Adding Midi CC for Logue Oscillator, using a trick.

We will misuse the k\_user\_osc\_param\_shiftshape parameter. This has MidiCC = 55.

Globally, we’ll be doing the following: If we receive a parameter change on k\_user\_osc\_param\_shiftshape, we check if it ‘could be from midi’ and if so, we interprete some consecutive values as a midi CC. Then the world is open to us.

First, how do we know ‘it could be from midi’.

Normally when you twist the knob on the NTS1, you will get value in 10 bit resolution. Midimessages will be upscaled (by the device, on midicc=55) to 10 bits, by multiplying them by 8.

So if we are only seeing messages divisible by 8, we can assume this is from midi, certainly if we start the messaging with two values, far apart, like MidiCC (55,3), MidiCC(55,100). We will see that we need 3 consecutive bytes.

We will just encode all 2 and 3 byte messages, such as e.g: (Set CC value to 20)

176 40 20

To fit in, we will omit the midi channel, and only process Control Change, Program Change, Channel Pressure and Piitchbend. Then we need 16 bits ( 2 + 7 + 7)

We will make sure two consecutive values are not the same and need one bit (x3) for that. So effectively we have 6 bits.

In all:

If the original message is 1sssmmmm 0abcdefg 0hijklmn then we will send:

55 <0sabcd00 or 0sabcd11> 0efghi01 0jklmn10

There are two bits for the s <00> | <11> thus we can implement 4 commands:

|  |  |  |
| --- | --- | --- |
| S | 00 | 11 (t) |  |
| 0 | 00 | Control Change |
| 1 | 00 | Program Change |
| 0 | 11 | Channel Pressure |
| 1 | 00 | Pitch Bend |

We allways know when there is a start byte and when there is a data byte (and which)

The framework decodes these messages for you (only Control Change is supported) and will send the result to OSC\_SetParameter(uint8\_t cc, uint8\_t value);

Note, that you have to encode midi messages to use them with this trick, something like:

void midiOut(int cc, int data1, int data2)

{

int s,t,abcd,efghi,jklmn;

s=0;

t=0;

abcd=(data1 >> 3) & 0xF;

efghi=((data1 & 7) << 2) + (data2 >> 5) & 3;

jklmn=data2 & 0x1F;

data2= (s << 6) + (abcd << 2);

WriteMidi(MIDI\_CC,55,data2);

data2=(efghi << 2) | 1;

WriteMidi(MIDI\_CC,55,data2);

data2=(jklmn << 2) | 2;

WriteMidi(MIDI\_CC,55,data2);

}