MIDI

Part of the lure of electronic music is the ability for one musician to perform highly complex compositions, or for the composer to hear his music without the need for performers at all. Splicing and digital editing allows this of course, but it is very tedious. As soon as analog synthesis became affordable, music engineers began looking for methods of automatic control for the systems.

Computer control was too expensive to contemplate in the early days (computer rental was over a million dollars), so a variety of techniques were tried: punched paper tape (Babbit's work on the RCA machine), recorded control signals (Subotnik's Butterflies) and elaborate digital sequencers (early Tangerine Dream). Some decent music was produced this way, but it was still hard work and the results were not really that complex. Electronic music that approaches orchestral music in scope had to wait for the appearance of cheap personal computers.

The first schemes (1974-84) for connecting synthesizers to computers were homemade or sold in small quantities by tiny companies. This led to a variety of systems that were mutually incompatible and so idiosyncratic that only their inventors could write software for them. The usual approach was to connect extra circuitry to the computer that either generated sounds directly or provided several channels of voltage control for modular synthesizers.

In 1983, several synthesizer manufacturers agreed on a communications protocol that would allow keyboard synthesizers to control each other (MIDI). This was very quickly picked up for computer applications, and today we have a mix and match situation, where any of several computers can be connected to one or more synthesizers, provided you have the proper software. MIDI is not perfect (the keyboard orientation and the rather slow data rate cause hassles), but it has provided an impetus for the development of software, has lowered the costs of computer assisted music, and has attracted many new musicians into the field.

The **M**usical **I**nstrument **D**ata **I**nterface specification defines both the organization of the information transmitted and the circuitry used to connect systems together. The wiring is similar to that used for microphone cables, two wires within a shield. (The MIDI connector has five pins on it, but two of those are not connected. This is done for economy: five pin DIN plugs, widely used overseas for stereo gear, cost less than the three pin model.) Exactly one input may be attached to each output. Multiples are not

allowed, but most devices have a "MIDI-THRU" output that simply passes data to the next device down the line. The basic configuration of equipment is a daisy-chain, with one master device controlling a series of slave synthesizers. An alternative arrangement is sometimes used where the data from the controller goes to a splitter box that feeds the data to several outputs, each connected to one synthesizer.

MIDI is a serial system. That means data is fed down a single wire one bit at a time. The bits are generated at the rate of 31,250 per second, but it takes ten bits to make a character and up to three characters to make a message, so it takes most of a millisecond to get anything said. As a rule, each action taken on the keyboard (such as releasing a key) generates a message. The typical message contains a channel number, a code for the key or other control affected, and descriptive data, such as key velocity. The channel number indicates which instruments are to respond to the data. There are sixteen channel numbers.

It is surprisingly easy to generate a lot of MIDI data. For instance, many keyboards have aftertouch; a feature that measures how hard you press on a key as you hold it down and feeds that information into the data stream. If you hit a chord and wiggle your wrists, you might generate several thousand bytes of data. This data may be vital, or it may be useless, depending on exactly how other instruments in the MIDI chain are voiced. When the data stream gets too full, bizarre things begin to happen. Instruments slow down, or messages can get lost. For this reason, many instruments and programs have a filter feature which removes selected types of data. You can even buy a special purpose box to do this.

Two streams of MIDI data cannot be mixed together in the simple manner two analog signals can. The group of bits that makes up a message must be kept intact or the meaning will be garbled. A device that combines MIDI signals, called a Merger, has a microprocessor in it that can recognize messages, assess a priority to them, knows how long the message should be, and prevents potential collisions by storing low priority messages until the output line is available. (This process is like switching freight trains onto a common track with out getting the cars mixed up.)

There are some other special tricks available in boxes. For instance there is a MIDI Delay which simply stores data a while before sending it along. If you connect an instrument's MIDI out to its own MIDI in through one of these, you get some complex echo effects. Another type of box is a Mapper which can change data to compensate for differences in synthesizers. For instance, instruments often vary in the number of presets the can store. If you are using a fancy machine to control several simple ones,

the fancy machine may implement all 128 preset locations, and the cheapies may only have 32. When you select preset 33 on the main synthesizer, it will send program change 33, which may have a unpredictable result on the slave. The mapper can be set to change that program 33 to anything you desire. [These features are also available as a part of better computer programs. Any synthesizer with more than 128 presets must have some sort of mapping feature.]

A type of box that is very popular is the MIDI patcher. This device has a lot of inputs and outputs, say eight of each. Controls on the box electrically switch inputs to various outputs, so you don't have to fish around for the MIDI cables to change your system configuration. A particularly intriguing feature is that a configuration can be assigned a program number, so that the patch can be controlled over the MIDI line.

Problems

The MIDI protocol is often badmouthed because the original intentions of the designers are misunderstood. The system was created to allow a simple, cheap, and universal interconnection scheme for instrument controllers and synthesizers. The specification was developed by a committee made up of representatives from several companies, and contains many compromises between various needs and opinions. The specification was inadvertently modified in translation to Japanese, but since the company that made the mistake sells more synthesizers than all other companies combined, their implementation became the standard. The MIDI committee is still active, and adds features to the specification from time to time.

Speed

The complaint heard most often about MIDI is that it is too slow. It takes one millisecond (1/1000 sec) to send the command that starts a note. This is musically imperceptible (in normal notation, MM=60,000) in simple pieces, but the delay across a twenty note chord can be noticed by a keen ear. The actual effect of this problem on the music is arguable (very few bands are together within twenty milliseconds). Probably the worst case for a performer is when the delay is unpredictably varied. The activities that generate the most frustration are elaborate computer controlled performances. The series connection MIDI system can clog up quickly when detailed control of a lot of instruments is attempted. The cure for this is to use a parallel connection scheme where the computer itself has several MIDI outputs.

Keyboardism

Another complaint is that MIDI sends the wrong information. It is clear that the standard was written with keyboard controllers in mind, and that is sensible, since the organ type keyboard is the most common controller for polyphonic single performer instruments. It is quite difficult but not impossible to design controllers with a continuous effect, such as a wind or bowed string instrument has, but the speed problem becomes extreme in such cases.

There is a proposal for a new standard, called "ZIPI" that addresses these two problems.

Stuck Notes

A perplexingly common occurrence is the stuck note. This happens because each note needs a separate message for **note on** and **note off**. If the note on is received, but the note off gets lost because of a loose cable, the note will sound forever. With many synthesizers the only way to get the note to shut up is to press many keys or turn the power off. (Most will quit if you change presets.)

Channels

The channelization scheme chosen causes a lot of confusion, but is not a problem. The channel numbers are really a tag on each command, and instruments have the option of ignoring commands that are not tagged a certain way. Difficulties arise when sending devices and receiving devices are not set to the same channel. The newer instruments can be set up to follow different channels with different voices, and this operation is often not clearly explained. The worst problem is that channel setting is usually hidden deep within an instrument's menus rather than on the front panel where it belongs.

Program Numbers

There is also some confusion about program numbers. The MIDI spec allows for 128 programs, numbered 0-127. Many manufacturers seem to feel that musicians are not ready to accept the concept of program zero, and number their buttons 1-128. Even worse are the systems that use funny numbering schemes, such as 88 meaning program 8 of bank 8.

The problems arise when one encounters a maverick corporation such as E-mu or Oberheim that calls a zero a zero; and when you need to enter program changes directly into a computer program. Of course the widespread belief that 128 programs are not enough has thrown another monkey wrench into the works as each company develops its own scheme for calling up to 1000 presets.

Modes

One of the most troublesome features is **omni mode**. A synthesizer set to omni will respond to any MIDI message, regardless of channel assignments. A typical problem this can cause is found when using Concertware: the player sends initial program changes for all eight voices at the beginning of a selection, even if there is nothing in some of the voices. A synthesizer in omni mode will respond to all of the program changes and wind up with the program number requested by voice eight. It is a good idea to check the mode of the synthesizer first off, since you don't know what the previous student was doing.(The only point to omni mode is to make synthesizers easy to demonstrate. I think it ought to be called "Salesman Mode".)

Overcoming these problems is a challenge, but is similar to challenges musicians are already familiar with. Here are a few guidelines to maintain sanity.

Use a simple configuration, and stay with it. The MIDI system is designed to have one master controller running a bunch of slaves. Mergers allow the use of two or more controllers, and switchers allow quick reconfiguration of the system, but there is usually little to be gained. The people who repatch the MIDI lines a lot are usually trying to use a black box sequencer and a keyboard as controllers at the same time.

Don't overload the system. Always filter out unnecessary information. Aftertouch, for instance should never be sent unless some device is responding to it. If you are playing with a sequenced track, the pedals are probably of interest only to the synthesizer you are playing.

Know the difference between OUT and THRU. OUT is information generated by the instrument. THRU is a copy of the input data. A few devices such as the Fadermaster provide a mix of the input and its own data at the OUT jack.

Take care of your cables. The MIDI connector is not noted for ruggedness and reliability. It is possible for a plug to look like it is in, but be loose enough to stop the data.

Read the manual. Read the Manual. READ THE MANUAL. Especially the part in the back that shows which MIDI features actually work. Pay particular attention to how to set the channel number and how to turn OMNI mode off.

Nuts And Bolts Of Midi

A MIDI message can consist of from one to several thousand bytes of data. The receiving instrument knows how many bytes to expect from the value of the first byte of the message. This byte is known as the status byte, the others are data bytes. Status bytes always have the most significant bit (msb) equal to one and data bytes have an msb of zero.[1] Because the msb of data bytes is always zero, actual values are limited to numbers less than 128. This restricts many things in the MIDI universe, such as the number of presets available.

Status bytes inform the receiver as to what to do with incoming data. Many of the commands include the channel number (0–15) as the four least significant bits of the status byte.

Commands are defined for about everything you would expect a synthesizer to do, to wit:

- **Note On**
- **Note Off**
- **Control Change**
- **Program Change**
- **Aftertouch** (for the entire keyboard, set by the heaviest push)
- **Polyphonic aftertouch** (values for each key down)
- **Pitch bend Note Messages**

Note On and Note Off

The most common status is **note on.** [The actual bit values are: 1001nnnn, where nnnn gives the channel number.] Note on is followed by two data bytes, the first is the note number, the second is key velocity. If a keyboard is not equipped to sense velocity, it is supposed to send the value 64. Not too surprisingly, there is a status called **note off**, with the same data format. Note off is actually not used very much. Instead, MIDI allows for a shorthand, known as **running status**. Once a note on[2] is received, an instrument interprets each pair of data bytes as instructions about a new note. If the velocity data is zero, the instrument performs a note off with velocity of 64.

This manner of thinking, requiring separate actions to start and stop a note, greatly simplifies the design of receiving instruments (the synthesizer does not have to keep time), but creates the potential for hung notes when a note off gets lost. The MIDI designers provided some features to compensate for this problem. There is a panic command,**all notes off**, which is generated by some keyboards and even some special effects boxes.

The note numbers start with 0 representing the lowest C. "Middle C" is supposed to be note 60. Middle C is usually known as "C4", but for some reason most manufactures call it C3.

Control Change

There is a group of commands called **control changes**, that relate to actions of things like foot pedals, modulation wheels, and sliders. Each command has two parts, defining which control to change and what to change it to. These are not very rigidly defined, so many systems allow assignment of controllers as part of preset definition. These are some of the official definitions: (numbers are actual data numbers)

- 1 Mod wheel
- 2 Breath controller
- 4 Foot controller
- 5 Portamento time
- 6 Data entry knob
- 7 Main Volume
- 8 Balance
- 10 Pan
- 11 Expression

A controller usually has a single data byte, giving a range of 0-127 as the value. This is rather coarse , so the controllers from 32 to 63 are reserved to give extra precision to those assigned from 0 to 31.

The numbers from 64 to 69 are switches or pedals:

- 64 Sustain 65
- Portamento
- 66 Sostenuto
- 67 Soft

The numbers from 98 to 101 allow extened control changes called NRPNs and RPNs.

There are some specialized control messages:

- 121 Reset all controllers
- 122 Local Control
- 123 All Notes Off
- 124 Omni Mode Off
- 125 Omni Mode On
- 126 Mono Mode On
- 127 Poly Mode On

Reset Controllers and **All Notes Off** have the obvious effects. What is not so obvious is that neither will work on a synthesizer set to Omni mode.

Local Control allows you to disconnect a keyboard from the synthesizer it is built into. The Keyboard still sends MIDI Data, and the synthesizer still responds to MIDI data, but pressing a key will not necessarily produce a sound. This is useful when you are using a computer based sequencer and want the computer to have total control of the sounds.

Controller 0 is the Bank change message. A bank change followed immediately by a program change should take you to a new sound on a different bank, but the actual use varies from instrument to instrument. details

Channel Modes

The modes take some explaining. When all this was set up, most synthesizer keyboards were monophonic, like the Moog. (Monophonic here means they would only play one note at a time.) A few instruments could play chords, these were Polyphonic. The original MIDI spec assumed you would use a MIDI channel to control each oscillator on an instrument or you would have instruments that would play chords from one channel. No one foresaw the current situation, where multitimbral synthesizers can play chords in response to several if not all of the MIDI channels.

There are four possible combinations of the mode messages:

- Omni On, Poly On or Mode 1: The synthesizer plays everything it gets.
- Omni On, Mono (Mode 2): The synthesizer plays only the most recent note.

- Omni Off, Poly (Mode 3): The synthesizer plays chords on one channel.
- Omni Off, Mono (Mode 4): The synthesizer plays the most recent note received on its base channel. It also plays the most recent note received on the next channel, and the one after that, until it's out of oscillators.

There is no message for Multi mode, so it has to be chosen from the synthesizer panel.

Program Change

The sound of a synthesizer is determined by the connections between the modules and settings of the module controls. Very few current models allow repatching of the digital subroutines that substitute for modules, but they have hundreds of controls to set. The settings are just numbers, and are stored in computer type memory. In a computer, a particular group of settings would be called a file. In synthesizers, it's a Patch, Preset, Voice, or Tone for different brands, but the official word is program. A MIDI message may call one of up to 128 of these by sending data of 0 to 127.

Most modern synthesizers have more than 128 presets. Different manufacturers and models implement a variety of ways to make these accessible by MIDI commands:

Maps On some instruments, 128 presets are called up by the Program Change commands, but you can choose ahead of time which presets are called by which command. You can assign preset 4 to Pgm Change 1, preset 205 to Pgm Change 2, and so forth. This kind of list is called a Map, and is occasionally used for other operations too.

Banks Many instruments organize the presets in groups of 64 or 128. Then you pick which group is in use at any time by pressing buttons on the instrument. At least one of the banks will be writeable, and you can copy presets into it if you want to combine some from different permanent banks[3]. Bank switching may be possible via MIDI, but the method for doing this is not standardized.

Performances Many instruments let you define a multi channel (or complex keyboard) setup that combines various presets. These Performance setups (also called Multis, or Mixes) are stored in a bank of their own. The Program Change command then picks among these. Performance setups can also have settings for processors, volume, pan, and so on.

(When an instrument is in multi channel performance mode, program changes may change the performance setup, or may change the program on a particular channel. This depends on a setting hidden somewhere in the MIDI setup of the instrument.)

Program changes have data values of 0 to 127, but are supposed to be called Programs 1-128. Many Synthesizer and Software companies do not[4], so you basically have to experiment to find out what will happen when a particular application sends a program change to a particular instrument.

Pitch Bend

Most of the wheels and knobs on a synthesizer generate control change messages, but one gets a status message of its own. This is the Pitch Bender. A dedicated message makes it possible to efficiently send a bend value of 14 bits. If you try to do pitch bend with only seven bits of precision, you either have to restrict the range or you get audible steps. Unfortunately, no manufacturer takes advantage of this.

Aftertouch

On many keyboards, if you lean into the key as you hold it down, you generate controller messages. This is a very expressive feature. On normal aftertouch (also known as Channel Pressure) the values sent correspond to the key with the most pressure.

Polyphonic Aftertouch

Polyphonic Aftertouch sends separate pressure information for each key. This is a tremendous amount of information, and only a couple of synthesizers respond to it.

System Messages

The preceding messages are **Channel Voice Messages** which apply only to instruments set to the specified channel. **System Messages** apply to all machines:

- **Song Pointer**
- **Song Select**
- **Start**
- **Stop**
- **Continue**
- **Clock**

- Midi Time Code
- Active Sensing
- System reset

With the first of these commands, several sequencers or computers can be cued to a preset point in a composition and run together. The clock command is a single byte that is "broadcast" by a master sequencer at the rate of 24 per quarter note. Sequencers can follow this clock and stay in tempo. This clock can be recorded on tape and played back with a suitable adapter. If this recording happens to be on a multi-track tape deck, complex sequences can be built up using many passes with a single synthesizer.

Song Select and **Song Pointer** cue up sequencers and drum machines, and **Start**, **Stop** and **Continue** control their operation.

An even more sophisticated synchronization system called **MIDI Time Code** is now available. In this system, time markers are recorded continuously on the tape. When the tape is played, sequencers will be automatically cued to match the tape. (This is a version of SMPTE time code, which does the same thing for video and audio editors.) Moreover, sequencers can be set to start doing their thing at arbitrary points in the composition, allowing such techniques as "slipping tracks" and eliminating the tedious process of composing long sequences of rests.

Active sensing warns an instrument if there is a serious malfunction. Once the active sensing command has been received, the instrument expects something on the MIDI line at least every 300 milliseconds (If the controller has nothing to say, it sends more active sensing messages.). If nothing is received the instrument shuts all notes off.

System Reset is supposed to return synthesizers to their power Up state. Hardly any recognize this.

The final group of commands are the **SYstem EXclusive commands**. These are commands that the manufacturer may define as they like. (Each manufacturer is assigned an ID code to prevent confusion.) The data stream may be arbitrarily long, terminating with a command known as End of Exclusive (**EOX**.) These messages are used for passing preset information, sequences, and even sound samples from one machine to another, and provide the foundation for the editor/librarian computer programs. Messages are not limited to program data; on the Yamaha instruments,

system exclusive commands can be used to control everything, including the power switch.

Extensions To Midi

The Midi Manufactures Association has not stopped their work. Since the initial definitions they have produced the following:

MIDI Time Code Described above, MTC made it possible to link MIDI systems to video and other time based operations.

Sample Dump Standard This allows samples to be transferred from one brand of sampler to another.

Standard MIDI File This one allows MIDI tracks recorded on one sequencer program to be used by another, even if it runs on a different kind of computer.

MIDI Show Control This defines ways to automate theatrical productions, synchronizing lighting effects, sound, and even fireworks.

MIDI Machine Control This allows remote control of audio and video recorders. With this and Time Code, you can run an entire studio from the computer.

And then there's....

General MIDI

General MIDI is a response to a problem that arose with the popularity of the Standard MIDI file. As composers began exchanging compositions (and selling them) in SMF format, they discovered that pieces would change when played on different synthesizers. That's because the MIDI program commands simply provide a number for a preset. What sound you get on preset four is anybody's guess.

General MIDI defines a standard list of voices. (This list is a sort of snapshot of the synthesizers that were popular in 1991. The easiest way to get it is to buy a GM compliant synthesizer.) Not only the names are standardized-- envelope times are defined so the right sort of textures are maintained. Standard MIDI also defines channel 10 as the percussion channel, and gives a map of the drum sound to associate with each note. A GM instrument may create these sounds in any manner, so there's still a lot of variation, but you no longer get a tuba when you expect a bass drum.

Most synths that support General MIDI do so by providing a bank titled GM. This is mostly a rearrangement of sounds from other banks.

General MIDI is most important in the soundcards that plug into PCs. These allow game programmers to create MIDI based scores instead of including recorded sounds for the music cuts.

General MIDI is coming to Macintosh computers as part of the expanded QuickTime system. Midi scores will be playable with no synthesizers at all!

Source: http://www.co-bw.com/Audio_MIDI_2.htm