MIDI on Linux

Matt Surico
Long Island Linux Users' Group
Tuesday, 12 April, 2005

Overview of Presentation

- What is MIDI?
- Putting together a MIDI Workstation running Linux
- Survey of Linux music tools
- Live demonstration of Rosegarden

MIDI

- Musical Instrument Digital Interface
- Standard for connecting together music-making devices
- A serial data protocol
- Origins in proprietary digital interfaces developed by manufacturers such as Roland, Oberheim, Fender Rhodes and Sequential Circuits

MIDI: What it is and is not

- MIDI ***is not*** audio data!!
- Instead, it is a series of instructions to a piece of equipment to perform actions (turn a specific note on or off, increase volume of a note, etc.)

MIDI

- First synth with MIDI: Prophet 600 in December 1982.
- First public demo of 2 synths connected together via MIDI at NAMM show, January, 1983
- Version 1.0 of MIDI standard published in October 1983
- The standard includes a **hardware specification** and a **data specification**

MIDI Hardware Specification

- Asynchronous serial interface
- 31.25 Kbaud (+/- 1%)
- Period = 320 microseconds per serial byte
- MIDI circuit is a 5 mA current loop
- Standard connector is a 5-pin DIN
- MIDI signal on pin 5, +5V bias on pin 4, pin 2 connected to cable shield (pins 1 & 3 are unused)

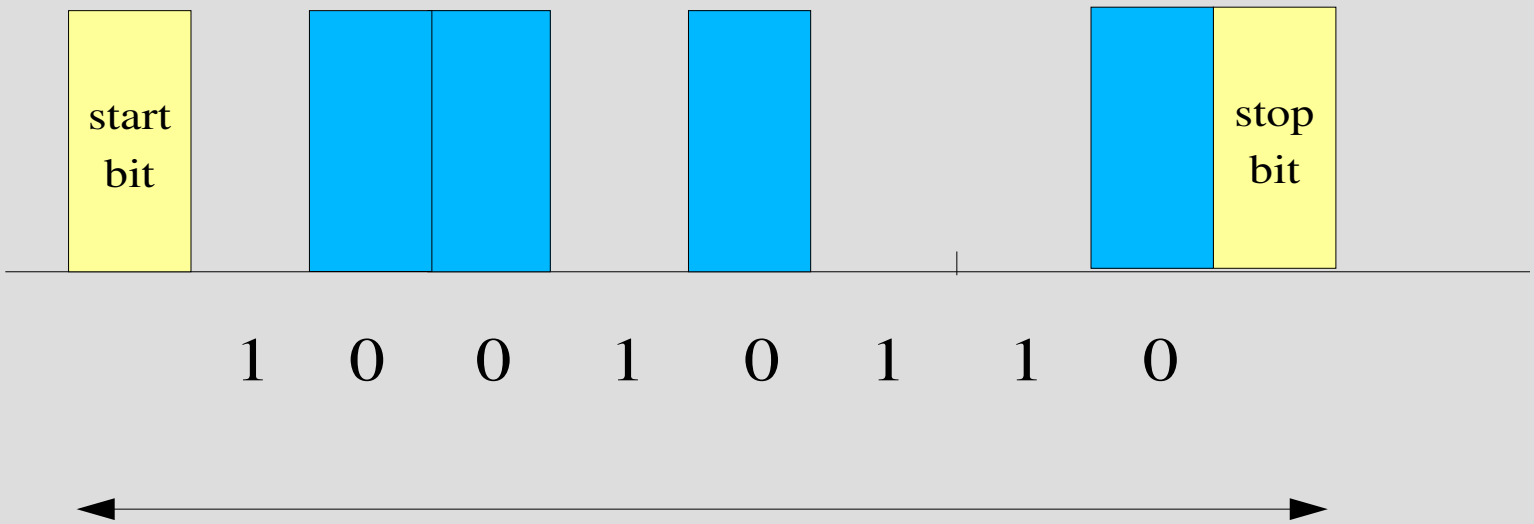
MIDI Data Specification

- 1 start bit, 8 data bits, 1 stop bit
- 8 data bits = 1 **byte**
- MIDI **messages** generally are 1, 2 or 3 bytes long
- Logical 0 is ”on”

MIDI Data Specification

- 8 data bits = 1 byte
- MIDI **messages** generally are 1, 2 or 3 bytes long
- First byte of the message is the **status byte**
- The status byte is the only MIDI message that has the 7^{th} bit set (i.e., 128 thru 255, or 0x80 thru 0xFF).
- Therefore, data bytes have values 0-127

MIDI Data Example



1/3125 second

1001 0110 = 150 decimal or 0x96

MIDI Messages

- Messages are sent to tell the equipment ”what to do”
- Remember: MIDI ***is not*** audio data!!
- Command set of MIDI includes messages to change a program (”patch”), sound a note value, apply a certain pressure to the note, and so forth

MIDI Message Example

- 10010000 01000101 01100101
- In hex, this 3-byte message is: 0x90 0x45 0x65
- First byte = status byte; any status byte starting with hex 9 is a note on
- Second byte = note number (0x45 = 69th note above the bottom C, which is F above Middle C)
- Third byte = velocity

MIDI Channels

- 16 logical channels
- Channel number encoded into bits 0-3 of the status byte
- From our previous example, 10010000 was the first byte, so 0x90 means ”status byte for channel 0”, or the first channel of 16.

Building a Linux Digital Audio Workstation

- Dell Inspiron 9200, with 1 GB RAM and a Pentium M 2.0 Ghz CPU
- 7200 RPM 60 GB hard drive
- Firewire (IEEE 1394) Port – currently poorly supported for use in music by Linux
- Two USB 2.0 ports

Goal of Linux DAW

- Create a working environment to produce music both in studio form (recording) as well as for live performances
- Use as little proprietary software as possible
- Keep costs low

Problems using Linux as a DAW

- Latency – need a tuned kernel
- Need professional quality audio managment on the PC
- Need professional sound sources
- Firewire support in its infancy under Linux

Linux Distributions for Making Music

- Most modern distributions will form a good ”backbone”
- Still need to tune the kernel
- There are suites of programs and applications packaged for running a Linux DAW
- Debian and RPM format

AGNULA

- AGNULA – A GNU/Linux Audio Distribution
- Consortium of several European universities, RedHat Linux, and the Free Software Foundation
- Completely customized distribution for making music
- Available in both Debian and RPM formats

Planet CCRMA

- Stanford University Center for Computer Research in Music and Acoustics
- Linux music applications wrapped in RPM format, distributed along with a low-latency kernel

Getting started installing the software

- Fedora Core 3 installed
- Planet CCRMA obtained from **http://ccrma.stanford.edu**
- Available for Fedora Core 1,2 and 3

Setting up

- Download/install the GPG signature for Planet CCRMA
- Install apt

```
[root@syrinx ~]# rpm --import
  http://ccrma.stanford.edu/planetccrma/RPM-GPG-
  KEY.planetccrma.txt
[root@syrinx ~]# rpm -Uvh
  http://ccrma.stanford.edu/planetccrma/apt/rpms/a
  pt-0.5.15cnc6-3.rhfc3.ccrma.i386.rpm
```

Setting up

- Do apt-get update, then install the core
- Note current kernel was 2.6.9-1.667 before the CCRMA modifications
- Can also try the ”bleeding edge kernel”

```
[root@syrinx ~]# apt-get update
[root@syrinx ~]# apt-get install planetccrma-core
[root@syrinx ~]# apt-get install planetccrma-core-
  edge
```

Contents of the CCRMA core

- Kernel 2.6.10 (bleeding edge is 2.6.11)
- ALSA goodies (tools, libraries, modules)
- GRUB bootloader configuration automagically updated

Installing CCRMA packages

- Can install RPMs one by one or ”go for broke” and install the whole kit

```
[root@syrinx ~]# apt-get install planetccrma-
  audioapps
```

What Audio/MIDI applications are available?

- Can install RPMs one by one or ”go for broke” and install the whole kit

```
[root@syrinx ~]# apt-get install planetccrma-
  audioapps
```

Components for successful Linux Audio/MIDI : ALSA

- ALSA – Advanced Linux Sound Architecture
- A huge improvement over the Open Sound System
- Potentially can support all types of audio interfaces
- Modular sound drives
- SMP and thread-safe design
- Uses alsa-lib (user space library) to make application programming easier
- Supports the older OSS API

Components for successful Linux Audio/MIDI : JACK

- JACK – Jack Audio Connection Kit
- Low latency audio server
- Designed for POSIX compliant operating systems
- Allows connectivity of a variety of applications to an audio device, and allows for sharing of audio between them
- Allows for applications to run ”normally” (as standalone apps) or within the JACK server (as a ”plugin”)

Components for successful Linux Audio/MIDI : LADSPA

- LADSPA – Linux Audio Developers Simple Plug-In API
- Allows for virtual instruments and effects (audio processors) to be used in a sequencer environment

MIDI Sequencers

- MusE
- anthem
- JAZZ++
- TekTracker
- Tutka
- seq24
- Rosegarden

MIDI Players and other Utilities

- TiMidity++
- Xpmidi
- ALSA Midi Patch Bay
- QmidiArp

Synthesizers

- FluidSynth (and Qsynth)
- Hydrogen – ultimate drum machine
- ZynAddSubFX – hybrid of additive and subtractive synthesis
- gmorgan
- amsynth – subtractive synthesis
- ALSA modular synth
- SpiralSynth

Audio Applications

- Audacity
- Ardour
- Galan
- pd
- qarecord

VSTs

- VST = Virtual Studio Technology
- A means of plugging in virtual instruments (VSTi) and audio processors to a music software environment (e.g., a sequencer)
- Developed by Steinberg
- Main platforms are Mac and PC

VSTs under Linux

- Running Windows compatible VSTs under Linux
  - vstserver project
  - fst server
  - dssi-vst wrapper
- vstserver:
  - Uses WINE (to run vstserver itself)
  - LADSPA host plugin for VST plugins (vst.so)
  - Includes standalone host application to run VSTis

Rosegarden Demo

- How does the hardware fit together?
- What software is running?
- How can we build a song snippet?