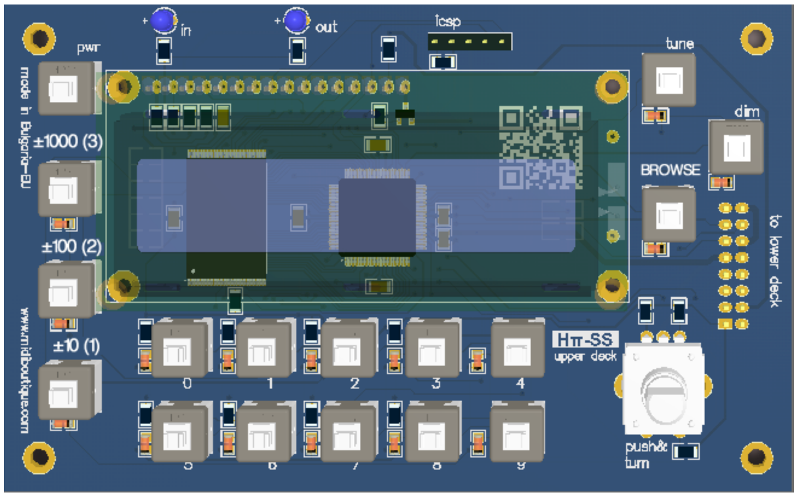
SCALE STATION (SS) SPECIFICATION

PUBLIC Version 9 · August 2017 · Aaron Andrew Hunt



NOTICE: This is a simplified public version of a provisional internal design document. Specifications may change at any time.

A. OVERVIEW AND COMPARISON WITH TBX1

1. Unit stores over 8000 x tuning tables. (Compare TBX1: 512 tables)

2. Each tuning table is 128 x 3 bytes per register: (note, bend MSB, bend LSB), and table names are 16 characters. This is the same as TBX1, except that TBX1 grouped 4 tables together into "layers". Scale Station offers instead any combination of any tuning table with any of 16 MIDI input channels (much more flexible than TBX1)

3. All data (including table names and presets) is stored in non-volatile RAM. Tuning names and preset data is not overwritten by firmware updates (which was a problem with TBX1).

4. Response / timing is up to 10x faster than TBX1.

5. There are 40 Presets, numbered 0..39, and 10 preset buttons, numbered 0..9, with helper buttons for selection. (Compare to TBX1: 15 presets with 15 buttons).

6. Has a 1/4" stereo jack for 2 pedals in one plug input for selecting presets forward / back (new, user-requested feature).

7. Each Preset has a MODE:

MONO: bend+note, 1 channel only - no dynamic channel allocation. Each input channel is assigned its own tuning table.
POLY: bend+note on selected channels, same as TBX1, except that each input channel is assigned its own tuning table.
MTS_: 128 note MTS sysex dump is sent when preset is pressed, unit passes MIDI input through unchanged
USER: user-defined sysex dump is sent when preset is pressed, unit passes MIDI input through unchanged

8. The 16 latching channel buttons of TBX1 are gone, they are instead assigned in the Preset dialog or via Sysex messages.

9. DIP switches are now settings stored in firmware, changed by sysex messages.

10. Standard cell-phone L-ion battery allows 8 hours operation, can be recharged using a standard micro-USB cell phone charger. (Compare to TBX1: 6 AA batteries or a 2.5mm barrel plug 12V 1A power supply).

11. Sysex message is shorter and variable in size (40% faster to program than TBX1).

12. SS retransmits CC messages (same as TBX1).

13. Has the same BYPASS latching switch which turns turns off processing, just passes MIDI through (same as TBX1).

14. Has a BROWSE function for quickly auditioning tunings without having to set a preset (new, user-requested feature).

B. ENCODER & PRESET HELPER BUTTONS

The button marked **±1000 (3)**
- when held down during preset programming, it changes the encoder step to ± 1000 (only while it is held down)
- when held down and a Preset button 0-9 is pressed, it adds 30 to the Preset button's number

Examples: holding down the **±1000** button and pressing Preset 0 activates preset **30**
holding down the **±1000** button and pressing Preset 9 activates preset **39**

The button marked **±100 (2)**
- when held down during preset programming, it changes the encoder step to **± 1000** (only while it is held down)
- when held down and a Preset button 0-9 is pressed, it adds **20** to the Preset button's number

Examples: holding down the **±100** button and pressing Preset 0 activates preset **20**
holding down the **±100** button and pressing Preset 2 activates preset **22**

The button marked **±10 (1)**
- when held down during preset programming, it changes the encoder step to **± 1000** (only while it is held down)
- when held down and a Preset button 0-9 is pressed, it adds **10** to the Preset button's number

Examples: holding down the **±10** button and pressing Preset 0 activates preset **10**
holding down the **±10** button and pressing Preset 3 activates preset **13**

C. ENCODER

Because there are so many tuning tables, we have the number buttons as helpers for the encoder knob.
When the user turns the encoder normally, values change by +/- 1 step. As discussed in the above section, the Encoder Helper buttons can change this step to +/- 10, +/- 100, or +/- 1000 steps.
This applies for any values input by the encoder, so for example ASCII values can skip by groups as well.
For any selection, the values “roll over” so when the maximum is reached going upwards, then values start again at 0. And when 0 is reached going downwards, values start over at the maximum.

D. DISPLAY

16x2 Characters line LCD backlit display. There is a latching switch to toggle backlight on/off (or dim/bright).
Startup display sequence: company name / product name shows for 1-2 seconds, then displays presets.

```
H-PI_INSTRUMENTS
_SCALE_STATION__

THIS_PRESET_NAME
00MODE_B000_P000
```

Description:

```
NNNNNNNNNNNNNNNN
AAMMMM_BKKK_TPPP

NNNNNNNNNNNNNNNN = 16-character name of the selected Preset
AA = number of selected Preset (00..39)
MODE = 4-char name of the MODE for the selected Preset (POLY, MONO, MTS_, USER)
BKKK = B followed by the 3-char Bank number for the selected Preset (000..127)
PKKK = P followed by the 3-char Patch number for the selected Preset (000..127)
```

E. SPECIAL DUAL PEDAL INPUT 1/4" JACK

The jack is a stereo input, so it is accepting 2 switches. The reason for this is hands-free preset switching which a lot of user asked for with TBX1.

Switch 1: step Preset forward (if Preset 3 is currently selected, this switch selects Preset 4)
Switch 2: step Preset backward (if Preset 8 is currently selected, this switch selects Preset 7)

Numbers "roll over" - stepping positively past 39 goes back to 0, stepping negatively past 0 goes to 39.

IMPORTANT NOTE: The unit senses the state of each switch at startup and considers the startup position as OFF.

F. CC input from MIDI

Incoming CC messages are retransmitted the same as TBX1.
NOTE: This includes "thinning" input filter in order not to produce "MIDI bottleneck" output.
Example 1:
1. user has a preset with OMNI input mode (all input channels set to the same tuning and transposition).
3. The user has selected the default output channel assignments, skipping channel 10 in output.
4. MIDI CC volume 7 is received on any channel.

Unit responds by transmitting the CC value to all 16 output channels, skipping channel 10.

G. MODES

A MODE is the property of a Preset which describes how the unit functions when the preset is selected.

POLY: [POLYphonic] - Default
Preset sends BANK and PATCH and selects 16 tuning tables (1 per input channel)
Unit sends bend + note on user-selected dynamic output channels
This function is the same as TBX1, except that each input channel is arbitrarily linked to a tuning table

MONO: [MONOphonic]
Preset sends BANK and PATCH and selects 16 tuning tables (1 per input channel)
Unit sends bend+note on 1 user-selected output channel, with no dynamic allocation

MTS_: [MIDI Tuning Standard, Full keyboard]
Preset sends BANK and PATCH and sends a 128-note (3 bytes per note) MTS Sysex when pressed
Unit passes MIDI input through unchanged (note that 2 of these bytes are not the same as stored table data).

USER: Preset sends BANK and PATCH and sends a User-Defined Sysex when pressed (defined though software)
Unit passes MIDI input through unchanged

H. POLY MODE

POLY mode is almost the same as TBX1. The only difference is that for every note input, the channel of the note determines which tuning table is looked up, according to the currently selected preset. In both POLY and MONO modes, midi input message follow a familiar path:

STAGE 1: get input message and prepare output messages

MIDI Controller output —> SS input
1. got note message
- Which input channel is this message from?
2. look up tuning table for this input channel according to the current Preset
- Which note is sent?
3. get BEND MSB, BEND LSB, and NOTE for this incoming MIDI note from the tuning table

STAGE 2: send output messages

1. for POLY mode, output functions the same as TBX1, according to output channels on/off states. The only difference is that each input channel has its own tuning table assignment.
2. for MONO mode, there is no dynamic channel allocation, only pitch bend + note on the MONO output channel, plus other constraints (see next section).

I. MONO MODE

This is a special mode for working with monophonic synthesizers, which requires its own special output routine.
Only one output channel is used, so this is not dynamic channel allocation.

Messages are still sent in the following order:

1. Pitch BEND
2. NOTE ON

Unlike dynamic allocation mode, pressing a new key for note ON does not cut off any previously held keys;
NOTE OFF messages are sent only when a key is lifted, as expected with normal MIDI.

The special part is this: each time a key is lifted and NOTE OFF is sent, we check
to see if any notes are being held. If the number of notes held > 0, then immediately after
sending the NOTE OFF for the lifted key, we must resend the BEND message for the last note played.
You see this means you have to keep an array of all notes played and the order in which they have been played,
so that whenever a key is lifted (NOTE OFF is received) the proper BEND value can be resent after sending
the tuning table NOTE OFF to output.

Example 1:
1. user plays middle C MIDI key 60
2. still holding the C, the user then plays G above it, MIDI key 67 (this sounds as monophonic portamento up)
3. user lifts G key 67 but still holds C key 60 (this sounds as portamento back down)

The messages sent in this example must be as follows:
1. BEND + NOTE ON (plus preset transposition value) for key 60
2. BEND + NOTE ON (plus preset transposition value) for key 67
3. NOTE OFF (plus preset transposition value) for key 67, followed immediately by BEND ONLY for key 60

The reason you see is that monophonic synths track held keys for portamento effects, and we need the right
pitch (BEND) when keys are lifted. Since only one key sounds at a time, this means we only have to
resend the bend message of the last note played. The next example shows further how it should work:

Example 2:
1. user plays middle C MIDI key 60
2. still holding C, user plays E above it, MIDI key 64
3. still holding C and E, user plays G above it, MIDI key 67
4. user lifts G key 67 but still holds C key 60 and E key 64
5. user lifts E key 64 but still holds C key 60
6. user plays D MIDI key 62
7. user lifts D key 62 still holding C key 60

The messages sent in this example are:
1. BEND + NOTE ON (plus preset transposition value) for key 60
2. BEND + NOTE ON (plus preset transposition value) for key 64
3. BEND + NOTE ON (plus preset transposition value) for key 67
4. NOTE OFF (plus preset transposition value) for key 67, followed immediately by BEND ONLY for key 64
5. NOTE OFF (plus preset transposition value) for key 64, followed immediately by BEND ONLY for key 60
6. BEND + NOTE ON for key 62
7. NOTE OFF (plus preset transposition value) for key 62, followed immediately by BEND ONLY for key 60

J. MTS_ MODE

This is a special mode for synthesizers supporting the MIDI Tuning Standard. In this mode, the unit does no
input processing. It only transmits a 128-note Sysex Tuning Dump when a Preset button is pressed. The tuning
dump data is in a 3-byte format (NOTE, Data1, Data2) where the data bytes are 12 * 128 * 128 = 196608 steps per octave,
zeroed at data (0,0), which is different from the internal SS data which is (Note, Bmsb, Blsb) 12 * 64 * 128 = 98304 steps per octave,
so the data must be transformed.

MTS Sysex. Program an MTS tuning table when a Preset in MTS_ mode is pressed
F0 - SysEx start
7E - Universal non-realtime
7F - All Devices

08 - MTS byte
01 - Bulk Tuning Dump
tt - tuning program number (0..127 / h00..7F)
<aa> - tuning name 16 ASCII characters
[nn, mm, bb] note data repeated 128 times
00 - (checksum, ignored)
F7 - SysEx end

K. USER MODE

This is a special mode for working with synthesizers that do not support MTS but have some other defined Sysex protocol. The mode globally stores the following parameters for building a user-defined sysex message (default values are given in parentheses and correspond to the MIDI Tuning Standard 1-byte scale/octave tuning dump):

1. BYTE:	number of header bytes, up to 32 or some other logical limit (5)
2. BYTE(S):	header byte actual values (h7E, h7F, h08, h06, h00)
3. BOOLEAN:	include program number? (yes)
4. BOOLEAN:	include tuning table name? (no)
5. BYTE:	tuning table start key: (60)
6. BYTE:	tuning table end key: (71)
7. BYTE:	per note byte format (0)
	0: 1 byte ± cents offset per note*
	1: 2 bytes ± pitch bend data per note (unchanged)
	2: 3 bytes note + pitch bend data per note (unchanged)
	3: 3 bytes MTS data per note**
	(other options can be added with firmware updates)
8. BOOLEAN:	include checksum byte? (yes)
9. BYTE:	checksum byte: (0)

The general form of the USER defined sysex message as sent when the user presses a USER mode Preset button is:

F0 - SysEx start
<user defined header bytes>
<optional program number> (0..127)
<optional tuning name> (16 ASCII chars)
    <series of (possibly transformed) data bytes from user-defined tuning table start-key to user-defined tuning table end-key>
<optional checksum byte defined value>
F7 - SysEx end

L. PRESETS

When a preset is pressed, the output behaviour depends on the MODE the user has assigned to that Preset.

1. for POLY and MONO, the behaviour is the same as TBX1 (Bank & Patch + Pitch Bend Range setting messages*)
2. for MTS_ and USER modes, the behaviour is new. Bank and Patch are sent, but no pitch bend setting messages. Instead, a sysex message is sent. For details, see the sections on these modes.

See also the section above on MODES. *range setting messages must respect the new GLOBAL Pitch Bend setting, see section N.

M. PRESET PROGRAMMING

NOTE: Tuning tables are 128 registers, 3 bytes each register: (NOTE, BEND MSB, BEND LSB) same as before.

A Preset consists of:

Name: 16 characters ASCII
MODE: POLY, MONO, MTS_, or USER
Bank: 0..127 or OFF
Patch: 0..127 or OFF
Tunings: 16 Tuning Tables, assigned one table per input channel
Output Channels: 16 output channels ON / OFF (instead of TBX1 hardware switches)

Default Values for new Presets:

Name: TUNING_PRESET_XX (where XX is the number of the Preset 00..39)
MODE: POLY
Bank: OFF
Patch: 000
Tunings: 0XX (where XX is the number of the Preset 00..39) assigned to all 16 channels
Output Channels: oooooooooXoooooo (o = ON, X = off … default is all ON except channel 10)

Presets are programmed as follows: (pressing any Preset button during programming exits at any time).

1. Press a Preset button, or a combination of the ±1000, ±100, or ±10 buttons with a Preset button together to select a Preset 0..39, and then press the programmer knob to begin.

2. Choose a mode. Display changes to:

```
PP__SELECT_MODE:
______MMMM______
```

Description:
PP = the Preset number
MMMM = the 4-char mode assigned to the Preset

Whatever mode has been programmed for this Preset is shown. Presets default to mode = POLY. For example, when the mode is POLY, then ______POLY______ flashes. Turning the encoder knob scrolls through the following options:

```
PP__SELECT_MODE:
______POLY______

PP__SELECT_MODE:
______MONO______

PP__SELECT_MODE:
______MTS_______

PP__SELECT_MODE:
______USER______
```

3. Assign a Bank. Display changes to:

```
PP__SELECT_BANK:
______BBB_______
```

Description:
PP = the Preset number
BBB = Bank number (000..127 or OFF) defaults to OFF for new Presets

Bank number OFF flashes. Turn the knob to find a desired bank. Press the knob to select and move on.

4. Assign a Patch. Display changes to:

```
PP_SELECT_PATCH:
______NNN_______
```

Description:
PP = the Preset number
NNN = Patch number (000..127 or OFF) defaults to 000 for new Presets

Patch number 000 flashes. Turn the knob to find a desired patch. Press the knob to select and continue.

5. Assign tuning tables. Display changes to:

```
PP_IN_CH01?_TTTT
ATUNINGTABLENAME
```

Description:
PP = the Preset number
TTTT = the tuning table number to assign
ATUNINGTABLENAME = 16-character tuning table name (changes with the table number)

The tuning table number flashes (the tuning table name does not flash).
The number TTTT is displayed as the table currently assigned to the given input channel for the preset.

User presses programmer knob to enter the tuning table. Display changes to:

```
PP_ASSIGN_ALL?__
______YES_______
```

______YES_______ flashes, and turning the knob alternates between NO and YES.

User presses programmer knob to select YES or NO.
If YES, then this tuning table is assigned to ALL input channels, and we continue with step 6.
If NO, then the display changes to:

```
PP_IN_CH02?_TTTT
ATUNINGTABLENAME
```

where TTTT defaults to the last table selected, and the process above is repeated for all 16 input channels.

6a. If POLY mode is selected:

```
PP_OUT_CHANNELS:
CCCCCCCCCCCCCCCC
```

Description:
PP = the Preset number
C = MIDI Channel (display is not shown as C, it is "o", or "X", or a blank space, see below)

The user is seeing output channels assigned to this preset. This works like the channel switches on TBX1.
All channels in the group default to ON except for channel 10 which defaults to OFF.

Each channel in the group can be turned ON and OFF by the user just like those switches on TBX1.
Channel ON (to be used in the dynamic allocation for this group) is shown by "o" = lower case letter o.
Channel OFF (to be skipped in this group) is shown by "X" upper case X.

Each output channel flashes its default value, and turning the knob toggles the value ON or OFF. Pushing the knob selects the value and moves right to the next channel. When all channels have been selected, this page is done.

NOTE: The user can choose to turn off all channels, effectively muting the unit for output on a given input channel, the same as physical switches on TBX1. For the case where all 16 channels are turned off, we flash an alert for 3 seconds:

```
YOU_MUST_ENGAGE_
AT_LEAST_1_CHAN_
```

Then go back to the first channel and make the user go through the whole process again.
In other words, we do not allow a situation where the user has turned off all output channels, as it makes no sense.

6b. for all other modes (MONO, MTS_, USER): 6a does not take place, instead we do this much simpler process.

Select the output channel. Display changes to:

```
PP_SELECT_OUTPUT
CHANNEL:_01_____
```

Channel number 01 flashes, the user turns the knob to select a value from 01 to 16. Push to select and continue.

6c. for MTS_ and USER modes only, we also need a program number (0-127) for the sysex dump.

Select a program number for the sysex dump. Display changes to:

```
PP_PROGRAM_NUM._
______000_______
```

Program number 000 flashes, the user turns the knob to select a value from 000 to 127. Push to select and continue.

7. Top line of display changes to:

```
PP_PRESET_NAME:_
```

Bottom line of display changes to:

```
TUNINGS_PRESET_X
```

Where "X" is the preset number.

To make this fast, first the whole default name TUNINGS_PRESET_X flashes.
If the user wants to edit the name, then they turn the knob.
If they want to use the default name, they just push the encoder knob and the process is over.

If they choose to edit the name, then each character in the name flashes starting with the leftmost character.
Turning the knob scrolls through characters, and pressing the knob selects the character, moving to the next one through all 16 characters left to right.
After the last character, the name is finished, and the programming process is over.

N. GLOBAL SETTINGS

These settings were formally handled by the DIP switches, with one additional setting.
We handle changing them via sysex messages (see final section), and we also provide a user dialog for setting them.

To initiate programming Global settings, the user presses and holds the encoder knob for 5 seconds. Pressing any Preset button or one of the Preset Helper buttons exits at any time.

1. Bank Select Byte Format

```
___BANK_FORMAT?_
___1_BYTE_CC0___
```

Bottom line flashes, the user turns the knob to select an option:

```
___1_BYTE_CC0___

___1_BYTE_CC32__

2_BYTES_CC0_CC32

2_BYTES_CC32_CC0
```

2. Pitch Bend Timing

```
___BEND_TIMING?_
______FAST______
```

Bottom line flashes, the user turns the knob to select an option:

```
______FAST______

______SLOW______
```

FAST means the unit sends all messages as fast as possible. SLOW means the unit pauses 30ms after sending the bend messages before sending the Note ON message. Press the knob to select and continue.

3. Sysex Retransmission

```
RETRANSMIT_SYSX?
______YES_______
```

YES flashes, and turning the knob alternates between NO and YES. Press the knob to select and continue.

4. Target Device Pitch Bend Range

```
PITCH_BEND_RANGE
_______01_______
```

01 flashes, and turning the knob navigates values between 1 and 24. Press the knob to select and exit.

NOTE: if this value is set to anything other than 1, the unit processes the tuning table bend bytes before sending them.

O. SYSEX UNIT PROGRAMMING MESSAGES

Message length varies with message type.

Messages are grouped into message blocks of 16 possible messages each, with undefined message IDs for future use.
(*Undefined*: 03 - 0F, 11 - 1F, 22 - 2F, 34 - 3F, etc.)

Messages are listed in order by block and ID below.

BLOCK 0: 3 TUNING TABLE MESSAGES

ID 00. program a microtonal tuning table name and all data: (6+2+16+384+1 = 409 bytes)
F0 - SysEx start
00 - First ID
21 - Second ID
7F - Third ID
1F - Device ID
00 - Message ID
    tt - tuning table number MSB (0..127 / h00..7F)
    tt - tuning table number LSB (0..127 / h00..7F)
    cc - ASCII table name char 1
    …
    cc - ASCII table name char 16
    nn - MIDI note OUT (0..127 / h00..7F) for note in 0
    pp - Pitch Bend MSB OUT (0..127 / h00..7F) for note in 0
    pp - Pitch Bend LSB OUT (0..127 / h00..7F) for note in 0
    …
    nn - MIDI note OUT (0..127 / h00..7F) for note in 127
    pp - Pitch Bend MSB OUT (0..127 / h00..7F) for note in 127
    pp - Pitch Bend LSB OUT (0..127 / h00..7F) for note in 127
F7 - SysEx end

ID 01. program a single microtonal note in a tuning table (6+2+4+1 = 13 bytes)
F0 - SysEx start
00 - First ID
21 - Second ID
7F - Third ID
1F - Device ID
01 - Message ID
    tt - tuning table number MSB (0..127 / h00..7F)
    tt - tuning table number LSB (0..127 / h00..7F)
    kk - key number IN (0..127 / h00..7F)
    nn - MIDI note OUT (0..127 / h00..7F)
    pp - Pitch Bend MSB OUT (0..127 / h00..7F)
    pp - Pitch Bend LSB OUT (0..127 / h00..7F)
F7 - SysEx end

ID 02. program a tuning table's 16-character ASCII name (6+2+16+1 = 25 bytes)

F0 - SysEx start
00 - First ID
21 - Second ID
7F - Third ID
1F - Device ID
02 - Message ID
    tt - tuning table number MSB (0..127 / h00..7F)
    tt - tuning table number LSB (0..127 / h00..7F)
    cc - ASCII char 1
    …
    cc - ASCII char 16
F7 - SysEx end

BLOCK 1: 1 PRESET MESSAGE

ID 10. program a Preset name and all data (6+1+16+1+1+1+1+1+32+16+1 = 77 bytes)
F0 - SysEx start
00 - First ID
21 - Second ID
7F - Third ID
1F - Device ID
10 - Message ID
    pp - Preset ID (0..39)
    cc - ASCII Preset Name, char 1
    …
    cc - ASCII Preset Name, char 16
    mm - Preset MODE (00: POLY, 01:MONO, 02:MTS_, 03:USER)
    bo - bank ON / OFF (0=OFF, any other value=ON)
    bb - bank number (0..127 / h00..80)
    po - patch ON / OFF (0=OFF, any other value=ON)
    pp - patch number (0..127 / h00..80)
    tt - tuning table number MSB (0..127 / h00..7F) for input channel 0
    tt - tuning table number LSB (0..127 / h00..7F) for input channel 0
    …
    tt - tuning table number MSB (0..127 / h00..7F) for input channel 15
    tt - tuning table number LSB (0..127 / h00..7F) for input channel 15
    vv - channel ON or OFF (0..nn / h00..nn) (0=OFF, any other value=ON) for channel 0
    …
    vv - channel ON or OFF (0..nn / h00..nn) (0=OFF, any other value=ON) for channel 15
F7 - SysEx end

BLOCK 2: 2 USER MODE PROGRAMMING MESSAGE

ID 20. program header bytes of a Preset's USER mode sysex message (6+1+(0 to 32)+1 = 8 to 40 bytes)
F0 - SysEx start
00 - First ID
21 - Second ID
7F - Third ID
1F - Device ID
20 - Message ID
    hh - count of header bytes: (0..32)

vv - header byte value 0

…

vv - header byte value hh

F7 - SysEx end

ID 21. program a Preset's USER mode sysex message options

F0 - SysEx start

00 - First ID

21 - Second ID

7F - Third ID

1F - Device ID

21 - Message ID

pp - include program number?: (0 = no, any other value = yes)

tt - include tuning table name? (0 = no, any other value = yes)

mm - include checksum byte? (0 = no, any other value = yes)

ss - tuning table start key: (0..127)

ee - tuning table end key: (0..127)

bb - byte formula: (0..3)

0: 1 byte ± cents offset per note

1: 2 bytes ± 0.012207 cents offset per note

2: 3 bytes MTS data per note

3: 3 bytes pitch bend data per note (unchanged)

(other options can be added with firmware updates)

F7 - SysEx end

BLOCK 3: 4 GLOBAL SETTINGS MESSAGES

ID 30. program the format of the Bank Select message (6+1+1 = 8 bytes)

F0 - SysEx start

00 - First ID

21 - Second ID

7F - Third ID

1F - Device ID

30 - Message ID

ss - format of the Bank Select message: (0..3)

0: send CC0, Value

1: send CC32, Value

2: send CC0, CC32, Value

3: send CC32, CC0, Value

F7 - SysEx end

ID 31. select the Pitch Bend response timing (6+1+1 = 8 bytes)

F0 - SysEx start

00 - First ID

21 - Second ID

7F - Third ID

1F - Device ID

31 - Message ID

tt - pitch bend response timing selection: (0..2)

0: as fast as possible

1: ~ 5 ms

2: ~ 30 ms

F7 - SysEx end

ID 32. select sysex retransmission behaviour (6+1+1 = 8 bytes)
F0 - SysEx start
00 - First ID
21 - Second ID
7F - Third ID
1F - Device ID
32 - Message ID
    ss - select sysex retransmission behaviour: (0=OFF, any other value=ON)
F7 - SysEx end

ID 33. set global Pitch Bend target device range (6+1+1 = 8 bytes)
F0 - SysEx start
00 - First ID
21 - Second ID
7F - Third ID
1F - Device ID
33 - Message ID
    rr - select global Pitch Bend target device range: (1..24) **NOTE**: 0=1, 1=1, and any value > 24 = 24
F7 - SysEx end

P. SCALE BROWSING

The button previously marked **BACK** is now marked **BROWSE**. The idea here is to give users a simple way to try out any of the different tunings in memory. Pressing this button enters a special unit state.

- the MODE, BANK, and PATCH of the currently selected preset are respected.
- whatever is the current preset TUNING on channel 1 is applied to all input channels (the preset is not changed)

After pressing this button, the display changes to:

```
_BROWSING:_TTTT_
ATUNINGTABLENAME
```

Description:

TTTT = the tuning table number (begins as the table assigned to the current preset on its channel 1)
ATUNINGTABLENAME = 16-character tuning table name (changes with the table number)

Nothing flashes because nothing is being programmed. The user can turn the encoder knob to select any tuning. The ±1000, ±100 and ±10 helper buttons apply to tuning selection as expected. Whatever tuning is currently shown while browsing is applied to all channels so that the user can simply play keyboard keys and hear the tuning.

- Pressing the encoder knob applies the tuning to all input channels of the current preset, exits the SCALE BROWSING state and returns the unit to normal functioning.
- Pressing the BROWSE button or any PRESET button exits the SCALE BROWSING state without changing any preset tunings, and returns the unit to normal functioning.

Q. RECEIVED MIDI MESSAGE BEHAVIOURS

1. **Realtime messages** - **POLY**: ignore realtime messages

- **MONO**, **MTS_**, **USER**: pass realtime messages through

2. **Program Change** - **POLY**: retransmit program changes on all active channels
   - **MONO**, **MTS_**, **USER**: pass program changes through

3. **Control Change** - **POLY**: retransmit CCs on all active channels (thin the data to avoid bottleneck)
   - **MONO**, **MTS_**, **USER**: pass CC through

4. **Pitch Bend** - **MTS_**, **USER**: pass pitch bend through

Pitch Bend - **POLY**, **MONO**: received value is combined with current value on active channel(s) so that notes already tuned with pitch bend from a tuning table can still smoothly bend up or down from their tuned position.

Example: receiving PB (65, 1)

No bend is (64, 0), so the incoming message means (MSB+1, LSB+1)

Say two channels are playing notes:

channel 0 is bending (63, 22)
channel 1 is bending (77, 45)

The incoming value is combined with the existing values, so

channel 0 becomes (63+1, 22+1) = (64, 23)
channel 1 becomes (77+1, 45+1) = (78, 46)

Of course, the above example is for concept illustration only. A general solution requires a smooth transition between LSB and MSB values by calculating bend in 16,384 steps per MIDI note:

5. **Polyphonic Aftertouch** - **POLY, MONO**: convert polyphonic aftertouch to channel aftertouch per active channel(s) (thin data to avoid bottleneck)
   - **MTS_**, **USER**: pass polyphonic aftertouch through

6. **Channel Aftertouch** - **POLY, MONO**: retransmit channel aftertouch on all active channels (thin data to avoid bottleneck)
   - **MTS_**, **USER**: pass channel aftertouch through