this is slower and less reliable than shared memory mode. Also, programmed-I/O mode is not as well tested when updating the drivers. You shouldn't use the programmed-I/O mode unless you need it for MS-DOS compatibility.

The 3c503's IRQ line is set in software, with no hints from an EEPROM. Unlike the MS-DOS drivers, the Linux driver has capability to autoIRQ: it uses the first available IRQ line in {5,2/9,3,4}, selected each time the card is ifconfig'ed. (Older driver versions selected the IRQ at boot time.) The ioctl() call in 'ifconfig' will return EAGAIN if no IRQ line is available at that time.

Some common problems that people have with the 503 are discussed in 3.6 (Problems with...).

If you intend on using this driver as a loadable module you should probably see 10.2 (Using the Ethernet Drivers as Modules) for module specific information.

Note that some old diskless 386 workstations have an on board 3c503 (made by 3Com and sold under different names, like 'Bull') but the vendor ID is not a 3Com ID and so it won't be detected. More details can be found in the Etherboot package, which you will need anyways to boot these diskless boxes.

5.1.3 Etherlink Plus 3c505

Status: Semi-Supported, Driver Name: 3c505

This is a driver that was written by Craig Southeren `geoffw@extro.ucc.su.oz.au`. These cards also use the i82586 chip. There are not that many of these cards about. It is included in the standard kernel, but it is classed as an alpha driver. See 3.1 (Alpha Drivers) for important information on using alpha-test ethernet drivers with Linux.

There is also the file `/usr/src/linux/drivers/net/README.3c505` that you should read if you are going to use one of these cards. It contains various options that you can enable/disable.

5.1.4 Etherlink-16 3c507

Status: Semi-Supported, Driver Name: 3c507

This card uses one of the Intel chips, and the development of the driver is closely related to the development of the Intel Ether Express driver. The driver is included in the standard kernel release, but as an alpha driver. See 3.1 (Alpha Drivers) for important information on using alpha-test ethernet drivers with Linux.

5.1.5 Etherlink III, 3c509 / 3c509B

Status: Supported, Driver Name: 3c509

This card is fairly inexpensive and has good performance for an ISA non-bus-master design. The drawbacks are that the original 3c509 requires very low interrupt latency. The 3c509B shouldn't suffer from the same problem, due to having a larger buffer. (See below.) These cards use PIO transfers, similar to a ne2000 card, and so a shared memory card such as a wd8013 will be more efficient in comparison.

The original 3c509 has a small packet buffer (4kB total, 2kB Rx, 2kB Tx), causing the driver to occasionally drop a packet if interrupts are masked for too long. To minimize this problem, you can try unmasking interrupts during IDE disk transfers (see `man hdparm`) and/or increasing your ISA bus speed so IDE transfers finish sooner.

The newer model 3c509B has 8kB on board, and the buffer can be split 4/4, 5/3 or 6/2 for Rx/Tx. This setting is changed with the DOS configuration utility, and is stored on the EEPROM. This should alleviate the above problem with the original 3c509.

3c509B users should use either the supplied DOS utility to disable the *plug and play* support, *and* to set the output media to what they require. The linux driver currently does *not* support the Autodetect media setting, so you *have* to select 10Base-T or 10Base-2 or AUI. Note that to turn off PnP entirely, you should do a `3C5X9CFG /PNP:DISABLE` and then follow that with a hard reset to ensure that it takes effect.

Some people ask about the “Server or Workstation” and “Highest Modem Speed” settings presented in the DOS configuration utility. Donald writes “These are only hints to the drivers, and the Linux driver does not use these parameters: it always optimizes for high throughput rather than low latency (‘Server’). Low latency was critically important for old, non-windowed, IPX throughput. To reduce the latency the MS-DOS driver for the 3c509 disables interrupts for some operations, blocking serial port interrupts. Thus the need for the ‘modem speed’ setting. The Linux driver avoids the need to disable interrupts for long periods by operating only on whole packets e.g. by not starting to transmit a packet until it is completely transferred to the card.”

Note that the ISA card detection uses a different method than most cards. Basically, you ask the cards to respond by sending data to an ID_PORT (port `0x100` to `0x1ff` on intervals of `0x10`). This detection method means that a particular card will *always* get detected first in a multiple ISA 3c509 configuration. The card with the lowest hardware ethernet address will *always* end up being `eth0`. This shouldn’t matter to anyone, except for those people who want to assign a 6 byte hardware address to a particular interface. If you have multiple 3c509 cards, it is best to append `ether=0,0,ethN` commands without the I/O port specified (i.e. use I/O=zero) and allow the probe to sort out which card is first. Using a non-zero I/O value will ensure that it does not detect all your cards, so don’t do it.

If this really bothers you, have a look at Donald’s latest driver, as you may be able to use a `0x3c509` value in the unused mem address fields to order the detection to suit your needs.

5.1.6 3c515

Status: Supported, Driver Name: 3c515

This is 3Com’s ISA 100Mbps offering, codenamed “CorkScrew”. A relatively new driver from Donald for these cards is included in the v2.2 kernels. For the most up to date information, you should probably look on the Vortex page:

Vortex `<http://cesdis.gsfc.nasa.gov/linux/drivers/vortex.html>`

5.1.7 3c523

Status: Semi-Supported, Driver Name: 3c523

This MCA bus card uses the i82586, and Chris Beauregard has modified the ni52 driver to work with these cards. The driver for it can be found in the v2.2 kernel source tree.

More details can be found on the MCA-Linux page at `http://glycerine.cetmm.uni.edu/mca/`

5.1.8 3c527

Status: Not Supported.

Yes, another MCA card. No, not too much interest in it. Better chances with the 3c529 if you are stuck with MCA.

5.1.9 3c529

Status: Supported, Driver Name: 3c509

This card actually uses the same chipset as the 3c509. Donald actually put hooks into the 3c509 driver to check for MCA cards after probing for EISA cards, and before probing for ISA cards, long before MCA support was added to the kernel. The required MCA probe code is included in the driver shipped with v2.2 kernels. More details can be found on the MCA-Linux page at:

`http://glycerine.cetmm.uni.edu/mca/`

5.1.10 3c562

Status: Supported, Driver Name: 3c589 (distributed separately)

This PCMCIA card is the combination of a 3c589B ethernet card with a modem. The modem appears as a standard modem to the end user. The only difficulty is getting the two separate linux drivers to share one interrupt. There are a couple of new registers and some hardware interrupt sharing support. You need to use a v2.0 or newer kernel that has the support for interrupt sharing.

Thanks again to Cameron for getting a sample unit and documentation sent off to David Hinds. Look for support in David's PCMCIA package release.

See 9.2 (PCMCIA Support) for more info on PCMCIA chipsets, socket enablers, etc.

5.1.11 3c575

Status: Unknown.

A driver for this PCMCIA card is under development and hopefully will be included in David's PCMCIA package in the future. Best to check the PCMCIA package to get the current status.

5.1.12 3c579

Status: Supported, Driver Name: 3c509

The EISA version of the 509. The current EISA version uses the same 16 bit wide chip rather than a 32 bit interface, so the performance increase isn't stunning. Make sure the card is configured for EISA addressing mode. Read the above 3c509 section for info on the driver.

5.1.13 3c589 / 3c589B

Status: Semi-Supported, Driver Name: 3c589

Many people have been using this PCMCIA card for quite some time now. Note that support for it is not (at present) included in the default kernel source tree. The "B" in the name means the same here as it does for the 3c509 case.

There are drivers available on Donald's ftp site and in David Hinds PCMCIA package. You will also need a supported PCMCIA controller chipset. See 9.2 (PCMCIA Support) for more info on PCMCIA drivers, chipsets, socket enablers, etc.

5.1.14 3c590 / 3c595

Status: Supported, Driver Name: 3c59x

These "Vortex" cards are for PCI bus machines, with the '590 being 10Mbps and the '595 being 3Com's 100Mbs offering. Also note that you can run the '595 as a '590 (i.e. in a 10Mbps mode). The driver is included in the v2.0 kernel source, but is also continually being updated. If you have problems with the driver in the v2.0 kernel, you can get an updated driver from the following URL:

Vortex `<http://cesdis.gsfc.nasa.gov/linux/drivers/vortex.html>`

Note that there are two different 3c590 cards out there, early models that had 32kB of on-board memory, and later models that only have 8kB of memory. Chances are you won't be able to buy a new 3c59x for much longer, as it is being replaced with the 3c90x card. If you are buying a used one off somebody, try and get the 32kB version. The 3c595 cards have 64kB, as you can't get away with only 8kB RAM at 100Mbps!

A thanks to Cameron Spitzer and Terry Murphy of 3Com for sending cards and documentation to Donald so he could write the driver.

Donald has set up a mailing list for Vortex driver support. To join the list, just do:

`echo subscribe | /bin/mail linux-vortex-request@cesdis.gsfc.nasa.gov`

5.1.15 3c592 / 3c597

Status: Supported, Driver Name: 3c59x

These are the EISA versions of the 3c59x series of cards. The 3c592/3c597 (aka Demon) should work with the vortex driver discussed above.

5.1.16 3c900 / 3c905 / 3c905B

Status: Supported, Driver Name: 3c59x

These cards (aka 'Boomerang', aka EtherLink III XL) have been released to take over the place of the 3c590/3c595 cards.

The support for the Cyclone 'B' revision was only recently added. To use this card with older v2.0 kernels, you must obtain the updated `3c59x.c` driver from Donald's site at:

Vortex-Page `<http://cesdis.gsfc.nasa.gov/linux/drivers/vortex.html>`

If in doubt about anything then check out the above WWW page. Donald has set up a mailing list for Vortex driver support announcements and etc. To join the list, just do:

`echo subscribe | /bin/mail linux-vortex-request@cesdis.gsfc.nasa.gov`

5.1.17 3c985

Status: Supported, Driver Name: acenic

This driver, by Jes Sorensen, is available in v2.2 kernels It supports several other Gigabit cards in addition to the 3Com model.

5.2 Accton

5.2.1 Accton MPX

Status: Supported, Driver Name: ne (+8390)

Don't let the name fool you. This is still supposed to be a NE2000 compatible card, and should work with the ne2000 driver.

5.2.2 Accton EN1203, EN1207, EtherDuo-PCI

Status: Supported, Driver Name: de4x5, tulip

This is another implementation of the DEC 21040 PCI chip. The EN1207 card has the 21140, and also has a 10Base-2 connector, which has proved troublesome for some people in terms of selecting that media. Using the card with 10Base-T and 100Base-T media have worked for others though. So as with all purchases, you should try and make sure you can return it if it doesn't work for you.

See 5.16.4 (DEC 21040) for more information on these cards, and the present driver situation.

5.2.3 Accton EN2209 Parallel Port Adaptor (EtherPocket)

Status: Semi-Supported, Driver Name: ?

A driver for these parallel port adapters is available but not yet part of the 2.0 or 2.1 kernel source. You have to get the driver from:

`http://www.unix-ag.uni-siegen.de/~nils/accton_linux.html`

5.2.4 Accton EN2212 PCMCIA Card

Status: Semi-Supported, Driver Name: ?

David Hinds has been working on a driver for this card, and you are best to check the latest release of his PCMCIA package to see what the present status is.

5.3 Allied Telesyn/Telesis

5.3.1 AT1500

Status: Supported, Driver Name: lance

These are a series of low-cost ethercards using the 79C960 version of the AMD LANCE. These are bus-master cards, and hence one of the faster ISA bus ethercards available.

DMA selection and chip numbering information can be found in 5.4.1 (AMD LANCE).

More technical information on AMD LANCE based Ethernet cards can be found in 8.5 (Notes on AMD...).

5.3.2 AT1700

Status: Supported, Driver Name: at1700

Note that to access this driver during `make config` you still have to answer 'Y' when asked "Prompt for development and/or incomplete code/drivers?" at the first. This is simply due to lack of feedback on the driver stability due to it being a relatively rare card. If you have problems with the driver that ships with the kernel then you may be interested in the alternative driver available at: `http://www.cc.hit-u.ac.jp/nagoya/at1700/`

The Allied Telesis AT1700 series ethercards are based on the Fujitsu MB86965. This chip uses a programmed I/O interface, and a pair of fixed-size transmit buffers. This allows small groups of packets to be sent back-to-back, with a short pause while switching buffers.

A unique feature is the ability to drive 150ohm STP (Shielded Twisted Pair) cable commonly installed for Token Ring, in addition to 10baseT 100ohm UTP (unshielded twisted pair). A fibre optic version of the card (AT1700FT) exists as well.

The Fujitsu chip used on the AT1700 has a design flaw: it can only be fully reset by doing a power cycle of the machine. Pressing the reset button doesn't reset the bus interface. This wouldn't be so bad, except that it can only be reliably detected when it has been freshly reset. The solution/work-around is to power-cycle the machine if the kernel has a problem detecting the AT1700.

5.3.3 AT2450

Status: Supported, Driver Name: pcnet32

This is the PCI version of the AT1500, and it doesn't suffer from the problems that the Boca 79c970 PCI card does. DMA selection and chip numbering information can be found in 5.4.1 (AMD LANCE).

More technical information on AMD LANCE based Ethernet cards can be found in 8.5 (Notes on AMD...).

5.3.4 AT2500

Status: Semi-Supported, Driver Name: rtl8139

This card uses the RealTek 8139 chip - see the section 5.31.5 (RealTek 8139).

5.3.5 AT2540FX

Status: Semi-Supported, Driver Name: eepro100

This card uses the i82557 chip, and hence may/should work with the eepro100 driver. If you try this please send in a report so this information can be updated.

5.4 AMD / Advanced Micro Devices

Carl Ching of AMD was kind enough to provide a very detailed description of all the relevant AMD ethernet products which helped clear up this section.

5.4.1 AMD LANCE (7990, 79C960/961/961A, PCnet-ISA)

Status: Supported, Driver Name: lance

There really is no AMD ethernet card. You are probably reading this because the only markings you could find on your card said AMD and the above number. The 7990 is the original 'LANCE' chip, but most stuff (including this document) refer to all these similar chips as 'LANCE' chips. (...incorrectly, I might add.)

These above numbers refer to chips from AMD that are the heart of many ethernet cards. For example, the Allied Telesis AT1500 (see 5.3.1 (AT1500)) and the NE1500/2100 (see 5.27.4 (NE1500)) use these chips.

The 7990/79c90 have long been replaced by newer versions. The 79C960 (a.k.a. PCnet-ISA) essentially contains the 79c90 core, along with all the other hardware support required, which allows a single-chip ethernet solution. The 79c961 (PCnet-ISA+) is a jumperless Plug and Play version of the '960. The final chip in the ISA series is the 79c961A (PCnet-ISA II), which adds full duplex capabilities. All cards with one of these chips should work with the lance.c driver, with the exception of very old cards that used the original 7990 in a shared memory configuration. These old cards can be spotted by the lack of jumpers for a DMA channel.

One common problem people have is the 'busmaster arbitration failure' message. This is printed out when the LANCE driver can't get access to the bus after a reasonable amount of time has elapsed (50us). This usually indicates that the motherboard implementation of bus-mastering DMA is broken, or some other device is hogging the bus, or there is a DMA channel conflict. If your BIOS setup has the 'GAT option' (for Guaranteed Access Time) then try toggling/altering that setting to see if it helps.

Also note that the driver only looks at the addresses: `0x300, 0x320, 0x340, 0x360` for a valid card, and any address supplied by an `ether=` boot argument is silently ignored (this will be fixed) so make sure your card is configured for one of the above I/O addresses for now.

The driver will still work fine, even if more than 16MB of memory is installed, since low-memory 'bounce-buffers' are used when needed (i.e. any data from above 16MB is copied into a buffer below 16MB before being given to the card to transmit.)

The DMA channel can be set with the low bits of the otherwise-unused dev->mem_start value (a.k.a. PARAM_1). (see 10.1.1 (PARAM_1)) If unset it is probed for by enabling each free DMA channel in turn and checking if initialization succeeds.

The HP-J2405A board is an exception: with this board it's easy to read the EEPROM-set values for the IRQ, and DMA.

See 8.5 (Notes on AMD...) for more info on these chips.

5.4.2 AMD 79C965 (PCnet-32)

Status: Supported, Driver Name: pcnet32

This is the PCnet-32 – a 32 bit bus-master version of the original LANCE chip for VL-bus and local bus systems. chip. While these chips can be operated with the standard `lance.c` driver, a 32 bit version (`pcnet32.c`) is also available that does not have to concern itself with any 16MB limitations associated with the ISA bus.

5.4.3 AMD 79C970/970A (PCnet-PCI)

Status: Supported, Driver Name: pcnet32

This is the PCnet-PCI – similar to the PCnet-32, but designed for PCI bus based systems. Please see the above PCnet-32 information. This means that you need to build a kernel with PCI BIOS support enabled. The '970A adds full duplex support along with some other features to the original '970 design.

Note that the Boca implementation of the 79C970 fails on fast Pentium machines. This is a hardware problem, as it affects DOS users as well. See the Boca section for more details.

5.4.4 AMD 79C971 (PCnet-FAST)

Status: Supported, Driver Name: pcnet32

This is AMD's 100Mbit chip for PCI systems, which also supports full duplex operation. It was introduced in June 1996.

5.4.5 AMD 79C972 (PCnet-FAST+)

Status: Unknown, Driver Name: pcnet32

This should also work just like the '971 but this has yet to be confirmed.

5.4.6 AMD 79C974 (PCnet-SCSI)

Status: Supported, Driver Name: pcnet32

This is the PCnet-SCSI – which is basically treated like a '970 from an Ethernet point of view. Also see the above information. Don't ask if the SCSI half of the chip is supported – this is the *Ethernet-HowTo*, not the SCSI-HowTo.

5.5 Ansel Communications

5.5.1 AC3200 EISA

Status: Semi-Supported, Driver Name: ac3200

Note that to access this driver during `make config` you still have to answer 'Y' when asked "Prompt for development and/or incomplete code/drivers?" at the first. This is simply due to lack of feedback on the driver stability due to it being a relatively rare card.

This driver is included in the present kernel as an alpha test driver. It is based on the common NS8390 chip used in the ne2000 and wd80x3 cards. Please see 3.1 (Alpha Drivers) in this document for important information regarding alpha drivers.

If you use it, let one of us know how things work out, as feedback has been low, even though the driver has been in the kernel since v1.1.25.

If you intend on using this driver as a loadable module you should probably see 10.2 (Using the Ethernet Drivers as Modules) for module specific information.

5.6 Apricot

5.6.1 Apricot Xen-II On Board Ethernet

Status: Semi-Supported, Driver Name: apricot

This on board ethernet uses an i82596 bus-master chip. It can only be at I/O address `0x300`. By looking at the driver source, it appears that the IRQ is also hardwired to 10.

Earlier versions of the driver had a tendency to think that anything living at `0x300` was an apricot NIC. Since then the hardware address is checked to avoid these false detections.

5.7 Arcnet

Status: Supported, Driver Name: arcnet (arc-rimi, com90xx, com20020)

With the very low cost and better performance of ethernet, chances are that most places will be giving away their Arcnet hardware for free, resulting in a lot of home systems with Arcnet.

An advantage of Arcnet is that all of the cards have identical interfaces, so one driver will work for everyone. It also has built in error handling so that it supposedly never loses a packet. (Great for UDP traffic!)

Avery Pennarun's arcnet driver has been in the default kernel sources since 1.1.80. The arcnet driver uses 'arc0' as its name instead of the usual 'eth0' for ethernet devices. Bug reports and success stories can be mailed to:

`apenwarr@foxnet.net`

There are information files contained in the standard kernel for setting jumpers and general hints.

Supposedly the driver also works with the 100Mbs ARCnet cards as well!

5.8 AT&T

Note that AT&T's StarLAN is an orphaned technology, like SynOptics LattisNet, and can't be used in a standard 10Base-T environment, without a hub that 'speaks' both.

5.8.1 AT&T T7231 (LanPACER+)

Status: Not Supported.

These StarLAN cards use an interface similar to the i82586 chip. At one point, Matthijs Melchior (`matthijs.n.melchior@att.com`) was playing with the 3c507 driver, and almost had something useable working. Haven't heard much since that.

5.9 Boca Research

Yes, they make more than just multi-port serial cards. :-)

5.9.1 Boca BEN (ISA, VLB, PCI)

Status: Supported, Driver Name: lance, pcnet32

These cards are based on AMD's PCnet chips. Perspective buyers should be warned that many users have had endless problems with these VLB/PCI cards. Owners of fast Pentium systems have been especially hit. Note that this is not a driver problem, as it hits DOS/Win/NT users as well. Boca's technical support number is (407) 241-8088, and you can also reach them at `75300.2672@compuserve.com`. The older ISA cards don't appear to suffer the same problems.

Donald did a comparitive test with a Boca PCI card and a similar Allied Telsyn PCnet/PCI implementation, which showed that the problem lies in Boca's implementation of the PCnet/PCI chip. These test results can be accessed on Don's www server.

Linux at CESDIS `<http://cesdis.gsfc.nasa.gov/linux/>`

Boca is offering a 'warranty repair' for affected owners, which involves adding one of the missing capacitors, but it appears that this fix doesn't work 100 percent for most people, although it helps some.

If you are *still* thinking of buying one of these cards, then at least try and get a 7 day unconditional return policy, so that if it doesn't work properly in your system, you can return it.

More general information on the AMD chips can be found in 5.4.1 (AMD LANCE).

More technical information on AMD LANCE based Ethernet cards can be found in 8.5 (Notes on AMD...).

5.10 Cabletron

Donald writes: 'Yes, another one of these companies that won't release its programming information. They waited for months before actually confirming that all their information was proprietary, deliberately wasting my time. Avoid their cards like the plague if you can. Also note that some people have phoned Cabletron, and have been told things like 'a D. Becker is working on a driver for linux' – making it sound like I work for them. This is NOT the case.'

Apparently Cabletron has changed their policy with respect to programming information (like Xircom) since Donald made the above comment several years ago – send e-mail to `support@ctron.com` if you want to verify this or ask for programming information. However, at this point in time, there is little demand for modified/updated drivers for the older E20xx and E21xx cards.

5.10.1 E10**, E10**-x, E20**, E20**-x

Status: Semi-Supported, Driver Name: ne (+8390)

These are NEx000 almost-clones that are reported to work with the standard NEx000 drivers, thanks to a ctron-specific check during the probe. If there are any problems, they are unlikely to be fixed, as the programming information is unavailable.

5.10.2 E2100

Status: Semi-Supported, Driver Name: e2100 (+8390)

Again, there is not much one can do when the programming information is proprietary. The E2100 is a poor design. Whenever it maps its shared memory in during a packet transfer, it maps it into the *whole 128K region!* That means you **can't** safely use another interrupt-driven shared memory device in that region, including another E2100. It will work most of the time, but every once in a while it will bite you. (Yes, this problem can be avoided by turning off interrupts while transferring packets, but that will almost certainly lose clock ticks.) Also, if you mis-program the board, or halt the machine at just the wrong moment, even the reset button won't bring it back. You will *have* to turn it off and *leave* it off for about 30 seconds.

Media selection is automatic, but you can override this with the low bits of the dev->mem_end parameter. See 10.1.1 (PARAM_2). Module users can specify an `xcvr=N` value as an `option` in the `/etc/conf.modules` file.

Also, don't confuse the E2100 for a NE2100 clone. The E2100 is a shared memory NatSemi DP8390 design, roughly similar to a brain-damaged WD8013, whereas the NE2100 (and NE1500) use a bus-mastering AMD LANCE design.

There is an E2100 driver included in the standard kernel. However, seeing as programming info isn't available, don't expect bug-fixes. Don't use one unless you are already stuck with the card.

If you intend on using this driver as a loadable module you should probably see 10.2 (Using the Ethernet Drivers as Modules) for module specific information.

5.10.3 E22**

Status: Semi-Supported, Driver Name: lance

According to information in a Cabletron Tech Bulletin, these cards use the standard AMD PC-Net chipset (see 5.4.1 (AMD PC-Net)) and should work with the generic lance driver.

5.11 Cogent

Here is where and how to reach them:

```
Cogent Data Technologies, Inc.
175 West Street, P.O. Box 926
Friday Harbour, WA 98250, USA.

Cogent Sales
15375 S.E. 30th Place, Suite 310
Bellevue, WA 98007, USA.

Technical Support:
Phone (360) 378-2929 between 8am and 5pm PST
Fax (360) 378-2882
Compuserve GO COGENT
Bulletin Board Service (360) 378-5405
Internet: support@cogentdata.com
```

5.11.1 EM100-ISA/EISA

Status: Semi-Supported, Driver Name: smc9194

These cards use the SMC 91c100 chip and may work with the SMC 91c92 driver, but this has yet to be verified.

5.11.2 Cogent eMASTER+, EM100-PCI, EM400, EM960, EM964

Status: Supported, Driver Name: de4x5, tulip

These are yet another DEC 21040 implementation that should hopefully work fine with the standard 21040 driver.

The EM400 and the EM964 are four port cards using a DEC 21050 bridge and 4 21040 chips.

See 5.16.4 (DEC 21040) for more information on these cards, and the present driver situation.

5.12 Compaq

Compaq aren't really in the business of making ethernet cards, but a lot of their systems have embedded ethernet controllers on the motherboard.

5.12.1 Compaq Deskpro / Compaq XL (Embedded AMD Chip)

Status: Supported, Driver Name: pcnet32

Machines such as the XL series have an AMD 79c97x PCI chip on the mainboard that can be used with the standard LANCE driver. But before you can use it, you have to do some trickery to get the PCI BIOS to a place where Linux can see it. Frank Maas was kind enough to provide the details:

“ The problem with this Compaq machine however is that the PCI directory is loaded in high memory, at a spot where the Linux kernel can't (won't) reach. Result: the card is never detected nor is it usable (sideline: the mouse won't work either) The workaround (as described thoroughly in http://www-c724.uibk.ac.at/XL/)

is to load MS-DOS, launch a little driver Compaq wrote and then load the Linux kernel using LOADLIN. Ok, I'll give you time to say 'yuck, yuck', but for now this is the only working solution I know of. The little driver simply moves the PCI directory to a place where it is normally stored (and where Linux can find it)."

More general information on the AMD chips can be found in 5.4.1 (AMD LANCE).

5.12.2 Compaq Nettelligent/NetFlex (Embedded ThunderLAN Chip)

Status: Supported, Driver Name: tlan

These systems use a Texas Instruments ThunderLAN chip Information on the ThunderLAN driver can be found in 5.36.1 (ThunderLAN).

5.13 Danpex

5.13.1 Danpex EN9400

Status: Supported, Driver Name: de4x5, tulip

Yet another card based on the DEC 21040 chip, reported to work fine, and at a relatively cheap price.

See 5.16.4 (DEC 21040) for more information on these cards, and the present driver situation.

5.14 D-Link

5.14.1 DE-100, DE-200, DE-220-T, DE-250

Status: Supported, Driver Name: ne (+8390)

Some of the early D-Link cards didn't have the `0x57` PROM signature, but the ne2000 driver knows about them. For the software configurable cards, you can get the config program from `www.dlink.com`. The DE2** cards were the most widely reported as having the spurious transfer address mismatch errors with early versions of linux. Note that there are also cards from Digital (DEC) that are also named DE100 and DE200, but the similarity stops there.

5.14.2 DE-520

Status: Supported, Driver Name: pcnet32

This is a PCI card using the PCI version of AMD's LANCE chip. DMA selection and chip numbering information can be found in 5.4.1 (AMD LANCE).

More technical information on AMD LANCE based Ethernet cards can be found in 8.5 (Notes on AMD...).

5.14.3 DE-528

Status: Supported, Driver Name: ne, ne2k-pci (+8390)

Apparently D-Link have also started making PCI NE2000 clones.

5.14.4 DE-530

Status: Supported, Driver Name: de4x5, tulip

This is a generic DEC 21040 PCI chip implementation, and is reported to work with the generic 21040 tulip driver.

See 5.16.4 (DEC 21040) for more information on these cards, and the present driver situation.

5.14.5 DE-600

Status: Supported, Driver Name: de600

Laptop users and other folk who might want a quick way to put their computer onto the ethernet may want to use this. The driver is included with the default kernel source tree. Bjorn Ekwall `bj0rn@blox.se` wrote the driver. Expect about 180kb/s transfer speed from this via the parallel port. You should read the README.DLINK file in the kernel source tree.

Note that the device name that you pass to `ifconfig` is *now* `eth0` and not the previously used `dl0`.

If your parallel port is *not* at the standard `0x378` then you will have to recompile. Bjorn writes: "Since the DE-620 driver tries to sqeeze the last microsecond from the loops, I made the irq and port address constants instead of variables. This makes for a usable speed, but it also means that you can't change these assignements from e.g. lilo; you _have_ to recompile..." Also note that some laptops implement the on-board parallel port at `0x3bc` which is where the parallel ports on monochrome cards were/are.

5.14.6 DE-620

Status: Supported, Driver Name: de620

Same as the DE-600, only with two output formats. Bjorn has written a driver for this model, for kernel versions 1.1 and above. See the above information on the DE-600.

5.14.7 DE-650

Status: Semi-Supported, Driver Name: de650 (?)

Some people have been using this PCMCIA card for some time now with their notebooks. It is a basic 8390 design, much like a NE2000. The LinkSys PCMCIA card and the IC-Card Ethernet are supposedly DE-650 clones as well. Note that at present, this driver is *not* part of the standard kernel, and so you will have to do some patching.

See 9.2 (PCMCIA Support) in this document, and if you can, have a look at:

Don's PCMCIA Stuff `<http://cesdis.gsfc.nasa.gov/linux/pcmcia.html>`

5.15 DFI

5.15.1 DFINET-300 and DFINET-400

Status: Supported, Driver Name: ne (+8390)

These cards are now detected (as of 0.99pl15) thanks to Eberhard Moenkeberg `emoenke@gwdg.de` who noted that they use 'DFI' in the first 3 bytes of the prom, instead of using `0x57` in bytes 14 and 15, which is what all the NE1000 and NE2000 cards use. (The 300 is an 8 bit pseudo NE1000 clone, and the 400 is a pseudo NE2000 clone.)

5.16 Digital / DEC

5.16.1 DEPCA, DE100/1, DE200/1/2, DE210, DE422

Status: Supported, Driver Name: depca

There is documentation included in the source file 'depca.c', which includes info on how to use more than one of these cards in a machine. Note that the DE422 is an EISA card. These cards are all based on the AMD LANCE chip. See 5.4.1 (AMD LANCE) for more info. A maximum of two of the ISA cards can be used, because they can only be set for `0x300` and `0x200` base I/O address. If you are intending to do this, please read the notes in the driver source file `depca.c` in the standard kernel source tree.

This driver will also work on Alpha CPU based machines, and there are various ioctl()s that the user can play with.

5.16.2 Digital EtherWorks 3 (DE203, DE204, DE205)

Status: Supported, Driver Name: ewrk3

These cards use a proprietary chip from DEC, as opposed to the LANCE chip used in the earlier cards like the DE200. These cards support both shared memory or programmed I/O, although you take about a 50%performance hit if you use PIO mode. The shared memory size can be set to 2kB, 32kB or 64kB, but only 2 and 32 have been tested with this driver. David says that the performance is virtually identical between the 2kB and 32kB mode. There is more information (including using the driver as a loadable module) at the top of the driver file `ewrk3.c` and also in `README.ewrk3`. Both of these files come with the standard kernel distribution. This driver has Alpha CPU support like depca.c does.

The standard driver has a number of interesting ioctl() calls that can be used to get or clear packet statistics, read/write the EEPROM, change the hardware address, and the like. Hackers can see the source code for more info on that one.

David has also written a configuration utility for this card (along the lines of the DOS program `NICSETUP.EXE`) along with other tools. These can be found on most Linux FTP sites in the directory `/pub/Linux/system/Network/management` – look for the file `ewrk3tools-X.XX.tar.gz`.

5.16.3 DE425 EISA, DE434, DE435, DE500

Status: Supported, Driver Name: de4x5, tulip

These cards are based on the 21040 chip mentioned below. The DE500 uses the 21140 chip to provide 10/100Mbs ethernet connections. Have a read of the 21040 section below for extra info. There are also some compile-time options available for non-DEC cards using this driver. Have a look at `README.de4x5` for details.

All the Digital cards will autoprobe for their media (except, temporarily, the DE500 due to a patent issue).

This driver is also Alpha CPU ready and supports being loaded as a module. Users can access the driver internals through ioctl() calls - see the 'ewrk3' tools and the de4x5.c sources for information about how to do this.

5.16.4 DEC 21040, 21041, 2114x, Tulip

Status: Supported, Driver Name: de4x5, tulip

The DEC 21040 is a bus-mastering single chip ethernet solution from Digital, similar to AMD's PCnet chip. The 21040 is specifically designed for the PCI bus architecture. SMC's new EtherPower PCI card uses this chip.

You have a choice of *two* drivers for cards based on this chip. There is the DE425 driver discussed above, and the generic 21040 'tulip' driver.

Warning: Even though your card may be based upon this chip, *the drivers may not work for you.* David C. Davies writes:

"There are no guarantees that either 'tulip.c' OR 'de4x5.c' will run any DC2114x based card other than those they've been written to support. WHY?? You ask. Because there is a register, the General Purpose Register (CSR12) that (1) in the DC21140A is programmable by each vendor and they all do it differently (2) in the DC21142/3 this is now an SIA control register (a la DC21041). The only small ray of hope is that we can decode the SROM to help set up the driver. However, this is not a guaranteed solution since some vendors (e.g. SMC 9332 card) don't follow the Digital Semiconductor recommended SROM programming format."

In non-technical terms, this means that if you aren't sure that an unknown card with a DC2114x chip will work with the linux driver(s), then make sure you can return the card to the place of purchase *before* you pay for it.

The updated 21041 chip is also found in place of the 21040 on most of the later SMC EtherPower cards. The 21140 is for supporting 100Base-? and works with the Linux drivers for the 21040 chip. To use David's `de4x5` driver with non-DEC cards, have a look at `README.de4x5` for details.

Donald has used SMC EtherPower-10/100 cards to develop the 'tulip' driver. Note that the driver that is in the standard kernel tree at the moment is not the most up to date version. If you are having trouble with this driver, you should get the newest version from Donald's ftp/WWW site.

Tulip Driver `<http://cesdis.gsfc.nasa.gov/linux/drivers/tulip.html>`

The above URL also contains a (non-exhaustive) list of various cards/vendors that use the 21040 chip.

Also note that the tulip driver is still considered an *alpha* driver (see 3.1 (Alpha Drivers)) at the moment, and should be treated as such. To use it, you will have to edit `arch/i386/config.in` and uncomment the line for `CONFIG_DEC_ELCP` support.

Donald has even set up a mailing list for tulip driver support announcements, etc. To join it just type:

`echo subscribe | /bin/mail linux-tulip-request@cesdis.gsfc.nasa.gov`

5.17 Farallon

Farallon sells EtherWave adaptors and transceivers. This device allows multiple 10baseT devices to be daisy-chained.

5.17.1 Farallon Etherwave

Status: Supported, Driver Name: 3c509

This is reported to be a 3c509 clone that includes the EtherWave transceiver. People have used these successfully with Linux and the present 3c509 driver. They are too expensive for general use, but are a great option for special cases. Hublet prices start at $125, and Etherwave adds $75-$100 to the price of the board – worth it if you have pulled one wire too few, but not if you are two network drops short.

5.18 Fujitsu

Unlike many network chip manufacturers, Fujitsu have also made and sold some network cards based upon their chip.

5.18.1 Fujitsu FMV-181/182/183/184

Status: Supported, Driver Name: fmv18x

According to the driver, these cards are a straight forward Fujitsu MB86965 implementation, which would make them very similar to the Allied Telesis AT1700 cards.

5.19 Hewlett Packard

The 272** cards use programmed I/O, similar to the NE*000 boards, but the data transfer port can be 'turned off' when you aren't accessing it, avoiding problems with autoprobing drivers.

Thanks to Glenn Talbott for helping clean up the confusion in this section regarding the version numbers of the HP hardware.

5.19.1 27245A

Status: Supported, Driver Name: hp (+8390)

8 Bit 8390 based 10BaseT, not recommended for all the 8 bit reasons. It was re-designed a couple years ago to be highly integrated which caused some changes in initialization timing which only affected testing programs, not LAN drivers. (The new card is not 'ready' as soon after switching into and out of loopback mode.)

If you intend on using this driver as a loadable module you should probably see 10.2 (Using the Ethernet Drivers as Modules) for module specific information.

5.19.2 HP EtherTwist, PC Lan+ (27247, 27252A)

Status: Supported, Driver Name: hp+ (+8390)

The HP PC Lan+ is different to the standard HP PC Lan card. This driver was added to the list of drivers in the standard kernel during the v1.1.x development cycle. It can be operated in either a PIO mode like a ne2000, or a shared memory mode like a wd8013.

The 47B is a 16 Bit 8390 based 10BaseT w/AUI, and the 52A is a 16 Bit 8390 based ThinLAN w/AUI. These cards have 32K onboard RAM for Tx/Rx packet buffering instead of the usual 16KB, and they both offer LAN connector autosense.

If you intend on using this driver as a loadable module you should probably see 10.2 (Using the Ethernet Drivers as Modules) for module specific information.

5.19.3 HP-J2405A

Status: Supported, Driver Name: lance

These are lower priced, and slightly faster than the 27247/27252A, but are missing some features, such as AUI, ThinLAN connectivity, and boot PROM socket. This is a fairly generic LANCE design, but a minor design decision makes it incompatible with a generic 'NE2100' driver. Special support for it (including reading the DMA channel from the board) is included thanks to information provided by HP's Glenn Talbott.

More technical information on LANCE based cards can be found in 8.5 (Notes on AMD...)

5.19.4 HP-Vectra On Board Ethernet

Status: Supported, Driver Name: lance

The HP-Vectra has an AMD PCnet chip on the motherboard. DMA selection and chip numbering information can be found in 5.4.1 (AMD LANCE).

More technical information on LANCE based cards can be found in 8.5 (Notes on AMD...)

5.19.5 HP 10/100 VG Any Lan Cards (27248B, J2573, J2577, J2585, J970, J973)

Status: Supported, Driver Name: hp100

This driver also supports some of the Compex VG products. Since the driver supports ISA, EISA and PCI cards, it is found under ISA cards when running `make config` on a kernel source.

5.19.6 HP NetServer 10/100TX PCI (D5013A)

Status: Supported, Driver Name: eepro100

Apparently these are just a rebadged Intel EtherExpress Pro 10/100B card. See the Intel section for more information.

5.20 IBM / International Business Machines

5.20.1 IBM Thinkpad 300

Status: Supported, Driver Name: znet

This is compatible with the Intel based Zenith Z-note. See 5.42.1 (Z-note) for more info.

Supposedly this site has a comprehensive database of useful stuff for newer versions of the Thinkpad. I haven't checked it out myself yet.

Thinkpad-info `<http://peipa.essex.ac.uk/html/linux-thinkpad.html>`

For those without a WWW browser handy, try `peipa.essex.ac.uk:/pub/tp750/`

5.20.2 IBM Credit Card Adaptor for Ethernet

Status: Semi-Supported, Driver Name: ? (distributed separately)

People have been using this PCMCIA card with Linux as well. Similar points apply, those being that you need a supported PCMCIA chipset on your notebook, and that you will have to patch the PCMCIA support into the standard kernel.

See 9.2 (PCMCIA Support) in this document, and if you can, have a look at:

Don's PCMCIA Stuff `<http://cesdis.gsfc.nasa.gov/linux/pcmcia.html>`

5.20.3 IBM Token Ring

Status: Semi-Supported, Driver Name: ibmtr

To support token ring requires more than only writing a device driver, it also requires writing the source routing routines for token ring. It is the source routing that would be the most time comsuming to write.

Peter De Schrijver has been spending some time on Token Ring lately. and has worked with IBM ISA and MCA token ring cards.

The present token ring code has been included into the first of the 1.3.x series kernels.

Peter says that it was originally tested on an MCA 16/4 Megabit Token Ring board, but it should work with other Tropic based boards.

5.21 ICL Ethernet Cards

5.21.1 ICL EtherTeam 16i/32

Status: Supported, Driver Name: eth16i

Mika Kuoppala (miku@pupu.elt.icl.fi) wrote this driver, and it was included into early 1.3.4x kernels. It uses the Fujitsu MB86965 chip that is also used on the at1700 cards.

5.22 Intel Ethernet Cards

Note that the naming of the various Intel cards is ambiguous and confusing at best. If in doubt, then check the `i8xxxx` number on the main chip on the card or for PCI cards, use the PCI information in the `/proc` directory and then compare that to the numbers listed here.

5.22.1 Ether Express

Status: Supported, Driver Name: eexpress

This card uses the intel i82586. Earlier versions of this driver (in v1.2 kernels) were classed as alpha-test, as it didn't work well for most people. The driver in the v2.0 kernel seems to work much better for those who have tried it, although the driver source still lists it as experimental and more problematic on faster machines.

The comments at the top of the driver source list some of the problems (and fixes!) associated with these cards. The slowdown hack of replacing all the `outb` with `outb_p` in the driver has been reported to avoid lockups for at least one user.

5.22.2 Ether Express PRO/10

Status: Supported, Driver Name: eepro

Bao Chau Ha has written a driver for these cards that has been included into early 1.3.x kernels. It may also work with some of the Compaq built-in ethernet systems that are based on the i82595 chip.

5.22.3 Ether Express PRO/10 PCI (EISA)

Status: Semi-Supported, Driver Name: ? (distributed separately)

John Stalba (stalba@ultranet.com) has written a driver for the PCI version. These cards use the PLX9036 PCI interface chip with the Intel i82596 LAN controller chip. If your card has the i82557 chip, then you *don't* have this card, but rather the version discussed next, and hence want the EEPro100 driver instead.

You can get the alpha driver for the PRO/10 PCI card, along with instructions on how to use it at:

EEPro10 Driver `<http://www.ultranet.com/~stalba/eep10pci.html>`

If you have the EISA card, you will probably have to hack the driver a bit to account for the different (PCI vs. EISA) detection mechanisms that are used in each case.

5.22.4 Ether Express PRO 10/100B

Status: Supported, Driver Name: eepro100

Note that this driver will *not* work with the older 100A cards. The chip numbers listed in the driver are i82557/i82558. For driver updates and/or driver support, have a look at:

EEPro-100B Page `<http://cesdis.gsfc.nasa.gov/linux/drivers/eepro100.html>`

To subscribe to the mailing list relating to this driver, do:

`echo subscribe | /bin/mail linux-eepro100-request@cesdis.gsfc.nasa.gov`

Apparently Donald had to sign a non-disclosure agreement that stated he could actually disclose the driver source code! How is that for sillyness on intel's part?

5.23 Kingston

Kingston make various cards, including NE2000+, AMD PCnet based cards, and DEC tulip based cards. Most of these cards should work fine with their respective driver. See *Kingston Web Page* `<http://www.kingston.com>`

The KNE40 DEC 21041 tulip based card is reported to work fine with the generic tulip driver.

5.24 LinkSys

LinkSys make a handful of different NE2000 clones, some straight ISA cards, some ISA plug and play and some even ne2000-PCI clones based on one of the supported ne2000-PCI chipsets. There are just too many models to list here.

LinkSys are linux-friendly, with a linux specific WWW support page, and even have Linux printed on the boxes of some of their products. Have a look at:

`http://www.linksys.com/support/solution/nos/linux.htm`

5.24.1 LinkSys Etherfast 10/100 Cards.

Status: Supported, Driver Name: tulip

Note that with these cards there have been several 'revisions' (i.e. different chipset used) all with the same card name. The 1st used the DEC chipset. The 2nd revision used the Lite-On PNIC 82c168 PCI Network Interface Controller, and support for this was merged into the standard tulip driver (as of version 0.83 and newer). More PNIC information is available at:

`http://cesdis.gsfc.nasa.gov/linux/drivers/pnic.html`

More information on the various versions of these cards can be found at the LinkSys WWW site mentioned above.

5.24.2 LinkSys Pocket Ethernet Adapter Plus (PEAEPP)

Status: Supported, Driver Name: de620

This is supposedly a DE-620 clone, and is reported to work well with that driver. See 5.14.6 (DE-620) for more information.

5.24.3 LinkSys PCMCIA Adaptor

Status: Supported, Driver Name: de650 (?)

This is supposed to be a re-badged DE-650. See 5.14.7 (DE-650) for more information.

5.25 Microdyne

5.25.1 Microdyne Exos 205T

Status: Semi-Supported, Driver Name: ?

Another i82586 based card. Dirk Niggemann `dirk-n@dircon.co.uk` has written a driver that he classes as "pre-alpha" that he would like people to test. Mail him for more details.

5.26 Mylex

Mylex can be reached at the following numbers, in case anyone wants to ask them anything.

```
MYLEX CORPORATION, Fremont
Sales:  800-77-MYLEX, (510) 796-6100
FAX:    (510) 745-8016.
```

They also have a web site: *Mylex WWW Site* `<http://www.mylex.com>`

5.26.1 Mylex LNE390A, LNE390B

Status: Supported, Driver Name: lne390 (+8390)

These are fairly old EISA cards that make use of a shared memory implementation similar to the wd80x3. A driver for these cards is available in the current 2.1.x series of kernels. Ensure you set the shared memory address below 1MB or above the highest address of the physical RAM installed in the machine.

5.26.2 Mylex LNP101

Status: Supported, Driver Name: de4x5, tulip

This is a PCI card that is based on DEC's 21040 chip. It is selectable between 10BaseT, 10Base2 and 10Base5 output. The LNP101 card has been verified to work with the generic 21040 driver.

See the section on the 21040 chip (5.16.4 (DEC 21040)) for more information.

5.26.3 Mylex LNP104

Status: Semi-Supported, Driver Name: de4x5, tulip

The LNP104 uses the DEC 21050 chip to deliver *four* independent 10BaseT ports. It should work with recent 21040 drivers that know how to share IRQs, but nobody has reported trying it yet (that I am aware of).

5.27 Novell Ethernet, NExxxx and associated clones.

The prefix 'NE' came from Novell Ethernet. Novell followed the cheapest NatSemi databook design and sold the manufacturing rights (spun off?) Eagle, just to get reasonably-priced ethercards into the market. (The now ubiquitous NE2000 card.)

5.27.1 NE1000, NE2000

Status: Supported, Driver Name: ne (+8390)

The ne2000 is now a generic name for a bare-bones design around the NatSemi 8390 chip. They use programmed I/O rather than shared memory, leading to easier installation but slightly lower performance and a few problems. Some of the more common problems that arise with NE2000 cards are listed in 3.4 (Problems with...)

Some NE2000 clones use the National Semiconductor 'AT/LANTic' 83905 chip, which offers a shared memory mode similar to the wd8013 and EEPROM software configuration. The shared memory mode will offer less CPU usage (i.e. more efficient) than the programmed I/O mode.

In general it is not a good idea to put a NE2000 clone at I/O address `0x300` because nearly *every* device driver probes there at boot. Some poor NE2000 clones don't take kindly to being prodded in the wrong areas, and will respond by locking your machine. Also `0x320` is bad because SCSI drivers probe into `0x330`.

Donald has written a NE2000 diagnostic program (ne2k.c) for all ne2000 cards. See 7.2 (Diagnostic Programs) for more information.

If you intend on using this driver as a loadable module you should probably see 10.2 (Using the Ethernet Drivers as Modules) for module specific information.

5.27.2 NE2000-PCI (RealTek/Winbond/Compex)

Status: Supported, Driver Name: ne, ne2k-pci (+8390)

Yes, believe it or not, people are making PCI cards based on the more than ten year old interface design of the ne2000. At the moment nearly all of these cards are based on the RealTek 8029 chip, or the Winbond 89c940 chip. The Compex, KTI, VIA and Netvin cards apparently also use these chips, but have a different PCI ID.

The latest v2.0 kernel has support to automatically detect all these cards and use them. (If you are using a kernel v2.0.34 or older, you should upgrade to ensure your card will be detected.) There are now two drivers to choose from; the original ISA/PCI `ne.c` driver, and a relatively new PCI-only `ne2k-pci.c` driver.

To use the original ISA/PCI driver you have to say 'Y' to the 'Other ISA cards' option when running `make config` as you are actually using the same NE2000 driver as the ISA cards use. (That should also give you a hint that these cards aren't anywhere as intelligent as say a PCNet-PCI or DEC 21040 card...)

The newer PCI-only driver differs from the ISA/PCI driver in that all the support for old NE1000 8 bit cards has been removed and that data is moved to/from the card in bigger blocks, without any intervening pauses that the older ISA-NE2000's required for reliable operation. The result is a driver that is slightly smaller and slightly more efficient, but don't get too excited as the difference will not be obvious under normal use. (If you really wanted maximum efficiency/low CPU use, then a PCI-NE2000 is simply a very poor choice.) Driver updates and more information can be found at:

`http://cesdis.gsfc.nasa.gov/linux/drivers/ne2k-pci.html`

If you have a NE2000 PCI card that is *not* detected by the most current version of the driver, please contact the maintainer of the NE2000 driver as listed in `/usr/src/linux/MAINTAINERS` along with the output from a `cat /proc/pci` and `dmesg` so that support for your card can also be added to the driver.

Also note that various card makers have been known to put 'NE2000 Compatible' stickers on their product boxes even when it is completely different (e.g. PCNet-PCI or RealTek 8139). If in doubt check the main chip number against this document.

5.27.3 NE-10/100

Status: Not Supported.

These are ISA 100Mbps cards based on the National Semiconductor DP83800 and DP83840 chips. There is currently no driver support, nor has anyone reported that they are working on a driver. Apparently documentation on the chip is unavailable with the exception of a single PDF file that doesn't give enough details for a driver.

5.27.4 NE1500, NE2100

Status: Supported, Driver Name: lance

These cards use the original 7990 LANCE chip from AMD and are supported using the Linux lance driver. Newer NE2100 clones use the updated PCnet/ISA chip from AMD.

Some earlier versions of the lance driver had problems with getting the IRQ line via autoIRQ from the original Novell/Eagle 7990 cards. Hopefully this is now fixed. If not, then specify the IRQ via LILO, and let us know that it still has problems.

DMA selection and chip numbering information can be found in 5.4.1 (AMD LANCE).

More technical information on LANCE based cards can be found in 8.5 (Notes on AMD...)

5.27.5 NE/2 MCA

Status: Semi-Supported, Driver Name: ne2

There were a few NE2000 microchannel cards made by various companies. This driver, available in v2.2 kernels, will detect the following MCA cards: Novell Ethernet Adapter NE/2, Compex ENET-16 MC/P, and the Arco Ethernet Adapter AE/2.

5.27.6 NE3200

Status: Not Supported.

This old EISA card uses a 8MHz 80186 in conjunction with an i82586. Nobody is working on a driver for it, as there is no information available on the card, and no real demand for a driver either.

5.27.7 NE3210

Status: Supported, Driver Name: ne3210 (+8390)

This EISA card is completely different from the NE3200, as it uses a Nat Semi 8390 chip. The driver can be found in the v2.2 kernel source tree. Ensure you set the shared memory address below 1MB or above the highest address of the physical RAM installed in the machine.

5.27.8 NE5500

Status: Supported, Driver Name: pcnet32

These are just AMD PCnet-PCI cards ('970A) chips. More information on LANCE/PCnet based cards can be found in 5.4.1 (AMD LANCE).

5.28 Proteon

5.28.1 Proteon P1370-EA

Status: Supported, Driver Name: ne (+8390)

Apparently this is a NE2000 clone, and works fine with Linux.

5.28.2 Proteon P1670-EA

Status: Supported, Driver Name: de4x5, tulip

This is yet another PCI card that is based on DEC's Tulip chip. It has been reported to work fine with Linux.

See the section on the 21040 chip (5.16.4 (DEC 21040)) for more driver information.

5.29 Pure Data

5.29.1 PDUC8028, PDI8023

Status: Supported, Driver Name: wd (+8390)

The PureData PDUC8028 and PDI8023 series of cards are reported to work, thanks to special probe code contributed by Mike Jagdis `jaggy@purplet.demon.co.uk`. The support is integrated with the WD driver.

5.30 Racal-Interlan

Racal Interlan can be reached via WWW at `www.interlan.com`. I believe they were also known as MiCom-Interlan at one point in the past.

5.30.1 ES3210

Status: Semi-Supported, Driver Name: es3210

This is an EISA 8390 based shared memory card. An experimetal driver is shipped with v2.2 kernels and it is reported to work fine, but the EISA IRQ and shared memory address detection appears not to work with (at least) the early revision cards. (This problem is not unique to the Linux world either...) In that case, you have to supply them to the driver. For example, card at IRQ 5 and shared memory `0xd0000`, with a modular driver, add `options es3210 irq=5 mem=0xd0000` to `/etc/conf.modules`. Or with the driver compiled into the kernel, supply at boot `ether=5,0,0xd0000,eth0` The I/O base is automatically detected and hence a value of zero should be used.

5.30.2 NI5010

Status: Semi-Supported, Driver Name: ni5010

You used to have to go get the driver for these old 8 bit MiCom-Interlan cards separately, but now it is shipped with the v2.2 kernels as an experimental driver.

5.30.3 NI5210

Status: Semi-Supported, Driver Name: ni52

This card also uses one of the Intel chips. Michael Hipp has written a driver for this card. It is included in the standard kernel as an 'alpha' driver. Michael would like to hear feedback from users that have this card. See 3.1 (Alpha Drivers) for important information on using alpha-test ethernet drivers with Linux.

5.30.4 NI6510 (not EB)

Status: Semi-Supported, Driver Name: ni65

There is also a driver for the LANCE based NI6510, and it is also written by Michael Hipp. Again, it is also an 'alpha' driver. For some reason, this card is not compatible with the generic LANCE driver. See 3.1 (Alpha Drivers) for important information on using alpha-test ethernet drivers with Linux.

5.30.5 EtherBlaster (aka NI6510EB)

Status: Supported, Driver Name: lance

As of kernel 1.3.23, the generic LANCE driver had a check added to it for the `0x52, 0x44` NI6510EB specific signature. Others have reported that this signature is not the same for all NI6510EB cards however, which will cause the lance driver to not detect your card. If this happens to you, you can change the probe (at about line 322 in lance.c) to printk() out what the values are for your card and then use them instead of the `0x52, 0x44` defaults.

The cards should probably be run in 'high-performance' mode and not in the NI6510 compatible mode when using the lance driver.

5.31 RealTek

5.31.1 RealTek RTL8002/8012 (AT-Lan-Tec) Pocket adaptor

Status: Supported, Driver Name: atp

This is a generic, low-cost OEM pocket adaptor being sold by AT-Lan-Tec, and (likely) a number of other suppliers. A driver for it is included in the standard kernel. Note that there is substantial information contained in the driver source file 'atp.c'.

Note that the device name that you pass to `ifconfig` was *not* `eth0` but `atp0` for earlier versions of this driver.

5.31.2 RealTek 8009

Status: Supported, Driver Name: ne (+8390)

This is an ISA NE2000 clone, and is reported to work fine with the linux NE2000 driver. The `rset8009.exe` program can be obtained from RealTek's WWW site at `http://www.realtek.com.tw` - or via ftp from the same site.

5.31.3 RealTek 8019

Status: Supported, Driver Name: ne (+8390)

This is a Plug and Pray version of the above. Use the DOS software to disable PnP and enable jumperless configuration; set the card to a sensible I/O address and IRQ and you should be ready to go. (If using the driver as a module, don't forget to add an `io=0xNNN` option to `/etc/conf.modules`). The `rset8019.exe` program can be obtained from RealTek's WWW site at `http://www.realtek.com.tw` - or via ftp from the same site.

5.31.4 RealTek 8029

Status: Supported, Driver Name: ne, ne2k-pci (+8390)

This is a PCI single chip implementation of a NE2000 clone. Various vendors are now selling cards with this chip. See 5.27.2 (NE2000-PCI) for information on using any of these cards. Note that this is still a 10+ year old design just glued onto a PCI bus. Performance won't be staggeringly better than the equivalent ISA model.

5.31.5 RealTek 8129/8139

Status: Semi-Supported, Driver Name: rtl8139

Another PCI single chip ethernet solution from RealTek. A driver for cards based upon this chip was included in the v2.0.34 release of linux. You currently have to answer 'Y' when asked if you want experimental drivers for v2.2 kernels to get access to this driver. For more information, see:

`http://cesdis.gsfc.nasa.gov/linux/drivers/rtl8139.html`

5.32 Sager

5.32.1 Sager NP943

Status: Semi-Supported, Driver Name: 3c501

This is just a 3c501 clone, with a different S.A. PROM prefix. I assume it is equally as brain dead as the original 3c501 as well. The driver checks for the NP943 I.D. and then just treats it as a 3c501 after that. See 5.1.1 (3Com 3c501) for all the reasons as to why you really don't want to use one of these cards.

5.33 Schneider & Koch

5.33.1 SK G16

Status: Supported, Driver Name: sk_g16

This driver was included into the v1.1 kernels, and it was written by PJD Weichmann and SWS Bern. It appears that the SK G16 is similar to the NI6510, in that it is based on the first edition LANCE chip (the 7990). Once again, it appears as though this card won't work with the generic LANCE driver.

5.34 SEEQ

5.34.1 SEEQ 8005

Status: Supported, Driver Name: seeq8005

This driver was included into early 1.3.x kernels, and was written by Hamish Coleman. There is little information about the card included in the driver, and hence little information to be put here. If you have a question, you are probably best off e-mailing hamish@zot.apana.org.au

5.35 SMC (Standard Microsystems Corp.)

The ethernet part of Western Digital was bought out by SMC many years ago when the wd8003 and wd8013 were the main product. Since then SMC has continued making 8390 based ISA cards (Elite16, Ultra, EtherEZ) and also added several PCI products to their range.

Contact information for SMC:

SMC / Standard Microsystems Corp., 80 Arkay Drive, Hauppage, New York, 11788, USA. Technical Support via phone: 800-992-4762 (USA) or 800-433-5345 (Canada) or 516-435-6250 (Other Countries). Literature requests: 800-SMC-4-YOU (USA) or 800-833-4-SMC (Canada) or 516-435-6255 (Other Countries). Technical Support via E-mail: `techsupt@ccmail.west.smc.com`. FTP Site: `ftp.smc.com`. WWW Site: *SMC* `<http://www.smc.com>`.

5.35.1 WD8003, SMC Elite

Status: Supported, Driver Name: wd (+8390)

These are the 8-bit versions of the card. The 8 bit 8003 is slightly less expensive, but only worth the savings for light use. Note that some of the non-EEPROM cards (clones with jumpers, or old *old* old wd8003 cards) have no way of reporting the IRQ line used. In this case, auto-irq is used, and if that fails, the driver silently assings IRQ 5. You can get the SMC setup/driver disks from SMC's ftp site. Note that some of the newer SMC 'SuperDisk' programs will fail to detect the real old EEPROM-less cards. The file `SMCDSK46.EXE` seems to be a good all-round choice. Also the jumper settings for all their cards are in an ASCII text file in the aforementioned archive. The latest (greatest?) version can be obtained from `ftp.smc.com`.

As these are basically the same as their 16 bit counterparts (WD8013 / SMC Elite16), you should see the next section for more information.

5.35.2 WD8013, SMC Elite16

Status: Supported, Driver Name: wd (+8390)

Over the years the design has added more registers and an EEPROM. (The first wd8003 cards appeared about ten years ago!) Clones usually go by the '8013' name, and usually use a non-EEPROM (jumpered) design. Late model SMC cards will have the SMC 83c690 chip instead of the original Nat Semi DP8390 found on earlier cards. The shared memory design makes the cards a bit faster than PIO cards, especially with larger packets. More importantly, from the driver's point of view, it avoids a few bugs in the programmed-I/O mode of the 8390, allows safe multi-threaded access to the packet buffer, and it doesn't have a programmed-I/O data register that hangs your machine during warm-boot probes.

Non-EEPROM cards that can't just read the selected IRQ will attempt auto-irq, and if that fails, they will silently assign IRQ 10. (8 bit versions will assign IRQ 5)

Cards with a non standard amount of memory on board can have the memory size specified at boot (or as an option in `/etc/conf.modules` if using modules). The standard memory size is 8kB for an 8bit card and 16kB for a 16bit card. For example, the older WD8003EBT cards could be jumpered for 32kB memory. To make full use of that RAM, you would use something like (for I/O=0x280 and IRQ 9):

```
LILO: linux ether=9,0x280,0xd0000,0xd8000,eth0
```

Also see 3.5 (8013 problems) for some of the more common problems and frequently asked questions that pop up often.

If you intend on using this driver as a loadable module you should probably see 10.2 (Using the Ethernet Drivers as Modules) for module specific information.

5.35.3 SMC Elite Ultra

Status: Supported, Driver Name: smc-ultra (+8390)

This ethercard is based on the 83c790 chip from SMC, which has a few new features over the 83c690. While it has a mode that is similar to the older SMC ethercards, it's not entirely compatible with the old WD80*3 drivers. However, in this mode it shares most of its code with the other 8390 drivers, while operating slightly faster than a WD8013 clone.

Since part of the Ultra *looks like* an 8013, the Ultra probe is supposed to find an Ultra before the wd8013 probe has a chance to mistakenly identify it.

Donald mentioned that it is possible to write a separate driver for the Ultra's 'Altego' mode which allows chaining transmits at the cost of inefficient use of receive buffers, but that will probably not happen.

Bus-Master SCSI host adaptor users take note: In the manual that ships with Interactive UNIX, it mentions that a bug in the SMC Ultra will cause data corruption with SCSI disks being run from an aha-154X host adaptor. This will probably bite aha-154X compatible cards, such as the BusLogic boards, and the AMI-FastDisk SCSI host adaptors as well.

SMC has acknowledged the problem occurs with Interactive, and older Windows NT drivers. It is a hardware conflict with early revisions of the card that can be worked around in the driver design. The current Ultra driver protects against this by only enabling the shared memory during data transfers with the card. Make sure your kernel version is at least 1.1.84, or that the driver version reported at boot is at least `smc-ultra.c:v1.12` otherwise you are vulnerable.

If you intend on using this driver as a loadable module you should probably see 10.2 (Using the Ethernet Drivers as Modules) for module specific information.

5.35.4 SMC Elite Ultra32 EISA

Status: Supported, Driver Name: smc-ultra32 (+8390)

This EISA card shares a lot in common with its ISA counterpart. A working (and stable) driver is included in both v2.0 and v2.2 kernels. Thanks go to Leonard Zubkoff for purchasing some of these cards so that linux support could be added for them.

5.35.5 SMC EtherEZ (8416)

Status: Supported, Driver Name: smc-ultra (+8390)

This card uses SMC's 83c795 chip and supports the Plug 'n Play specification. It also has an *SMC Ultra* compatible mode, which allows it to be used with the Linux Ultra driver. For best results, use the SMC supplied program (avail. from their www/ftp site) to disable PnP and configure it for shared memory mode. See the above information for notes on the Ultra driver.

For v1.2 kernels, the card had to be configured for shared memory operation. However v2.0 kernels can use the card in shared memory or programmed I/O mode. Shared memory mode will be slightly faster, and use less CPU resources as well.

5.35.6 SMC EtherPower PCI (8432)

Status: Supported, Driver Name: de4x5, tulip

NB: The EtherPower II is an entirely different card. See below! These cards are a basic DEC 21040 implementation, i.e. one big chip and a couple of transceivers. Donald has used one of these cards for his development of the generic 21040 driver (aka `tulip.c`). Thanks to Duke Kamstra, once again, for supplying a card to do development on.

Some of the later revisons of this card use the newer DEC 21041 chip, which may cause problems with older versions of the tulip driver. If you have problems, make sure you are using the latest driver release, which may not yet be included in the current kernel source tree.

See 5.16.4 (DEC 21040) for more details on using one of these cards, and the current status of the driver.

Apparently, the latest revision of the card, the EtherPower-II uses the 9432 chip. It is unclear at the moment if this one will work with the present driver. As always, if unsure, check that you can return the card if it doesn't work with the linux driver *before* paying for the card.

5.35.7 SMC EtherPower II PCI (9432)

Status: Semi-Supported, Driver Name: epic100

These cards, based upon the SMC 83c170 chip, are entirely different than the Tulip based cards. A new driver has been included in kernels v2.0 and v2.2 to support these cards. For more details, see:

`http://cesdis.gsfc.nasa.gov/linux/drivers/epic100.html`

5.35.8 SMC 3008

Status: Not Supported.

These 8 bit cards are based on the Fujitsu MB86950, which is an ancient version of the MB86965 used in the Linux at1700 driver. Russ says that you could probably hack up a driver by looking at the at1700.c code and his DOS packet driver for the Tiara card (tiara.asm). They are not very common.

5.35.9 SMC 3016

Status: Not Supported.

These are 16bit I/O mapped 8390 cards, much similar to a generic NE2000 card. If you can get the specifications from SMC, then porting the NE2000 driver would probably be quite easy. They are not very common.

5.35.10 SMC-9000 / SMC 91c92/4

Status: Supported, Driver Name: smc9194

The SMC9000 is a VLB card based on the 91c92 chip. The 91c92 appears on a few other brand cards as well, but is fairly uncommon. Erik Stahlman (erik@vt.edu) has written this driver which is in v2.0 kernels, but not in the older v1.2 kernels. You may be able to drop the driver into a v1.2 kernel source tree with minimal difficulty.

5.35.11 SMC 91c100

Status: Semi-Supported, Driver Name: smc9194

The SMC 91c92 driver is supposed to work for cards based on this 100Base-T chip, but at the moment this is unverified.

5.36 Texas Instruments

5.36.1 ThunderLAN

Status: Supported, Driver Name: tlan

This driver covers many Compaq built-in ethernet devices, including the NetFlex and Netelligent groups. It also supports the Olicom 2183, 2185, 2325 and 2326 products.

5.37 Thomas Conrad

5.37.1 Thomas Conrad TC-5048

This is yet another PCI card that is based on DEC's 21040 chip.

See the section on the 21040 chip (5.16.4 (DEC 21040)) for more information.

5.38 VIA

You probably won't see a VIA networking card, as VIA make several networking chips that are then used by others in the construction of an ethernet card. They have a WWW site at:

`http://www.via.com.tw/`

5.38.1 VIA 86C926 Amazon

Status: Supported, Driver Name: ne, ne2k-pci (+8390)

This controller chip is VIA's PCI-NE2000 offering. You can choose between the ISA/PCI `ne.c` driver or the PCI-only `ne2k-pci.c` driver. See the PCI-NE2000 section for more details.

5.38.2 VIA 86C100A Rhine II (and 3043 Rhine I)

Status Supported, Driver Name: via-rhine

This relatively new driver can be found in current 2.0 and 2.1 kernels. It is an improvement over the 86C926 NE2000 chip in that it supports bus master transfers, but strict 32 bit buffer alignment requirements limit the benefit gained from this. For more details and driver updates, see:

`http://cesdis.gsfc.nasa.gov/linux/drivers/via-rhine.html`

5.39 Western Digital

Please see 5.35 (SMC) for information on SMC cards. (SMC bought out Western Digital's network card section many years ago.)

5.40 Winbond

Winbond don't really make and sell complete cards to the general public – instead they make single chip ethernet solutions that other companies buy, stick onto a PCI board with their own name and then sell through retail stores.

5.40.1 Winbond 89c840

Status: Semi-Supported, Driver Name: winbond-840

This driver isn't currently shipped with the kernel, as it is in the testing phase. It is available at:

`http://cesdis.gsfc.nasa.gov/linux/drivers/test/winbond-840.c`

5.40.2 Winbond 89c940

Status: Supported, Driver Name: ne, ne2k-pci (+8390)

This chip is one of the two commonly found on the low price PCI ne2000 cards sold by lots of manufacturers. Note that this is still a 10+ year old design just glued onto a PCI bus. Performance won't be staggeringly better than the equivalent ISA model.

5.41 Xircom

For the longest time, Xircom wouldn't release the programming information required to write a driver, unless you signed your life away. Apparently enough linux users have pestered them for driver support (they claim to support all popular networking operating systems...) so that they have changed their policy to allow documentation to be released without having to sign a non-disclosure agreement. Some people have said they they will release the source code to the SCO driver, while others have been told that they are no longer providing information on 'obsolete' products like the earlier PE models. If you are interested and want to check into this yourself, you can reach Xircom at 1-800-874-7875, 1-800-438-4526 or +1-818-878-7600.

5.41.1 Xircom PE1, PE2, PE3-10B*

Status: Not Supported.

Not to get your hopes up, but if you have one of these parallel port adaptors, you may be able to use it in the DOS emulator with the Xircom-supplied DOS drivers. You will have to allow DOSEMU access to your parallel port, and will probably have to play with SIG (DOSEMU's Silly Interrupt Generator).

5.41.2 Xircom PCMCIA Cards

Status: Semi-Supported, Driver Name: ????

Some of the Xircom PCMCIA card(s) have drivers that are available with David Hinds PCMCIA package. Check there for the most up to date indformation

5.42 Zenith

5.42.1 Z-Note

Status: Supported, Driver Name: znet

The built-in Z-Note network adaptor is based on the Intel i82593 using *two* DMA channels. There is an (alpha?) driver available in the present kernel version. As with all notebook and pocket adaptors, it is under the 'Pocket and portable adaptors' section when running `make config`. Also note that the IBM ThinkPad 300 is compatible with the Z-Note.

5.43 Znyx

5.43.1 Znyx ZX342 (DEC 21040 based)

Status: Supported, Driver Name: de4x5, tulip

You have a choice of *two* drivers for cards based on this chip. There is the DE425 driver written by David, and the generic 21040 driver that Donald has written.

Note that as of 1.1.91, David has added a compile time option that may allow non-DEC cards (such as the Znyx cards) to work with this driver. Have a look at `README.de4x5` for details.

See 5.16.4 (DEC 21040) for more information on these cards, and the present driver situation.

5.44 Identifying an Unknown Card

Okay, so your uncle's cousin's neighbour's friend had a brother who found an old ISA ethernet card in the AT case he was using as a cage for his son's pet hampster. Somehow you ended up with the card and want to try and use it with linux, but nobody has a clue what the card is and there isn't any documentation.

First of all, look for any obvious model numbers that might give a clue. Any model number that contains 2000 will most likely be a NE2000 clone. Any cards with 8003 or 8013 on them somewhere will be Western/Digital WD80x3 cards or SMC Elite cards or clones of them.

5.44.1 Identifying the Network Interface Controller

Look for the biggest chip on the card. This will be the network controller (NIC) itself, and most can be identified by the part number. If you know which NIC is on the card, the following might be able to help you figure out what card it is.

Probably still the most common NIC is the National Semiconductor DP8390 aka NS32490 aka DP83901 aka DP83902 aka DP83905 aka DP83907. And those are just the ones made by National! Other companies such as Winbond and UMC make DP8390 and DP83905 clone parts, such as the Winbond 89c904 (DP83905 clone) and the UMC 9090. If the card has some form of 8390 on it, then chances are it is a ne1000 or ne2000 clone card. The second most common 8390 based card are wd80x3 cards and clones. Cards with a DP83905 can be configured to be an ne2000 *or* a wd8013. Never versions of the genuine wd80x3 and SMC Elite cards have an 83c690 in place of the original DP8390. The SMC Ultra cards have an 83c790, and use a slightly different driver than the wd80x3 cards. The SMC EtherEZ cards have an 83c795, and use the same driver as the SMC Ultra. All BNC cards based on some sort of 8390 or 8390 clone will usually have an 8392 (or 83c692, or ???392) 16 pin DIP chip very close to the BNC connector.

Another common NIC found on older cards is the Intel i82586. Cards having this NIC include the 3c505, 3c507, 3c523, Intel EtherExpress-ISA, Microdyne Exos-205T, and the Racal-Interlan NI5210.

The original AMD LANCE NIC was numbered AM7990, and newer revisions include the 79c960, 79c961, 79c965, 79c970, and 79c974. Most cards with one of the above will work with the Linux LANCE driver, with the exception of the old Racal-Interlan NI6510 cards that have their own driver.

Newer PCI cards having a DEC 21040, 21041, 21140, or similar number on the NIC should be able to use the linux tulip or de4x5 driver.

Other PCI cards having a big chip marked RTL8029 or 89C940 or 86C926 are ne2000 clone cards, and the ne driver in linux version v2.0 and up should automatically detect these cards at boot.

5.44.2 Identifying the Ethernet Address

Each ethernet card has its own six byte address that is unique to that card. The first three bytes of that address are the same for each card made by that particular manufacturer. For example all SMC cards start with `00:00:c0`. The last three are assigned by the manufacturer uniquely to each individual card as they are produced.

If your card has a sticker on it giving all six bits of its address, you can look up the vendor from the first three. However it is more common to see only the last three bytes printed onto a sticker attached to a socketed PROM, which tells you nothing.

You can determine which vendors have which assigned addresses from RFC-1340. Apparently there is a more up to date listing available in various places as well. Try a WWW or FTP search for `EtherNet-codes` or `Ethernet-codes` and you will find something.

5.44.3 Tips on Trying to Use an Unknown Card

If you are still not sure what the card is, but have at least narrowed it down some, then you can build a kernel with a whole bunch of drivers included, and see if any of them autodetect the card at boot.

If the kernel doesn't detect the card, it may be that the card is not configured to one of the addresses that the driver probes when looking for a card. In this case, you might want to try getting `scanport.tar.gz` from your local linux ftp site, and see if that can locate where your card is jumpered for. It scans ISA I/O space from `0x100` to `0x3ff` looking for devices that aren't registered in `/proc/ioports`. If it finds an unknown device starting at some particular address, you can then explicity point the ethernet probes at that address with an `ether=` boot argument.

If you manage to get the card detected, you can then usually figure out the unknown jumpers by changing them one at a time and seeing at what I/O base and IRQ that the card is detected at. The IRQ settings can also usually be determined by following the traces on the back of the card to where the jumpers are soldered through. Counting the 'gold fingers' on the backside, from the end of the card with the metal bracket, you have IRQ 9, 7, 6, 5, 4, 3, 10, 11, 12, 15, 14 at fingers 4, 21, 22, 23, 24, 25, 34, 35, 36, 37, 38 respectively. Eight bit cards only have up to finger 31.

Jumpers that appear to do nothing usually are for selecting the memory address of an optional boot ROM. Other jumpers that are located near the BNC or RJ-45 or AUI connectors are usually to select the output media. These are also typically near the 'black box' voltage converters marked YCL, Valor, or Fil-Mag.

A nice collection of jumper settings for various cards can be found at the following URL:

Ethercard Settings `<http://www.slug.org.au/NIC/>`

5.45 Drivers for Non-Ethernet Devices

There are a few other drivers that are in the linux source that present an *ethernet-like* device to network programs, while not really being ethernet. These are briefly listed here for completeness.

`dummy.c` - The purpose of this driver is to provide a device to point a route through, but not to actually transmit packets.

`eql.c` - Load Equalizer, enslaves multiple devices (usually modems) and balances the Tx load across them while presenting a single device to the network programs.

`ibmtr.c` - IBM Token Ring, which is not really ethernet. Broken-Ring requires source routing and other uglies.

`loopback.c` - Loopback device, for which all packets from your machine and destined for your own machine go. It essentially just moves the packet off the Tx queue and onto the Rx queue.

`pi2.c` - Ottawa Amateur Radio Club PI and PI2 interface.

`plip.c` - Parallel Line Internet Protocol, allows two computers to send packets to each other over two joined parallel ports in a point-to-point fashion.

`ppp.c` - Point-to-Point Protocol (RFC1331), for the Transmission of Multi-protocol Datagrams over a Point-to-Point Link (again usually modems).

`slip.c` - Serial Line Internet Protocol, allows two computers to send packets to each other over two joined serial ports (usually via modems) in a point-to-point fashion.

`tunnel.c` - Provides an IP tunnel through which you can tunnel network traffic transparently across subnets

`wavelan.c` - An Ethernet-like radio transceiver controlled by the Intel 82586 coprocessor which is used on other ethercards such as the Intel EtherExpress.

제 6 절 케이블들, 동축선, 꼬임선

스크래치(?)에서 네트워크를 시작한다면, 여러분은 아마도 thin 이더넷 (BNC 커넥터를 가진 RG58 동축 케이블)이든 10baseT (8개의 선으로 된 RJ-45 '전화' 커넥터를 가진 텔코 방식의 꼬임선 케이블)이든 어느것을 사용할지 결정해야만 한다. N 커넥터를 가진 RG-5 케이블의, 구시대 thick 이더넷은 더이상 찾아볼수도 없다.

케이블에 대한 소개는 다음을 보라. 2.6 (Type of cable...) 또 *comp.dcom.lans.ethernet*의 FAQ에는 케이블이나 그런것들에 대한 수많은 유용한 정보 들을 찾을수 있다. rtfm.mit.edu의 `/pub/usenet-by-hierarchy/`에 FTP로 들어가 보면 그 뉴스그룹의 FAQ를 찾을수 있다.

6.1 Thin Ethernet (thinnet)

Thin 이더넷 케이블을 별로 비싸지 않다. 만일 여러분이 직접 자신의 solid-core RG58A를 만든 다면, 0.27/m이다. 그리고 표준 RG58AU는 0.45/m이다. 꼬인 BNC 커넥터는 개당 2가 되지않으며 다른 물품들도 가격은 비슷하다. 각 케이블의 끝에는 50옴짜리 터미네 이터로 막아주어야 하는데, 이것은 쌍으로 개당 2이다. 또한 '뿌리'가 없으면 여러분의 케이블은 쓸수 없다 – 'T' 커넥터는 반드시 이더넷 카드에 직접 연결되어야 한다.

thinnet을 사용하는 데는 두가지 단점이 있다. 첫째는 그 속도가 10Mb/sec 까지 제한되어 있다 는 것이다 - 꼬임선은 100Mb/sec 까지 된다. 두번째 단점은 만일 여러분이 머신들이 큰 루프를 만들며 연결시켰다면, 몇몇 멍청이가 그 티(T 커넥터:역자주) 주변의 케이블을 떼어내서 루프를 깨트리고, 50옴 터미네이터의 용량대신 무한대의 전기저항 (개방형 회로)으로 전체 네트워크를 정지시키게 된다. 만일 여러분이 전체 서브넷을 죽이지 않고 카드에서 티 조각을 제거한다고 하 더라도, 오랫동안 티에서 케이블을 제거하지 못한다는 것을 알아두라. 물론 실제 티에서 케이블 을 떼어내면서 머신을 방해할 것이다. 8-) 그리고 만일 여러분이 두 머신으로 구성된 작은 네트 워크로 한다해도, *여전히* 티와 50옴 터미네이터가 필요하다. – 여러분은 *절대*로 케 이블과 카드를 연결할수 없다!

또한 단 한가닥 선만 카드에 연결된 *것처럼 보이*는 몇몇 멋진 케이블 시스템도 있다. 그러나 실제로는 두가닥 케이블을 납이 따로 따로 나누고 외부 껍질로 싸여져 있는 것이며, 타원형의 교차부분은 납으로 되어있다. 루프의 돌아오는 지점에는, 여러분의 카드에 연결되는 BNC 커넥터가 나누어 진다. 그래서 여러분은 두선의 케이블과 BNC T를 가진 것과 동일하지만, 이 경우에는, 사용자가 티의 한쪽에서 케이블을 제거하는 것이 불가능하며, 네트워크를 방해하 지 않는다.

6.2 꼬임선

꼬임선 네트워크는 실제로 50이상하는 허브가 필요하고, 케이블만의 가격도 thinnet보 다 높다. 여러분은 여러분의 전화선을 사용하며 수많은 불만을 무시할수도 있지만, 그런것은 이제 거의 설치하지 않고 있다.

다른말로 하면, 모든 100Mb/sec 이더넷 계획들은 꼬임선을 사용하고, 기업에 설치되는 새로운 것들은 꼬임선을 사용한다. 또한 Russ Nelson은 '새로 설치하는 것들은 카테고리 5 선을 사용해야 된다. 다른 것들은 여러 분의 설치자들의 시간을 낭비하기만 하는 것이고, 100Base-무엇이든 카테고리 5가 필요할 것이 다.'라고 더했다.

```
Pin 번호                   지정값
----------                 ----------
1                          데이타 출력(+)
2                          데이타 출력(-)
3                          데이타 입력(+)
4                          전화용으로 보존
5                          전화용으로 보존
6                          데이타 입력 (-)
7                          전화용으로 보존
8                          전화용으로 보존
```

만일 여러분이 케이블을 만들고자 한다면, 다음의 내용대로 하면 된다. 서로 다른 신호 쌍들은 반드시 UTP 케이블의 최소 임피던스/손실 요구량을 얻도록 같은 꼬임선 위에 있어야 한다. 여러 분이 위의 표를 보고 있다면, 1+2 와 3+6가 두개의 다른 신호 쌍임을 알수 있다. 1+3 과 2+6 이 아니다!!!!!! 짧은 거리에서, 10MHz는 그러한 에러들을 무시할수 있을 것이다, 만일 그것이 단 지 짧은 거리라면 말이다. 100MHz에서는 생각도 하지 말자.

일반적인 수정 코드에서는, 그 끝이 'A' 와 'B'라면, 꼬임선 한쌍을 각각 입출력으로(전기저항 문제로), 핀과 핀끼리 곧바로 연결시켜 할 것이다. 이것은 1A 가 1B로, 2A 가 2B로 3A 가 3B로 그리고 6A가 6B로 연결된다는 뜻이다. 1A-1B와 2A-2B가 꼬임선이어야 한다. 그리고 3A-3B와 6A-6B가 또하나의 꼬임선이다.

만일 여러분이 허브를 가지고 있지 않고, '널 케이블'을 만들고자 한다면, 극의 바꿈없이, 'A'의 입력은 'B'의 출력이 되고 'A'의 출력은 'B'의 입력이 되게 만들면 된다. 이것은 1A를 3B로(A의 +출력을 B의 +입력으로) 그리고 2A를 6B로(A의 -출력을 B의 -입력으로) 연결하면 된다. 이 두 선은 반드시 꼬임선이어야 한다. 카드/플러그 'A'의 출력은 카드/플러그 'B'의 입력이 되는 것이다. 그리고 나서 3A를 1B에(A의 +입력을 B의 +출력으로) 그리고 또 6A를 2B에(A의 -입력을 B의 -출력으로) 연결하면 된다. 이 두번째것도 꼬임선이어야 한다. 카드/플러그 'A'의 입력은 카드/플러그 'B'의 출력이 되는 것이다.

그래서 만약 여러분이 일반적인 수정 코드를 생각한다면 그 끝을 쪼개서, 새 플러그안에 Rx 와 TX 꼬임선의 자리를 바꾸고, 꼬아내리면, '널' 케이블이 되는 것이다. 복잡하지 않다. 여러분은 한 카드의 Tx 신호를 두번째 그리고 그 다음의 Rx 로 보내려고 하는 것이다.

10BaseT가 표준으로 승인되기전에, 위와 같은 선구조로, RJ-45 커넥터를 사용하는 다른 네트워 크 포맷들이 있었다. SynOptics의 LattisNet과 AT&T의 StarLAN이 그 예다. 몇몇 경우에는, (초기 3C503 카드같은) 카드가 다른 종류의 허브와 통신하기 위해서 점퍼를 설정해줄수 있었다. 그러나 이런 구형 네트워크 형식을 위해 만들어진 대부분의 카드들은 표준 10BaseT 네트워크/허 브들과 함께 제대로 작동하지 않았다. (만일 그 카드들이 AUI 포트도 갖고 있었다면, AUI 와 10BaseT 단자로 연결해 사용하지 못할 이유도 없다.)

6.3 Thick Ethernet

Thick 이더넷은 가장 쓸모없으며, 단지 기존 설비들에 대한 호환을 위해 남겨져 사용되고 있다. 여러분은 3짜리 N-to-BNC 커넥터를 사용해서 thick 과 thin 이더넷을 함께 연결하여 확장시킬수 있다. 이것은 기존의 thicknet을 확장할수 있는 최선의 방법이다. 이경우에 원래 올바른 (하지만 비싼) 해결책은 리피터를 사용하는 것이다.

제 7 절 소프트웨어 설정과 카드 점검

대부분의 경우에, 소프트웨어로 설정을 하고 EEPROM에 저장된다면, 여러분은 보통 DOS로 부팅해서, 판매자가 제공한 DOS 프로그램을 사용해서 카드의 IRQ, I/O, mem_addr등을 정할려 할 것이다. 게다가, 운좋게도 여러분은 그것을 단 한번만 설정하면 된다. 만일 여러분이 카드용 DOS 소프트웨어를 가지고 있지 않다면, 여러분의 카드 제조업체의 WWW 사이트를 찾아보기 바란다. 사이트의 이름을 모른다면, 추측해보라. 즉. 'www.판매자.com', 여기서 '판매자'는 여러분의 카드 제조업체 이름을 말한다. 이것은 SMC, 3Com, 그리고 아주 많은 다른 제조업 체들에도 적용된다.

몇몇 카드들은 리눅스용 설정유틸리티가 있고, 그 목록은 여기에 있다. Donald는 리눅스하에서 돌아가는 조금 작은 카드 점검 프로그램들을 만들었다. 이들은 그가 수많은 드라이버들을 만드는 동안에 디버깅 툴들의 결과로 만들어졌다. 멋진 메뉴방식의 인터페이스를 기대하지는 말라. 그들 대부분은 사용하기 위해서 그 소스 코드를 읽어야만 한다. 비록 여러분의 특정 카드가 제대로 점검이 되지 않을지라도, 여러분은 `cat /proc/net/dev`를 쳐서 몇가지 정보를 얻어낼수 있다. – 적어도 여러분의 카드가 부팅시에 찾아졌다고 가정한다.

어떤 경우에는, 이들 프로그램을 root로 사용해야 하고(포트 입출력을 허용하기 위해서), 아 마도 그렇게 하기 전에 우선은 `ifconfig eth0 down`를 쳐서 이더넷 카드를 닫아야 할 것이다.

7.1 이더넷 카드들을 위한 설정 프로그램들

7.1.1 WD80x3 카드들

wd80x3 카드를 가진 사람들은, 리눅스 ftp 사이트상에서 `wdsetup-06a.tar.gz` 안의 `wdsetup`를 찾을수 있을 것이다. 이 프로그램은 실제로 유지되지 않고 있으며, 오랜동안 갱신되지 않고 있다. 만일 제대로 작동하면 아주좋고, 그렇지 않으면, 여러분의 카드와 함께 가지고 있는 DOS 버전을 사용하라. 여러분이 DOS 버전을 가지고 있지 않으면, SMC의 ftp 사이트에서 SMC 설치/드라이버 디스크를 얻을수 있다. 물론, 이 유틸리티를 사용하기 위해서는 EEPROM 카드를 가지고 있어야만 한다. 오래된, *구형* wd8003 카들들, 그리고 몇몇 wd8013 호환제품들은 카드설정시 점퍼를 사용 하기도 한다.

7.1.2 Digital / DEC 카드들

Digital EtherWorks 3 카드는 DOS 프로그램인 NICSETUP.EXE와 비슷한 방식으로 설정할수 있다. David C. Davies가 만든 EtherWorks 3용 여러 툴들은 드라이버와 합쳐져 있다. 여러분의 지역 리눅스 FTP 사이트의 /pub/linux/system/Network/management 디렉 토리안에 ewrk3tools-X.XX.tar.gz라는 이름의 화일들을 찾을수 있다.

7.1.3 NE2000+ 또는 AT/LANTIC 카드들

몇몇 Nat Semi DP83905 제품들 (AT/LANTIC 와 NE2000+ 같은)은 소프트웨어로 설정할수 있다. (이 카드들도 wd8013 카드를 에뮬레이트할수 있다는 것을 알아두라!) 여러분은 cesdis.gsfc.nasa.gov에 있는 Donald의 ftp 서버에서 이 카드를 설정하기 위해 /pub/linux/setup/atlantic.c 화일을 가져올수 있다. 또한, Kingston DP83905 카드용 설정 프로그램은 모든 카드들에서 쓸수 있을 것 같은데, 그들은 여러분이 그들을 사용하려고 하기 전에는 판매자의 특정 주소를 검사하지 않기 때문이다. 다음의 URL을 따라가서 20XX12.EXE와 INFOSET.EXE 를 가져오라. *Kingston Software* <http://www.kingston.com/download/etherx/etherx.htm>

여러분이 만일 잘못된 설정값을 주면 문제가 발생할수 있으므로, NE2000+카드의 설정은 조심 하라. 전형적인 예로 EEPROM 내의 부트 롬이(마치 ROM이 없는것처럼 VGA 카드 설정 내용과 충돌을 일으킬수 있다. 그러면 여러분이 컴퓨터를 켰을 때 삑소리가 나고 화면에는 아무것도 나타나지 않는다.

여러분은 이것을 다음과 같은 방법으로 복구할 수 있다. 머신에서 카드를 제거하고, 부팅해서 CMOS 설정에 들어간다. ‘Display Adapter’ 항목을 ‘Not Installed’로 바꾸고 기본 부팅 드라이 브를 ‘A:’(여러분의 플로피 드라이브)로 바꾼다. 또 ‘Wait for F1 if any Error’를 ‘Disabled’ 로 바꾼다. 이렇게 하면 컴퓨터는 사용자의 간섭없이 부팅될 것이다. 이제 부팅가능한 DOS 플로 피를 만들고 (‘format a: /s /u’) 위 아카이브 20XX12.EXE에서 default.exe 를 플로피로 복사한다. 그런 후에 echo default > a:autoexec.bat라고 치면 여러분 이 이 플로피로 부팅했을때 자동으로 실행해서 카드를 기본설정으로 바꾸어놓는다. 머신을 끄고, ne2000+ 카드를 다시 설치한후, 새 부트 플로피를 넣고, 파워를 켠다. 아직 여전히 삑소 리가 나겠지만, 결국에는 플로피로 부팅되면서 플로피에 불이 들어오는 것을 볼수 있을 것이다. 플로피가 멈출때까지 한 1-2분을 기다린 후, default.exe 프로그램의 실행이 끝났다고 알 려지면 컴퓨터의 파워를 내린다. 다시 켰을 때, 제대로 돌아가는 화면을 보게되고, 여러분의 CMOS 설정을 돌려놓은뒤, 카드의 EEPROM 설정을 여러분이 원하는 값으로 바꾸면 된다.

만일 여러분이 DOS 플로피를 구할수 없으면, default.exe 프로그램이 자동으로 실행되는 DOS 부팅 디스크 대신에, 위의 모든 작업들이 리눅스 부트 디스크에서 자동 적으로 실행되는 Donald의 atlantic 프로그램(올바른 명령행 전환과 함께)을 구할수 있다.

7.1.4 3Com 카드들

3Com Etherlink III 계열의 카드들(즉, 3c5x9)은 Donald의 다른 설정 유틸리티를 사용해서 설정할수 있다. 여러분은 cesdis.gsfc.nasa.gov에 있는 /pub/linux/setup/3c5x9setup.c 화일을 구해서 이러한 카드들을 설정할수 있다. (DOS 3c5x9B 설정 유틸리티는 Etherlink III 계열의 새로운 “B” 시리즈에 관련된 더 많은 옵션들이 있다는 것을 알아두라.)

7.2 이더넷 카드들을 위한 점검 프로그램들

Donald가 작성한 모든 점검 프로그램들은 이 URL에서 구할수 있다.

Ethercard Diagnostics <ftp://cesdis.gsfc.nasa.gov/pub/linux/diag/index.html>

Allied Telesis AT1700 – cesdis.gsfc.nasa.gov상의 /pub/linux/diag/at1700.c를 찾아보라.

Cabletron E21XX – cesdis.gsfc.nasa.gov상의 /pub/linux/diag/e21.c를 찾아보라.

HP PCLAN+ – cesdis.gsfc.nasa.gov상의 /pub/linux/diag/hp+.c를 찾아보라.

Intel EtherExpress – cesdis.gsfc.nasa.gov상의 /pub/linux/diag/eexpress.c를 찾아보라.

NE2000 카드들 – cesdis.gsfc.nasa.gov상의 /pub/linux/diag/ne2k.c를 찾아보라. 거기에는 또한 이제는 일반적인 NE2000-PCI 호환제품들을 위한 PCI 버전도 있다.

RealTek (ATP) Pocket adaptor – cesdis.gsfc.nasa.gov상의 /pub/linux/diag/atp-diag.c를 찾아보라.

다른 모든 카드들 – cat /proc/net/dev 와 dmesg 라고 치고 질문에서 카드에 대해 커널이 가진 유용한 정보가 무엇인지 보라.

제 8 절 기술적 정보

이 문서는 카드가 어떻게 작동하는가를 조금이라도 더 이해하길 원하거나, 현재 드라이버들을 가지고 놀거나 현재 지원되지 않는 카드의 고유한 드라이버를 만들려고 하는이들에게 매우 유용 할 것이다. 만일 이 카테고리에 빠져들기 싫다면 이 장을 넘어가도 된다.

8.1 Programmed I/O vs. 공유 메모리 vs. DMA

만일 여러분이 이미 백투백 패킷들을 보내고 받을수 있다면, 선을 통해 더 많은 것을 넣을수는 없다. 모든 현대적인 이더넷 카드는 백투백 패킷들을 받을수 있다. 리눅스의 DP8390 드라이버들 (wd80x3, SMC-Ultra, 3c503, ne2000, 등등)은 백투백 패킷들을 매우 잘 보내고(현재의 인터럽트 대기시간에 의존한다) 3c509와 AT1500 하드웨어는 자동으로 백투백 패킷들을 보내는데 전혀 문 제가 없다.

ISA 버스는 5.3MB/sec (42Mb/sec)의 속도를 가질수 있으며, 10Mbps 이더넷에는 충분해 보인다. 100Mbps 카드들의 경우에는 네트워크 대역폭의 이점을 얻기 위해서는 더 빠른 버스가 필요한 것이 당연하다.

8.1.1 Programmed I/O (e.g. NE2000, 3c509)

Pro: 어떠한 강제적인 시스템 자원의 사용도 없으며, 아주 조금의 I/O 레지스터를 사용하고, 16M 제한을 가지지 않는다.

Con: 보통 전송률이 가장 느리고, CPU는 항상 기다려야하며, 끼어든 패킷에 대한 접근은 보통 힘들거나 불가능하다.

8.1.2 공유 메모리 (e.g. WD80x3, SMC-Ultra, 3c503)

Pro: 간단하고, programmed I/O 보다 빠르며, 패킷들에 대한 무작위 접근이 허용된다. 어디에서 사용되냐면, 리눅스 드라이버들이 들어온 IP 패킷들이 카드에서 복사되므로 그들의 체크섬을 계 산하고, 그로인해 동급의 PIO 카드에 비해 CPU 점유율이 적어진다.

Con: 메모리 공간을 사용하며 (DOS 사용자들에게는 매우 큰거지만, 리눅스상에서는 특별히 문제 되지 않는다), 그리고 여전히 CPU를 잡아둔다.

8.1.3 Slave (normal) Direct Memory Access (e.g. 리눅스에는 없다!)

Pro: 실제 데이타 전송시에는 CPU 를 놓아준다.

Con: 경계선 상태를 확인하고, 인접한 버퍼들을 재배치하며, DMA 리지스터들을 프로그래밍하여 모든 기술들중에 가장 느리다. 또한 부족한 DMA 채널을 사용하며, 저수준 메모리 버퍼의 정렬을 필요로 한다.

8.1.4 Bus Master Direct Memory Access (e.g. LANCE, DEC 21040)

Pro: 데이타 전송중에는 CPU가 자유로우며, 버퍼들과 함께 배열할수 있으며, ISA 버스상에서 잃어버리는 아주 조금이나 아니면 CPU시간을 전혀 필요로하지 않기도 한다. 대부분의 버스 마스 터링 리눅스 드라이버들은 현재 '카피브레이크' 구조를 사용하는데 이것은 카드가 큰 패킷들을 커널 네트워킹 버퍼에 직접 넣는 장소이고, 작은 패킷들은 CPU가 복사함으로서 이후로 진행되는 캐시를 준비하게 한다.

Con: (ISA 버스 카드들에만 해당된다) 카드에는 저수준 메모리 버퍼와 DMA채널이 필요하다. Any bus-master will have problems with other bus-masters that are bus-hogs, such as some primitive SCSI adaptors. A few badly-designed motherboard chipsets have problems with bus-masters. And a reason for not using *any* type of DMA device is using a 486 processor designed for plug-in replacement of a 386: these processors must flush their cache with each DMA cycle. (This includes the Cx486DLC, Ti486DLC, Cx486SLC, Ti486SLC, etc.)

8.2 Writing a Driver

The only thing that one needs to use an ethernet card with Linux is the appropriate driver. For this, it is essential that the manufacturer will release the technical programming information to the general public without you (or anyone) having to sign your life away. A good guide for the likelihood of getting documentation (or, if you aren't writing code, the likelihood that someone else will write that driver you really, really need) is the availability of the Crynwr (nee Clarkson) packet driver. Russ Nelson runs this operation, and has been very helpful in supporting the development of drivers for Linux. *Net-surfers* can try this URL to look up Russ' software.

Russ Nelson's Packet Drivers `<http://www.crynwr.com/crynwr/home.html>`

Given the documentation, you can write a driver for your card and use it for Linux (at least in theory). Keep in mind that some old hardware that was designed for XT type machines will not function very well in a multitasking environment such as Linux. Use of these will lead to major problems if your network sees a reasonable amount of traffic.

Most cards come with drivers for MS-DOS interfaces such as NDIS and ODI, but these are useless for Linux. Many people have suggested directly linking them in or automatic translation, but this is nearly impossible. The MS-DOS drivers expect to be in 16 bit mode and hook into 'software interrupts', both incompatible with the Linux kernel. This incompatibility is actually a feature, as some Linux drivers are considerably better than their MS-DOS counterparts. The '8390' series drivers, for instance, use ping-pong transmit buffers, which are only now being introduced in the MS-DOS world.

(Ping-pong Tx buffers means using at least 2 max-size packet buffers for Tx packets. One is loaded while the card is transmitting the other. The second is then sent as soon as the first finished, and so on. In this way, most cards are able to continuously send back-to-back packets onto the wire.)

OK. So you have decided that you want to write a driver for the Foobar Ethernet card, as you have the programming information, and it hasn't been done yet. (...these are the two main requirements ;-) You should start with the skeleton network driver that is provided with the Linux kernel source tree. It can be found in the file /usr/src/linux/drivers/net/skeleton.c in all recent kernels. Also have a look at the Kernel Hackers Guide, at the following URL: *KHG* `<http://www.redhat.com:8080/HyperNews/get/khg.html>`

8.3 Driver interface to the kernel

Here are some notes on the functions that you would have to write if creating a new driver. Reading this in conjunction with the above skeleton driver may help clear things up.

8.3.1 Probe

Called at boot to check for existence of card. Best if it can check un-obtrsively by reading from memory, etc. Can also read from I/O ports. Initial writing to I/O ports in a probe is *not good* as it may kill another device. Some device initialization is usually done here (allocating I/O space, IRQs,filling in the dev->??? fields etc.) You need to know what io ports/mem the card can be configured to, how to enable shared memory (if used) and how to select/enable interrupt generation, etc.

8.3.2 Interrupt handler

Called by the kernel when the card posts an interrupt. This has the job of determining why the card posted an interrupt, and acting accordingly. Usual interrupt conditions are data to be rec'd, transmit completed, error conditions being reported. You need to know any relevant interrupt status bits so that you can act accordingly.

8.3.3 Transmit function

Linked to dev->hard_start_xmit() and is called by the kernel when there is some data that the kernel wants to put out over the device. This puts the data onto the card and triggers the transmit. You need to know how to bundle the data and how to get it onto the card (shared memory copy, PIO transfer, DMA?) and in the right place on the card. Then you need to know how to tell the card to send the data down the wire, and (possibly) post an interrupt when done. When the hardware can't accept additional packets it should set the dev->tbusy flag. When additional room is available, usually during a transmit-complete interrupt, dev->tbusy should be cleared and the higher levels informed with `mark_bh(INET_BH)`.

8.3.4 Receive function

Called by the kernel interrupt handler when the card reports that there is data on the card. It pulls the data off the card, packages it into a sk_buff and lets the kernel know the data is there for it by doing a netif_rx(sk_buff). You need to know how to enable interrupt generation upon Rx of data, how to check any relevant Rx status bits, and how to get that data off the card (again sh mem, PIO, DMA, etc.)

8.3.5 Open function

linked to dev->open and called by the networking layers when somebody does `ifconfig eth0 up` - this puts the device on line and enables it for Rx/Tx of data. Any special initialization incantations that were not done in the probe sequence (enabling IRQ generation, etc.) would go in here.

8.3.6 Close function (optional)

This puts the card in a sane state when someone does `ifconfig eth0 down`. It should free the IRQs and DMA channels if the hardware permits, and turn off anything that will save power (like the transceiver).

8.3.7 Miscellaneous functions

Things like a reset function, so that if things go south, the driver can try resetting the card as a last ditch effort. Usually done when a Tx times out or similar. Also a function to read the statistics registers of the card if so equipped.

8.4 Technical information from 3Com

If you are interested in working on drivers for 3Com cards, you can get technical documentation from 3Com. Cameron has been kind enough to tell us how to go about it below:

3Com's Ethernet Adapters are documented for driver writers in our 'Technical References' (TRs). These manuals describe the programmer interfaces to the boards but they don't talk about the diagnostics, installation programs, etc that end users can see.

The Network Adapter Division marketing department has the TRs to give away. To keep this program efficient, we centralized it in a thing called 'CardFacts.' CardFacts is an automated phone system. You call it with a touch-tone phone and it faxes you stuff. To get a TR, call CardFacts at 408-727-7021. Ask it for Developer's Order Form, document number 9070. Have your fax number ready when you call. Fill out the order form and fax it to 408-764-5004. Manuals are shipped by Federal Express 2nd Day Service.

There are people here who think we are too free with the manuals, and they are looking for evidence that the system is too expensive, or takes too much time and effort. So far, 3Com customers have been really good about this, and there's no problem with the level of requests we've been getting. We need your continued cooperation and restraint to keep it that way.

8.5 Notes on AMD PCnet / LANCE Based cards

The AMD LANCE (Local Area Network Controller for Ethernet) was the original offering, and has since been replaced by the 'PCnet-ISA' chip, otherwise known as the 79C960. Note that the name 'LANCE' has stuck, and some people will refer to the new chip by the old name. Dave Roberts of the Network Products Division of AMD was kind enough to contribute the following information regarding this chip:

'Functionally, it is equivalent to a NE1500. The register set is identical to the old LANCE with the 1500/2100 architecture additions. Older 1500/2100 drivers will work on the PCnet-ISA. The NE1500 and NE2100 architecture is basically the same. Initially Novell called it the 2100, but then tried to distinguish between coax and 10BASE-T cards. Anything that was 10BASE-T only was to be numbered in the 1500 range. That's the only difference.

Many companies offer PCnet-ISA based products, including HP, Racal-Datacom, Allied Telesis, Boca Research, Kingston Technology, etc. The cards are basically the same except that some manufacturers have added 'jumperless' features that allow the card to be configured in software. Most have not. AMD offers a standard design package for a card that uses the PCnet-ISA and many manufacturers use our design without change. What this means is that anybody who wants to write drivers for most PCnet-ISA based cards can just get the data-sheet from AMD. Call our literature distribution center at (800)222-9323 and ask for the Am79C960, PCnet-ISA data sheet. It's free.

A quick way to understand whether the card is a 'stock' card is to just look at it. If it's stock, it should just have one large chip on it, a crystal, a small IEEE address PROM, possibly a socket for a boot ROM, and a connector (1, 2, or 3, depending on the media options offered). Note that if it's a coax card, it will have some transceiver stuff built onto it as well, but that should be near the connector and away from the PCnet-ISA.'

A note to would-be card hackers is that different LANCE implementations do 'restart' in different ways. Some pick up where they left off in the ring, and others start right from the beginning of the ring, as if just initialised.

8.6 Multicast and Promiscuous Mode

Another one of the things Donald has worked on is implementing multicast and promiscuous mode hooks. All of the *released* (i.e. **not** ALPHA) ISA drivers now support promiscuous mode.

Donald writes: 'I'll start by discussing promiscuous mode, which is conceptually easy to implement. For most hardware you only have to set a register bit, and from then on you get every packet on the wire. Well, it's almost that easy; for some hardware you have to shut the board (potentially dropping a few packet), reconfigure it, and then re-enable the ethercard. OK, so that's easy, so I'll move on something that's not quite so obvious: Multicast. It can be done two ways:

1. Use promiscuous mode, and a packet filter like the Berkeley packet filter (BPF). The BPF is a pattern matching stack language, where you write a program that picks out the addresses you are interested in. Its advantage is that it's very general and programmable. Its disadvantage is that there is no general way for the kernel to avoid turning on promiscuous mode and running every packet on the wire through every registered packet filter. See 8.7 (The Berkeley Packet Filter) for more info.

2. Using the built-in multicast filter that most etherchips have.

I guess I should list what a few ethercards/chips provide:

```
Chip/card  Promiscuous  Multicast filter
----------------------------------------
Seeq8001/3c501  Yes     Binary filter (1)
3Com/3c509      Yes     Binary filter (1)
8390            Yes     Autodin II six bit hash (2) (3)
LANCE           Yes     Autodin II six bit hash (2) (3)
i82586          Yes     Hidden Autodin II six bit hash (2) (4)
```

1. These cards claim to have a filter, but it's a simple yes/no 'accept all multicast packets', or 'accept no multicast packets'.

2. AUTODIN II is the standard ethernet CRC (checksum) polynomial. In this scheme multicast addresses are hashed and looked up in a hash table. If the corresponding bit is enabled, this packet is accepted. Ethernet packets are laid out so that the hardware to do this is trivial – you just latch six (usually) bits from the CRC circuit (needed anyway for error checking) after the first six octets (the destination address), and use them as an index into the hash table (six bits – a 64-bit table).

3. These chips use the six bit hash, and must have the table computed and loaded by the host. This means the kernel must include the CRC code.

4. The 82586 uses the six bit hash internally, but it computes the hash table itself from a list of multicast addresses to accept.

Note that none of these chips do perfect filtering, and we still need a middle-level module to do the final filtering. Also note that in every case we must keep a complete list of accepted multicast addresses to recompute the hash table when it changes.

8.7 The Berkeley Packet Filter (BPF)

The general idea of the developers is that the BPF functionality should not be provided by the kernel, but should be in a (hopefully little-used) compatibility library.

For those not in the know: BPF (the Berkeley Packet Filter) is an mechanism for specifying to the kernel networking layers what packets you are interested in. It's implemented as a specialized stack language interpreter built into a low level of the networking code. An application passes a program written in this language to the kernel, and the kernel runs the program on each incoming packet. If the kernel has multiple BPF applications, each program is run on each packet.

The problem is that it's difficult to deduce what kind of packets the application is really interested in from the packet filter program, so the general solution is to always run the filter. Imagine a program that registers a BPF program to pick up a low data-rate stream sent to a multicast address. Most ethernet cards have a hardware multicast address filter implemented as a 64 entry hash table that ignores most unwanted multicast packets, so the capability exists to make this a very inexpensive operation. But with the BPF the kernel must switch the interface to promiscuous mode, receive _all_ packets, and run them through this filter. This is work, BTW, that's very difficult to account back to the process requesting the packets.

제 9 절 랩탑/노트북 컴퓨터와의 네트워킹

여러분의 랩탑을 네트워크 상에 올리는 방법에는 여러가지가 있다. 여러분은 SLIP 코드(그리고 시리얼 라인 속도로 움직이는)를 사용할수도 있다; PCMCIA 슬롯이 내장된 노트북을 가질수도 있다; 그리고 ISA 이더넷 카드가 꼽혀있는 도킹 스테이션을 가진 랩탑을 가질수도 있다; 아니면 병렬 포트 이더넷 아답터를 사용할수도 있다.

9.1 SLIP 사용하기

이것은 가장 저렴한 해결책이다, 하지만 가장 어렵다. 또한, 매우 높은 전송률을 얻을수는 없을 것이다. SLIP은 정말로 이더넷 카드와 관련된 것이 아니기 때문에 여기서는 더이상 논하 지 않을 것이다. NET-2 하우투를 보기 바란다.

9.2 PCMCIA 지원

여러분이 가진 하드웨어가 무엇인지 정확하게 알아내보고 (예를 들어 카드 제조사, PCMCIA 칩 콘트롤러 제조업체) 랩탑 사용자들에게 물어보라. 말할 필요도 없겠지만, 모든것이 간단할 것이라고는 기대하지 말라. 사소한 것들이나, 커널 패치 등에 관심을 가져라. 어느날엔가는 여러분이 'make config'라고 쳐넣어야 할 것이다 8-)

현재, 데이타북 TCIC/2와 인텔 i82365 두개의 PCMCIA 칩셋이 지원되고 있다.

tsx-22.mit.edu안의 /pub/linux/packages/laptops/에는 수많은 프로그램들이 있으므로 그 곳에서 여러분은 쓸만한 것을 찾을수 있을 것이다. 여기에는 PCMCIA 이더넷 카드 드라이버들 부터 PCMCIA 콘트롤러 칩과 통신하는 프로그램까지 다룬다. 이들 드라이버에 대해 알아둘 것은 이들은 보통 특정 PCMCIA 칩(예를 들어 인텔 82365나 TCIC/2)과 관련된 것이라는 것이다.

NE2000 호환 카드들에 있어서는, 몇몇 사람들이 DOS상에서 카드를 설정하고, DOS 커맨드 프롬프트에서 `loadlin`를 통해 리눅스로 부팅해서 성공했다.

PCMCIA 지원을 원하는 리눅스 사용자들이 찾는 것은 실제적으로 그 과정이 진행되고 있다. David Hinds는 이러한 노력을 하는 개척자이다. 그의 최신 PCMCIA 지원 패키지는 다음장소에서 구할 수 있다.

PCMCIA Package `<ftp://cb-iris.stanford.edu/pub/pcmcia>`

`pcmcia-cs-X.Y.Z.tgz` 같은 화일들을 찾아라. 여기서 X.Y.Z는 최신의 버전 숫자이다. 이 또한 `tsx-11.mit.edu`에 매우 잘 업로드되고 있다.

유저 레벨에서 진행되는 Donald의 PCMCIA 이너블러가 작동하는 것과 David Hinds의 커널 레벨 해법을 잘 알아두라. 아마 더 많이 사용되고 지속적인 개발하에 있는 David의 패키지를 가장 좋다고 여길 것이다.

9.3 도킹 스테이션의 ISA 이더넷 카드

보통 250 정도하는 랩탑용 도킹 스테이션은 두개의 풀 사이즈 ISA 슬롯과, 두개의 시리 얼, 그리고 하나의 패러렐 포트를 제공한다. 대부분의 도킹 스테이션은 랩탑 배터리의 전원을 소모시키며, 여러분이 짧은 ISA 카드를 사용한다면 몇몇 도킹스테이션은 별도의 배터리를 추가 할수 있을 것이다. 여러분은 비싸지 않은 이더넷 카드를 추가해서 풀 스피드의 이더넷 성능을 즐길수 있다.

9.4 포켓/패러렐 포트 아답터들.

'포켓' 이더넷 아답터들도 여러분의 필요에 맞을수 있을 것이다. 단, 알아둘 점은 전송 속도가 패러렐 포트 인터페이스의 한계치(아마 200kB/s가 최고지?)이내로 밖에는 나오지 않 는다는 것이다.

또한 가장 여러분을 구속하는 것은 전원 공급장치이다. 여러분은 가끔 랩탑의 키보드로부터 전원을 끌어들이는 케이블을 사거나 만들어서 아답터가 달린 벽돌로부터 피할수 있을것이다. (다음을 보라 5.31.1 (keyboard power))

두개의 지원되는 포켓 아답터에 관해서는 5.14.5 (DE-600 / DE-620) 와 5.31.1 (RealTek) 를 보라

제 10 절 Miscellaneous.

관련된 자료들이지만 다른 어느곳에도 맞지 않는 자료들은 여기에 싣는다. 이것이 적절하지 않을지도 모르고, 일반적인 관심거리가 아닐지도 모르지만, 어쨌든 여기에 넣어둔다.

10.1 커널에 이더넷 인수 넘겨주기

부팅시에 커널에 넘겨줄수 있는 두가지 일반적인 커널 명령어가 있다.(`ether` 와 `reserve`). 이것은 LILO 나 loadlin, 또는 선택적 인수들을 받아들일수 있는 어떤 다른 부팅 유틸리티에서 도 사용할수 있다.

예를 들어, 명령어가 'blah' 이고 3개의 인수를 요구한다면(123, 456, 그리고 789라면), LILO에서는 다음과 같이 사용한다.

`LILO: linux blah=123,456,789`

부팅시 인수들에 대한 더 많은 정보는 (그리고 완전한 리스트), 다음의 장소를 보라. *BootPrompt-HOWTO* `<http://metalab.unc.edu/mdw/HOWTO/BootPrompt-HOWTO.html>`

10.1.1 `ether` 명령어

`ether=` 인수는 커널에 직접 만들어져 적재된 드라이버의 경우에 사용된다. `ether=` 인수는 모듈 드라이버에는 *절대 아무 효과가 없다*. 대부분의 일반적인 형식은 다음과 같다.

```
ether=IRQ,BASE_ADDR,PARAM_1,PARAM_2,NAME
```

모든 인수들은 선택이다. 첫번째 숫자아닌 인수가 이름으로 받아들여진다.

IRQ: 말그대로다. IRQ 값 '0' (보통 기본값)은 자동IRQ를 의미한다. IRQ 설정이 base_addr보다 먼저 나오는 것은 역사적 사건이다 – 이것은 다른것이 바뀔때면 어느때든지 고쳐질 것이다.

BASE_ADDR: 이것도 말그대로다. '0'값 (보통 기본값)은 이더넷 카드의 카드별 특정 주소값을 검색하는 것이다.

PARAM_1: 이것은 원래 WD80*3과 같은 공유메모리 이더넷 카드에서 메모리 시작을 재정의 하기 위해 사용되었다. 몇몇 드라이버들은 이 값의 낮은 4비트를 디버그 메세지 레벨 설정하는데 사용 한다. 0 – default, 1-7 – level 1..7, (7은 최다 메세지) 8 – level 0 (메세지 없음). 또한, LANCE 드라이버는 이 값의 낮은 4비트를 DMA 채널 선택에 사용한다. 그렇지 않으면 auto-DMA이다.

PARAM_2: 3c503 드라이버는 이것으로 내장과 외장 트랜시버를 선택한다. 0 – default/internal, 1 – AUI external. Cabletron E21XX 카드도 PARAM_2의 낮은 4비트를 출력 장치 선택에 사용한다. 그렇지 않으면 자동으로 찾는다.

NAME: 네트워크 디바이스를 선택할 때 이 값으로 참조를 한다. 표준 커널은 버스에 꽂혀있는 이더넷 카드에게는 'eth0', 'eth1', 'eth2' 그리고 'eth3'라는 이름을 사용하고, 패러렐 포트 '포켓' 이더넷 아답터에게는 'atp0'를 사용한다. arcnet 드라이버는 'arc0'라는 이름을 사용한다. 기본 설정은 검색해내는 단일 이더넷 카드에 'eth0'라는 이름을 붙인다. 여러개의 카드들은 LILO 인자들을 사용해서 그들의 주소값을 설정해야만 사용할수 있다. 1.0 커널은 LANCE-기반의 이더넷 카드들을 특별한 경우로 다룬다. LILO 인수가 무시되고, LANCE 카드들은 항상 'eth0'로 시작하는 'eth<n>' 이름으로 연결된다. 추가적으로 LANCE가 아닌 이더넷 카드들은 'eth<n+1>'로 정해지고, 보통 'eth0'는 'ether=0,-1,eth0' 와 같이 검색할수 없게 된다. (그렇다, 이건 버그다.)

10.1.2 `reserve` 명령어

이 두번째 lilo 명령어도 위의 'ether='처럼 사용된다. 예를 들어 lilo.conf 안에 부트시 선택하는 특정한 이름을 추가한다.

```
reserve=IO-base,extent{,IO-base,extent...}
```

몇몇 머신에서 이것은 특정 영역내에서 장치검사로부터(자동 검색) 장치 드라이버를 보존하기 위해서 필요하다. 이것은 엉성하게 만들어진 하드웨어, 즉 잘못 인식되거나, 이전의 검색에서 설정이 바뀐 하드웨어, 또는 커널이 초기화 시키길 원하지 않는 하드웨어가 부팅중 머신을 *얼게* 만들기 때문이다 (몇몇 이더넷 카드들 처럼).

보존하는 부팅시 인수 주소들은 특정한 I/O 포트 영역이 검색되지 않는데서 생기는 문제때문이다. 그 영역은 마치 장치가 이미 그 주소에서 찾아진것처럼, 커널의 포트 등록 테이블안에 보존되어진다. 이 메카니즘은 모든 머신들에게 필요한 것이 아니란 것을 알아두기 바란다. 단지 문제가 있거나 특별한 경우에만 이것을 사용할 필요가 있다.

특정 영역내의 I/O 포트들은 장치 검색으로부터 보호된다. 이 때문에 몇몇 드라이버들은 NE2000으로 인해 시스템을 정지시키거나, 다른 몇몇 장치들을 본래의 것과 다르게 인식하는 일이 생긴다. 그러므로 만일 다른 특정한 부트 인수들이 주어지지 않으면, 정확한 드라이버가 보존된 영역내의 장치를 검색할수 없다. 그래서 `reserve`가 몇몇 다른 부트 인수 와 함께 가장 자주 사용된다. 특정 장치를 보호하기 위해 **보존**하는 영역을 정하 면 여러분은 그 장치의 정확한 검색을 정해주어야만 한다. 대부분의 드라이버들은 정확한 주소가 주어지면 포트 등록 테이블을 무시하게 된다.

예를 들어, 다음과 같은 부트 라인은

```
LILO: linux reserve=0x300,32 ether=0,0x300,eth0
```

이더넷 카드를 제외하고 다른 모든 장치 드라이버들이 0x300-0x31f를 검색하지 못하게 한다.

보통 부팅시 명령은 인자를 11개 까지밖에 줄수 없기 때문에, 여러분은 `reserve` 키워드당 5개만을 정할수 있다. 여러개의 `reserve` 명령은 특별히 복잡한 요청이 있을을 경우에 사용하게 된다.

10.2 이더넷 드라이버들을 모듈로 사용하기

대부분의 리눅스 배포판에 적재되는 커널은 아주 소수의 드라아버만을 그안에 담고 있다. 그대신 독립적이고 동적으로 적재가능한 모듈로써 드라이버들을 제공하고 있다. 이 모듈 드라이 버들은 관리자가 보통 `modprobe(8)` 명령을 사용해서 적재하거나, 또는 어떤 경우 에는 `modprobe`를 호출하는 'kerneld' (2.0에서) 이나 'kmod'를 통해 자동으로 적 재된다.

여러분의 특정한 배포판은 이더넷 설정을 위해 아주 멋진 그래픽 설정 툴들을 제공할지도 모른다. 가능하면 먼저 그들을 사용하고 시험해봐라. 이 아래에서는 이 설정 프로그램들이 무엇 에 근거하며, 무엇을 바꾸는지에 대해 설명한다.

어떠한 모듈들이 사용되고, 각 모듈에는 어떤 옵션이 지원되는지에 대한 제어 정보들은 보통 `/etc/conf.modules` 화일에 저장되어 있다. 이 화일에서 관심분야(이더넷 카드들에 대한)의 두 주요 옵션은 `alias`와 `options`이다. `modprobe` 명령은 모듈정보에 관해서 이 화일을 참고한다.

실제 모듈 자신은 보통 `/lib/modules/'uname -r'/net` 디렉토리에 들어있다. 여기서 `uname -r` 명령은 커널 버전을 돌려준다 (예. 2.0.34). 여러분은 그곳에서 여러분의 카드에 맞는 모듈을 찾을수 있을 것이다.

여러분이 여러분의 `conf.modules` 화일에 가장 먼저 할 일은 `eth0` 네트워크 인터페이스를 사용하기 위해 어떤 드라이버가 사용될지 `modprobe`에게 말해주는 것이다. 여기서 `alias` 명령을 사용한다. 예를 들어, 여러분이 `smc-ultra.o` 드라이버 모듈을 사용하는 ISA SMC EtherEZ 카드를 가지고 있다면, 이 드라이버를 `eth0`에 `alias`하기 위해서는 다음과 같은 줄을 추가 하면 된다.

```
alias eth0 smc-ultra
```

여러분에게 필요한 다른 하나는 특정 모듈 (또는 모듈 별명:module alias)에 어떠한 옵션이 사용되었는지를 알려주는 `options` 줄이다. 위의 예제를 이어보면, 여러분이 `options` 행 없이 단지 `alias` 행만을 사용할 경우, 커널은 ISA 카드 를 자동 검색으로 찾는 것은 좋은 생각이 아니라며 여러분에게 경고할 것이다 (`dmesg`를 보라). 이 경고를 없애려면, 모듈에게 카드가 무슨 I/O 주소를 가지고 있는지 말해주는 행을 추가해줘야 한다. 예제의 경우 16진수 주소 `0x280`라고 적고 있다.

```
options smc-ultra io=0x280
```

대부분의 ISA 모듈들은 `insmod` 명령행 중에 `io=0x340` 와 `irq=12` 같은 인수를 받아들인다. 여러분이 카드의 탐색을 피하기 위해 이러한 인자들을 주는 것은 *필요하고* 아니면 적어도 *강하게 추천한다.* PCI 와 EISA 장치들과는 다르게 대부분의 ISA 장치들을 자동 검색할 현실적으로 안전한 방법이 없다. 그리고 드라이버를 모듈로 사용할 때는 반드시 피해야만 한다.

각 모듈들이 받아들이는 모든 옵션들의 목록은 다음 파일에서 찾아볼수 있다.

`/usr/src/linux/Documentation/networking/net-modules.txt`

여러분의 특정 카드에서 사용할수 있는 옵션이 무엇인지 찾아서 읽어보길 권한다. 모든 8390 기반의 드라이버들과 PLIP 드라이버처럼 하나의 모듈로 여러개의 장치들을 제어할수 있는 모듈을 위해 몇몇 모듈들은 콤마로 구분된 여러개의 값들을 지원한다. 예를 들어,

```
options 3c503 io=0x280,0x300,0x330,0x350 xcvr=0,1,0,1
```

위에서는 card 2와 4는 외부 단자를 사용해서 하나의 모듈이 4개의 3c503 카드를 제어하고 있다. '='이나 콤마 주변에 공백이 있어서는 안된다.

또 알아둘 것은 *사용중인* 모듈은 제거할수 없다는 것이다. 이것은 여러분이 모듈을 제거하기 전에 `ifconfig eth0 down` (이더넷 카드 죽이기)를 사용해야 한다는 것이 다.

`lsmod` 명령은 무슨 모듈이 적재되었고, 어느것이 사용중인지 알려주며, `rmmod`는 그들을 제거한다.

10.3 관련 문서

이 정보의 많은 부분은 가치있는 정보자원들이 있는 comp.os.linux 뉴스그룹의 게시물을 저장한 것이다. 다른 유용한 정보들은 Donald 자신이 만든 작은 화일들에서 온 것이다. 물론 만약 여러분이 이더넷 카드를 설치하고나면, 여러분이 사용할 소프트웨어를 실제적으로 설정할 수 있는 NET-2 하우투를 읽어보고자 할 것이다. 또한, 자신에게 약간이라도 해커다운 면이 있다 면, 여러분은 언제나 드라이버 소스 화일들로부터 추가적인 정보들을 얻을수 있을 것이다. 그 안에는 실제 코드가 시작되기 전에 보통 어떤 중요한 점들에 대해 설명한것이 있다.

그들에게 리눅스 세계에서 정보를 찾는 것들은 별로 특별한 것도 아니다.(예를 들어, 10BaseT가 무엇이고, AUI가 무엇이며, 허브가 하는 이은 무엇인지 등등.) 나는 *comp.dcom.lans.ethernet*과/또는 *comp.sys.ibm.pc.hardware.networking* 뉴스그룹을 이용하길 권장한다. `dejanews.com`에 있는 그러한 뉴스그룹 아카이브들 은 쓸모없는 정보자원일수 있다. 여러분은 RTFM(모든 뉴스그룹 FAQ들을 가지고 있는)로부터 뉴스그룹 FAQ를 가져올수 있다. 그 URL은 다음과 같다.

Usenet FAQs `<ftp://rtfm.mit.edu/pub/usenet-by-hierarchy/>`

여러분은 또한 다음의 URL에서 흔히 'Ethernet-HomePage'라고 말하는 곳을 볼수 있다.

Ethernet-HomePage `<http://wwwhost.ots.utexas.edu/ethernet/ethernet-home.html>`

10.4 Disclaimer and Copyright

이 문서는 복음성가가 *아니다*. 그러나 여러분이 찾을수 있는 가장 시대에 맞는 정보일 것이다. 여러분 자신을 빼고는 여러분의 하드웨어에 무슨 일이 일어나든지 아무도 책임을 지지 않는다. 만일 여러분의 이더넷카드나 다른 어떤 하드웨어에서 연기가 난다고 해도(...거의 불 가능하다!) 우리는 아무런 책임을 지지 않는다. 즉, 이 문서에 포함된 정보로 인해 일어난 어떠 한 손실에 대해서도 저자는 아무런 책임을 지지 않는다.

번역을 하는 사람들이 고려할 것이다. 첫째, 다른 출력 포맷을 생성할수 있으므로 SGML 소스 (하우투 메인 사이트에서 FTP로 사용할수 있다)를 번역하라. 여러분이 번역한 오리지날 영문 SGML 소스의 사본을 간직하라! 갱신된 하우투가 나오면, 그 버전의 새 SGML 소스를 가져와서, 간단히 `diff -u old.sgml new.sgml`라고 치면 무엇이 바뀌었는지 정확하게 보여주고 여러분은 SGML 소스를 전부 다시읽고 재번역할 필요없이 쉽게 바꾸어서 합칠수 있다.

만일 여러분이 출판된 문서에 반영하길 원하면, 연락해서 (e-mail을 통해서) 가능한 가장 최신 의 정보들을 있을 것이다. 과거에는 리눅스 하우투 문서의 철지난 버전들이 출판되어서, 이미 갱신된 버전에는 답해져 있는 질문들로 개발자들을 시달리게 한 적이 있다.

10.5 끝맺음

여러분이 만일 이 글에서 오타나, 아니면 철지난 정보들을 찾아내게 되면, e-mail을 보내 주기 바란다. 이것은 매우 크고, 전체적으로 보기는 쉽다. 만약 여러분이 수정한 것을 보냈는 데, 다음 버전에 포함되지 않았다면, 주저말고 다시 보내길 바란다. 그건 아마도 내가 받은 스팸과 쓸모없는 메일의 바다에서 잃어버렸을 것이기 때문이다.

Thanks!

Paul Gortmaker, `p_gortmaker@yahoo.com`