리눅스 SMP 하우투

David Mentre Mentre

도운이

양유성

yooseong (at) kldp.org

임은재

eunjea (at) kldp.org

v1.9, 13 January 2000

이 하우투 문서는 리눅스에서 SMP 사용에 대한 문제, 해결책들에 관해 써졌습니다.

1. 소개

리눅스는 SMP (Symmetric Multi-Processors) 머쉰에서 작동합니다. SMP 지원은 커널 버젼 2.0에서 시작되어 꾸준히 개선되고 있습니다. 2.2.x 대의 커널에서는 보다 안정되고 빠른 속도를 가져 올수 있게 되었습니다.

본 문서는 David Mentr (David.Mentre@irisa.fr (mailto:David.Mentre@irisa.fr))에 의해 관리되고 있으며, 다음 주소에서 최근 버젼을 구할수 있습니다.

- http://www.irisa.fr/prive/mentre/smp-howto/ (http://www.irisa.fr/prive/mentre/smp-howto/) (프랑스)
- http://www.phy.duke.edu/brahma/smp-faq/ (http://www.phy.duke.edu/brahma/smp-faq/) (미국)

만약 당신이 이 문서에 대한 조언, 수정 사항등이 있다면 SGML 버젼 (http://www.irisa.fr/prive/mentre/smp-howto/smp-howto.sgml) 에 대한 diff 형식으로 보내 주십시요. 그러나 텍스트 형태의 문서도 환영입니다. 만약 당신이 이 하우투에 대해 저에게 이메일을 보내신다면 제목에 `[Linux SMP HOWTO]` 를 넣어주시면 더 빠른 답변을 받으실수 있습니다. ;)

이 하우투 문서는 **Chris Pirih**의 첫번째 문서 (http://www.ihoc.net/linux-smp-faq-draft.html) 문서로부터 개선되고 있습니다.

저자들은 이문서상의 오류와 생략으로 발생할수 있는 모든 손해와 문제들에 대해 책임지지 않습니다.

2. 모든 아키텍처와 관련된 질문들

2.1. 커널 관련

1. 리눅스가 멀티 스레드를 지원하나요**?** 만약 내가 두개 이상의 프로세스를 실행 시킨다면**,** 모든 **CPU**들에 분산 처리가 될까요**?**

   네, 프로세스와 커널-스레드들은 프로세서들에게 분산될 것입니다, 그러나 유저 스페이스 스레드들은 그렇지 않습니다.

2. 어떤 아키텍처들에서 **SMP**를 지원 하나요**?**

   Alan Cox로 부터:

   > SMP는 커널 2.0 이상 에서 인텔 MP1.1/1.4 를 지원하는 hypersparc(SS20 또는 그외), 인텔 486, 펜티엄 또는 그 이상에서 작동합니다. **Richard Jelinek**가 덧붙임: 4 개의 CPU 에서 테스트 되었고, MP 표준에 따르면 이론적으로 16 CPU 까지 지원할수 있습니다.
   >
   > 리눅스 커널 2.2.x 이상에서 UltraSparc, SparcServer, Alpha 와 PowerPC 등을 지원합니다.

   Ralf B押hle로 부터:

   > MIPS, m68k 와 ARM 은 SMP를 지원하지 않습니다. m68k 와 ARM은 아마 영원히 지원하지 않을것입니다.
   >
   > 저는 MIPS-SMP 박스가 생기는 데로 SMP 지원을 위해 해킹을 해보려고 합니다.

3. 리눅스 **SMP** 커널은 어떻게 만드나요**?**

   많은 리눅스 배포본들이 SMP 커널 패키지를 포함하고 있지 않기 때문에 (역자주: 사실 요즘 배포본들은 대부분 포함하고 있지만,) 당신이 직접 만들어야 합니다. 만약 당신이 아직까지 커널 컴파일을 해본적이 없다면 이것은 좋은 이유가 될것입니다. 커널 컴파일에 대한 설명은 이 문서의 목적에 벗어나므로, 더 자세한 내용은 Linux Kernel Howto 를 참고하세요. (**C. Polisher**)

   커널 2.0 이상 (2.1.132을 제외한) 에서는 커널의 주 Makefile (`/usr/src/linux/Makefile`)에서 `SMP=1` 라인의 주석을 풀어주면 됩니다.

커널 2.2.x 이상에서는 Processor type and features ---> [*] Symmetric multi-processing support "Symmetric multi-processing support"를 yes 로 해줍니다. (**Michael Elizabeth Chastain**).

그리고

Character devices ---> [*] Enhanced Real Time Clock Support

위와 같이 "RTC support" 를 yes 로 해줍니다. (**Robert G. Brown**). RTC 지원은 우리가 알고 있는 문제인 SMP 시스템의 시간의 느려짐을 해결하지는 않을것입니다, 그러나 부팅시 시간을 읽을때의 정지현상을 예방해 줍니다. 또한 RTC 기능은 몇몇 오리지날 인텔 메인보드에서 두번째 CPU를 인식하는데 필요합니다 (**Richard Jelinek**).

그리고

x86 커널에서는 APM (advanced power management) 기능을 넣지 마십시요! APM 과 SMP는 호환하지 않습니다. 그리고 당신의 시스템은 분명히(최소한 아마도 `;)`) 부팅시에 문제를 일으킬 것입니다 (**Jakob Oestergaard**). 2.1.x 이상의 SMP 커널에서는 APM 기능은 꺼집니다. 기본적으로 APM 은 SMP 시스템에서 미정이며, 무슨일이든지 일어날 수 있습니다. (**Alan Cox**)

그리고

x86 커널은 "MTRR (Memory Type Range Register)" 기능을 커널에 넣습니다. 이것은 몇몇 버그가 있는 BIOS에서 두번째 프로세서의 캐쉬 메모리가 작동하지 않는것을 해결 해 줍니다.

당신은 커널과 모든 관련 모듈들을 SMP 모드로 다시 컴파일 해야합니다. `make modules` 과 `make modules_install` 을 잊지 마십시요 (**Alan Cox**).

만약 모듈 적재 오류가 생긴다면 당신은 아마 모듈들을 컴파일 하지 않았거나 재 인스톨 하지 않았을 것입니다. 또한, 몇몇 2.2.x 대의 커널에서 SMP 커널에서 일반커널로의 재 컴파일시 문제가 있다는 보고가 있었습니다. 이것을 해결하려면 **.config** 파일을 저장해(다른 곳에) 놓은 다음, **make mrproper** 한후, 백업해놓은 **.config** 파일을 복사한후 커널을 재 컴파일 합니다. (**Wade Hampton**). 커널 컴파일후 lilo 실행을 잊지 마십시요.

요약:

```
make config # 또는 menuconfig 또는 xconfig
make dep
make clean
make bzImage # 또는 원하는 것으로(make zlilo,...)
# 커널 이미지를 복사한후(/boot/에) lilo를 실행
make modules
make modules_install
```

4. 비 **SMP** 커널을 어떻게 만드나요**?**

커널 2.0 대에서는 Makefile (/usr/src/linux/Makefile) 에서 `SMP=1` 라인을 주석 처리합니다.

2.2 대에서는 커널 설정시 "Symmetric multi-processing support" 에 no 로 대답하면 됩니다 (**Michael Elizabeth Chastain**).

당신은 커널과 관련 모듈 모두를 재 컴파일, 인스톨 해야합니다. `make modules` 와 `make modules_install` 그리고 lilo를 실행 시키는 것을 잊지 마십시요.

5. **SMP** 커널의 작동 여부는 어떻게 확인하나요**?**

   cat /proc/cpuinfo

   dual PentiumII 의 전형적인 결과:

```
processor       : 0
cpu             : 686
model           : 3
vendor_id       : GenuineIntel
[...]
bogomips        : 267.06

processor       : 1
cpu             : 686
model           : 3
vendor_id       : GenuineIntel
[...]
bogomips        : 267.06
```

6. 세밀한 락킹과 멀티스레딩 상태로 전환되는 커널의 상태는**?**

   리눅스 2.2 커널은 시그널 처리와 인터럽트와 몇몇 I/O 의 세밀한 락(lock)처리가 되어있다. 나머지는 천천히 이식되고 있다. 모든 스케줄링은 SMP에 안전하다.

   2.3 (다음 버전인 2.4) 커널은 아주 세밀한 락킹 기능을 가지고 있다. 2.3 커널에서 커다란 커널 락의 사용은 기본적으로 사라지고 대부분의 리눅스 커널의 하부 시스템들은 충분히 스레드화 된다: 네트워킹, VFS, VM, IO, block/page 캐쉬, 스케줄링, 인터럽트, 시그널 등등. (**Ingo Molnar**)

7. 리눅스 **SMP** 가 프로세서의 유사성을 지원하나요**?**

   일반적인 커널

   아니요 & 네. 프로세스들을 특정 CPU 위에서 실행하게 하는 길은 없습니다. 그러나 리눅스 스케쥴러는 각 과정들을 위해 프로세서 성향을 가집니다. 그것은 프로세스들을 특정 CPU들에 연결시키게 하는 경향이 있습니다.

   패치

   네. 관련 사이트 PSET - Processor Sets for the Linux kernel (http://isunix.it.ilstu.edu/~thockin/pset/): "이 프로젝트의 목적은 pset의 상호 호환성과 기능을 만들어 줍니다. (SGI에 의해 정의된 - 부분적으로 IRIX 6.4 커널에서 삭제된). 이것은 사용자들이 특정 프로세서(들)의 위에서 프로세스들을 동작하도록 결정하는 것을 가능하게 합니다. 그리고 스레드들은 분리된 프로세서들, 타이밍, 안전 (root 만의 CPU?) 에서 사용될수 있습니다. 그리고 아마도 더 많은 것들도."

이것은 syscall sysmp()에 중점을 둡니다. 이 기능은 어느 기능이 요청되는가에 따라 많은 매개 변수들이 있습니다. 기능들은 다음을 포함합니다.

- 프로세스/스레드를 특정 CPU에 고정하는것
- 몇가지 프로세스들을 실행하는 CPU의 능력을 한정하는것
- 집중되는 실행에서 CPU를 한정 시키는것 (restricting a CPU from running at all)
- 오로지 한개의 프로세스(부 프로세스들을 포함)를 실행하도록 하는것
- CPU의 상태에 대한 정보를 얻는것
- 바운드 되어 있을지도 모르는 프로세스들의 생성과 파괴 (creating/destroying sets of processors, to which processes may be bound)

8. **SMP 버그는 어디에 보고해야 하나요?**

   `linux-smp@vger.rutgers.edu.`

9. **SMP의 성능은 어떤가요?**

   만약 당신이 SMP 시스템의 성능을 측정하고 싶다면 Cameron MacKinnon에 의해 만들어진 http://www.phy.duke.edu/brahma/benchmarks.smp에서 몇가지 테스트를 해볼수 있습니다.

2.2. 사용자의 측면

1. **나에게 SMP가 정말 필요한가요?**

   만약 당신이 그렇게 물어봐야 한다면 아마도 아닐것 입니다. :) 일반적으로, 멀티 프로세서 시스템은 한개의 프로세서를 가진 시스템에 비해 더 낳은 퍼포먼스를 보여줍니다. 그러나 분명히 파악해야 할것은 CPU 의 수이외에 많은 다른 요인들을 고려할 필요가 있습니다. 예를 들어, 주어진 시스템의 프로세서가 느린 디스크 드라이브 때문에 쉬고 있다면, 이 시스템은 "input/output bound"이며, 프로세서의 추가로 얻는 이익은 적을 것입니다. 만약 시스템이 많은 프로세스들을 동시에 실행하고 있다면 프로세서의 추가로 얻는 이득은 많아집니다. 복수의 프로세서들이 사용될때 SCSI 디스크 드라이브들은 매우 효과적일수 있습니다.(**C. Polisher**)

2. **두개의 300 MHz 프로세서와 한개의 600 MHz 프로세서는 같은 능력을 수행하는지?**

   이것은 수행되는 어플리케이션에 따라 다릅니다. 그러나 대부분의 경우는 아닙니다. SMP 는 단일 프로세서에 비해 약간의 오버헤드를 추가합니다. (**Wade Hampton**). :)

3. **어떻게 하면 여러개의 CPU의 성능을 출력해 볼수 있나요?**

   Samuel S. Chessman의 덕택으로 몇가지 유용한 유틸리티들이 다음에 있습니다:

문자:

http://www.cs.inf.ethz.ch/~rauch/procps.html

근본적으로 이것은 procps v1.12.2 이다. 그리고 SMP를 위한 약간의 패치들.

2.2.x 대를 위한 패치는 이곳에 (**Gregory R. Warnes**) http://queenbee.fhcrc.org/~warnes/procps

그래픽:

xosview-1.5.1 는 SMP 를 지원합니다. 그리고 2.1.85 이상의 커널에서 /proc/stat 에 cpuX 항목이 있는 경우.

xosview 의 공식 사이트는: http://lore.ece.utexas.edu/~bgrayson/xosview.html

Kumsup Lee 에 의한 2.2.x 커널 패치들이 이곳에 있습니다. http://www.ima.umn.edu/~klee/linux/xosview-1.6.1-5a1.tgz

이외 여러가지 패치들이 http://www-isia.cma.fr/~forissie/smp_kernel_patch/에 있습니다.

당신은 xosview 로 프로세스 스케쥴링을 정확하게 모니터링 할수는 없습니다. xosview 자체가 하나의 프로세스 이며, 스케쥴링에 영향을 주므로 (**H. Peter Anvin**).

4. 커널 컴파일시 **1**개의 이상의 프로세스를 실행 시키려면**?**

   다음과 같이 합니다:

```
      # make [modules|zImage|bzImages] MAKE="make -jX"
 X 는 CPU 숫자입니다.
 주의 : 이것은 "make dep" 에서는 적용되지 않습니다.
```

   2.2 대 커널에서는 `/usr/src/linux/Documentation/smp.txt` 문서를 참고하세요.

   복수의 프로세서들을 사용하기 위한 충분한 메모리와 입출력 속도(하드 디스크등의) 가 아니라면 컴파일과정에 더 지연을 일으킬수 있습니다. `make MAKE="make -j 2" -j 2` 는 실제로 단일 프로세서에서도 효과를 볼수 있습니다. (**Ralf B押hle**).

5. 왜 **`time`** 명령어가 부정확하게 작동하는지**?** (**Joel Marchand**)

   2.x 대의 커널에서 `time` 명령어의 결과는 부정확합니다. 유저와 시스템의 합은 맞습니다만, 유저와 시스템 사이에 spreading (배치? 발생?)되는 시간은 정확하지 않습니다.

   더 자세하게: 부트 CPU 이외의 프로세서들에 의해 사용되는 시간들이 시스템의 시간과 같다고 생각되기 때문이다. 만일 당신이 프로그램의 시간을 잰다면, 사용자 시간과 시스템의 시간을 더한다면 그것은 거의 정확할 것이다. (시스템 시간을 계산하는 시간을 제외한) (**Jakob 兎tergaard**).

   이 버그는 2.2 커널에서 수정되었습니다.

2.3. SMP 프로그래밍

Jakob 兎tergaard 에 의해

SMP 리눅스를 위한 다중 스레드 프로그래밍에 대한 섹션입니다.

2.3.1. 병렬 처리 방법들

1. POSIX 스레드
2. PVM / MPI Message Passing Libraries
3. fork() -- 다중 프로세스

fork() 와 PVM/MPI는 일반적으로 메모리를 공유하지 않아, IPC 또는 메시징 API 에 의해 소통되기도 하며, 이것은 이번장에서 더이상 설명되지는 않을 것이며, 이것들은 단일 프로세서와 클러스터에서도 사용되는 것이므로 SMP 에 특정되어 있지도 않습니다.

오로지 POSIX 스레드만이 시스템 자원을 공유하는 것(특히 메모리)과 같은 다중 스레드를 제공한다. 이것은 SMP 머신을 특별하게 하는 것이며, 많은 프로세서들이 메모리를 공유하는 것이 가능하다. SMP에서 양쪽(또는 그이상)의 프로세서를 사용하기 위해서는 커널-스레드-라이브러리를 사용한다. 좋은 라이브러리는 LinuxThreads - Xavier Leroy에 의해 만들어진 pthread 라이브러리 (http://pauillac.inria.fr/~xleroy/linuxthreads/)이다. 새로운 리눅스 배포본들은 이 라이브러리를 기본으로 포함하고 있다. 그러므로, 당신은 커널 스레드 사용을 위해 따로 패키지를 사용할 필요가 없다.

어플리케이션 수준에서 커널-스레딩을 사용하지 않는 스레드들 (그리고 POSIX 스레드들)의 실현이 있다. 이 스레드 꾸러미들은 한개의 과정에서 스레딩을 유지한다. 그러므로 SMP를 이용하지 말라. 그러나 그들은 많은 적용에 좋고, 한개의 프로세서 시스템에 관한 커널-스레드들 보다 실제로 더 빠른 경향이 있다.

다중-스레딩은 Un*x 세계에서 인기가 없었습니다. 몇가지 이유로, 복수의 프로세스 또는 스레드를 필요로 하는 어플리케이션을 위해, 대부분은 fork()를 사용하여 씌여졌습니다. 그러므로, 스레드 사용에 접근할때, 서로 호환되지 않는(thread-ready하지 않은) 라이브러리, 컴파일러 그리고 디버거등이 문제가 됩니다. GNU/Linux 또한 예외는 아닙니다. 다음의 몇장에서 현재 가능한 것과 그렇지 않은 것을 설명합니다.

2.3.2. C 라이브러리

오래된 C 라이브러리는 스레드에 안전하지 않습니다. GNU LibC (**glibc**), 또한 **libc6**로 알려진 라이브러리를 사용하는 것은 매우 중요합니다. 이전 버전들도 당연히 사용가능은 하나, 당신을 좀더 괴롭혀 시스템 업그레이드의 원인이 될것입니다. 아마도 :)

만약 프로그램의 디버깅을 위해 GDB 를 사용하고자 한다면 다음을 보십시요.

2.3.3. 언어, 컴파일러 그리고 디버거

GNU/Linux 를 위한 풍부한 프로그래밍 언어가 있습니다, 그리고 그중에 대부분은 어떻게 든 스레드(심지어 Ada 와 자바와 같은 언어들도) 를 사용할수 있습니다.

이 장에서는 C 와 C++ 에 관해서만 기술할것입니다. 만약 당신이 다른 언어로 SMP 프로그래밍의 경험이 있다면 알려주세요.

GNU C 와 C++, EGCS C 와 C++ 컴파일러들은 표준의 C 라이브러리에서 스레드를 잘 지원한다. (**glibc**). 그러나 여기에 약간의 이슈들이 있다.

1. C 와 C++ 의 컴파일중, **-D_REENTRANT** 를 컴파일러 커맨드 라인에서 정의한다. 이것은 에러 처리 기능을 위해 필요하다. (errno variable과 같은).
2. C++ 를 사용할때에 만약 두개의 스레드가 동시에 throw exceptions 한다면, 이 프로그램은 segfault 될것입니다. 그리고, 컴파일러는 스레드-안전 하지 않은 코드를 생성할 것입니다. 회피 방법은 pthread_mutex_lock(&global_exception_lock) 을 모든 constructor(s) 클래스 throw()에 넣는다. , 그리고 상응하는 pthread_mutex_unlock(...) 를 추가합니다. 이것은 보기 좋지는 않으나, 작동은 합니다. 이 방법은 **Markus Ferch**에 의해 제시 되었습니다.

GNU 디버거 **GDB** 버전 4.18은 스레드를 바르게 취급할 것입니다. 대부분의 리눅스 배포본들이 패치된(thread-aware)한 gdb 를 제공합니다.

단지, 스레드와 일하기 위해 **glibc**를 패치하는 것은 불필요합니다. 만약 당신이 소프트웨어를 디버깅 할 필요가 없다면(개발 시스템을 제외한, 대부분의 시스템에서는 그럴것이다.) **glibc**의 패치는 불필요 합니다.

코어 덤프는 복수의 스레드들에 의해 생기지 않습니다. 어떻게든, 코어 덤프는 프로그램 전체가 아닌 현재 실행중인 스레드에 붙여집니다. 그러므로, 무엇이든지 디버깅을 할때 디버거 상에서 실행시키세요. Note that core-dumps are of no use when using multiple threads. Somehow, the core dump is attached to one of the currently running threads, and not to the program as a whole. Therefore, whenever you are debugging anything, run it from the debugger.

힌트**:** 만약 스레드가 100% CPU time을 잡아먹고 있다면, 그 이유를 알아낼수 없을것입니다. 이 경우 좋은 방법이 있습니다. : GDB 상이 아니라, 쉘상에서 바로 프로그램을 실행 시킵니다. top 으로 그 프로그램의 PID를 알아냅니다. 다음 GDB를 다음과 같이 실행합니다. **gdb** 프로그램 **pid**. 이것은 GDB를 지정한 PID 프로세스에 적용하게 하며, 스레드는 멈출 것 입니다. 이제 당신은 그 스레드에 해당하는 GDB 세션과 **bt**를 사용할수 있으며, 무엇이 일어나고 있는지 알수 있습니다.

2.3.4. 다른 라이브러리들

ElectricFence: 이 라이브러리는 스레드에서 안전하지 않습니다. 그러나 이것에 mutex lock을 삽입함으로써 SMP 환경에서의 사용이 가능해 집니다.

2.3.5. SMP 프로그래밍에 대해 참고할수 있는 것들

1. 병렬 프로그래밍에 관한 더 많은 정보를 어디서 찾을수 있나요?

   리눅스 병렬 처리 HOWTO (http://yara.ecn.purdue.edu/~pplinux/PPHOWTO/pphowto.html)

   많은 쓸모있는 정보를 여기서 찾을수 있습니다. 리눅스를 이용한 병렬 처리 (http://yara.ecn.purdue.edu/~pplinux/)

   또한 리눅스 스레드 FAQ (http://linas.org/linux/threads-faq.html)

2. 스레드화 프로그램과 라이브러리들은 어디에?

   다음을 보세요: 리눅스 멀티 스레드 프로그램 (http://www.informatik.uni-bremen.de/~hollow/mthread.html) (저는 하이퍼 링크를 좋아합니다 그거 아세요? ;))

   참고가 될 라이브러리들:

   OpenGL Mesa 라이브러리

   David Buccarelli 와 **Andreas Schiffler** 그리고 **Emil Briggs**의 덕택 으로 다중 스레드 버젼(몇몇 OpenGL 벤치마크에 의하면 5-30% 의 속도 증가를 제공하는 버젼이 있습니다. [1998-05-11]). 다중 스레드는 현재 실험적인 옵션으로서 메사 라이브러리에 포함 되었습니다. 더 자세한 것은 Mesa 라이브러리 (http://www.ssec.wisc.edu/~brianp/Mesa.html)을 참고하세요.

   BLAS

   펜티엄 프로 최적화 BLAS 그리고 인텔 리눅스를 위한 FFTs (http://www.cs.utk.edu/~ghenry/distrib/)

   멀티 스레드 BLAS 는 지금은 존재하지 않습니다만, 다중 프로세스 라이브러리는 1998-05-27 에 설계 되었습니다. Blas News (http://www.cs.utk.edu/~ghenry/distrib/blasnews).

   GIMP

   Emil Briggs (다중 스레드 메사를 만들고 있는 사람중 하나인) 에 의해 다중 스레드 GIMP 플러그인들이 있습니다. http://nemo.physics.ncsu.edu/~briggs/gimp/index.html

3. x86 아키텍처와 관련된 질문들

3.1. 왜 내 컴퓨터에서 작동하지 않나요?

1. **Cyrix, AMD** 등의 인텔외의 **CPU**에서 **SMP** 를 사용할수 있나요**?**

   짧은 대답**:** 아니요.

   긴 대답**:** 인텔은 APIC SMP안 에 대한 소유권을 주장 하고 있습니다. 그리고 위 회사들이 그 안을 사용하고 있지 않고 있습니다. (이것은 미래에 바뀔수 있겠지요). 사이릭스와 AMD는 소유권이 보호되지 않는 OpenPIC를 지원합니다만, 현재까지 그것을 사용하는 마더보드가 존재하지 않습니다.

2. 왜 오래된 내 **Compaq** 에서 작동하지 않나요**?**

   MP1.1/1.4 호환 모드로 맞추어 놓으세요.

   "Configure Hardware" -> "View / Edit details" -> "Advanced mode" (F7 일 겁니다.) "APIC mode" 설정에서 "full Table mode"로 합니다. 이것은 컴팩의 공식적인 권장사항 입니다.(**Daniel Roesen**)

   (**Adrian Portelli**)은 다음과 같이 했습니다 :

   a. 서버 부팅시 F10을 누르면 시스템 설정으로 들어갑니다.
   b. 엔터를 누르고 스플레쉬 화면을 지나갑니다.
   c. 재빨리 CTRL+A 를 누릅니다.
   d. "Advanced Mode" 설정 메세지가 나타날 것입니다.
   e. "Configure Hardware" -> "View / Edit details" 를 선택하고,
   f. 설정 화면이 나타나면
   g. "APIC Mode" 까지 스크롤 한다음 "Fully Mapped"를 선택합니다.
   h. 저장하고 리부팅합니다.

3. 왜 **ALR**에서 작동하지 않나요**?**

   Robert Hyatt로부터: ALR Revolution quad-6 는 매우 안전해 보인다. 몇몇 오래된 revolution quad (P6 프로세서가 없는)는 불확실...

4. 왜 **SMP** 가 느리죠**?** 또는 왜 한개의 **CPU**가 다른 **CPU**에 비해 매우 낮은 보고밉스 값을 나타내지요**?**

   Alan Cox 로 부터: 만약 프로세서들 중 하나의 보고밉스 값이 매우 낮다면, 캐쉬가 작동하지 않는것 입니다. 당신의 마더보드는 아마도 버그가 있는 BIOS를 사용하고 있을것입니다. 패치(BIOS 업그레이드?)를 하던지 돌려보내든지, 새로 사던지 하세요.

2.0 커널 (> 2.0.36) 에서 MTRR 패치는 이 문제를 해결해 줄것입니다. (커널 설정에서 "Handle buggy SMP BIOSes with bad MTRR setup" 를 선택하세요).

마지막 버젼의 2.2 대 커널들은 버그가 있는 SMP BIOS 문제를 알아서 처리할것 이라고 생각합니다.

5. **IBM** 머쉰에서 문제들이 있다고 들었습니다**.**

   몇몇 IBM 의 EBDA 에서 MP1.4 bios 블럭을 가집니다. 이것은 허락되지만 2.2 커널 이하에서는 지원되지 않습니다.

   오래된 486SLC IBM SMP 박스에서 Linux/SMP 는 하드웨어 FPU 가 필요합니다.

6. 인텔 **MP 1.4**가 **1.1** 규정에 비해 이점이 있나요**?**

   아뇨 (Alan 에 의하면 :)), 1.4 는 stricker specs of 1.1 일 뿐이다.

7. **SMP** 에서 왜 시계가 그렇게 빨리 가지요**?**

   2.0 대의 커널에서 알려진 문제이다, 2.2 대의 커널로 업그레이드를 고려해라.

   Jakob Oestergaard 로 부터: 또는, xntpd의 실행을 고려하세요. 이것은 당신의 시간을 정확하게 맞춰 줄겁니다. (커널에서의 RTC 지원도 이 현상을 막아준다라고 저는 생각합니다. 저의 경우 이것은 해당되었구요. 그러나 확실하지 않으므로 이것은 그저 행운일지도 모르지요.)

   이것을 예방할 수정이 2.2.x 대의 커널에 있었습니다.

8. 왜 내 프로세서들의 번호가 **0** 과 **1**이 아닌 **0** 과 **2** 로 되지요**?**

   CPU 번호는 마더보드 제조업체들에 의해 할당되는 것이며, 이것은 아무 의미도 가지지 않습니다. 무시하세요.

9. 내 **quad-Xeon** 시스템이 부팅시 정지됩니다**.**

   (**Doug Ledford**) LILO 를 LARGE_EBDA 지원하게 재컴파일 하십시요. 그리고 커널 빌드시 항상 make bzImage 로 하십시요. 이것은 인텔 다중 지온 보드의 SMP 부팅시 정지 를 막아줍니다. 그러나 이것은 LILO 에서 root= 옵션이 더이상 작동 하지 않습니다. 그러므로 확실히 rdev 로 당신의 커널이 정확한 루트 파티션을 사용하도록 해야 합니다.

   (**Robert M. Hyatt**) 3개의 CPU를 사용한다면, 네번째 소켓에 터미네이터가 있나요?

10. 부팅시 **IOAPIC** 시그날과 함께 정지 됩니다**.**

    부팅 옵션에 "noapic" 를 넣거나(**John Aldrich**) 그리고(또는) "reboot=bios" 를 사용합니다. (**Terry Shull**).

11. 내 시스템이 **NFS**에 많은 부하가 걸렸을때 정지 됩니다**.**

    커널 버전 2.2.x 이상과 knfsd 패치를 사용 해보십시요. 이것은 현재 조사중입니다. (**Wade Hampton**)

12. 내 시스템이 **oops** 메시지없이 정지 됩니다**.**

만약 당신이 커널 2.2.11 또는 2.2.12를 사용하고 있다면 마지막 버젼의 커널을 사용하십시요. 2.2.13에서 몇가지 SMP 관련 패치가 있었습니다. 몇몇 사람들에게서 이 버젼(2.2.11과 2.2.12)가 SMP 모드에서 안정적이지 않다는 보고가 있었습니다 (NFS 문제들도). 시리얼 콘솔을 사용해 커널의 oops 메세지를 캡춰해 볼수 있습니다. (**Wade Hampton**)

계속 문제가 있다면(그리고, 다른 사항들 조차 도움이 되지 않았다면), 당신은 2.3 대의 커널을 시도해 볼수 있습니다. 이 버젼의 커널들은 더 많고, 강력한 SMP/APIC 코드를 가지고 있습니다. 그리고 automatic hard-lockup-prevention code 는 그저 조용히 멎어 (시스템이) 버리는 것이 아니라 쓸모있는 oopses 메세지를 남길것입니다. (**Ingo Molnar**)

(**Osamu Aoki**) 가 : 당신은 또란 BIOS 와 관련된 모든 전력 절약 모드를 불가능하게 하게 해야합니다. 다음은 올바른 설정의 예입니다. (듀알 Celeron 466 / Abit BP6):

```
POWER MANAGEMENT SETUP.
  ACPI:              Disabled
  POWER MANAGEMENT:  Disabled
  PM CONTROL by APM: No
```

만약 절전 모드가 켜져 있다면, 무작위적인 시스템 다운이 일어날수 있습니다.

13. 시스템 정지 디버깅

이절은 **Wade Hampton**에 의해 씌여졌음.

시스템 정지 디버깅의 좋은 수단은 Andrea Arcangeli에 의한 ikd 패치를 사용하는 것입니다. ftp://ftp.suse.com/pub/people/andrea/kernel-patches
(ftp://ftp.suse.com/pub/people/andrea/kernel-patches/)

몇개의 디버깅 옵션이 있는데, soft lockup 옵션은 사용하지 마세요. 새로운 SMP 머쉰들은 NMI oopser상에서 커널 디버깅 옵션을 사용합니다. NMI oopser 의 작동 확인은 `/cat /proc/interrupts` 의 결과에 NMI 가 있는지 보면 됩니다. 이제 시스템이 정지되면 당신은 oops 메세지를 얻을수 있을것입니다.

또한, %eip 옵션을 시험해 보아도 좋습니다. 이것은 커널이 커널 함수가 불려질때마다, 콘솔상으로 %eip 주소를 출력해 줍니다. 시스템이 정지될때, 다음 두번째 칼럼에 의한 첫번째 칼럼을 적어둔후, System.map 파일에서 그 주소를 찾아 봅니다. 이것은 콘솔모드에서만 할수 있습니다.

또한 시리얼 콘솔은 커널 정지를 디버깅하는데 대단히 편리합니다. (단지 SMP 커널만이 아닌.)

14. 로그의 **"APIC error interrupt on CPU#n, should never happen"** 메시지

다음과 같은 메세지는

```
APIC error interrupt on CPU#0, should never happen.
... APIC ESR0: 00000002
... APIC ESR1: 00000000
```

잘못된 체크섬 에러를 가르킵니다. 이것은 리눅스(하드웨어 체크섬 부분의 APIC 메세지)에 기인할수 없습니다. 이것은 추가(주변 기기?) 하드웨어에 의한 것일지도 모릅니다. 시스템의 불안정함이 보일때까지는 이것은 문제가 되지 않습니다. - APIC 메세지는 배달될때 까지 재시도 됩니다. (**Ingo Molnar**)

3.2. 충돌의 가능한 원인들

이번 장에서는 SMP 머쉰의 비정상적인 작동의 원인들을 찾을수 있을것이다. (**Jakob tergaard**)

- 냉각

  Ralf Bhle 로부터 : [이 경우 팬들의 크기에 관련이 있었다] 공기의 흐름이 중요합니다. 너무 작은 케이스는 문제를 일으킬수 있습니다. 반대로 쓸데없이 큰 케이스도 문제의 소지가 있습니다. 일반적인 타워케이스가 데스크탑들 보다 약간 냉각효율이 좋다고 봅니다. 요컨대, 좋은 케이스는 공기역학적으로 설계되어 있겠지요.

  당연히 여러분은 전자상가에서 다른 팬을 추가할수 있습니다. 또한 여러분은 메인보드에 장착되어 있는 lm 센서로 CPU와 메인보드의 온도,전압등을 모니터링 할수 있습니다. (http://www.netroedge.com/~lm78) 이것은 과열문제를 도와줄수 있습니다. (**Wade Hampton**)

- 나쁜 메모리

  싸구려 램을 사지 마세요. 그리고 다른 램 모듈들을 혼합해서 사용하지 마십시요.

  특히 Tyan 마더보드들은 램 속도와 관련하여 문제가 있습니다. (다음장의 Tyan 마더보드들에 대한 해결책을 보세요.)

  CPU가 8ns 램을 사용해야 하는 경우, 10ns의 PC100램을 사용한 마더보드들에서 버그 보고가 있었습니다. (**Wade Hampton**)

- 다른 스텝핑을 가진 프로세서들의 나쁜 조합

  `/proc/cpuinfo` 을 확인해서 프로세서들이 같은 스테핑(stepping)을 가지고 있는지 확인해 봅니다.

- 만약 당신의 시스템이 불안정 하다면 오버클럭을 하지 마세요!

  만약 안정적이라 하더라도, 오버클럭은 하지 않는게 좋습니다.

  Ralf Bhle로부터 : 오버클럭킹은 미묘한 문제들을 일으킵니다. 좋은 예로, 나의 오버클럭한 오래된 기계들중에 640x400 의 프랙탈 픽셀들을 그려내는데 오류를 일으킵니다. 이 문제들은 도구를 사용하여 비교하면 나타납니다. 그러므로, 오버 클럭킹은 절대 (**never, nuncas, jamais, niemals**) 하지 마세요.

- **2.0.x** 대의 커널과 **fast ethernet** (**Robert G. Brown**)

  2.0.x 커널에서 놓은 성능의 빠른 이더넷 시스템이 중요한(그리고 알려진) 문제를 넥트웍 인터럽트 핸들에서 가지고 있습니다.

  해결책은 마지막 개발 버전의 100BT 드라이버를 다음에서 구하는 것입니다. CESDIS 리눅스 이더넷 드라이버 사이트 (http://cesdis.gsfc.nasa.gov/linux/drivers/) (SMPCHECK을 정의한).

- **440FX** 칩셋에서의 버그 (**Emil Briggs**)

만약 당신의 시스템의 마더보드가 440FX 칩셋을 사용하며, 시스템이 정지되는 문제가 있다면 칩셋의 문서화된 정오표에 의한 것일수 있습니다.

참조 : 인텔 440FX PCIset 82441FX (PMC) 와 82442FX (DBX) 규격의 업데이트. pg. 13

http://www.intel.com/design/pcisets/specupdt/297654.htm

이 문제는 BIOS 이 업그레이드 (또는 커널 패치)로 해결할수 있습니다. 그리고 실제로 David Wragg 는 Richard Gooch 의 MTRR 패치를 포함하는 커널 패치를 썼습니다. 더 많은 정보는 다음을 참고:

http://nemo.physics.ncsu.edu/~briggs/vfix.html

- 리눅스 **SMP**로 부팅하기전에 **emm386.exe**를 실행시키지 마세요**.**

  Mark Duguid 로 부터, 특히나 W6LI 마더보드에서는. `;)`

- 만약 당신의 시스템이 리부팅후 멎어 버린다면**,** 두가지 원인이 있을수 있습니다**. (BIOS** 와 메모리와 관련된**) (Jakob 兎tergaard)**

  - 만약 BIOS 의 설정중 "memory hole at 16M" 또는 "OS/2 memory > 64MB" 을 disable 로 하십시요, 리눅스는 이 옵션들에 반응하지 않습니다.
  - 만약 당신이 64MB 이상의 메모리를 가지고 있다면, 그리고 당신이 lilo 설정에 수동으로 메모리양을 적어 주었다면, 그 설정을 실제의 메모리양에서 1MB 적게 적어 주세요. 예를 들어 128MB 를 가지고 있다면, append="mem=127M"

- **IRQ** 와 관련된 문제들중 알아야 할것

  몇몇 카드들이 인식되지 않거나, IRQ 충돌 현상이 있다면 카드들을 서로 다른 슬롯으로 옮겨 보거나, IRQ 를 바꿔봅니다.

  hASCII 에 의해 : 리로 설정 파일에서 "append="hisax=9,2,3"" (ISDN +Hisax 지원을 위한) 을 지웁니다. (커널 2.1.xx). 2.0.xx 에서는 문제 없음.

  BIOS 설정에서 "MP 1.4 mode" 또는 "route PCI interrupts through IOAPIC", 또는 "OS Type" 와 같은 설정들을 DOS 또는 Novell 로 설정하지 마세요. (**Ingo Molnar**).

- 플로피와 사운드 카드가 동시에 사용할때

  만약 플로피를 사용하려 할때(예를 들어 사운드를 플레이 하면서) 시스템 정지가 일어 난다면, drivers/pci/quirks.c 파일을 다음과 같이 고칩니다. `/int isa_dma_bridge_buggy = 1;` 이 문제는 내 Dell WS400 dual PII/300, 2.2.x, SMP에서 일어 났습니다. (**Wade Hampton**).

3.3. 마더보드 특정 정보

주의할점: 몇몇 많은 특정 정보는 다음 사이트에서 찾을 수 있다. 리눅스 SMP를 동작시킬 수 있는 마더보드 (http://www.nlug.org/smp/)

3.3.1. 마더보드에 알려진 문제들

- 지금까지 알려진 것은 없다.

3.4. 저가의 **SMP** 리눅스 박스**(**듀얼 셀러론 박스**)**

(**St묹hane <65533>olivet**)

현재로 살만한 가장 저가의 SMP 리눅스 박스는 듀얼 셀러론 시스템이다. 그러한 시스템은 인텔에 따르면 공식적으로는 가능하지 않다고한다. 2세대 셀러론 (128kb L2 캐시)를 고려하는게 좋다.

3.4.1. 듀얼 셀러론 박스를 작동시키는게 가능할까요**?**

인텔에서부터의 공식적인 대답**:** 가능하지 않습니다, 셀러론은 SMP 모드에서는 작동할 수 없습니다.

현실적인 대답**:** 가능하지만 슬롯 1 프로세서에 대해 하드웨어 변경을 요구합니다. 변형은 Tomohiro Kawada의 페이지 듀얼 셀러론 시스템 (http://kikumaru.w-w.ne.jp/pc/celeron/index_e.html) 에서 확인할 수 있다. 물론, 이러한 종류의 변형은 제품에 대한 보증을 기대하지 말아야 한다. 몇몇 셀러론의 버젼들은 370 소켓 포맷에 적용이 가증하다. 그러한 경우에 변형은 슬롯 1 어댑터에 소켓 370 위에서 이루어 질 수도 있고 SMP 사용에 맞추어서 미리 만들어진채로 팔리기도 한다. (**Andy Poling**, **Hans - Erik Skyttberg**, **James Beard**)

두개의 셀러론을 소켓 370 포맷으로 집어넣는 마더보드(ABIT BP6)가 있습니다. (**Martijn Kruithof**, **Ryan McCue**). ABIT 컴퓨터 BP6는 테스트를 해봤으며 듀얼 ppga 소켓 370 을 이용 리눅스에 적용했다. (**Andre Hedrick**).

3.4.2. 어떻게 하면 리눅스가 듀얼 셀러론 시스템에서 동작하나요**?**

잘 작동합니다.

3.4.3. 셀러론 프로세서들은 쉽게 오버클럭을 할 수 있다고 알려져 있는데 듀얼 시스템도 가능한가요**?**

동작할 수 있습니다**.** 하지만 이러한 종류의 시스템을 오버클럭 한다는 것은 하나의 프로세서 시스템에서의 오버클럭만큼 쉽지 않습니다. 생산적인 시스템을 위해서는 그리 썩 좋은 생각은 아니다. 개인적인 사용을 위한 것이라면 듀얼 300A를 450Mhz 로 안정적으로 쓰고 있다는 보고가 있습니다.(많은 사람들이 보고 하고 있음)

3.4.4. 그리고 4개의 셀러론 시스템은 만드는 중인가요?

불가능 합니다. 셀러론 프로세서들은 펜티엄 II와 거의 같은 특성을 갖고 있기 때문이다. 만일 여러분이 여러분의 시스템에 2개 이상의 프로세서를 원한다면 여러분은 펜티엄 프로나 펜티엄 제온, 펜티엄 III(?)를 고려해야 할 것이다.

3.4.5. 셀러론과 펜티엄 II 프로세서와 섞어 쓰는 것은 어떤가요?

재사용이 가능한 셀러론 프로세서와 펜티엄 프로세서를 같은 환경에서 사용한다면 이론적으로 가능하다.

Alexandre Charbey가 그런 시스템을 만든 적이 있음:

- Asus P2B-D motherboard, proc 1: Celeron 366, proc 2: Pentium II 400@266
- 66Mhz and 75Mhz 버스 진동수
- 가장 빠른 프로세서(셀러론의 경우에서)는 두번째 슬롯에 위치해 있어야 한다. 가장 빠른 프로세서와의 교체는 엄청난 실패를 가져온다.

4. 스팍 구조에 관한 질문들

4.1. 어떤 스팍 머신이 지원되나?

다음의 사이트UltraLinux (http://ultra.linux.cz/)를 살펴보면 (오직 SMP 시스템에 관해서):

- UltraSPARC PCI 기반의 워크스테이션들: Ultra60, Ultra450
- UltraSPARC SBUS 기반의 서버들: Enterprise 1, 2, 150
- UltraSPARC SBUS 기반의 큰 서버들: Enterprise 3000, 4000, 5000, 6000, 10000
- UltraSPARC PCI 기반의 서버들: Enterprise 250, 450
- SPARC sun4m SMP 머신들 (**Anton Blanchard**)

UltraLinux는 14개의 CPU 머신에서도 작동한다. (참고 사이트 dmesg 결과입니다 (http://lwn.net/1998/1210/a/dm-sparc.html)).

4.2. 스팍 SMP 지원과 관련된 문제

(**David Miller**) 특별한 걱정은 없습니다.

유일하게 알려진 문제는 고치려 하지 않았지만 만일 32비트 시스템 (즉 ultrasparc이 아니라는 의미) SMP 커널을 형성시키고자 하면 커널은 sun4c 시스템에서 작동하지 않는다.

4.3. 최신 커널(2.2)이 갖고 있는 SMP의 특정한 제한사항

(**David Miller**) include/linux/tasks.h 에 버그가 있는데 NR_CPUS를 지원하는 하드웨어에 대한 상한 선으로 UltraSparc에서 64로 하는 것이 좋다. :-)

5. PowerPC 구조의 특정한 문제들

5.1. 어떠한 PPC 머신들이 지원되나?

- PowerSurge 보드들 (UMAX s900을 포함해서)
- PowerMac
- Motorola MTX: 지원은 아직 개발중이다. 패치들은 아직 커널에 포함되어 있지 않다. (**Troy Benjegerdes**)

(**Cort Dougan**) PPC RS/6000 시스템들도 아직 지원이되지 않는다.

5.2. PPC SMP 지원에 관련된 특정 문제들

문제는 없다. 대개 SMP 컴파일에는. 대개는 UP나 SMP 둘중에 하나에 모듈이 특정화되어있다. 커널을 재컴파일하라. (**Paul Mackerras**)

6. 알파 구조의 특정문제들

6.1. 어떠한 알파머신들이 지원되나?

(**Geerten Kuiper**) SMP는 대부분의 AXP 서버에서는 잘 작동한다.

(**Jay A Estabrook**) SMP는 대부분의 컴팩 제품에는 동작하는 것 같다. 두개 이상의 CPU를 갖는 박스는 다음을 포함한다:

- AS2000/2100 (SABLE)

- AS4000/4100 (RAWHIDE)
- DS20 (DP264)

포함하지 않는 것은:

- AS2100A (LYNX)
- TurboLaser bigboys (8200/8400)

6.2. 알파 SMP 지원에 관련된 특정문제들

없습니다.

7. 유용한 점들

7.1. 다양한 것들

- 리눅스를 이용한 병렬처리 (http://yara.ecn.purdue.edu/~pplinux/)
- 리눅스 병렬처리 HOWTO (http://yara.ecn.purdue.edu/~pplinux/PPHOWTO/pphowto.html)
- (오래되었지만) 리눅스 SMP 홈페이지 (http://www.uk.linux.org/SMP/title.html)
- 리눅스-smp 메일링 리스트

  구독하려면, `subscribe linux-smp`라는 내용으로 majordomo@vger.rutgers.edu (mailto:majordomo@vger.rutgers.edu)의 주소로 보내라.

  unsubscribe하고 싶지 않다면, `unsubscribe linux-smp`라는 내용으로 majordomo@vger.rutgers.edu (mailto:majordomo@vger.rutgers.edu)으로 메일을 보내면 된다.

  리눅스 SMP 모음 (http://www.linuxhq.com/lnxlists/linux-smp/)

  progressive-comp.com의 리눅스 SMP모음 (http://www.progressive-comp.com/Lists/?l=linux-smp&r=1&w=2#linux-smp)

- Xavier Leroy가 만든 pthread 라이브러리 (http://pauillac.inria.fr/~xleroy/linuxthreads/)
- 리눅스 SMP가 작동하는 것으로 알려진 마더보드 (http://www.nlug.org/smp/)
- procps (http://www.cs.inf.ethz.ch/~rauch/procps.html)
- 커널 2.2.x를 위한 procps 패치 (http://queenbee.fhcrc.org/~warnes/procps)
- xosview (http://lore.ece.utexas.edu/~bgrayson/xosview.html)

- 커널 2.2.x를 위한 xosview (http://www.ima.umn.edu/~klee/linux/xosview-1.6.1-5a1.tgz)
- 리눅스의 SMP 성능 (http://www.phy.duke.edu/brahma/benchmarks.smp)
- CESDIS 리눅스 이더넷 디바이스 드라이버 사이트 (http://cesdis.gsfc.nasa.gov/linux/drivers/)
- 듀얼 셀러론 시스템 (http://kikumaru.w-w.ne.jp/pc/celeron/index_e.html)

7.2. 멀티스레드 프로그램과 라이브러리

- 리눅스 스레드 FAQ (http://linas.org/linux/threads-faq.html)
- 리눅스에서 멀티스레드 프로그램 (http://www.informatik.uni-bremen.de/~hollow/mthread.html)
- 인텔 펜티엄 프로에 최적화된 BLAS와 FFT (http://www.cs.utk.edu/~ghenry/distrib/) Blas 뉴스 (http://www.cs.utk.edu/~ghenry/distrib/blasnews)
- Mesa 라이브러리 (http://www.ssec.wisc.edu/~brianp/Mesa.html)
- GIMP를 위한 병렬 플러그인 (http://nemo.physics.ncsu.edu/~briggs/gimp/index.html)

7.3. 특정 SMP 패치들

- Forissier 커널 패치들 (http://www-isia.cma.fr/~forissie/smp_kernel_patch/)
- 440FX 칩셋의 버그를 위한 패치 (http://nemo.physics.ncsu.edu/~briggs/vfix.html)
- MTRR 패치 (http://www.atnf.csiro.au/~rgooch/kernel-patches.html)
- PSET - 리눅스 커널을 위한 프로세서 세트 (http://isunix.it.ilstu.edu/~thockin/pset/)
- Ingo Molnar SMP 패치 (http://www.redhat.com/~mingo/)

7.4. 586/686 머신을 위한 병렬/최적화 컴파일러 (Sumit Roy)

- 펜티엄 컴파일러 그룹 (http://www.goof.com/pcg/) pgcc의 창시자들
- Absoft (http://www.absoft.com/) 포트란 90과 77 컴파일러
- Portland Group 회사 (http://www.pgroup.com/)는 supports the OpenMP (http://www.openmp.org)라는 리눅스에서 포트란 코드를 병렬화 시키는 표준을 지원한다.

- Pacific-Sierra 연구소 (http://www.psrv.com/)는 자유롭게 사용할 수 있는 F90 컴파일러를 SMP 리눅스를 위해 개발했다.
- Applied Parallel Research (http://s006.infomall.org/index.html)는 현재 윈도우즈 NT를 위해 병렬 컴파일러를 개발한다.
- KAI (http://www.kai.com)는 OpenMPI 실행이 가능한 리눅스를 위해 C++-컴파일러를 갖고 있다. 이를 Guide_OpenMP라 한다. 더 많은 정보는 http://www.kai.com/parallel/kappro/guide에서 찾을 수 있다. (**Gero Wedemann**)

8. Glossary

- **SMP**는 병렬 다중 프로세서(Symmetric Multi-Processors)의 약자이다.
- **APIC**는 향상된 프로그램이 가능한 인터럽트 컨트롤러(Advanced Programmable Interrupt Controller)이다.
- **thread** 스레드라는 것은 하나의 프로세스에서 프로세서의 활동성을 나타내는 것이다. 동일한 프로세스는 다중의 스레드를 가질 수 있다. 그러한 스레드들은 프로세스 주소공간을 공유하고 데이터 또한 공유할 수 있다.
- **pthread** Posix 스레드로 Posix 표준에 의해 정의된 것이다.
- **APM** 향상된 전원관리(Advanced Power Managment)

9. 새로운 것은 무엇인가霧妹買盂罵靺氓罵買

v1.9, 2000년 1월 13일

- 모든 BIOS 파워절전 특성들을 사용하지 말기를 바란다.(**Osamu Aoki**)
- 컴팩 서버에 향상된 설정 모드로 접근하는 방법을 설명해주십시오. (**Adrian Portelli**)

v1.8, 1999년 11월 8일

- 4개의 셀러론 마더보드는 hoax였고 지난 단락에 있음. (**Simen Timian Thoresen**)

v1.7, 1999년 11월 6일

- 새로운 도입(**C. Polisher** aka cp)
- 숫자가 잘못된 것과 문법적 오류 삭제
- 커널 컴파일에 관한 도입단락
- SMP 필요성에 관한 도입단락
- KAI 최적화 컴파일러에 관한 참조 (**Gero Wedemann**)
- 4개의 셀러론 보드가 존재한다. (**Jeffrey H. Ingber**)

v1.6, 1999년 10월 21일

- xosview 스케줄링에 관한 추가된 정보
- "APIC error interrupt on CPU#n에 관한 APIC 에러 인터럽트"에 관한 추가된 정보
- 하드 lockup에 관한 추가된 정보
- "최대 성능을 얻는 방법"에 관한 내용 삭제
- 다른 x86 프로세서를 포함하는 (셀러론과 P-II) 듀얼 시스템에 관한 정보

v1.5, 1999년 10월 4일

- 좀더 정확한 PSET에 대한 기술

v1.4, 1999년 9월 30일

- MTRR 지원을 x86 SMP 커널에서 지원기능을 활성화 시키는 가에 관한 내용

v1.3, 1999년 9월 29일

- 아주 아주 많은 문법적 오류와 오타 교정 (**Wade Hampton**)
- 2.2/2/4/2/0 의 차이에 관련된 짤막한 도입내용 추가
- 커널을 재컴파일 하는 방법에 관한 내용추가
- SMP/UP 모듈 문제에 관련된 내용 추가
- 사용자와 커널 스레드에 관계된 Posix 스레드 부분에 대한 내용첨가
- NFS 와 커널 lock에 관한 새로운 목록
- 메세지 없이 커널 lock을 하는 것에 관한 목록 추가

- lockup 문제해결에 관한 목록추가
- 발열문제에 관한 새로운 내용추가
- 본저자가 잊어버린 기타의 업데이트 자료들
- 플로피로 접근 하는 것과 사운드에 관한 내용추가

v1.2, 1999년 9월 27일

- 이름 변화: 이문서는 HOWTO가 된다. (**Guylhem Aznar**)

v1.1, 1999년 9월 26일

- 첫번째 Chris Pirih의 FAQ 초안 링크
- IRQ와 관련된 문제들 확장

v1.00, 1999년 9월 25일

- 아주 오랜만에 첫번째 업그레이드
- 전체 FAQ 압축: 2.4가 곧 출시
- Ingo Molnar로부터 얻은 커널 locking 정보
- "어떻게 하면 SMP에서 응용프로그램이 작동하나요"라는 항목 삭제
- "제 SMP 시스템이 항상 락이 걸려 있는데"라는 항목 삭제 outdated
- "여러분은 2.0.35 실행하지 않나요?"라는 항목 삭제
- "몇몇 하드웨어는 문제를 일으키는 것으로 알려져 있는데"라는 항목삭제 outdated
- "알려진 문제가 있는 마더보드들"이란 부분을 새로 처음부터 시작
- "알려진 문제가 없는 마더보드들"이란 부분 삭제
- 업데이트된 듀얼 셀러론 부분 추가
- "SPARC sun4m SMP 머신들" 부분을 SMP 스팍 머신을 위해 추가 (**Anton Blanchard**)
- "부팅도중 머신이 IOPANIC 문제를 일으키고 멈춰버리는 경우" 부분을 " 왜 내 머신에서는 작동하지 않는가?"라는 부분으로 추가 이동
- "SMP 성능은 어떤가요?"라는 항목 추가
- "왜 제 오래된 컴팩은 작동 안하나요?" 항목 추가
- 오래된 점들 보강
- Ingo SMP 패치 테스트에 관한 내용 추가

v0.54, 1999년 3월 13일

- SMP 알파 시스템에 관한 추가된 부분

v0.53, 1999년 3월 8일

- SMP PowerPC 시스템에 관한 내용 추가

v0.52, 1999년 3월 7일

- SMP 스팍 시스템에 관한 부분추가

v0.51, 1999년 3월 6일

- 추가된 듀얼셀러론 부분
- Adaptec 부분 삭제
- procps 링크 업데이트
- xosview 링크 업데이트
- 네개의 Xeon 부트중 멈추는 현상에 관한 추가적인 대답
- gd를 위한 glibc 패치에 관한 내용추가:반드시 레드햇 5.2에 추가

v0.50, 1999년 2월 3일

- "리눅스에서 멀티스레드 프로그램"링크 업데이트

v0.49, 1999년 1월 13일

- CONFIG_SMP에 관한 내용 업데이트. Documentation/smp에 .txt를 추가 (**Michael Elizabeth Chastain**)

v0.48, 1998년 12월 10일

- 오타 수정. 이메일 주소 수정

v0.47, 20 november 1998

- MTRR 패치로서 2.0.3 내용추가 (BogoMips 문제와 관련된)

v0.46, 1998년 11월 10일

- Epox KP6-LS 마더보드에 관한 내용 업데이트

v0.45, 1998년 10월 25일

- /proc/stat 파일에 관한 내용 수정
- CESDIS 이더넷 리눅스 드라이버 사이트 추가

v0.44, 1998년 10월 14

- 리눅스 **SMP**에서 작동하는 소문난 마더보드들 의 링크 업데이트
- 리눅스 커널 2.0 SMP 시스템의 시간을 맞추는 방법에 관한 Jakob의 설명

v0.43, 1998년 9월 9일

- 3.1절의 첫번째 문제 업데이트
- mt-Mesa 링크 업데이트: 다중-스레드가 Mesa 배포에 실험적으로 들어가 있다.

v0.42, 1998년 9월 2일

- 3.3절의 조그만 업데이트
- 다중스레드 Mesa 와 SMP 성능에 관한 두개의 링크 정보가 오래됨
- 스레드와 C++에서 예외에 관한 목록 업데이트 (3.3절)

v0.41, 1998년 9월 1일

- Jakob兗tergaard가 쓴 "3.3 SMP 프로그래밍" 추가
- 3.3절에 있던 "3.2 사용자 측면" 섹션의 내용에서 몇가지 옮겨옴

v0.40, 1998년 8월 27일

- 3.1절 업데이트, 7목록: 프로세서 경향성

v0.39, 1998년 8월 27일

- Tyan 마더보드를 위한 Award BIOS 버전 추가 (**hASCII**)
- 충돌에 관한 부분에 대한 목록추가
- Asus P2B-DS의 좋은 지원 (**Ulf Rompe**)
- smp-list 모음에 새내용 추가(**Hank Leininger**)

v0.38, 1998년 8월 8일

- 리눅스 스레드 FAQ에관한 내용추가

v0.37, 1998년 7월 30일

- **Emil Briggs**는 Gimp를 위한 병렬 플러그인에 대해 작업중 ("스레드된 프로그램이나 라이브러리 있나요?",절과 "사용자 측면")

v0.36, 1998년 6월 26일

- **Jakob 免tergaard** 덕택에 , "충돌의 두가지 가능한 원인"부분의 변화
  - 2.0.33에서 2.0.35로 변화
  - "시스템 중단의 BIOS와 관련된 문제들"추가

v0.35, 1998년 7월 14일

- 문제없는 보드중의 하나인 N440BX Server Board에 관한 내용추가
- BIOS 업그레이드와 함께오는 GigaByte 마더보드에 대한 성공이야기 추가
- "최고의 성능을 얻는 방법"에 관한 내용 추가

v0.34, 1998년 6월 10일

- "586/686 머신을 위한 병렬화/최적화 컴파일러"의 내용을 "유용한 점들"부분에 추가 **Sumit Roy**
- 오타수정 "Asus P/I-UP5"는 원래"Asus P/I-P65UP5"이었음.

v0.33, 1998년 6월 3일

- GigaByte DLX 마더보드의 성공스토리
- Tyan 마더보드를 위한 팁, BIOS 옵션중에서 "DRAM Fast Leadoff"기능을 죽여라

v0.32, 1998년 5월 27일

- Asus P/I-UP5 보드를 문제 없는 마더보드 부분에 추가

v0.31, 1998년 5월 18일

- Elitegroup P6LX2-A이 2.1.100과 2.2.101과 작동됨.
- 버그들은 다음주소로 보고하면 됨`linux-smp@vger.rutgers.edu`

v0.30, 1998년 5월 12일

- SuperMicro보드가 문제없는 보드 부분에 추가

v0.29, 1998년 5월 11일

- GigaByte 686 보드를 2.1.101 커널을 이용하여 성공한 이야기
- "사용자 측면"에 새로운 항목추가:"스레드 프로그램이나 라이브러리"는 있나요?"
- OpenGL Mesa 라이브러리가 다중 스레드 지원 자세한 것은 그 부분을 살펴보기 바람.

v0.28, 1998년 5월 9일

- 이 FAQ가 미국 미러사이트 생김
- Gigabyte 686의 두가지 혼동되는 내용이 병합됨

v0.27, 1998년 5월 5일

- Adaptech과 TeckRam 드라이버를 위한 새로운 정보
- SMP가 Micronics W6-LI 보드에서 동작

10. 기여한 사람들

이 HOWTO를 유지하는데 도움을 준 많은 분들께 감사:

1. Tigran A. Aivazian
2. John Aldrich
3. Niels Ammerlaan
4. H. Peter Anvin
5. Osamu Aoki
6. Guylhem Aznar
7. Ralf B押hle
8. James Beard
9. Troy Benjegerdes
10. Anton Blanchard
11. Emil Briggs
12. Robert G. Brown
13. Alexandre Charbey
14. Michael Elizabeth Chastain
15. Samuel S. Chessman
16. Alan Cox
17. Andrew Crane
18. Cort Dougan
19. Mark Duguid
20. St昱hane <65533>olivet
21. Jocelyne Erhel
22. Jay A Estabrook
23. Byron Faber

24. Mark Garlanger
25. hASCII
26. Wade Hampton
27. Andre Hedrick
28. Claus-Justus Heine
29. Benedikt Heinen
30. Florian Hinzmann
31. Moni Hollmann
32. Robert M. Hyatt
33. Jeffrey H. Ingber
34. Richard Jelinek
35. Tony Kocurko
36. Geerten Kuiper
37. Martijn Kruithof
38. Doug Ledford
39. Kumsup Lee
40. Hank Leininger
41. Ryan McCue
42. Paul Mackerras
43. Cameron MacKinnon
44. Joel Marchand
45. David Maslen
46. Chris Mauritz
47. Jean-Francois Micouleau
48. David Miller
49. Ingo Molnar
50. Ulf Nielsen
51. Jakob Oestergaard
52. C Polisher
53. Adrian Portelli
54. Matt Ranney
55. Daniel Roesen

56. Ulf Rompe
57. Jean-Michel Rouet
58. Volker Reichelt
59. Sean Reifschneider
60. Sumit Roy
61. Thomas Schenk
62. Terry Shull
63. Chris K. Skinner
64. Hans - Erik Skyttberg
65. Szakacsits Szabolcs
66. Jukka Tainio
67. Simen Timian Thoresen
68. El Warren
69. Gregory R. Warnes
70. Gero Wedemann
71. Christopher Allen Wing
72. Leonard N. Zubkoff