Understanding the Sample Dump Standard

Norman Weinberg

IN THE AUGUST 1990 ISSUE OF ***Percussive Notes***, there was a reference to the MIDI Sample Dump Standard. This portion of the MIDI specification was developed to provide a method for swapping sample data between different samplers, even if they were produced by different manufacturers. Yes, it is possible to take samples from your friend's Ensoniq EPS and move them to your own sampler.

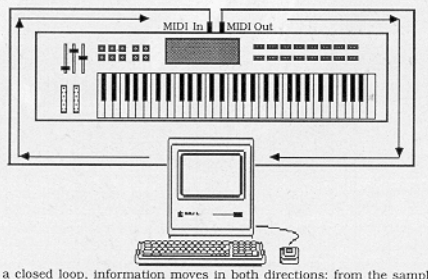
In order to accomplish this little feat of magic, you're going to need a few things. In addition to a sampler, you'll need a computer with software that is designed to work with the messages that make up the Sample Dump Standard. One method would be to hook your friend's sampler up to your computer and use the software to download a sample and store it to disk. Then connect your own sampler to the computer and upload the same sample. Easy!

Another method of swapping samples involves the use of a modem to download samples from a music bulletin board. Then you can send these samples to your own machine, granting you access to literally thousands of samples!

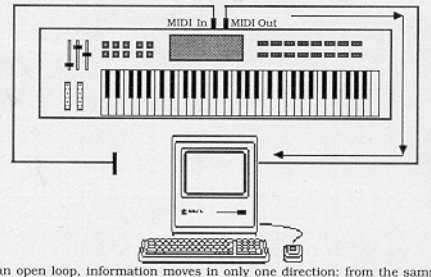
There are many different parts to the MIDI specification, and the Sample Dump Standard is a sub-set of "system exclusive" messages. In fact, they fall under another sub-set called "universal non-real-time" system exclusive messages. What this means is that these messages are "universal" (they will work for all MIDI gear that adheres to the MIDI specification) and they don't operate in "real-time"—i.e. as the music is being played.

When working with the Sample Dump Standard, two different hardware configurations are possible: a closed loop, or an open loop. Figure One shows the difference between the two loop types. In an open loop, the sending unit (often called the "source") sends messages but gets no response from the receiving unit (often called the "destination"). Communications in a closed loop system (often called "handshaking") move in both directions. Handshaking improves the speed of transmission and insures proper error recovery. In a closed loop system, the destination unit sends its own information back to the source. In other words, the two units are constantly talking to one another—as if a little conversation were going on between the two devices. In Figure One, the source is the keyboard sampler and the destination device is a computer.

FIGURE ONE



In a closed loop, information moves in both directions: from the sampler to the computer, and from the computer back to the sampler.



In an open loop, information moves in only one direction: from the sampler to the computer. The computer sends no data back to the sampler.

The Sample Dump Standard consists of seven different commands, and follows an exact series of messages. The seven commands are shown in Example No. 1 and the ordering of messages is shown in Example No. 2. All Sample Dump Standard messages begin with the system exclusive command of 1111 0000, followed by the I.D. number used for all universal non-real-time messages (7E in hex format or 126 in decimal). Next follows a byte which designates the device channel. The device channel should not to be confused with the normal MIDI *voice* channels. There are 16 MIDI *voice* channels, but there are 128 different *device* channels. In a monster electronic music system, a single command unit (most likely a computer) could handle all the non-real-time messages for 127 different MIDI devices (numbered from 0-126). If the number in this position was 127 (7F in hex), then the message would be directed to all devices connected in the system.

Following the device channel byte, there is a "Sub-I.D." byte which specifies one of the seven sample dump messages. After the Sub-I.D. comes the data.

There can be any number of data bytes contained in this section depending upon the message type. Last, but certainly not least, is the "end of exclusive" command of F7H (numbers written in hex format are usually followed by a capital "H") which lets every device in the system know that the system exclusive messages are completed.

Now, let's get down and dirty with the seven different Sub-I.D. commands that have been defined in the MIDI specification.

SAMPLE DUMP HEADER—01H

The sample dump header (See Example No. 3) is sent one time at the beginning of the sample dump. This header has only one purpose, to tell the receiving device everything it needs to know about the data that is going to follow. Using the information contained in the header, the receiving device can determine if it has enough memory to accept the dump, and learns how to interpret all of the digital bits and bytes that describe the sample. Let's break this thing apart, analyze it, and see what makes it tick.

Whenever you're reading owner's manuals or technical information about MIDI messages, you might see things that look like "##," "yy," "pp," "xx," or any other type of weird letters. These are abbreviations, and are just another way of saying that the information at this point is a variable.

Often, these letters are mnemonic to help you remember and understand what they are. The sample start point might be abbreviated as "ss," and "pp" may stand for program number. Some manufacturers, when describing how their MIDI implementation operates, move straight down the letters of the alphabet like: "jj," "kk," "ll," "mm," and so on.

F0—This command is the system exclusive status byte that lets every device in the system know that system exclusive messages are about to follow.

7E—This is the I.D. for universal non-real-time system exclusive commands (the message that signifies the Sample Dump Standard).

00-7F—This is the device channel number. It can specify any or all of 127 different devices that might be hooked up to the system.

01—This is the Sub-I.D. number which specifies the sample dump header.

##— These two bytes are the indication of the sample number. All current samplers can store several sounds, and this command specifies which sample you are going to work with. Because this is a two-byte command, this message can isolate any one of 16,384 different samples.

bb—This stands for "significant" bits, and can be any number between 08H to 1CH. Samplers use different "resolution" rates to measure the sounds they are digitizing. Resolution is one of the measurements of how accurately the sampler is doing its job.

Some of the first samplers used an eight-bit resolution, and could measure sounds by assigning a value between 0 and 255. Twelve-bit samplers can assign values from 0 to 4,095, and sixteen-bit machines can work with values between 0 to 65,535. If a sample was created using eight-bit resolution, then the number at this position would be 08H (0000 1000). A sample that had sixteen-bit resolution would have 10H in this position (0001 0000). The highest value of 1CH would be used for a sample created with 28-bit resolution.

pp—This is an abbreviation for the sample period in nanoseconds. This number is an indication of how fast the samples were made, and is often called the "sample rate". A sampler working with a rate of 40,000 samples per second, will take 40,000 sound measurements for every second of sound. The Sample Dump Standard needs three bytes to convey this information that is combined to express a value between 0

EXAMPLE NO. 1—The seven commands that make up the Sample Dump.

Hex	Binary	Decimal	Definition
01	0000 0001	001	Sample Dump Header
02	0000 0010	002	Sample Dump Data Header
03	0000 0011	003	Sample Dump Request
7C	0111 1100	124	Wait
7D	0111 1101	125	Cancel
7E	0111 1110	126	Ack (Acknowledge)
7F	0111 1111	127	Nak (Not Acknoledged)

EXAMPLE NO. 2—Basic Format for all Sample Dump Messages.

Hex	Binary	Decimal	Definition
F0	1111 0000	240	System Exclusive Status Byte
7E	0111 1110	126	I.D. for Non-Real Time Commands
00-7E	0??? ????	???	Device Channel Number
00-7E	0??? ????	???	Sub-I.D. (the actual Sample Dump Command)
00-7E	0??? ????	???	Series of Data Bytes
F7	1111 0111	247	EOX (End of Exclusive) Status Byte

EXAMPLE NO. 3—The Sample Dump Header (01H)

Hex	Binary	Decimal	Definition
F0	1111 0000	240	System Exclusive Status Byte
7E	0111 1110	126	I.D. For Non-Real Time
00-7F	0??? ????	???	Device Channel Number
01	0000 0001	001	Sub-I.D. For Sample Dump Header
00-7F			Series Of Data Bytes
##	0??? ????		Sample Number LSB
##	0??? ????		Sample Number MSB
bb	000? ????		Significant Bits
pp	0??? ????		Sample Period LSB
pp	0??? ????		Sample Period NSB
pp	0??? ????		Sample Period MSB
ll	0??? ????		Sample Length LSB
ll	0??? ????		Sample Length NSB
ll	0??? ????		Sample Length MSB
ss	0??? ????		Sustain Loop Start Point LSB
ss	0??? ????		Sustain Loop Start Point NSB
ss	0??? ????		Sustain Loop Start Point MSB
ee	0??? ????		Sustain Loop End Point LSB
ee	0??? ????		Sustain Loop End Point NSB
ee	0??? ????		Sustain Loop End Point MSB
tt	0000 000?		Loop Type
F7	1111 0111	247	EOX (End of Exclusive) Status Byte

and 2,097,151. In the example, you can see that the Least Significant Byte is sent first, the "NSB" stands for "Next Significant Byte," and the Most Significant Byte is sent last.

ll—The three data bytes in this position are the sample length "in word as". *In words* means that if the sample resolution was twelve-bit, then one word would be twelve bits long. A sampler that works at 32,000 samples per second would send 32,000 twelve-bit "words" of information for each second of sound. A sample that was three seconds in length would contain 96,000 words. Again, in order to express larger numbers, three bytes are combined.

ss—These three data bytes are combined to express the sustain loop's starting position. This is the sample number, *in words*, where the sample loop begins.

ee—These three data bytes indicate the sustain loop's end point. As in the sustain loop starting position, the end point is also expressed *in words*.

tt—Some samplers can play their loops in different ways. This is an indication of the loop type. If the number 00H is in this position (0000 0000), then the loop is forward. If this number is 01H (0000 0001), then the loop is backward. A value of 7FH at this position would indicate that the loop was turned off.

F7—The EOX status byte.

EXAMPLE NO. 4—The Sample Dump Data Packet (02H)

Hex	Binary	Decimal	Definition
F0	1111 0000	240	System Exclusive Status Byte
7E	0111 1110	126	I.D. For Non-Real Time
00-7F	0??? ????	???	Device Channel Number
02	0000 0010	002	Sub-I.D. For Sample Dump Packet
00-7F			Series Of Data Bytes
cc	0??? ????		Packet Count
dd	0??? ????		Sample Data (120 bytes)
xx	0??? ????		Checksum of previous 124 bytes
F7	1111 0111	247	EOX (End of Exclusive) Status Byte

EXAMPLE NO. 5—The Sample Dump Request (03H)

Hex	Binary	Decimal	Definition
F0	1111 0000	240	System Exclusive Status Byte
7E	0111 1110	126	I.D. For Non-Real Time
00-7F	0??? ????	???	Device Channel Number
03	0000 0010	003	Sub-I.D. For Sample Dump Request
00-7F			Series Of Data Bytes
##	0??? ????		Sample Number LSB
##	0??? ????		Sample Number MSB
F7	1000 0111	247	EOX (End of Exclusive) Status Byte

SAMPLE DUMP DATA PACKET—02H

The sample dump data packet (See Example Number 4) contains the actual sample data. After receiving the sample dump header, the destination device now knows how to interpret the flow of information. Each packet consists of 127 bytes. By using small packets, devices that don't have a great deal of internal memory can receive the data packet, process it, and then ask for more data.

F0, 7E, 00-7F—These are the same three bytes that are used for all non-real-time commands. They are the system exclusive status byte, the non-real-time Sub I.D., and the device channel number.

02—This is the Sub-I.D. number which specifies the sample dump data packet.

cc—This number can be anything between 0 and 127. It is a running count of the data packet number. As an example, data packet number one would have 0000 0001 in this position, while data packet twenty-seven would have 0001 1011. If there are more than 128 different data packets, then this running count number starts over again with zero.

dd—This is the data itself. Each data packet consists of 120 bytes of information that can work together to form either 30, 40, or 60 different "words". The exact number of words is determined by the sample format. If the sample uses eight-bit to fourteen-bit resolution, then two data bytes form a single word, and there will be 60 of these in each data packet. A sample with fifteen-bit to 21-bit resolution will require three bytes for each word (40 words per packet), and samples that use 22-bit to 28-bit resolution will need four bytes for each word (30 words per packet). Information contained in these bytes is left-justified, and any unused bits are filled with a "0".

xx—This is a checksum. Checksums are a form of error detection used by computers. This number is a result of the previous 124 bytes of information. When the source device sends this checksum, the receiving device figures out its own checksum. If the two numbers match, then everything is OK. If the numbers don't match, then the receiving device will know that an error has occurred, and will ask the source device to send the data packet again.

F7—This is the EOX status byte that must end every system exclusive message.

SAMPLE DUMP REQUEST—03H

The sample dump request (Example Number 5) simply asks the receiving device to dump the sample.

F0, 7E, 00-7F—These are the system exclusive status byte, the non-real-time Sub-I.D., and the device channel number.

03—The Sub-I.D. number for a sample dump request is 03H.

##—This is the sample number that is being requested. After receiving this command, the sampler will check to see if this sample number is valid. If it is, it will begin the sample dump. If it isn't, then the message is ignored. Because this is a two-byte number, it can specify any of 16,384 different samples.

F7—EOX status byte.

SAMPLE DUMP HANDSHAKING MESSAGES—7CH, 7DH, 7EH, 7FH

Looking at Example Number 6, you will see four different types of handshaking messages. These messages are sent from the destination device back to the source device. They are used only in a closed loop system, and serve to keep the source device informed about what is happening inside the destination.

Except for the Sub-I.D. byte, the format for all handshaking messages is the same. All messages contain the system exclusive status byte, the non-real-time I.D., the device channel number, and a data packet number. At this time, only four different types of handshaking messages have been defined.

Wait (7CH or 0111 1100)—When the destination device sends this message back to the source device, the source will wait until it receives another instruction. In English, this means "pause the dump until you receive another message." This command might be used whenever the destination device has to pause in order to transfer data from its internal memory onto a disk.

Cancel (7DH or 0111 1101)—This message will stop the dump. The packet number in this command is the packet on which the dump is stopped.

Nak (7EH or 0111 1110)—Nak is a computer term for "not acknowledged". This message tells the source device that a specific data packet was not received correctly, and to send that packet again.

Ack (7FH or 0111 1111)—The ack message means that the specific data packet *was* received correctly, and that the source device can send the next data packet. Ack is another computer term for "acknowledged".

SAMPLE DUMP COMMUNICATIONS

There is one more term that we should define before we continue with the exploration of the Sample Dump Standard. There is a period of time in which nothing happens in the source device. This is a waiting period during which the source device listens for the destination device's answer. This time period is called a "time out". If the source device hears an answer from the destination device, it will process the response and do whatever the destination device asks. If the source device doesn't hear anything during this time out, then it assumes that there is an open loop configuration, and will continue with the sample dump procedure. Some MIDI devices require the closed loop in order to operate. For example, the E-Mu Systems *SP-12* drum machine will respond with a "No Disk Connected" message if it finds an open loop and will go no further.

Now that we have taken a look at how devices communicate with non-real-time system exclusive messages (closed loop or open loop), and we have torn apart all of the currently defined Sample Dump Standard commands, let's see how this communication might actually take place. Example Number 7 is a diagram of two devices, source and destination, in the process of a sample dump.

Message One—The first message is the dump request (03H or 0000 0011). It can come from the destination unit through MIDI or it can originate from the front panel of the source device.

Message Two—After receiving the sample dump request, the source unit sends the sample dump header.

Message Three—The third message is sent by the destination unit during the source unit's time out period. This first time out period must last at least two seconds. By having two full seconds to respond, the destination unit has enough time to check whether or not it can accept the sample. The destination unit's answer can be one of three different defined messages of ACK, WAIT, or CANCEL, send some other piece of information, or not answer at all.

Message Four—The fourth message is sent by the source device after it receives and processes the third message from the destination. As Example Number 7 shows, a message of ACK or NONE will cause the source to send the first data packet. Receiving a message of CANCEL or any other message will cause the sample dump to stop, and a message of WAIT will place the source unit in a "holding stage" until it receives another message.

Message five—This message is sent from the destination unit during the source unit's time out period. After the header's time out period of two seconds, all other time out periods last only 20 ms. These messages can include all the ones that were used before (ACK, WAIT, CANCEL), and one more. The NAK message will tell the source device

EXAMPLE NO. 6—Sample Dump Handshaking Messages

WAIT

Hex	Binary	Decimal	Definition
F0	1111 0000	240	System Exclusive Status Byte
7E	0111 1110	126	I.D. For Non-Real Time
00-7F	0??? ????	???	Device Channel Number
7C	0111 1100	124	Sub-I.D. For Wait
00-7F	0??? ????	???	Packet Number
F7	1000 0111	247	EOX (End of Exclusive) Status Byte

CANCEL

Hex	Binary	Decimal	Definition
F0	1111 0000	240	System Exclusive Status Byte
7E	0111 1110	126	I.D. For Non-Real Time
00-7F	0??? ????	???	Device Channel Number
7D	0111 1101	125	Sub-I.D. For Cancel
00-7F	0??? ????	???	Packet Number
F7	1000 0111	247	EOX (End of Exclusive) Status Byte

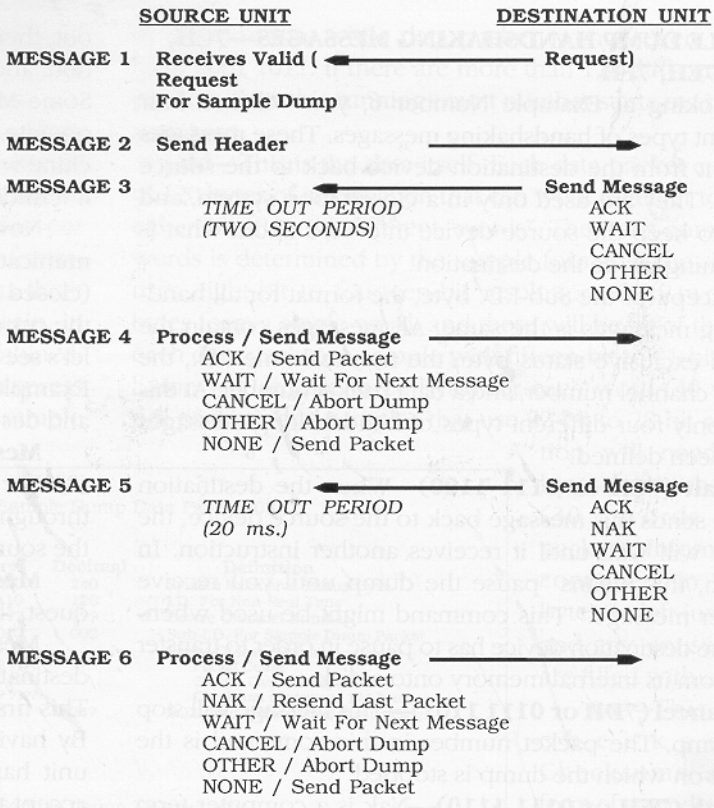
NAK

Hex	Binary	Decimal	Definition
F0	1111 0000	240	System Exclusive Status Byte
7E	0111 1110	126	I.D. For Non-Real Time
00-7F	0??? ????	???	Device Channel Number
7E	0111 1110	126	Sub-I.D. For Nak
00-7F	0??? ????	???	Packet Number
F7	1000 0111	247	EOX (End of Exclusive) Status Byte

ACK

Hex	Binary	Decimal	Definition
F0	1111 0000	240	System Exclusive Status Byte
7E	0111 1110	126	I.D. For Non-Real Time
00-7F	0??? ????	???	Device Channel Number
7F	0111 1111	127	Sub-I.D. For Ack
00-7F	0??? ????	???	Packet Number
F7	1000 0111	247	EOX (End of Exclusive) Status Byte

EXAMPLE NO. 7—Sample Dump Communications



that a specific data packet was not received correctly.

Message Six—The sixth message would be sent from the source device after it has received and processed the information in message five. If the NAK message was received, then the source device will send the same data packet again.

If there are any more data packets remaining (and, you can be sure that there will be), then message types five and six are repeated. During each time out period of 20 ms., the destination unit sends its handshaking message and the source unit then answers by sending another packet, sending the last packet again, stopping the dump, or waiting. This continues until all of the data packets are sent and received. The Sample Dump Standard suggests that there be one final handshaking message following the last data packet.

So, there you have it...all the gory details of the Sample Dump Standard. I hope that this article has taken some of the mystery out of the MIDI specification. ■

Norman Weinberg *is an Associate Professor of Music at Del Mar College in Corpus Christi, and serves as Principal Timpanist with the Corpus Christi Symphony Orchestra. He has taught at the Ruben Academy of Music in Jerusalem, The University of Missouri at Kansas City, and Indiana University.*

His last book, ***The Complete Electronic Drummer,*** *has recently been published by Modern Drummer Publications, and* ***The Last MIDI Book,*** *was published by Alexander Publications in March, 1988. Weinberg has published articles in several journals including* ***Modern Drummer, Percussive Notes, Percussive Notes Research Edition, The Instrumentalist*** *and* ***Rhythm.*** *He has compositions published by Southern Music Company. Also, he was a guest clinician at the 1988 Percussive Arts Society International Convention in San Antonio, Texas.*