Answers to Frequently Asked Questions about ZIPI
Author(s): Matthew Wright
Source: *Computer Music Journal*, Vol. 18, No. 4 (Winter, 1994), pp. 92-96
Published by: The MIT Press
Stable URL: http://www.jstor.org/stable/3681362
Accessed: 10-04-2017 04:06 UTC

Matthew Wright
Center for New Music and Audio Technologies (CNMAT)
Department of Music, University of California, Berkeley
1750 Arch St.
Berkeley, California 94720 USA
Matt@CNMAT.Berkeley.edu

Answers to Frequently Asked Questions about ZIPI

What Is the Equivalent of a MIDI Channel?

MIDI channels are part of MIDI's address space, so MPDL's address space mechanism fills the same role that MIDI channels fill. MPDL's address space is a three-level hierarchy, with notes, instruments, and families, each of which can be sent any control message. Exactly which of the three levels of MPDL's address space corresponds to a MIDI channel depends on how you think of MIDI channels.

In MIDI, the channel is the only possible recipient of pitch-bend and continuous controller measurements. Because of this, many people play only one note per MIDI channel, giving them complete control over that note. MIDI channels used in that way are equivalent to MPDL notes, because individual MPDL notes can be sent pitch, amplitude, and other continuous controllers.

Other users allocate one MIDI channel per synthesizer patch and play multiple notes on each MIDI channel. If they send any pitch-bend or continuous controller messages to the channel, these affect all sounding notes on that channel. In that case, you could think of a ZIPI instrument as being equivalent to a MIDI channel because you can play multiple notes inside the same instrument, and because you can send control messages at the instrument level that affect all the sounding notes of that instrument.

Remember that MPDL's address space has yet a third level, the family, which is a collection of instruments. Nothing in MIDI really corresponds to this.

Computer Music Journal, 18:4, pp. 92–96, Winter 1994

How Do You Set up Which Instrument Has Which Sound?

Typically, you send program change messages to each instrument. Then all the notes on a given instrument will be played from the same patch. Thus choosing a particular instrument on which to play a note will be the same thing as choosing a patch for that note. In multitimbral MIDI synthesizers, there is one patch per MIDI channel and it can be set either from the synthesizer's front panel or from MIDI; sending ZIPI program change messages is the equivalent.

Why Can't You Dynamically Reassign Instruments to New Families?

Remember that a ZIPI instrument is nothing more than an address; it is a place to which you send messages. You say, for example, "Make instrument 3 of family 7 louder." If instrument 3 from family 7 became instrument 6 from family 9, nothing about the new address would be the same as the old one. As an example, if I lived on 21 Main Street and then moved to 1750 Arch Street, you would not say that 21 Main Street changed into 1750 Arch Street! Instead, you'd say that *I* moved from one address to another; the two addresses in question would be the same as they always were.

Similarly, if you want to move an instrument to another family, you do not move the address. Instead, you move the parameters that are associated with that instrument, such as the patch, the pan amount, and the loudness. You send program change, pan, and loudness messages to the new in-

strument, reset the old instrument, and start sending messages to the new one.

Why Can an Instrument Be in Only One Family?

Allowing an instument to be in more than one family would raise a lot of tricky questions that would be very confusing to most users. Suppose that the piano instrument is in both the string and the percussion families. What happens to the piano if the string family is panned to the left and percussion is panned to the right? There are issues like these for every parameter; there would have to be rules about how to combine the family parameter values for an instrument in two families.

On the other hand, you can have two instruments in two different families playing the same patch. You can make instrument 88 of the string family be a piano and instrument 88 of the percussion family also be a piano. Notes played on the piano instrument in the string family would be affected by messages sent to the string family, and notes played on the piano instrument in the percussion family would be affected by messages sent to the percussion family. With this system, there is no ambiguity about combining string family parameters with percussion family parameters.

How Do I Layer Sounds from Multiple Synthesizers?

Many people use MIDI to make multiple synthesizers play the same notes in unison; this is sometimes called "layering" a sound. This is quite easy to do with ZIPI as well. If you want all of the synthesizers on a ZIPI network to play the same things in unison, the controller can send ZIPI packets that are addressed to all devices on the ring. A single articulation message would then be seen by every synthesizer, and all would play the note.

If you want only some of the synthesizers on a ZIPI network to play in unison, it is a little more tricky. One possibility is for the controller to repeat all of the control messages, sending each synthesizer its own copy of the message. With ZIPI's high bandwidth, this will not cause a performance problem in most cases. Another possibility is to send ZIPI packets addressed to all devices on the ring, but to be clever about the note address you choose. Suppose you want synthesizers A, B, and C to play a part in unison, and synthesizers D, E, and F not to play it. You could pick an instrument (e.g., instrument 6 of family 11) and send program change messages to synthesizers A, B, and C to give that instrument address the desired patches. Then you could send program change messages with the value zero for that instrument address to the other synthesizers, ensuring that notes played on that instrument will result in silence from those synthesizers. Then you could broadcast control messages to all synthesizers, on the given instrument, and only the desired synthesizers would play it.

How Can I Have a Large Virtual Orchestra Implemented on a Group of Timbre Modules?

There are two easy methods for accomplishing this. The first applies if each family of the orchestra will only be played by a single synthesizer. Simply decide ahead of time which synthesizer will play each family, and then decide on the MPDL note addresses for each instrument. For example, synthesizer A could implement the string section, with violins as instrument 1 of family 1, second violins as instrument 2, etc. Synthesizer B could implement the woodwind section in family 1 and the brass section in family 2. ZIPI packets are addressed to a unique network device, so messages for the clarinets would be addressed to synthesizer B and messages for violas would be addressed to synthesizer A. Messages for the entire string section would go to family 1 of synthesizer A.

The second method will work even if the instruments in a family are spread out over multiple synthesizers. You just have a single address space across all synthesizers, like the one shown in

Table 1. Address Space for a ZIPI Orchestra

Family 1: strings
Instrument 1: first violins
Instrument 2: second violins
Instrument 3: violas
Family 2: winds
Instrument 1: oboe
Instrument 2: flute
Instrument 3: clarinet
Family 3: brass
Instrument 1: trumpet
Instrument 2: French horn

Table 1. Each instrument might be on a different synthesizer, e.g., violas and oboes on synthesizer A, violins and trumpets on synthesizer B, and flute and horn on synthesizer C. On synthesizer A, set family 1 instrument 3 to be a viola, family 2 instrument 1 to be an oboe, and all other instruments to program change value zero, indicating silence. On synthesizer B, you would set up violins and trumpets in the right addresses and leave everything else as silence.

Now you can broadcast all ZIPI messages to all devices. A message sent to instrument 1 of family 3 will be ignored by synthesizers A and C because they have silence associated with that instrument. On synthesizer B, however, that is the address for trumpet, so a trumpet would sound. A message sent to family 2 would be remembered at the family level by both synthesizer A and synthesizer B because both implement wind instruments.

Will It Take a Long Time to Pass the Token All the Way around My 17 Timbre Modules?

No. Timbre modules will never have messages of their own to send, except in rare cases like patch dumps or network initialization, so when the token gets to a timbre module, the ZIPI hardware in the timbre module sees that it has no message to send and it declines the token immediately, passing it to the next device on the ring. This takes 1.5 clock cycles, so at ZIPI's minimum speed of 250 kBaud, it will take 6 μsec. For 17 timbre modules, the delay to pass the token around the ring is 102 μsec.

Can I Daisy-Chain Controllers in ZIPI?

There is no need to daisy-chain anything in ZIPI. Any devices plugged into the same hub are on the same ZIPI network and can send messages to one another. (If multiple hubs are connected, all devices connected to any of the hubs will be on the same ZIPI network.) It is possible to have 10 keyboard controllers on the same ZIPI network; the number of controllers has no impact on network connectivity.

How Do I Translate ZIPI to MIDI and Vice Versa?

It is not as easy as it might seem. When going from MIDI to ZIPI's MPDL, it is pretty obvious what to do with note-on, note-off, controller 7 (amplitude), and pitch-bend. However, there are a number of other issues. Should key-on velocity map to amplitude, loudness, brightness, some combination of these, or something else? What should continuous controller 4 map to? There is nothing difficult about MIDI to ZIPI translation; it is just that there are lots of ways to do it, and you must make lots of assumptions.

Going from ZIPI to MIDI is more difficult because of MIDI's limitations. How do you translate into MIDI a single note that starts at the lowest pitch on a cello and makes a glissando up three octaves? What if the other three notes played by the same cello are constant in pitch? What do you do with the output of a ZIPI guitar controller that would swamp MIDI's data rate? What if you need more than 16 channels? Which MIDI continuous controller is for even/odd harmonic balance? Translation from ZIPI to MIDI means deciding which information to throw away, so there can never be a perfect translator. Nevertheless, we imagine that people will do the best they can with this problem and that there will be commercially available translators when ZIPI gains wide acceptance.

Some Scales Have More Than 127 Notes! Why Impose That Limit?

In ZIPI's address space, saying that an instrument has 127 notes means that 127 of the tones played by a synthesizer can be controlled together by sending messages to that instrument. It does not mean that the instrument can produce only 127 distinct pitches! Pitch for a note is given by the pitch controller, which has two data bytes. It is possible to represent 65,536 different pitches in ZIPI—a resolution of better than 0.2 cents over a range of more than 10 octaves.

250 kBaud? That's Not Even 10 Times MIDI! Why So Slow?

MIDI's speed is 31.25 kBaud; ZIPI's speed is variable, starting at 250 kBaud, which is eight times faster than MIDI. The serial hardware used to run ZIPI can run at up to 20 MBaud, 640 times faster than MIDI. We chose 250 kBaud as the minimum data rate to accommodate slow computers without DMA that must service an interrupt for each byte. On a 16-MHz Motorola 68000—a popular processor for musical applications—it takes seven or eight instructions to service an interrupt, which is around 4 μsec. At 250 kBaud, a byte comes every 32 μsec, so if it takes 4 μsec to service the interrupt, that's one-eighth of the CPU time just to copy incoming ZIPI messages into a buffer. Considering that there has to be enough time left over to process the messages and turn them into music, it would be difficult to handle a faster Baud rate on low-end processors.

ZIPI uses its bandwidth more efficiently than MIDI, however, so the effective amount of information transmitted is larger by more than a factor of eight. MIDI has 10-bit bytes, only eight of which hold musical data, so it is only 80 percent efficient. One could make the case that MIDI's control byte is also overhead, in which case MIDI would be only 70 percent efficient. In ZIPI, there are seven overhead bytes per frame, but no overhead bits per byte. For large frames like the ones produced by a ZIPI guitar controller, that is 93 percent efficient. (Making this comparison on large frames is fair because only large amounts of data require large bandwidth.)

ZIPI also has provisions to transmit efficiently the kinds of data that tend to take up lots of MIDI bandwidth. For example, one expensive operation in MIDI is controlling a whole group of channels. The only way to do this is to send the same control message multiple times, once for each channel. In ZIPI, you can send one message to an entire family, and it will apply to all of the instruments of that family, avoiding duplicated control messages.

Another expensive operation in MIDI is applying some function to the value of a controller—for example, specifying vibrato by explicitly setting the pitch-bend, or making an instrument decrescendo by explicitly setting the channel's amplitude. These require a constant stream of parameter values every few msec. In ZIPI, the modulation feature allows you to say, "Start a sinusoidal variation of pitch, with a rate of 8 Hz and a depth of 10 cents," or "Start an exponential decay of loudness, to reach pianissimo in 3.2 sec," in a single message. From then on, the correct vibrato or decrescendo will occur without using any more ZIPI bandwidth.

How Could a Synthesizer Store Parameter Values for One Million Addresses?

There are 63 ZIPI families, 8,001 ZIPI instruments, and 1,016,127 ZIPI notes, for a total of 1,024,191 ZIPI addresses. Conceptually, a synthesizer must store parameter values, e.g., loudness, for each of these addresses. If a synthesizer implements pitch, articulation, and loudness, that results in 5 bytes per address. The most straightforward way to store this data, a huge array, would use more than 5 Mb of memory just to store parameter values. Obviously, that is a ridiculous amount of memory for a synthesizer to use for this purpose.

In practice, a ZIPI controller will not send information to anywhere near one million different addresses. So the synthesizer can use a clever data

structure to store and retrieve parameter values, dynamically allocating space only for those addresses that are actually used. When the controller resets addresses that it no longer plans to use (via zero-valued allocation priority messages), the synthesizer can reclaim that memory. In pathological cases, in which the controller sets values for thousands of ZIPI addresses, the synthesizer can give up on parameter values that have not been accessed recently or can use some other heuristic to decide what to throw away.

The authors will distribute to ZIPI developers C-language source code that implements these algorithms.

Why Not Save Some Wires and Encode the Data on the Clock Line?

It would be possible to encode the clock onto the data line, eliminating the need for the clock wire and associated opto-isolator. The receiving device could use a digital phase-locked loop for clock recovery. However, this would limit maximum data rates to around 1 MBaud and would reduce communications reliability, which we consider to be an unacceptable trade-off.

How Do I Turn on Omni Mode?

MIDI synthesizers can have a state called "omni mode" in which they respond to all messages on all MIDI channels. When omni mode is off, they respond to messages only on a single MIDI channel. ZIPI has no equivalent to MIDI's omni mode. It is assumed that all ZIPI synthesizers will respond to messages in all parts of ZIPI's address space, i.e., that they will always be in omni mode. (But see the answers to the questions "How Do I Layer Sounds from Multiple Synthesizers?" and "How Can I Have a Large Virtual Orchestra Implemented on a Group of Timbre Modules?" for a demonstration of how to make a ZIPI synthesizer play notes from only part of the address space.)