(Unofficial) Akai Professional MidiMix Communications Protocol Guide v0.04

by Julian Ceipek

MIDI Messages from the MidiMix

During standard operation, the MidiMix sends a message every time a button is pressed, slider is moved, or dial is turned. These standard messages each consist of 3 bytes.

Continuous Controller (CC) Data Message

This message gets sent by the dials/knobs, sliders, and buttons configured to send CC messages (see Outbound Configuration Commands for how to make buttons behave like CC elements).

byte #	value (hex)	value (unsigned decimal)	description
1	0xB<Chan>	176+<Chan>	The most significant half of the byte is a constant 0xB. The least significant half indicates the channel. In MIDI programs, channels are usually identified as 1 - 16. In the message, channel 1 is identified as 0x00, or 0, and 16 as 0x0F, or 15.
2	<CC ID>		CC ID (0 - 127 or 0x00 - 0x7F).
3	0x00 - 0x7F	0 - 127	Value of the continuous element

Note On/Off (NT) Messages

This message gets sent by all buttons by default. They can be configured to send CC messages instead (see Outbound Configuration Commands for how to do this). Note that the SEND ALL button does not send any NT or CC messages of its own. Instead, pressing it sends a CC message for each dial and slider (it does not send CC messages for any buttons, even if they are configured to behave as CC elements).

byte #	value (hex)	value (unsigned decimal)	description
1	0x9<Chan> or 0x8<Chan>	144+<Chan> or 128+<Chan>	The most significant half of the byte indicates whether it is an on (0x9) or off (0x8) message. The least significant half Indicates the channel. In MIDI programs, channels are usually identified as 1 - 16. In the message, channel 1 is identified as 0x00, or 0, and 16 as 0x0F, or 15.
2	<NT ID>		NT ID (0 - 127 or 0x00 - 0x7F).
3	0x7F	127	Always 0x7F.

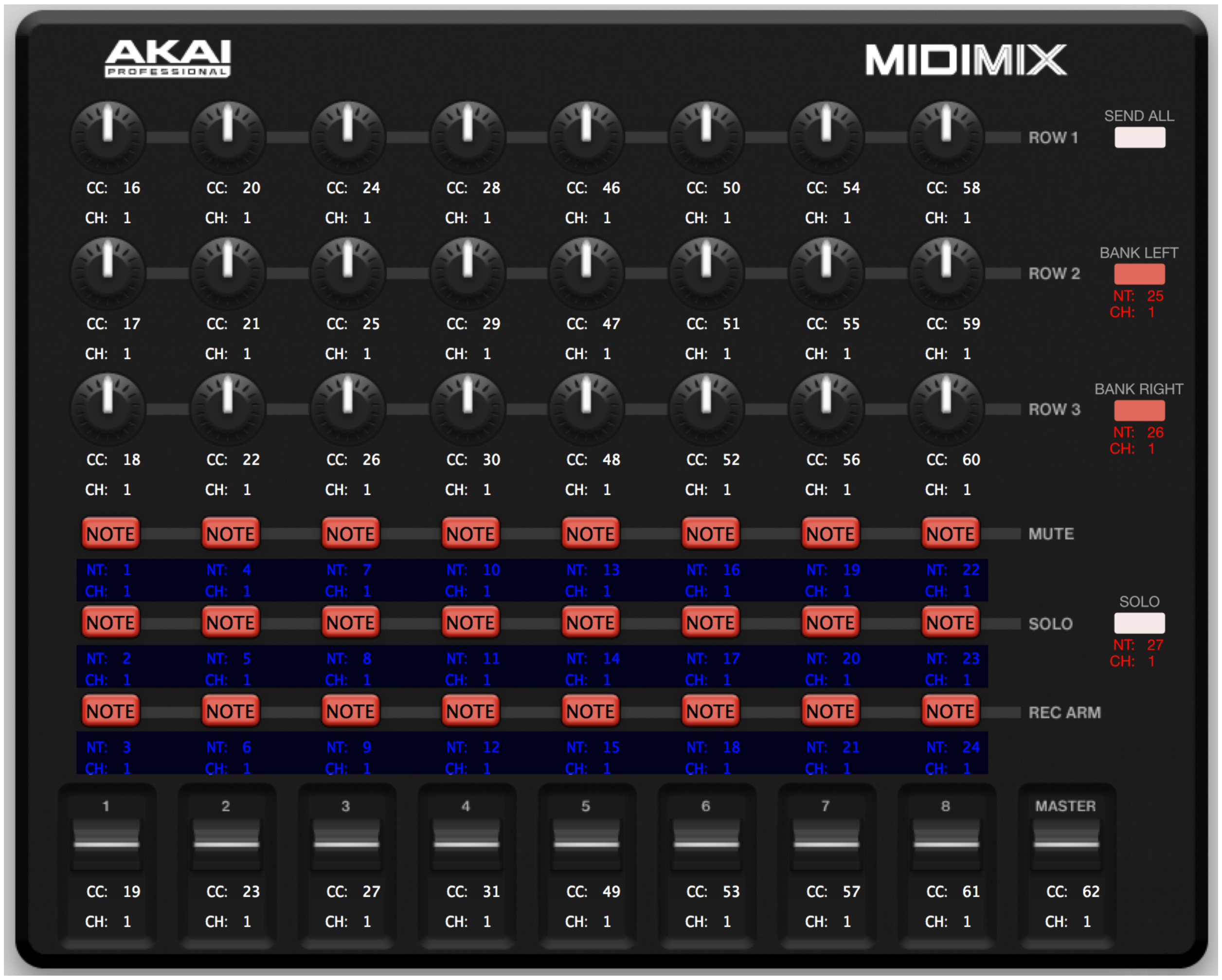
MIDI Messages to set the MidiMix Button LEDs (sent from host computer to MidiMix)

Each MUTE and REC ARM button has an LED behind it that can be turned on or off. While the SOLO button is pressed, the MUTE buttons set their LEDs based on their MUTE+SOLO identifiers (see the Element Identifiers below). The SEND ALL and SOLO buttons do not have physical LEDs according to Akai support.

byte #	value (hex)	value (unsigned decimal)	description
1	0x90	144	The LED change state command is the same as the Note On message sent by the MidiMix when a channel 0 button is pressed. Other Akai products also support using the 0x80 to turn off an LED, but the MidiMix does not appear to understand that.
2	<original NT ID>		The original NT identifier of the button, even if it has been configured to send a different NT or CC ID when pressed.
3	0x00 - 0x7F	0 - 127	Send a non-zero value to turn on the button's LED or 0x00 to turn it off (light intensity appears to be unaffected). Send a non-zero value to turn on the button's LED or 0x00 to turn it off (light intensity appears to be unaffected). Some values greater than 0x7F seem to be supported, but may be misinterpreted as other MIDI commands because anything greater than 0x7F has an active most significant bit, which is reserved in the MIDI spec.

```
const midi = require('midi');
let output = new midi.output();
output.openPort(0);
let set_bank_left_on = [0x90, 0x19, 0x7F];
output.sendMessage(set_bank_left_on);
```

Default Element Identifiers



These are the default element identifiers for the MidiMix. Element/Channel identifiers in blue can be configured but maintain their default values when sending messages to control their LEDs. Element/Channel identifiers in red cannot be configured with the messages in this document. White Element/Channel identifiers can be freely configured but cannot receive messages. The SOLO row in this diagram represents the SOLO+MUTE button combination. While the SOLO button is pressed, the MUTE button ids will temporarily change to match the SOLO+MUTE identifiers.

System Exclusive (SysEx) Messages

"Universal" MIDI messages

These messages should be understood by devices across manufacturers, not just by the MidiMix.

Device Inquiry Request/Response

Send this message to a specific device or all devices connected to your host computer to identify the MIDI device(s).

byte #	value (hex)	value (unsigned decimal)	description
1	0xF0	240	SysEx message start
2	0x7E	126	Non-Realtime message type
3	0x00 - 0x7F	0 - 127	??? Other devices seem to use this to tell devices with a specific id to respond, and use 0x7F to target all devices. The MidiMix seems to respond no matter what.
4	0x06	6	General information message sub-id
5	0x01	1	Identity request message sub-id
6	0xF7	247	SysEx message terminator

```
const midi = require('midi');
let output = new midi.output();
output.openPort(0);
let inquiry_request = [0xF0, 0x7E, 0x7F, 0x06, 0x01, 0xF7];
output.sendMessage(inquiry_request);
```

The device(s) addressed by the inquiry message will send back a response. The following is the response the MidiMix will send. Other devices will send different responses (even of different lengths), but the order of the first 6 bytes should be the same.

byte #	value (hex)	value (unsigned decimal)	description
1	0xF0	240	SysEx message start
2	0x7E	126	Non-Realtime message type
3	0x00 - 0x7F	0 - 127	Always seems to match byte #3 of the inquiry message.
4	0x06	6	General information message sub-id
5	0x02	2	Identity reply message sub-id
6	0x47	71	Akai manufacturer's ID
7	0x31	49	MidiMix model ID
8	0x00	0	(<byte 8> * 128 + <byte 9>) represents the number of bytes in the <data bytes> section. Here that is 0*128+25 = 25. The MIDI standard reserves the most significant bit in a byte, so only the lowest 7 bits of each byte can be used to indicate the number; treating the bytes as a short won't work.
9	0x19	25	
10	0x00	0	I have no idea what this data represents, and it may be different with your own MidiMix. You might be able to use this to distinguish between multiple MidiMixes connected to the same computer. Byte 14 might be the device id, based on
11	0x00	0	
12	0x00	0	
13	0x11	17	
14	0x00	0	
15	0x00	0	

16	0x00	0	the APC40 protocol, but the other bytes don't seem to match up well.
17	0x00	0	
18	0x00	0	
19	0x41	65	
20	0x00	0	
21	0x00	0	
22	0x00	0	
23	0x00	0	
24	0x00	0	
25	0x00	0	
26	0x00	0	
27	0x00	0	
28	0x00	0	
29	0x00	0	
30	0x00	0	
31	0x00	0	
32	0x00	0	
33	0x00	0	
34	0x00	0	
35	0xF7	247	SysEx message terminator

General format of Akai-specific SysEx MIDI messages

Messages of this type are specific to Akai products. The specific messages may vary for different products.

byte #	value (hex)	value (unsigned decimal)	description
1	0xF0	240	SysEx message start
2	0x47	71	Akai manufacturer's ID
3	<Device ID>		This is usually 0x00 but may change if you have multiple devices connected
4	0x31	49	Product model ID
5	<Message ID>		Message type identifier -- for example, 0x66 indicates a configuration request message
6	<Number of data bytes, MSB>		(MSB * 128 + LSB) represents the number of bytes in the <data bytes> section. The MIDI standard reserves the most significant bit in a byte, so only the lowest 7 bits of each byte can be used to indicate the number; treating the bytes as a short won't work.
7	<Number of data bytes, LSB>		
8...n-1	<data bytes>		Data bytes containing more information (depending on the message type). The number of bytes in this section is conveyed by bytes #6 and #7.
n	0xF7	247	SysEx message terminator

Outbound Configuration Commands (sent from host computer to MidiMix)

Introduction (0x60)

Send this message to tell the firmware of the MidiMix what the version number of the software application is so that the firmware could respond differently if the application gets updated (specifically used in conjunction with Ableton Live). When you open Ableton Live, it sends this message to the MidiMix with mode 1 (0x41).

byte #	value (hex)	value (unsigned decimal)	description
1	0xF0	240	SysEx message start
2	0x47	71	Akai manufacturer's ID
3	<Device ID>		This is usually 0x00 but may change if you have multiple devices connected
4	0x31	49	MidiMix model ID
5	0x60	96	Introduction message ID
6	0x00	0	Number of bytes in the data section. 0*128+4 = 4.
7	0x04	4	
8	0x40 or 0x41	64 or 65	Application/Configuration Mode identifier. Other Akai Professional products use 0x40 for mode 0 (Generic), 0x41 for mode 1 (Ableton Live), and 0x41 for mode 2 (Alternate Ableton Live Mode). I have yet to see any operational difference using different modes.
9	<Version High>		Parts of the software version. Ableton Live 9.7.4 sends 0x09,0x07,0x04.
10	<Version Low>		
11	<Bugfix Level>		
12	0xF7	247	SysEx message terminator

```
const midi = require('midi');
let output = new midi.output();
output.openPort(0);
let introduction = [0xF0, 0x47, 0x00, 0x31, 0x60, 0x00, 0x04, 0x41, 0x09, 0x07, 0x04,
0xF7];
output.sendMessage(introduction);
```

Configuration Set (0x64)

Send this message to change the MidiMix configuration. It contains information about how each button, slider, and dial is currently configured. The MidiMix Editor sends this when you click `File -> Send To Hardware`.

	byte #	**value (hex)**	**value (unsigned decimal)**	**description**
	1	0xF0	240	SysEx message start
	2	0x47	71	Akai manufacturer's ID
	3	<Device ID>		This is usually 0x00 but may change if you have multiple devices connected
	4	0x31	49	MidiMix model ID
	5	0x64	100	Configuration set message ID
	6	0x01	1	Number of bytes in the data section. 1*128+10 = 138.
	7	0x0A	10	
DATA	**8...55**	<Dial Config>		A set of two bytes for each of the dial. See Configuration Sections.
	56...73	<Slider Config>		A set of two bytes for each of the 9 sliders. See Configuration Sections.
	74...97	<MUTE Button Config>		A set of three bytes for each MUTE button. See Configuration Sections.
	98...121	<REC ARM Button Config>		A set of three bytes for each REC ARM button. See Configuration Sections.
	122...145	<MUTE+SOLO Button Config>		A set of three bytes for each MUTE button used while the SOLO button is pressed. See Configuration Sections.

	146	0xF7	247	SysEx message terminator

Request Existing Configuration (0x66)

Send this message to ask the MidiMix to send a configuration response message (0x67) that contains information about how each button, slider, and dial is currently configured. The MidiMix Editor sends this when you click `File -> Load From Hardware`.

byte #	**value (hex)**	**value (unsigned decimal)**	**description**
1	0xF0	240	SysEx message start
2	0x47	71	Akai manufacturer's ID
3	<Device ID>		This is usually 0x00 but may change if you have multiple devices connected
4	0x31	49	MidiMix model ID
5	0x66	102	Request existing configuration message ID
6	0x00	0	This should be indicating how many data bytes there are, but there are 0, so maybe it indicates something else? The MidiMix seems to respond the same way if a different value is supplied here...
7	0x01	1	
8	0xF7	247	SysEx message terminator

```
const midi = require('midi');
let output = new midi.output();
output.openPort(0);
let request_configuration = [0xF0, 0x47, 0x00, 0x31, 0x66, 0x00, 0x01, 0xF7];
output.sendMessage(request_configuration);
```

Inbound Configuration Commands (sent by MidiMix)

Configuration Response (0x67)

The MidiMix sends this in reply to a "request existing configuration message" (0x66). It contains information about how each button, slider, and dial is currently configured and is identical to the configuration set message (0x64) with the exception of the message ID. The MidiMix Editor save files (with the extension `.midimix`) use this format as well. This makes loading from a file the same as loading from a device.

	byte #	**value (hex)**	**value (unsigned decimal)**	**description**
	1	0xF0	240	SysEx message start
	2	0x47	71	Akai manufacturer's ID
	3	<Device ID>		This is usually 0x00 but may change if you have multiple devices connected
	4	0x31	49	MidiMix model ID
	5	0x67	103	Configuration response message ID
	6	0x01	1	Number of bytes in the data section. 1*128+10 = 138.
	7	0x0A	10	
DATA	**8...55**	<Dial Config>		A set of two bytes for each of the dial. See Configuration Sections.
	56...73	<Slider Config>		A set of two bytes for each of the 9 sliders. See Configuration Sections.
	74...97	<MUTE Button Config>		A set of three bytes for each MUTE button. See Configuration Sections.
	98...121	<REC ARM Button Config>		A set of three bytes for each REC ARM button. See Configuration Sections.
	122...145	<MUTE+SOLO Button Config>		A set of three bytes for each MUTE button used while the SOLO button is pressed. See Configuration Sections.

	146	0xF7	247	SysEx message terminator

Configuration Sections (included in configuration set/response messages)

<Dial Config>

The 24 MidiMix dials/knobs each have an associated continuous controller (CC) identifier ranging from 0 to 127 inclusive, and a channel ranging from 1 to 16 inclusive. The configuration response (0x67) and configuration set (0x64) messages include a dial configuration section to get/set these values.

Dial in <column, row>	**byte #**	**value (hex)**	**value (unsigned decimal)**	**description**
<1,1>	**8**	0x00 - 0x0F	0 - 15	Channel, 0 indexed -- channel 1 is represented as 0 in the message, and 16 is represented as 15.
	9	0x00 - 0x7F	0 - 127	CC identifier
<1,2>	**10**	0x00 - 0x0F	0 - 15	Channel
	11	0x00 - 0x7F	0 - 127	CC identifier
<1,3>	**12**	0x00 - 0x0F	0 - 15	Channel
	13	0x00 - 0x7F	0 - 127	CC identifier
<2,1>	**14**	0x00 - 0x0F	0 - 15	Channel
	15	0x00 - 0x7F	0 - 127	CC identifier
<2,2>...<8,2>	**16...53**	...	...	...
<1,8>	**54**	0x00 - 0x0F	0 - 15	Channel
	55	0x00 - 0x7F	0 - 127	CC identifier

<Slider Config>

Like the dials, the 9 MidiMix sliders each have an associated continuous controller (CC) identifier ranging from 0 to 127 inclusive, and a channel ranging from 1 to 16 inclusive. The configuration response (0x67) and configuration set (0x64) messages include a slider configuration section to get/set these values.

Slider in column	byte #	value (hex)	value (unsigned decimal)	description
1	**56**	0x00 - 0x0F	0 - 15	Channel, 0 indexed -- channel 1 is represented as 0 in the message, and 16 is represented as 15.
	57	0x00 - 0x7F	0 - 127	CC identifier
2	**58...69**	...	...	...
8	**70**	0x00 - 0x0F	0 - 15	Channel
	71	0x00 - 0x7F	0 - 127	CC identifier
master	**72**	0x00 - 0x0F	0 - 15	Channel
	73	0x00 - 0x7F	0 - 127	CC identifier

<MUTE Button Config>

The 8 MidiMix MUTE buttons each have an associated Note (NT) or continuous controller (CC) identifier ranging from 0 to 127 inclusive, and a channel ranging from 1 to 16 inclusive. They also have a mode (0x00 or 0x01) that configures them to behave like buttons or continuous controllers. In the latter case, they will act like sliders when reporting data (but they aren't pressure sensitive, so they'll report fully on or fully off). The configuration response (0x67) and configuration set (0x64) messages include a MUTE button configuration section to get/set these values.

Button in column	**byte #**	**value (hex)**	**value (unsigned decimal)**	**description**
1	**74**	0x00 - 0x0F	0 - 15	Channel, 0 indexed -- channel 1 is represented as 0 in the message, and 16 is represented as 15.
	75	0x00 or 0x01	0 or 1	Button mode -- 0x00 means behave like a note; 0x01 means behave like a CC
	76	0x00 - 0x7F	0 - 127	CC/NT identifier
2	**77...94**	...	...	...
8	**95**	0x00 - 0x0F	0 - 15	Channel
	96	0x00 or 0x01	0 or 1	Button mode
	97	0x00 - 0x7F	0 - 127	CC/NT identifier

<REC ARM Button Config>

The 8 MidiMix REC ARM buttons can be configured just like the MUTE buttons.

Button in column	byte #	value (hex)	value (unsigned decimal)	description
1	**98**	0x00 - 0x0F	0 - 15	Channel
	99	0x00 or 0x01	0 or 1	Button mode
	100	0x00 - 0x7F	0 - 127	CC/NT identifier
2	**101...118**	...	...	...
8	**119**	0x00 - 0x0F	0 - 15	Channel
	120	0x00 or 0x01	0 or 1	Button mode
	121	0x00 - 0x7F	0 - 127	CC/NT identifier

<MUTE+SOLO Button Config>

The 8 MidiMix MUTE buttons behave like a new row of buttons while the SOLO button is pressed. These virtual buttons can be configured just like the REC ARM and MUTE buttons.

Button in column	byte #	value (hex)	value (unsigned decimal)	description
1	**122**	0x00 - 0x0F	0 - 15	Channel
	123	0x00 or 0x01	0 or 1	Button mode
	124	0x00 - 0x7F	0 - 127	CC/NT identifier
2	**125...142**	...	...	...
8	**143**	0x00 - 0x0F	0 - 15	Channel
	144	0x00 or 0x01	0 or 1	Button mode
	145	0x00 - 0x7F	0 - 127	CC/NT identifier