A Guide to the MIDI Messaging Protocol in Two Pages

by Jeff Russ 2015

A complete MIDI message

Each midi message is 3 bytes long: 1 status byte followed by 2 data bytes.
The first bit of each byte is used to identify whether it's a status byte or a data byte.
All status bytes begin with 1 and all data byte begin with 0

```
1??? ????    0??? ????    0??? ????
```

There for in hex the status byte is 80 and up and the data byte is 7F and down

Status's:

For statuses 80 … EF, The second hex digit in the status message is the channel number. The 16 MIDI channels map out to 0 through F.

example: 80 … 8F are all note off commands for each of the 16 MIDI channels.

The first hex digit specifies what kind of message it it. Statuses beginning with 8 through E (7 possible values) are channel-specific messages. The first hex is the type of message and the second hex is the channel. F0 … FF map to various global and sysex messages.

Status 80 … EF

```
8_  note off
9_  note on
A_  poly aftertouch
B_  control/mode change
C_  program change
D_  channel aftertouch
E_  pitch bend
```

Status F0 … FF

```
F0  System Exclusive            F8  Timing clock
F1  MIDI Time Code Qtr. Fram    F9  Undefined (Reserved)
F2  Song Position Pointer       FA  Start
F3  Song Select (Song #)        FB  Continue
F4  Undefined   (Reserved)      FC  Stop
F5  Undefined   (Reserved)      FD  Undefined (Reserved)
F6  Tune request                FE  Active Sensing
F7  end of SysEx (EOX)          FF  System Reset
```

Close look at Data Bytes

Since data bytes waste the first bit, making it 0, we have 128 possible values
Which is 7-bit:

00 … 7F which is 0 … 127 in decimal

Close look at status 80 … AF The polyphonic messages

The first 3 types of messages are polyphonic. The second byte (1st data byte) specifies the note.
Since the first bit is wasted, we have 7 bits which gives us 128 values and 128 different notes.

Status Byte	Data Byte 1	Data Byte 2
8_ note off	note (0-127)	Velocity (0-127)
9_ note on	note (0-127)	Velocity (0-127)
A_ poly aftertouch	note (0-127)	Pressure (0-127)

Close look at status B0 … BF The control change (CC) messages

Status Byte	Data Byte 1	Data Byte 2
B_ CC message	CC# (0-127)	CC Value (0-127)

Most Defined (and therefore most used) Control Changes (CC numbers)

```
0   Bank Select (MSB)   (followed by cc32 & Program Change)
1   Modulation Wheel
4   Foot Pedal  (MSB)
6   Data Entry  (MSB    if you follow cc100=0 & cc101=0 this is pitch bend range
7   Volume      (MSB)   Note: CC7 and 11 both adjust the volume.
                        Use cc7 as you would the control on the amplifier -
                        set it and leave it at the beginning of the MIDI track
10  Pan position(MSB)
11  Expression  (MSB)   Note: CC7 and 11 both adjust the volume. Use cc11 for volume
                        changes during the track (crescendo, diminuendo,
32  Bank Select (LSB)   (see cc0)
64  Hold Pedal (on/off) Nearly every synth will react to 64 (sustain pedal)
65  Portamento (on/off)
71  Resonance           (aka Timbre)
74  Frequency Cutoff    (aka Brightness )
91  Reverb Level
93  Chorus Level
```

It's probably best not to use the group below for assigning controllers.

```
96 Data Button increment            97 Data Button decrement
98 Non-registered Parameter (LSB)   99 Non-registered Parameter (MSB)
100 Registered Parameter (LSB)      101 Registered Parameter (MSB)
```

Do not use these no matter what unless you want to invoke these functions

```
120 All Sound Off               121 All Controllers Off
122 Local Keyboard (on/off) You might actually crash your keyboard.
123 All Notes Off
```

Close look at status C0 … DF Program Change and Monophonic Aftertouch

None of these messages used the last byte aka the second data byte. Therefore they simply Identify themselves as a Program Change message (C_) or Aftertouch (D_) followed by the channel number in placed of _.

Status Byte	Data Byte 1	Data Byte 2
C_ Program Change	Program(0-127)	none
D_ Aftertouch	Pressure(0-127)	none

Close look at status E0 … EF Pitch Bend (14-bit !)

Pitch bend is unusual in that it uses both of the data bytes to get finer resolution. The first bits are of course still wasted but that leaves us with 14 bits to work which is 16,384 possible values instead of 128. If a pitch bend wheel were to span two octaves, one up and one down, we would have 682 possible values per half step which is 7 per each cent. Not bad!

Status Byte	Data Byte 1	Data Byte 2
E_ Pitch Bend	LSB (0-127)	MSB (0-127)

Overview of status 80 … EF Channel Data

Here is a full message for the a middle C in channel 1 with a velocity of 60:

90 3C 3C

Here is the note off message for that same note:

80 3C 3C

Note that our velocity is not zero as it typically is for a "note off message." This is that way for the rarely used feature of MIDI called aftertouch. This can cause hanging notes on systems that expect a velocity of 00 to turn off a note. It's advised to always send note off message as 8_ __ 00 unless you are dealing with a target that supports aftertouch.