MIDI Sysex Messages for the Eventide Factor series pedals

This technote assumes familiarity with the MIDI System Exclusive Message format, and the use of hex (hexadecimal) numbers.

Be aware that Sysex messages are mainly of interest to "power users," and cannot be generated by simple pedalboards. For this reason, we can offer little information or support beyond this document.

The Factor pedals accept some Standard Non-Real Time messages as well as proprietary messages using the following forms.

Standard Non-Real Time

0xF0 0x7E <channel number> <sub-ID#1> <sub-ID#2> ….. 0xF7

the 0xF0 and 0xF7 are standard MIDI for start of system exclusive, and end of system exclusive. Note that 0xF0 (for example) is a hexadecimal representation of the decimal value 240, while 0xF7 (sometimes known as EOX) is decimal 247.

<channel number> must be either 0x7F or equal to the unit's configured SYSEX ID (0 in update mode).

<sub-ID#1> <sub-ID#2> give more information about the message type.

Supported Types:

General Information Request: sub-ID#1 = 0x06
Identity Request: sub-ID#2 = 0x01

This message may be sent in either normal or update modes and causes an ***Identity Reply*** message to be sent.

General Information Request: sub-ID#1 = 0x06
Identity Reply: sub-ID#2 = 0x02

This message is the reply to an ***Identity Request.*** The format is as follows.

0xF0 0x7E <channel> 0x06 0x02 mm ff ff dd dd ss ss ss ss <text> 0xF7

0xF0 0x7E <channel>	Header
0x06	General Information (sub-ID#1)
0x02	Identity Reply (sub-ID#2)
mm	Manufacturer's ID – Eventide 0x1C
ff ff	Device Family Code (14 bits, LSB first)
dd dd	Device Family Member Code (14 bits, LSB first)
ss ss ss ss	Software Revision Level.
<text>	A long XML format string that gives further information on the unit.

Eventide proprietary messages

0xF0 <EVENTIDE> <H4000> <id> <message_code> <lots-o-bytes> 0xF7

<**EVENTIDE**> 0x1C (decimal 28)

<**H4000**> 0x70 (decimal 112).

<**id**> device id number. If this is zero, all units will listen to the message. Otherwise it must match the unit's configured SYSEX ID.

<**message_code**> tells us what message this is. The various messages are described below.

<**lots-o-bytes**> is the rest of the message. This **data** depends on the type of message. Not all messages have 'lots-o-bytes'.

With many messages, a "byte" is actually two bytes. Since MIDI allows only 7 bits of data, we split an 8 bit byte into two 4 bit *nibbles* and send the nibbles. The most significant nibble is sent first.

Message codes:

SYSEXC_OK 0x00

Data: None. This message is returned by the unit in response to assorted commands. It simply says everything was ok with that last command.
Response: None.

SYSEXC_ERROR 0x0D

Data: This message is returned by the unit in response to assorted commands. The message indicates that an error occured with the last command. <lots-o-bytes> may contain a ascii text error message (not split into nibbles)
Response: None.

SYSEXC_PROGRAM_DUMP 0x15

Data: A program in binary form, the format being the same as SYSEXC_FILES_DUMP.
Response: None.

SYSEXC_VALUE_PUT 0x2d

Data: Send a new value for a parameter.
Parameter 1 (ascii hex) is the key
Parameter 2 (ascii number) is the new value
Response: A SYSEXC_VALUE_DUMP message.

SYSEXC_VALUE_DUMP 0x2e

Data: A value of a parameter in ascii hex
Response: N/A

SYSEXC_OBJECTINFO_WANT 0x31

Data: Same as SYSEXC_VALUE_WANT. A ascii key value in hex.
Response: A SYSEXC_VALUE_DUMP message.

SYSEXC_VALUE_WANT 0x3b

Data: A ascii key value in hex.
Response: A SYSEXC_VALUE_DUMP message.

SYSEXC_TJ_PRESETS_WANT 0x48

Data: None.
Response: The unit sends a SYSEXC_TJ_PRESETS_DUMP message

SYSEXC_TJ_PRESETS_DUMP 0x49

Data: A dump of the all the presets in the unit. See Appendices for more information.
Response: n/a

SYSEXC_TJ_SYSVARS_WANT 0x4c

Data: None.
Response: The unit sends a SYSEXC_TJ_SYSVARS_DUMP message.

SYSEXC_TJ_SYSVARS_DUMP 0x4d

Data: A dump of the current system parameters. This message is to be used for backup purposes and will not be further documented.
Response: n/a

SYSEXC_TJ_PROGRAM_WANT 0x4e

Data: None.
Response: The unit sends a SYSEXC_TJ_PROGRAM_DUMP message.

SYSEXC_TJ_PROGRAM_DUMP 0x4f

Data: A dump of the current preset. See Appendices for more information.
Response: n/a

SYSEXC_TJ_ALL_WANT 0x50

Data: None.
Response: The unit sends a SYSEXC_TJ_ALL_DUMP.

SYSEXC_TJ_ALL_DUMP 0x51

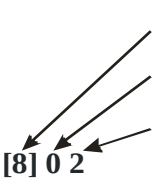
Data: The entire state of the unit (except MIDI maps). This is a single message, comprising a SYSEXC_TJ_SYSVARS_DUMP followed by a SYSEXC_TJ_PRESETS_DUMP
Response: None.

Appendix A – Program Dump Format

The following information will vary according to the software version of the pedal.

[8] 0 2
3 3f20 0 7fe0 102 3ee0 7ec2 36c0 65e0 1a80 7fe0 3ff0
3ea0 3f20 0 0 4000 7fe0 60 5dc0 0 3ee0 7fa0 0 0 0 0 7fe0 19a0 1a80 7fe0 7fe0 0 18eb 0
65000.000 65000.000 65000.000 65000.000
C_1322

The first line in this example has the following fields:



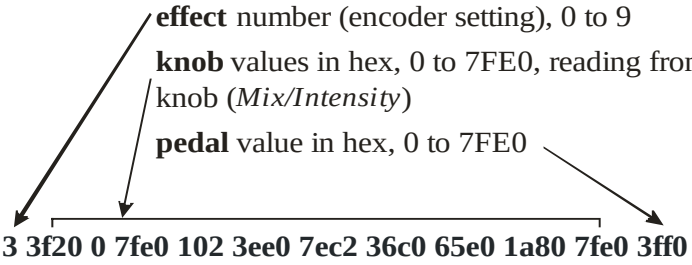
preset number, starting at 1

algorithm number. This is usually 0

dump format number. This is currently always 2. If it changes, the dump format may be different

[8] 0 2

The second line in this example has the following fields:



effect number (encoder setting), 0 to 9

knob values in hex, 0 to 7FE0, reading from bottom left knob (*Xknob*,*D-Mod*) to top left knob (*Mix/Intensity*)

pedal value in hex, 0 to 7FE0

3 3f20 0 7fe0 102 3ee0 7ec2 36c0 65e0 1a80 7fe0 3ff0

The first 20 values on the next line are in pairs, one pair for each knob and cover the pedal to knob mapping. The first value in each pair is the knob value when the pedal is at minimum (up), the second is the knob value when the pedal is at maximum (down). They are between 0 and 7FE0. They are in the same order as the knob values above.

3ea0 3f20 0 0 4000 7fe0 60 5dc0 0 3ee0 7fa0 0 0 0 0 7fe0 19a0 1a80 7fe0 7fe0 0 18eb 0

The remaining three values are:
unused - leave at 0
tempo*100
tempo **on/off** (1/0)

The next 4 values are too hard to explain and have functions that will vary from product to product. They will either be 65000.00 (inactive) or some small floating point value. Either leave them as they are or set them inactive.

65000.000 65000.000 65000.000 65000.000

The final line is a **checksum,** which is the integer sum of all the values except for the first line. This is used as an integrity check - a preset with a bad checksum will be ignored. If your arithmetic is not very good, use the value **C_XXXX** instead, and it will be assumed correct.

C_1322

Appendix B – System Variables

A large amount of information is stored in the form of *system variable*. For example, most of the information to be found under **System Mode** is stored as system variables. In addition, various control states are also indicated by system variables, such as *bypass*, MIDIclock present, etc.

Because these system variables are so all-embracing, they should be changed with care and some forethought. The user has the consolation that any damage done can always be undone by "Restoring Factory System Settings," as described in the User Manual (UM).

From the above, it will be clear that the following is intended for expert users only, and neophytes or MIDIots should read no further. Little further support is available, and these should be used entirely at your own risk.

These values are the way the unit work internally and were not designed as a user interface – we make them available in the hope that they will be either useful or interesting.

Be aware that the operation of these variables may change without notice with a different software version, and new ones may be added in the future. We do, however, undertake that none will be removed, so that you can assume that their *key values* (id numbers) will remain valid in the future. These numbers apply to all Eventide Factor, Space and H9 pedals, although some may be non-functional on different units.

Each command is a MIDI System Exclusive message of the following form:

```
F0
1C          // manuf ID - Eventide
70          // model ID – may change
00          // channel ID – may be ignored. 0 selects all channels.
xx          // command – see below
…..         // data payload – see below
F7          // EOX
```

All the following messages cause a reply message. All numbers sent in the data field are ascii *hexadecimal* numbers with optional leading zeros and without preceding "0x" or trailing"H".
Values returned are ascii *decimal* unless stated otherwise.

Commands:

either SYSEXC_OBJECTINFO_WANT (0x31) or SYSEXC_VALUE_WANT (0x3B)

data format: "XXXX" where XXXX is the key number (defined below). Example "01A2".

This command will return a SYSEXC_VALUE_DUMP (0x2E) message containing the data value according to the key.

either SYSEXC_VALUE_PUT (0x2d) or SYSEXC_USEROBJECT_SHORT (0x3C)

data format: "XXXX YY" where XXXX is the key number (defined below) and YY is a data value. Example "01A2 1B".

This command will write the data according to the key and return a SYSEXC_VALUE_DUMP (0x2E) message containing the updated value.

Key Values:

The key value determines the parameter to be read or written. If the key value is unknown or unsuitable for the command, the SYSEXC_VALUE_DUMP message will contain no data. The parameter may be read-only or read/write.

Read-only parameters:

#define TJ_ROOT_KEY 0

#define **tj_version_key** (TJ_ROOT_KEY+0)
returns a version number string of the form "OSV E.M CGV B.V S1V 8.8 S2V 3.5"

#define **tj_switch_key** (TJ_ROOT_KEY+1)
returns a binary number giving the current value of the switches in the following order:

TF/MF/PF/SPC

ki_switch1	left foot switch
ki_switch2	middle foot switch
ki_switch3	right foot switch
ki_save	always 0
ki_enc_sw	encoder switch
ki_enc_a	ignore
ki_enc_b	ignore
ki_pedal_sense	
ki_in1_sense	
ki_in2_sense	
ki_out1_sense	
ki_out2_sense	
ki_foot_tip	
ki_foot_ring	
ki_foot_sense	
ki_out_gain	

H9
ki_in1_sense
ki_in2_sense
ki_out2_sense
ki_out1_sense,
ki_codec_ovfl
ki_codec_ovfr
ki_pedal_sense
ki_dummy1
ki_presets
ki_z
ki_y
ki_x,
ki_hot
ki_switch1
ki_switch3
ki_enc_sw,
ki_foot_tip
ki_foot_ring,

Example: "0000000110110010".

#define **tj_pedal_key** (TJ_ROOT_KEY+2)
returns pedal value between 0 and 100 as a decimal integer.

Read/write parameters:

The following may be either read or written. They fall into five types – Boolean (0 or 1), Byte (0 to FF), Word (0 to FFFF), Dummy1 (variable) and Dummy2 (variable). The final key value is given by adding the parameter offset value to the ***type key value***.

Changes to the first three types below will be written into the unit's NVRAM. Values marked as "not used" are subject to future change and should not be written. Writing invalid values may result in improper system operation.

Programmers will recognize the following as enumerated types and may derive some insights as to how this stuff works.

Example: to set sp_startup_mode to sp_preset_mode:
`sp_startup_mode` has value `2`
key value is *tj_sysvars_byte_key* + `sp_startup_mode`
data value is `sp_preset_mode = 1`
send text string "202 1"

Appendix B – System Variables

Boolean Parameters:

type key value is *tj_sysvars_boolean_key* (TJ_ROOT_KEY+0x100)

0:	sp_tap_sp	not used
1:	sp_exp_sp	not used
2:	sp_bypass	set bypass state
3:	sp_kill_dry,	set killdry state
4:	sp_midi_in	MIDI in type: 0 is MIDI, 1 is USB.
5:	sp_midi_out	not used
6:	sp_midi_pgm_change	not used
7:	sp_tap_syn,	tempo enabled
8:	sp_was_catchup	not used
9:	sp_old_midiout_mode	not used
10:	sp_midiclock_enable	MIDIclock enabled
11:	sp_tx_midi_cc	transmit MIDI cc from knob changes etc
12:	sp_tx_midi_pchg	transmit MIDI program change
13:	sp_global_mix	read MIX from system
14:	sp_demo	not used
15:	sp_was_preset_spill	not used
16:	sp_global_tempo	read TEMPO from system
17:	sp_mod_display	show modulation display (MF only)
18:	sp_midiclock_out	generate MIDIclock
19:	sp_midiclock_filter	filter received MIDIclock
20:	sp_pedal_locked	pedal calibration 1:disabled, 0:enabled (not H9)
21:	sp_bluetooth_disabled	Bluetooth 1:disabled, 0:enabled (H9 only)
22:	sp_x_unlocked	X switch in unlocked (expert) mode (H9 only)
23:	sp_y_unlocked	Y switch in unlocked (expert) mode (H9 only)
24:	sp_z_unlocked	Z switch in unlocked (expert) mode (H9 only)
25:	sp_pedal_cal_disabled	pedal calibration 1:disabled, 0:enabled (H9 only)
26:	sp_alg_masks	not used
27:	sp_ui_tempo_mode	switch 3 is in TAP mode (H9 only)
28:	sp_blue_midi_enable	enable MIDI over Bluetooth (non-standard)
29:	sp_global_inswell	make input swell global
30:	sp_global_outswell	make output swell global
31:	sp_send_PC_on_rx_PC Change received	send MIDI Program Change when a MIDI Program

Byte Parameters:

type key value is *tj_sysvars_byte_key* (TJ_ROOT_KEY+0x200)

0: sp_bypass_mode
- 0: sp_bypass_mode_dsp
- 1: sp_bypass_mode_relay
- 2: sp_bypass_mode_dsp_plus_delay
- 3: sp_bypass_mode_mute

1: sp_input not used

Appendix B – System Variables

2: sp_startup_mode
    0: sp_effect_mode
    1: sp_preset_mode
3: sp_midi_rx_channel MIDI receive channel 0 to 15
4: sp_sysex_id MIDI sysex ID number 0 to 255
5: sp_midi_pgm_offset not used
6: sp_num_banks_lo minimum bank (TF/MF/PF) or preset (SPC,H9)
7: sp_spare_byte, not used
8: sp_midi_tx_channel MIDI transmit channel 0..15
9: sp_dump_type
    0: sp_dump_all
    1: sp_dump_current_preset
    2: sp_dump_presets
    3: sp_dump_system
10: sp_num_banks maximum bank (TF/MF/PF) or preset (SPC,H9) (1 less than actual value).
11: sp_tap_average tap tempo averaging length (number of taps)

The following are source values for remote control

0: sps_off
1: sps_tip AUX TIP
2: sps_rng AUX RING
3: sps_tip_ring AUX TIP+RING
4: sps_bnd MIDI Pitch Bend
5: sps_cc0 CC #0
6: sps_cc1, etc CC #1
……..
133: sps_tip_momentary
134: sps_rng_momentary
135: sps_tip_ring_momentary

The following are destinations for MIDI remote control

12: sp_byp_src bypass
13: sp_bankup_src increment bank
14: sp_bankdown_src decrement bank
15: sp_tap_src tap
16: sp_pfs_src performance switch
17: sp_snap_src not used

18: sp_kb1_src knob 1
19: sp_kb2_src,
20: sp_kb3_src
21: sp_kb4_src
22: sp_kb5_src
23: sp_kb6_src
24: sp_kb7_src
25: sp_kb8_src

26: sp_kb9_src
27: sp_kb10_src, knob 10

28: sp_mode_src preset/perf mode (not SPC,H9)
29: sp_lp1_src looper record (TF,H9 only)
30: sp_lp2_src looper play (TF,H9 only)
31: sp_lp3_src looper stop (TF,H9 only)
32: sp_pedal_src expression pedal
33: sp_not_byp_src active

The following are destinations for AUX remote control
34: sp_byp_src2
35: sp_bankup_src2
36: sp_bankdown_src2
37: sp_tap_src2
38: sp_pfs_src2
39: sp_snap_src2
40: sp_kb1_src2
41: sp_kb2_src2
42: sp_kb3_src2
43: sp_kb4_src2
44: sp_kb5_src2
45: sp_kb6_src2
46: sp_kb7_src2
47: sp_kb8_src2
48: sp_kb9_src2
49: sp_kb10_src2
50: sp_mode_src2
51: sp_lp1_src2
52: sp_lp2_src2
53: sp_lp3_src2

MIDI alternate mode
54: sp_alt1_src
55: sp_alt2_src
56: sp_alt3_src

AUX alternate mode
57: sp_alt1_src2
58: sp_alt2_src2
59: sp_alt3_src2

60: sp_pedal_src2 not used
61: sp_spare_byte2 not used

62: sp_tuner_audio_mode 0: sp_tuner_audio_bypass 1:sp_tuner_audio_mute
63: sp_tuner_meter_mode 1: sp_tuner_meter_bargraph 1:sp_tuner_meter_cents

64: sp_tuner_cal offset to Note A from 440 in cents

65: sp_tuner_src tuner ON/OFF from MIDI
66: sp_tuner_src2 tuner ON/OFF from AUX

67: sp_byp_toggle_src toggle BYPASS from MIDI

68: sp_instrument 0: guitar, 1:bass
69: sp_knob_mode 0: km_normal 1:km_catchup 2:km_locked
70: sp_expert_mode not used
71: sp_auxtip_mode H9 EXP: 0:spa_pedal 1:spa_switch 2:spa_both
72: sp_preset_input_gain reduce level at preset input
73: sp_preset_spill fade audio when changing presets (PF only)
74: sp_progup_src not used
75: sp_progdown_src not used
76: sp_progup_src2 not used
77: sp_progdown_src2 not used
78: sp_bankup_load_src MIDI increment program and load remote source
79: sp_bankdn_load_src MIDI decrement program and load remote source
80: sp_bankup_load_src2 AUX increment program and load remote source
81: sp_bankdn_load_src2 AUX decrement program and load remote source

various looper remotes

82: sp_empty_looper_src MIDI empty looper remote source
83: sp_empty_looper_src2 AUX empty looper remote source
84: sp_dub_src MIDI dub looper remote source
85: sp_dub_src2 AUX dub looper remote source
86: sp_lp_dir_src MIDI reverse looper remote source
87: sp_lp_dir_src2 AUX reverse looper remote source
88: sp_lp_8ve_src MIDI octave looper remote source
89: sp_lp_8ve_src2 AUX octave looper remote source
90: sp_ingain_src MIDI ingain remote source
91: sp_outgain_src MIDI outgain remote source
92: sp_routing_mode 0: srm_normal 1:srm_4_wire 2:srm_wet_dry
93: sp_midiout_mode 0: sp_midi_out_out 1:sp_midi_out_thru
2: sp_midi_out_thru_clock 3:sp_midi_out_merge

Word Parameters:

type key value is *tj_sysvars_word_key* (TJ_ROOT_KEY+0x300)

0: sp_os_version
1: sp_mix_knob
2: sp_tempo

knob min/max values for EXP

3: sp_kb1_min
4: sp_kb1_max
5: sp_kb2_min

6: sp_kb2_max
7: sp_kb3_min
8: sp_kb3_max
9: sp_kb4_min
10: sp_kb4_max,
11: sp_kb5_min,
12: sp_kb5_max,
13: sp_kb6_min,
14: sp_kb6_max,
15: sp_kb7_min,
16: sp_kb7_max,
17: sp_kb8_min,
18: sp_kb8_max,
19: sp_kb9_min,
20: sp_kb9_max,
21: sp_kb10_min,
22: sp_kb10_max,

knob min/max values for EXP

23: sp_kb1_min
24: sp_kb1_max
25: sp_kb2_min
26: sp_kb2_max
27: sp_kb3_min
28: sp_kb3_max
29: sp_kb4_min
30: sp_kb4_max,
31: sp_kb5_min,
32: sp_kb5_max,
33: sp_kb6_min,
34: sp_kb6_max,
35: sp_kb7_min,
36: sp_kb7_max,
37: sp_kb8_min,
38: sp_kb8_max,
39: sp_kb9_min,
40: sp_kb9_max,
41: sp_kb10_min,
42: sp_kb10_max,
43: sp_input_gain, // in 0.1 dB steps
44: sp_output_gain, // in 0.1 dB steps
45: sp_version, //encoded (v[0] << 12) + (v[1] << 8) + v[2] (x.y.z[a] - not including a)
46: sp_pedal_cal_min,
47: sp_pedal_cal_max,

Appendix B – System Variables

Dummy Parameters:

type key value is tj_sysvars_dummy1_key (TJ_ROOT_KEY+0x400)

0: sp_preset_dirty
1: sp_preset_outgain
2: sp_product_type
3: sp_transient_preset
4: sp_x_switch
5: sp_y_switch
6: sp_z_switch
7: sp_slow
8: sp_inswell_enabled
9: sp_outswell_enabled