The Linux MIDI-HOWTO

Phil Kerr

phil (at) plus24.com

도운이

송주성

venal (at) hanmail.net

임은재

eunjea (at) kldp.org

v1.10, 2002 4월

이 문서는 리눅스에서 미디(MIDI)를 사용하는데 필요한 소프트웨어와 하드웨어, 그리고 그 절차등에 대해서 설명하였습니다.

고친 과정

고침 v1.12002년 4월 24일고친이 venal

1.1 업데이트

고침 v1.02002년 2월 1일 고친이 eunjea

1.0 번역

1. 소개

이 문서는 다음의 내용을 가지고 있다:

- 미디 인터페이스 설정
- 미디 파일 연주
- 시퀀싱 (Sequencing)
- 외부 미디 장비 조작
- 소프트웨어 사운드 신디사이저와 미디

- 미디 셈플 코드

1.1. 문서의 목표

이 문서의 초점은 미디 인터페이스를 사용한 외부 미디 기기의 사용이다.

2. 문서의 저작권

3. 최신 문서

이 문서의 공식 버전은 Linux Documentation Project <http://www.linuxdoc.org/>에서 얻을 수 있다.

가장 최신의 문서는 <http://www.midi-howto.com>에 있다.

이 문서의 마지막 번역본은 <http://kldp.org/~eunjea/>에 있다.

4. 감사의 글

이 HOWTO는 Hideki Saito가 쓴 MIDI-SB mini-HOWTO를 기초로 하였다. 문서를 리눅스 커뮤니티에 공개한 그에게 감사드린다.

이 문서에 포함된 셈플 코드들은 Linux Audio Developers (LAD) 메일링 리스트에 기초하였다. 소스를 사용하게 해 준 그들에게 감사드린다.

5. 책임의 한계

이 문서의 정보를 이용함으로써 생길 수 있는 위험은 여러분 스스로가 책임져야 한다.

본 저자는 문서의 내용이 야기할 수 있는 어떠한 결과에 대해서도 책임을 지지 않는다.

이 문서의 개념, 예제, 정보들을 이용함으로써 생기는 위험은 전적으로 여러분 자신이 책임져야 한다.

특별히 명시하지 않은 이상 모든 저작권은 그것의 소유자가 가지고 있다.

이 문서에서 사용한 용어는 상표권에 대한 특별한 허가를 받지 않았다.

단순히 상품이나 상표의 이름을 부르기 위해 사용할 때는 승인을 받을 필요가 없을 것 같아서이다.

시스템에 설치 하기 전에, 그리고 적절한 간격으로 시스템을 백업할 것을 강력히 권하는 바이다.

6. 미디의 배경(기초)

미디는 하드웨어와 소프트웨어의 프로토콜로 전자 악기의 통신에 사용된다. 1982년경에 첫번째 릴리즈가 있었으며, 사실상 전자 악기의 표준이 되었다. 이 프로토콜은 케이블과 악기를 위한 물리적인 하드웨어 변수를 정의하고, 음악, 타이밍 정보 데이터를 교환하는 통신 프로토콜들을 제공한다.

미디 이전에는 아날로그(후에는 기초적인 디지털 방식) 통신 프로토콜이 사용되었다. 1974년 이전에는 전선을 통한 전기적인 신호로 건반들을 연결했다. 1980 - 1981년 이후, 롤랜드가 DCB라는 기본 디지털 프로토콜을 만들었다. MIDI Manufacturers Association (MMA)이 표준 프로토콜과 물리적인 연결을 정의함으로써 규정에 따르는 모든 미디 기기들의 연결이 가능하게 되었다.

1985년경 부터 미디 인터페이스가 가정용 컴퓨터용으로 나왔고 곧 시퀀싱 프로그램들도 등장했다.

7. 미디 장치 설정

미디 장치는 사운드 카드에 포함되거나 별도의 장치로 존재한다. 외부 미디 인터페이스는 시리얼 포트나 USB 포트에 연결되기도 한다.

첫번째 가장 중요한것은 당신의 카드가 지원되는 것인지 확인하는 것이다!

http://www.alsa-project.org/soundcards.php3

http://www.4front-tech.com/osshw.html

리눅스 배포본의 설정을 변경하여 미디 장치들을 설치한다. 간단하게, 리눅스를 잘 지원하는 카드들은 OS를 설치할 때 설정이 될 것이다.

리눅스 커널은 OSS 드라이버와 2.5 버전에서는 ALSA 드라이버를 가지고 있다. 대부분의 배포본들은 사운드 카드 설정을 위한 툴을 제공하는데, 사운드 카드의 미디 포트를 사용한다면 이것도 설정할 수 있다. 레드헷은 sndconfig, 수세는 yast, 맨드레이크는 DrakConf를 사용한다.

만약 이런 툴들이 미디 인터페이스를 설정할 수 없거나 문제가 있다면 다음 단계를 따른다:

lsmod의 출력 결과가 미디 관련 모듈을 포함하고 있는가? 다음은 OSS 기반 시스템에서의 전형적인 출력 예이다.

```
[root@beatbox]# lsmod
Module                  Size  Used by
lockd                  32208   1  (autoclean)
sunrpc                 54640   1  (autoclean) [lockd]
autofs                  9456   2  (autoclean)
usb-ohci               12624   0  (unused)
usbcore                43632   1  [usb-ohci]
hisax                 470096   0  (autoclean) (unused)
isdn                  104208   0  (autoclean) [hisax]
slhc                    4544   0  (autoclean) [isdn]
eepro100               16144   1  (autoclean)

#---- 사운드 카드 관련 모듈
    opl3                   11376   2
    mad16                   7968   1
    ad1848                 16848   1  [mad16]
    sb                     34752   1  [mad16]
    uart401                 6384   1  [mad16 sb]
    sound                  58368   0  [opl3 mad16 ad1848 sb uart401]

soundlow                 464   0  [sound]
soundcore               2800   6  [sb sound]
nls_cp437               3952   2  (autoclean)
vfat                    9408   1  (autoclean)
fat                    30432   1  (autoclean) [vfat]
ide-scsi                7664   0
```

mpu401, olp3, uart401, oss를 보라.

USB 장치를 사용하고 있다면 USB 모듈을 확인하는 것을 잊지 말라.

sndstat 파일을 확인해 본다:

```
[root@beatbox]# cat /dev/sndstat
OSS/Free:3.8s2++-971130
Load type: Driver loaded as a module
Kernel: Linux mega 2.2.17-21mdk #1 Thu Oct 5 13:16:08 CEST 2000 i686
Config options: 0

Installed drivers:

Card config:

Audio devices:
0: MAD16 WSS (82C930) (DUPLEX)
```

```
Synth devices:
0: Yamaha OPL3

Midi devices:
0: Mad16/Mozart

Timers:
0: System clock

Mixers:
0: MAD16 WSS (82C930)
```

여기서 mad16이 미디 장치라는 것을 알 수 있으며 이것은 위의 lsmod 출력에서도 볼 수 있었다.

만약 미디와 관련된 것이 아무것도 출력되지 않는다면 /etc/modules.conf 파일을 확인해본다.

```
[root@beatbox]# cat /etc/modules.conf
alias net-pf-4 ipx
pre-install pcmcia_core /etc/rc.d/init.d/pcmcia start
alias usb-interface usb-ohci
alias parport_lowlevel parport_pc
alias block-major-11 scsi_hostadapter
pre-install plip modprobe parport_pc ; echo 7 > /proc/parport/0/irq
alias scsi_hostadapter ide-scsi
alias eth0 eepro100
alias eth1 hisax

#---- 사운드 카드
    alias sound-slot-0 mad16
    options sound dmabuf=1
    alias midi opl3
    options opl3 io=0x388
    options sb support=1
    options mad16 io=0x530 irq=5 dma=0 dma16=1 mpu_io=0x300 mpu_irq=7 joy-
stick=1
```

다음은 미디 모듈이 로드되었을때 /proc/modules 을 확인한 결과이다.

```
[root@mega /proc]# cat modules
0000-001f : dma1
0020-003f : pic1
0040-005f : timer
0060-006f : keyboard
0070-007f : rtc
0080-008f : dma page reg
00a0-00bf : pic2
00c0-00df : dma2
```

```
00f0-00ff : fpu
0170-0177 : ide1
01f0-01f7 : ide0
02f8-02ff : serial(auto)

#---- 미디 장치
    0300-0303 : MPU-401 UART

0376-0376 : ide1
0388-038b : Yamaha OPL3
03c0-03df : vga+
03f6-03f6 : ide0
03f8-03ff : serial(auto)
0530-0533 : MAD16 WSS config
0534-0537 : MAD16 WSS
de00-de1f : Intel Speedo3 Ethernet
f000-f007 : ide0
f008-f00f : ide1
```

위와 비슷한 결과가 확인되어야 한다. 그렇지 않다면 미디 드라이버를 설치해야할 필요가 있다.

만약 ALSA 0.5x 드라이버를 사용할 것 이라면 Valentijn Sessink가 쓴 Alsa-sound-mini-HOWTO 문서를 참고한다:

http://www.linuxdoc.org/HOWTO/mini/Alsa-sound.html

ALSA 드라이버 0.9x 이후 버전은 Madhu Maddy의 ALSA-HOWTO 문서를 읽어보라.

http://www.alsa-project.org/alsa-doc/alsa-howto/

7.1. ALSA 0.9 빠른 설치

아래는 ALSA 0.9 드라이버와 대부분의 미디 프로그램의 설치를 위해 필요한 라이브러리들을 설치하기 위한 예시이다.

```
[root@beatbox] # tar jxvf alsa-driver....tar.bz2
[root@beatbox] # cd alsa-driver.....
[root@beatbox] # ./configure

messages - no errors

[root@beatbox] # make

messages - no errors

[root@beatbox] # make install

messages - no errors
```

```
[root@beatbox] # ./snddevices
```

이제 /etc/modules.conf 파일을 편집하는데, 다른 배포본에서는 다른 디렉토리에 있을 수 있는 ALSA 파일을 편집한다. 파일에는 미디가 아닌 다른 장치들을 위한 정보들도 있으니 주의해서 다루기 바란다.

일반적으로 설정 파일에는 ALSA나 OSS를 위한 예전 설정들이 적혀 있을 것이므로, 지우든지 커맨트 처리를 하든지 해야 한다.

아래는 OSS 기반의 ALSA 설정을 위한 전형적인 modules.conf 파일이다.

```
alias char-major-116 snd
alias char-major-14 soundcore

alias snd-card-0 (MIDI/Sound card)
alias sound-slot-0 snd-card-0

alias sound-service-0-0 snd-mixer-oss
alias sound-service-0-1 snd-seq-oss
alias sound-service-0-3 snd-pcm-oss
alias sound-service-0-12 snd-pcm-oss
```

(MIDI/Sound card) 부분에 적합한 카드 이름을 적는다. 이 정보는 ALSA 웹사이트에서 찾을 수 있다.

이제 ALSA 기반 프로그램들을 이용하기 위한 라이브러리 파일들을 설치해야한다. 이 파일들은 alsa-libs 패키지에 있다.

alsa-drivers 와 alsa-libs 의 쌍으로 매치되는 것을 확인하라!

```
[root@beatbox] # tar jxvf alsa-libs....tar.bz2
[root@beatbox] # cd alsa-libs.....
[root@beatbox] # ./configure

messages - no errors

[root@beatbox] # make

messages - no errors

[root@beatbox] # make install
```

이제 설정이 끝났다 :)

간단한 C 프로그램으로 체크해 볼 수 있다. 이 프로그램이 컴파일되고 실행되면 시스템 설정이 제대로 된 것이다.

```
// 이렇게 컴파일하라: gcc alsatest.c -o alsatest -lasound
```

```
#include <stdio.h>
#include <alsa/asoundlib.h>

int main (int argc, char *argv[])
{
  snd_seq_t *seq_handle;

  if (snd_seq_open(&seq_handle, "hw", SND_SEQ_OPEN_DUPLEX, 0) < 0) {
    fprintf(stderr, "Error opening ALSA sequencer.\n");
    exit(1);
  }

printf("The ALSA libraries are installed.\n");
return 0;
}
```

7.2. 지연(Latency)

미디는 실시간 프로토콜로 latency는 심각한 문제가 될 수 있다.

현재 몇몇 개발자들이 이것과 관련되어 리눅스가 미디 플랫폼으로 사용하기 적합하도록 커널 해킹을 하고 있다.

기존의 리눅스 커널도 별 무리없이 사용가능 하지만 low-latency 패치를 적용하는 것이 좋다. 자세한 정보는 다음 링크에서 찾을 수 있다:

http://www.gardena.net/benno/linux/audio/

8. 소프트웨어

리눅스 상에서 미디에 대한 관심이 높아져 감에 따라 많은 소프트웨어들이 개발되고 있으므로, 여기에서 모든 미디 소프트웨어를 다루지는 못할 것이다. 하지만, 적합한 어플리케이션을 고르는데 도움이 될것 이다.

만약 여기에 나열되지 않은 미디 어플리케이션을 개발중이거나 알고 있다면 저자에게 알려주기 바란다.

8.1. 드라이버

8.1.1. Alsa 드라이버

Alsa (Advanced Linux Sound Architecture)는 몇가지 부분으로 구성되어 있다. 그 첫번째는 완전히 모듈화된 사운드 드라이버로 모듈의 자동 로딩, devfs, isapnp 설정과 아날로그/디지털 오디오 접근, 믹서, 신디사이저, DSP, MIDI 와 오디오 하드웨어의 타이밍 콘트롤러등이다. 또한 이것은 커널 수준의 시퀀서를 포함하고 있으며, OSS/Free 어플리케이션과 호환되고, 어플리케이션 (클라이언트/서버, 플러그인, PCM 공유/멀티 플랙싱, PCM metering등)을 객체지향 C 라이브러리와 ALSA 커널 드라이버의 기능들로 지원한다.

http://www.alsa-project.org/

8.1.2. OSS

OSS 사운드 카드 드라이버는 대부분의 유명한 사운드 카드를 리눅스와 FreeBSD에서 지원한다. 이 드라이버는 디지털 오디오, 미디, 신디사이저, 믹서들을 포함하고 있으며, Open Sound System API를 따른다. OSS는 사용자 편의의 GUI를 제공하여 편리하게 설치와 설정을 할 수 있다. 이것은 200 가지 이상의 사운드 카드를 지원하고 자동으로 인식한다. 또한 Plug-n-Play를 지원하고 PCI 사운드 카드와 full duplex 오디오를 지원한다.

http://www.opensound.com/

8.1.3. Notemidi

Notemidi는 노트북/랩탑 컴퓨터의 RS-232 시리얼 포트를 사용한 미디 출력 드라이버이다. 노트미디는 MIDIator MS-124W, 로랜드 사운드 캔버스 모듈, 야마하 MU-x 시리즈 사운드 모듈과 사용 할 수 있다.

http://www.michaelminn.com/linux/notemidi

8.2. 미디 파일 연주기

8.2.1. KMid

KMid는 리눅스와 FreeBSD를 위한 X11 / KDE 기반의 미디 연주기이다. 이것은 가라오케 파일의 텍스트를 표시하고 색상을 변화시켜 쉽게 따라 부를 수 있는 기능도 가지고 있다. KMid는 /dev/sequencer를 출력 장치로 지원하여 신디, AWE, FM, GUS 카드를 지원한다.

http://perso.wanadoo.es/antlarr/kmid.html

8.2.2. Pmidi

Pmidi는 간단한 명령행 프로그램으로 ALSA 시퀀서를 통해 미디 파일을 연주한다.

http://www.parabola.demon.co.uk/alsa/pmidi.html

8.2.3. TiMidity++

TiMidity는 미디 - WAVE 변환기로, Gravis Ultrasound(*)-호환 패치 파일을 사용하여 일반적인 미디 파일을 디지털 오디오 데이터로 변환해준다. 오디오 데이터는 어떠한 소리 장치를 통해서도 연주될 수 있으며, 디스크에 저장할 수도 있다. 빠른 컴퓨터에서는 실시간으로 연주가 가능하다.

http://www.goice.co.jp/member/mo/timidity/

8.3. Sequencers

8.3.1. Brahms

Brahms는 시퀀서와 음악 기보 프로그램으로 악보-, 피아노롤-, 드럼-, 마스터 트랙 편집기 등의 편집 모드를 제공한다. 미디의 수/출입 또한 지원하며 aRts-0.3.4 와 사용하여 웨이브 파일을 연주하고 소프트웨어 신디사이저로의 미디버스를 사용한 미디 이벤트 출력도 가능하다.

KooBase로도 알려져 있다.

http://brahms.sourceforge.net/

8.3.2. Anthem

Anthem은 진보적인 오픈 소스 미디 시퀀서이다. Anthem은 세련되고 객체 지향적인 노래 기법을 사용한 녹음, 편집, 연주등이 가능하다.

http://anthem.sourceforge.net/

8.3.3. Jazz++

JAZZ++는 리눅스와 윈도우를 위한 미디 시퀀서이다.

http://www.jazzware.com/cgi-bin/Zope.cgi/jazzware/

8.3.4. Linux Ultimate Music Editor

UltiMusE-LX (the Ultimate Music Editor)는 작곡 프로그램으로 당신이 음악을 쓰는데 사용될 수 있다. 당신은 마우스/키보드로 악보를 그릴 수 있으며, 대부분의 표준 기보법을 지원하고, 미디 장비 패치 변환, 이벤트, 리얼타임 클럭등을 지원한다.

http://hometown.aol.com/knudsenmj/myhomepage/umuselx.htm

8.3.5. Melys

Melys는 Advanced Linux Sound Architecture (ALSA)를 위한 미디 시퀀서 어플리케이션이다. Melys는 ALSA의 시퀀서 지원을 사용하며 GNOME 환경에서 작동한다.

http://www.parabola.demon.co.uk/melys/

8.3.6. MidiMountain Sequencer

MidiMountain은 표준 미디 파일 편집기이다. 사용하기 쉬운 인터페이스로 미디 파일을 초보자도 쉽게 편집하고 만들수 있으며, 피아노 롤에서 부터 바이너리 exclusive 메세지까지 모든 표준 미디 프로토콜의 편집이 가능하다.

http://www.midimountain.com/

8.3.7. MusE

MusE는 Qt 2.1 기반의 미디 시퀀서로 녹음과 편집이 가능하다. 시퀀서가 연주중에 이벤트를 피아노롤과 악보 편집기에서 실시간으로 수정하는 것이 가능하다. 녹음된 미디 이벤트는 그룹과 부분으로 편곡 수정기에서 다룰 수 있다.

http://muse.seh.de/

8.3.8. Rosegarden

로즈가든은 미디 시퀀서와 악보 편집기이다.

http://www.all-day-breakfast.com/rosegarden/

8.4. MIDI Trackers

8.4.1. tektracker

ttrk (tektracker)는 트랙커형의 편집기를 가지고 있는 콘솔용 미디 시퀀서이다. 뮤트 단추와 루프 제어가 가능한 연주기를 내장하고 있다. ttrk는 미디 클럭 펄스를 보내는 것과 싱크가 가능하다.

http://div8.net/ttrk/

8.4.2. ShakeTracker

ShakeTracker는 트랙커 인터페이스와 완전한 기능의 미디 시퀀서를 지향한다. 이것은 대부분의 트랙커 효과를 잘 지원하며 Impulse 트랙커를 사용하던 사용자들은 쉽게 적응할 것이다.

http://reduz.com.ar/shaketracker/

8.5. 드럼 편집기

8.5.1. DrumPatterns

드럼패턴은 오픈 소스로 웹 지향적인 드럼 패턴 생성기이다. 이것은 드럼 패턴과 진보적인 리듬의 기초등을 가르치는 목적으로 사용될 수 있다. 악보, 미디로의 출력이 가능하며 몇시간 분량의 예제를 포함하고 있다.

http://www.linux-france.org/prj/drumpatterns/index-en.html

8.6. 패치 편집기

8.6.1. JSynthLib

JSynthLib는 자바로 씌여진 오픈 소스 신디사이저 패치 편집기 / 라이브러리안이다.

8.7. 소프트웨어 신디

Spiral Synth는 폴리포닉 아날로그 신디사이저를 모델로 하여, 아날로그 신디사이저의 소리나 잡음을 만들어 낼 수 있다. 또한 당신은 이것을 사용하여 이상한 소리를 만들어 낼 수도 있다. 미디를 지원하고 표준 OSS/Free 소리 출력 (/dev/dsp)를 사용한다.

http://www.pawfal.org/SpiralSynth/

8.7.1. UltraMaster Juno-6

UltraMaster Juno-6는 Roland Juno-6 폴리포닉 신디사이저를 재현한 소프트웨어이다. 80년대초 스타일의 arpeggiator, 코러스를 실시간 64비트 내장 ULTRANALOG 웨이브 합성음으로 지원하며, 100개 이상의 자신만의 패치를 저장할 수 있다. 모든 변수들은 GUI와 외부 미디 콘트롤러를 통해 실시간으로 조절 가능하다.

http://www.ultramaster.com/juno6/index.html

8.8. 플러그인

8.8.1. xmms-midi

x11amp를 위한 미디 파일 지원 (timidity 사용). 설정 다디얼로그 창을 통해 허술한 믹서 인터페이스가 제공된다.

http://ban.joh.cam.ac.uk/~cr212/xmms-midi/

8.9. 개발

8.9.1. sfront

Sfont는 능률적인 C 프로그램으로 MPEG 4 구조를 가지는 오디오 (MP4-SA) 비트스트림을 지원한다. MP4-SA는 algorithmic 사운드의 표준으로 오디오 시그널 프로세싱 언어 (SAOL)과 악보 언어(SASL과 전통적인 미디 파일 포맷)로 이루어진다. sfont는 리눅스 상에서 실시간, low-latency 오디오 입/출력, 사운드 카드로 부터 로컬 미디 입력과 RTP 와 SIP를 사용한 네트워크상의 미디 입력을 지원한다. MP4-SA에 관한 문서를 웹 사이트에서 읽을 수 있다.

http://www.cs.berkeley.edu/~lazzaro/sa/index.html

8.9.2. jMax

jMax는 dataflow circuit을 디자인할 수 있게 해준다. 기본 데이터 형식은 integers, symbols, lists등이 사용될 수 있으며, 이것은 이벤트-드라이븐 시스템으로 미디 처리에 사용 된다. 두번째 시스템 부분(DSP)는 circuit의 시그널 흐름을 지속하게 해주며, 이것은 PCM 사운드(예: 마이크로폰, 소리 파일등)에서 매우 유용하다. 시스템은 동적 라이브러리에 의해 확장이 가능하고, 당신은 데이터 프로세서, 데이터 형, GUI 요소, 디바이스 형식등을 추가할 수 있다. 데이터 프로세서는 circuit에 따라 디자인되고 재 사용할 수 있다.

http://www.ircam.fr/equipes/temps-reel/jmax/

8.9.3. TSE3

TSE3는 C++로 씌여진 강력한 오픈 소스 시퀀서 엔진이다. 이것이 '시퀀서 엔진'인 이유는 유저 인터페이스가 없기 때문이다. 시퀀서 어플리케이션 또는 멀티미디어 패키지는 TSE3 라이브러리와 연결되어 사용자에게 미디 시퀀싱 기능을 제공할 수 있다.

http://TSE3.sourceforge.net/

8.9.4. KeyKit

KeyKit은 다중 처리 프로그래밍 언어 처리기 (awk에서 영감을 얻은)로 실시간, algorithmic 미디 조작에 쓰일 수 있다. KeyKit의 GUI는 이러한 algorithmic 음악 실험을 위한 도구를 제공하며, 다중 트랙 시퀀서와 드럼 패턴 편집기를 포함하고 있다. GUI와 도구들은 모두 KeyKit 언어 자체로 씌여져 있다. 이것은 사용자가 직접 새로운 도구를 추가 할 수 있다는 것을 의미한다.

http://nosuch.com/keykit/

9. 미디 개발

미디 어플리케이션 개발을 위해선 좋은 예제들이 종종 필요하다. For those looking to develop MIDI applications, good examples are often needed to get you started.

다음의 예제들은 LAD 메일링 리스트에 예제나 문서/참고자료로 포스팅 됐던 것들이다. The following examples were posted to the LAD mailing list in a thread about examples and documentation/tutorials.

아래는 Dr. Matthias Nagorni의 ALSA 0.9 시퀀서 클라이언트 샘플이다. 이것은 입력 가능한(writeable) 시퀀서 클라이언트를 만들고, Pitchbender-, 7-Bit Controller-, Note-event 들을 stderr를 통해 출력한다.

더 많은 예제들을 이 문서의 링크 섹션에 있는 Matthias의 사이트를 통해 입수할 수 있다.

다음과 같이 컴파일하라:

```
[phil@beatbox] $ gcc seqdemo.c -o seqdemo -lasound
```

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <alsa/asoundlib.h>

snd_seq_t *open_seq();
void midi_action(snd_seq_t *seq_handle);

snd_seq_t *open_seq() {

  snd_seq_t *seq_handle;
  int portid;

  if (snd_seq_open(&seq_handle, "hw", SND_SEQ_OPEN_DUPLEX, 0) < 0) {
    fprintf(stderr, "Error opening ALSA sequencer.\n");
    exit(1);
  }
  snd_seq_set_client_name(seq_handle, "ALSA Sequencer Demo");
  if ((portid = snd_seq_create_simple_port(seq_handle, "ALSA Se-
quencer Demo",
            SND_SEQ_PORT_CAP_WRITE|SND_SEQ_PORT_CAP_SUBS_WRITE,
            SND_SEQ_PORT_TYPE_APPLICATION)) < 0) {
    fprintf(stderr, "Error creating sequencer port.\n");
    exit(1);
  }
  return(seq_handle);
}

void midi_action(snd_seq_t *seq_handle) {
```

```
  snd_seq_event_t *ev;

  do {
    snd_seq_event_input(seq_handle, &ev);
    switch (ev->type) {
      case SND_SEQ_EVENT_CONTROLLER:
        fprintf(stderr, "Control event on Channel %2d: %5d       \r",
                ev->data.control.channel, ev->data.control.value);
        break;
      case SND_SEQ_EVENT_PITCHBEND:
        fprintf(stderr, "Pitchbender event on Channel %2d: %5d   \r",
                ev->data.control.channel, ev->data.control.value);
        break;
      case SND_SEQ_EVENT_NOTEON:
        fprintf(stderr, "Note On event on Channel %2d: %5d       \r",
                ev->data.control.channel, ev->data.note.note);
        break;
      case SND_SEQ_EVENT_NOTEOFF:
        fprintf(stderr, "Note Off event on Channel %2d: %5d      \r",
                ev->data.control.channel, ev->data.note.note);
        break;
    }
    snd_seq_free_event(ev);
  } while (snd_seq_event_input_pending(seq_handle, 0) > 0);
}

int main(int argc, char *argv[]) {

  snd_seq_t *seq_handle;
  int npfd;
  struct pollfd *pfd;

  seq_handle = open_seq();
  npfd = snd_seq_poll_descriptors_count(seq_handle, POLLIN);
  pfd = (struct pollfd *)alloca(npfd * sizeof(struct pollfd));
  snd_seq_poll_descriptors(seq_handle, pfd, npfd, POLLIN);
  while (1) {
    if (poll(pfd, npfd, 100000) > 0) {
      midi_action(seq_handle);
    }
  }
}
```

아래는 Nick Dowell 가 만든 ALSA 0.9 MIDI 리다이렉터(redirector) 이다.

```
/* ALSA Sequencer MIDI redirector.
   인풋을 미디 채널을 통해 설정된 아웃풋으로 리다이렉트한다.
   (Linux-Audio-Dev 에서 Nathaniel Virgo 에게서 요청받은 것임 ;-)
```

```
   Dr. Matthias Nagorni 의 ALSA seq 예제를 기초로 했음

   Nick Dowell <nixx@nixx.org.uk>
   */

#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <alsa/asoundlib.h>

int
main()
{
  snd_seq_t *seq_handle;
  snd_seq_event_t *ev;
  int i;
  int portid;              /* input port */
  int oportid[16];         /* output ports */
  int npfd;
  struct pollfd *pfd;
  char txt[20];

  if (snd_seq_open(&seq_handle, "hw", SND_SEQ_OPEN_DUPLEX, 0) < 0) {
    fprintf(stderr, "Error opening ALSA sequencer.\n");
    exit(1);
  }

  snd_seq_set_client_name(seq_handle, "MIDI Redirect");

  /* open one input port */
  if ((portid = snd_seq_create_simple_port
       (seq_handle, "Input",
        SND_SEQ_PORT_CAP_WRITE | SND_SEQ_PORT_CAP_SUBS_WRITE,
        SND_SEQ_PORT_TYPE_APPLICATION)) < 0) {
    fprintf(stderr, "fatal error: could not open input port.\n");
    exit(1);
  }
  /* open 16 output ports for the MIDI channels */
  for (i=0; i<16; i++){
    sprintf( txt, "MIDI Channel %d", i );
    if ((oportid[i] = snd_seq_create_simple_port
         (seq_handle, txt,
          SND_SEQ_PORT_CAP_READ | SND_SEQ_PORT_CAP_SUBS_READ,
          SND_SEQ_PORT_TYPE_APPLICATION)) < 0) {
      fprintf(stderr, "fatal error: could not open output port.\n");
      exit(1);
    }
  }
```

```
  npfd = snd_seq_poll_descriptors_count(seq_handle, POLLIN);
  pfd = (struct pollfd *)alloca(npfd * sizeof(struct pollfd));
  snd_seq_poll_descriptors(seq_handle, pfd, npfd, POLLIN);

  while (1)  /* main loop */
    if (poll(pfd, npfd, 1000000) > 0){
      do {
        snd_seq_event_input(seq_handle, &ev);
        snd_seq_ev_set_source( ev, oportid[ev->data.control.channel] );
        snd_seq_ev_set_subs( ev );
        snd_seq_ev_set_direct( ev );
        snd_seq_event_output_direct( seq_handle, ev );
        snd_seq_free_event(ev);
      } while (snd_seq_event_input_pending(seq_handle, 0) > 0);
    }
  return 0;
}
```

아래는 Craig Stuart Sapp이 작성한, OSS의 /dev/midi 인터페이스에 데이터를 보내는 예제이다.

더 많은 예제들을 이 문서의 링크 섹션에 나와 있는 Craig 의 사이트에서 구할 수 있다.

```
//
// Programmer:    Craig Stuart Sapp [craig@ccrma.stanford.edu]
// Creation Date: Mon Dec 21 18:00:42 PST 1998
// Last Modified: Mon Dec 21 18:00:42 PST 1998
// Filename:      ...linuxmidi/output/method1.c
// Syntax:        C
// $Smake:        gcc -O -o devmidiout devmidiout.c && strip devmidiout
//

#include <linux/soundcard.h>
#include <unistd.h>
#include <fcntl.h>
#include <stdio.h>

int main(void) {
   char* device =  "/dev/midi" ;
   unsigned char data[3] = {0x90, 60, 127};

   // step 1: open the OSS device for writing
   int fd = open(device, O_WRONLY, 0);
   if (fd < 0) {
      printf("Error: cannot open %s\n", device);
      exit(1);
   }

   // step 2: write the MIDI information to the OSS device
```

```
   write(fd, data, sizeof(data));

   // step 3: (optional) close the OSS device
   close(fd);

   return 0;
}
```

10. 유용한 링크

다음은 리눅스와 미디와 관련된 링크들이다.

http://www.bright.net/~dlphilp/linuxsound/ 리눅스용 사운드 & 미디 소프트웨어. 미디와 오디오 관련 정보가 풍부하다.

http://www.linuxdj.com/audio/lad/ 리눅스 오디오 개발자 메일링 리스트. 개발에 관한 논의들이 있음.

http://www.linuxdj.com/audio/quality 리눅스 오디오 음질 HOWTO. 미디 카드와 리눅스 오디오 스튜디오를 만드는데 도움이 된다.

http://www.alsa-project.org/ Alsa 프로젝트 - Audio/MIDI Driver

http://www.4front-tech.com/ OSS - Audio/MIDI Driver

http://www.gardena.net/benno/linux/audio/ Low Latancy 패치

http://ccrma-www.stanford.edu/~craig/articles/linuxmidi/ 리눅스에서의 미디 프로그래밍 소개. Craig Stuart Sapp는 OSS 드라이버를 위해 C/C++를 사용한 기초적인 미디 유틸리티들에 대해 정보를 제공하고 있다.

http://www.suse.de/~mana Dr. Matthias Nagorni의 ALSA 0.9 미디와 PCM 에 관한 C 프로그램 예제모음.

11. 피드백

이 HOWTO는 계속 진행중이다. 리눅스상에서의 새로운 또는 개발중인 미디 관련 어플리케이션에 관한 정보는 환영한다.

미디 카드는 매우 다양하며, 어떤 경우에는 설정에 어려움이 있을 수 있다. 정보를 찾아 볼 수 있는 가장 좋은 곳은 뉴스 그룹과 리눅스 오디오 메일링 리스트이다.

12. 역자 추가

역자는 MPU-401 인터페이스와 신디사이저 사운드 모듈들, 기타 프로세서등을 가지고 있는데 얼마 전까지만 해도 리눅스에서는 쓸만한 프로그램이 없어서 윈도우에서 케이크 워크나 큐베이스를 사용했었습니다. 하지만, 지금은 오픈 소스의 빠른 발전으로 이 모든것이 모두 리눅스에서도 가능하게 되었습니다. :)

한글로 번역하지 않고 그대로 영문을 옮겨 쓴 부분은 그것이 더 이해하기 쉬울 것이라고 생각했기 때문입니다. 미디와 관련 장비를 다루어 보신 분들은 다 이해하실 것이라고 생각합니다. ;)

이 문서의 마지막 번역본은 http://kldp.org/~eunjea/에 있으며 오류/수정 사항은 <eunjea (at) kldp.org>로 알려 주세요.

MIDI : Musical Instrument Digital Interface