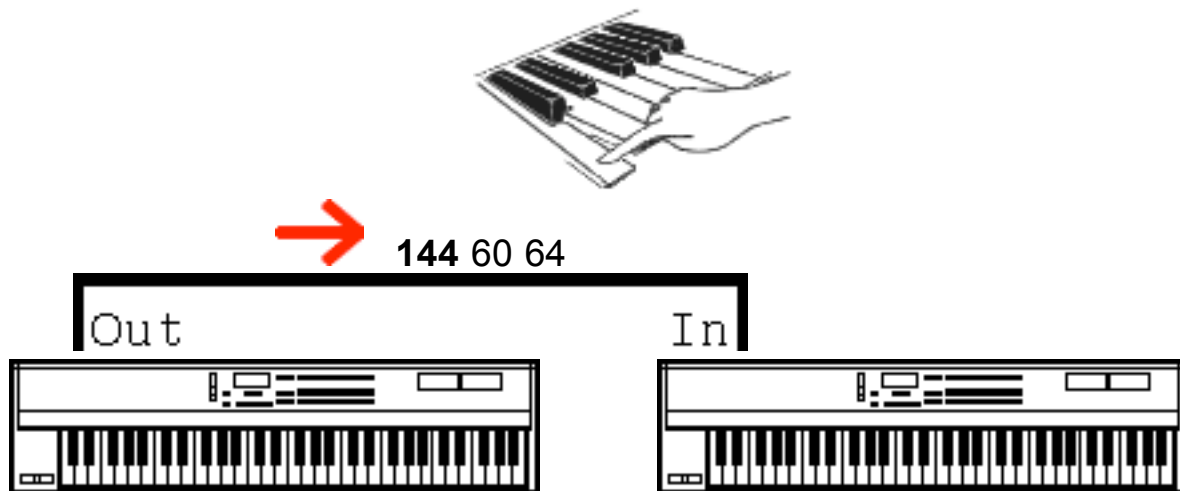


The MIDI Protocol

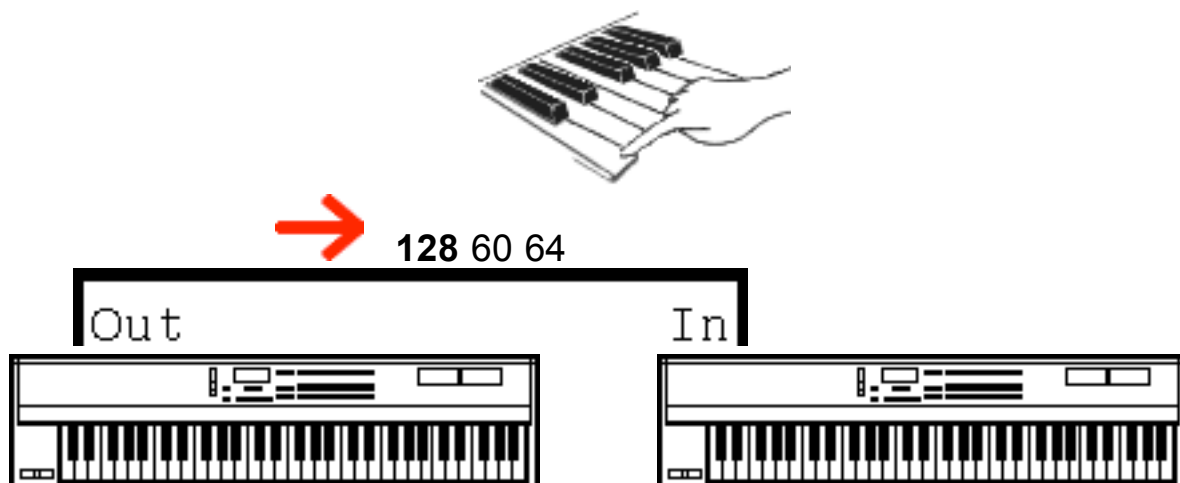
A few music manufacturers got together in 1983 and created **MIDI**, which stands for **M**usical **I**nstrument **D**igital **I**nterface.

MIDI consists of both a simple hardware interface, and a transmission protocol. The MIDI protocol defines what data is sent from one musical device to another in order to communicate a musical event. For example, when a key is pressed on a MIDI keyboard, a “Note On” message is sent from the keyboard's MIDI output and then through a MIDI cable and received by another MIDI instrument's MIDI input. This event describes which note was played and how fast (the velocity) it was pressed. The receiving device then interprets this event and plays the appropriate sound.

Pressing a key down will always send a **Note On** message out the MIDI out jack on the keyboard.

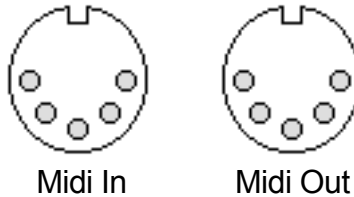


Letting your finger up will always send a **Note Off** message out the MIDI out jack on the keyboard.



MIDI Connectors

The hardware **MIDI connectors** on an instrument are **female 5-pin DIN jacks**. There are separate jacks for incoming MIDI signals and outgoing MIDI signals.



MIDI cables (with male 5-pin DIN connectors) are used to connect various instruments, controllers and/or computer interfaces together, so that they can pass MIDI signals to each other. You connect the MIDI OUT of one instrument to the MIDI IN of another instrument, and vice versa.

MIDI Command Set

MIDI is more than just **Note On** and **Note Off** messages. There's a message that tells an instrument to move its **pitch wheel** and by how much. There's a message that tells the instrument to press or release its **sustain pedal**. There's a message that tells the instrument to change its **volume** and by how much. There's a message that tells the instrument to change its **patch** (ie, maybe from an organ sound to a guitar sound). These are only a few of the many available messages in the MIDI command set. And just like with Note On and Note Off messages, these other messages are **automatically sent out the MIDI out jack when a musician plays the instrument**.

MIDI Data Format

MIDI communication is achieved through "**messages**" consisting of one or more (usually 2 or 3) bytes. A byte ranges between 0 and 255 as shown below in various representations:

decimal (base 10)
0 to 255

binary (base 2)
00000000 to 11111111

hexadecimal (base 16)
00 to FF

You should become familiar with binary and hexadecimal numbers to better understand the organization of the MIDI protocol. The reason is this: Computers and synthesizers communicate in binary and their CPU chips are optimized to process data in 8 bit units (bytes), 16 bit units (words), and 32 bit units (longs). Computer programmers use the hexadecimal system (base 16) because it represents multiples of 8 bits that the computer uses to process data, and it's easier to "think" in hex than in binary. You may encounter MIDI represented in hexadecimal from time to time

MIDI Messages

The first byte in the **message** is called the **status byte**. The remaining bytes are called **data bytes**.

	decimal	binary (8 bits = 1 byte)	hexadecimal
DATA bytes:	0 to 127	00000000 to 01111111	00 to 7F
STATUS bytes:	128 to 255	10000000 to 11111111	80 to FF

There are 8 categories of status bytes (also called status messages). Status messages are sent by playing keys, pushing buttons, or moving wheels on the synthesizer. They can also be sent by the computer. The left four bits indicate the type of message, the right four bits indicate the MIDI channel.

EXAMPLES:

This is a MIDI “Note On” message to play Middle C on MIDI channel 1

90 3C 6B (hexidecimal)

status byte: **9** = Note On (144 in decimal)
0 = MIDI Channel 1 (MIDI channels 1-16 are represented as 0-F)
data byte: **3C** = Note Number (note number 60 in decimal, or Middle C)
data byte: **6B** = Velocity Value (107 in decimal)

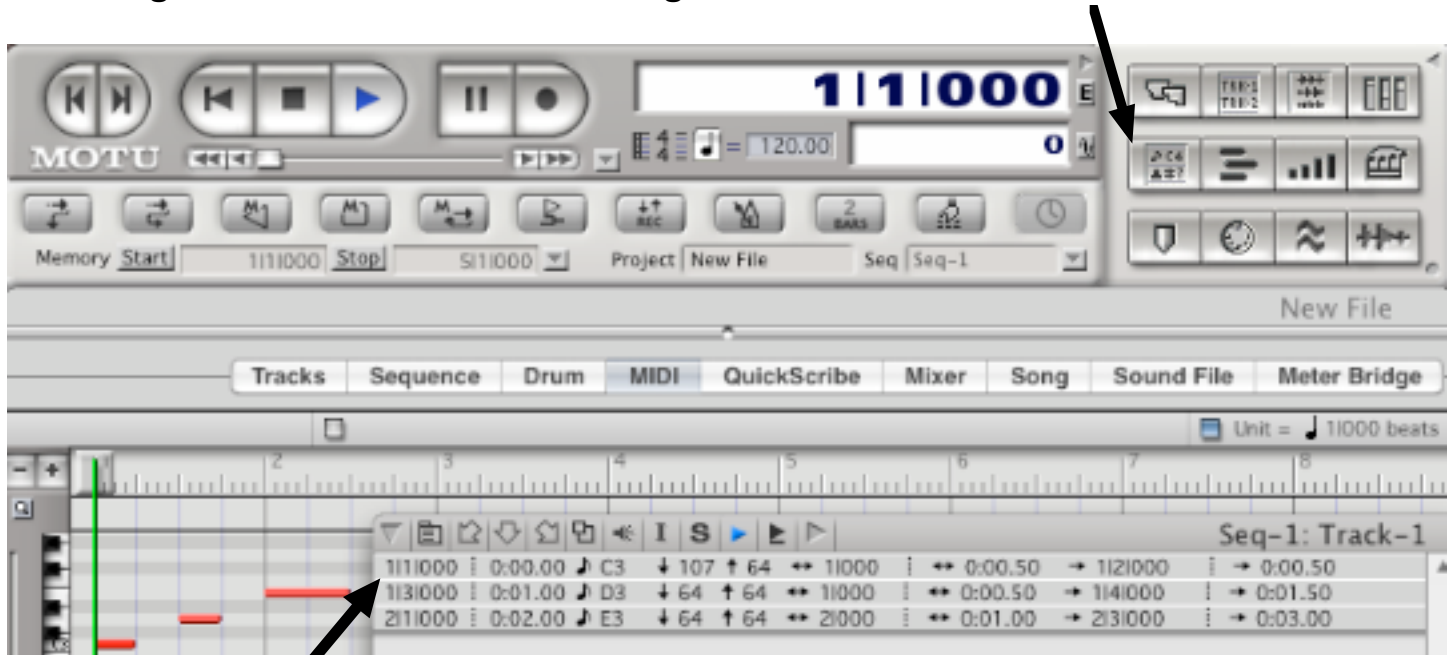
This is a MIDI “Note Off” message to turn off Middle C on MIDI channel 1

80 3C 6B (hexidecimal)

status byte: **8** = Note Off (128 in decimal)
0 = MIDI Channel 1 (MIDI channels 1-16 are represented as 0-F)
data byte: **3C** = Note Number (note number 60 in decimal / Middle C)
data byte: **40** = Velocity Value (ignored)

Note: A “Note On” with a velocity of zero also works as Note Off message

In Digital Performer you can view the MIDI data in the List Edit view by selecting a track or note and clicking on the List Edit button.



The three notes in the MIDI window are displayed in list form in the List Edit window above. This includes Note-On time (in bars|beats|tics and SMPTE), a Note-On icon, the Note Number, Velocities (for Note-On and Note-Off), duration (in bars|beats|tics and SMPTE), and Note-Off time (in bars|beats|tics and SMPTE).

MIDI command Set

Status byte	Meaning	Data byte 1	Data byte 2
80 to 8F	(Note Off / Channel #)	Note Number	velocity
90 to 9F	(Note On / Channel #)	Note Number	velocity
A0 to AF	(Aftertouch / Channel #)	Note Number	touch
B0 to BF	(Continuous controller / Chan #)	controller #	value
C0 to CF	(Patch change / Channel #)	instrument #	-
D0 to DF	(Channel Pressure / Channel #)	pressure	-
E0 to EF	(Pitch bend / Channel #)	value (lsb)	value (msb)
F0 to FF	(System Exclusive commands)		

MIDI Sequencers and Standard MIDI Files

When a key is pressed on the keyboard, a "note on" message is generated in real time. In these real time applications, there is no need for timing information to be sent along with the MIDI messages. However, if the MIDI data is to be stored as a data file, and/or edited using a sequencer, then some form of "time-stamping" for the MIDI messages is required. The Standard MIDI File specification provides a standardized method for handling time-stamped MIDI data. This standardized file format for time-stamped MIDI data allows different applications, such as sequencers, scoring packages, and multimedia presentation software, to share MIDI data files.

System Exclusive Commands: (non-musical commands)

Status byte	Meaning	Data byte
F0	start of system exclusive message	variable
F1	MIDI Time Code (Sys Common)	
F2	Song Position Pointer (Sys Common)	
F3	Song Select (Sys Common)	
F4	???	
F5	???	
F6	Tune Request (Sys Common)	
F7	end of system exclusive message	0
F8	Timing Clock (Sys Realtime)	
FA	Start (Sys Realtime)	
FB	Continue (Sys Realtime)	
FC	Stop (Sys Realtime)	
FD	???	
FE	Active Sensing (Sys Realtime)	
FF	System Reset (Sys Realtime)	

