MIDI IN WEB3D

IMPLEMENTED IN ONE WEB3D BROWSER

Topics:

- References
- MIDI 2 transport packets vs MIDI 1 messages
- strategies and conversions for MIDI2 readiness, and MIDI1 back support
- frame speed routing and MIDI-thread direct transport
- Proposed node set
- ecmascript MIDI 2 packet parsing and construction
- demo scenes
- use of midi as an event sequencer for general routing

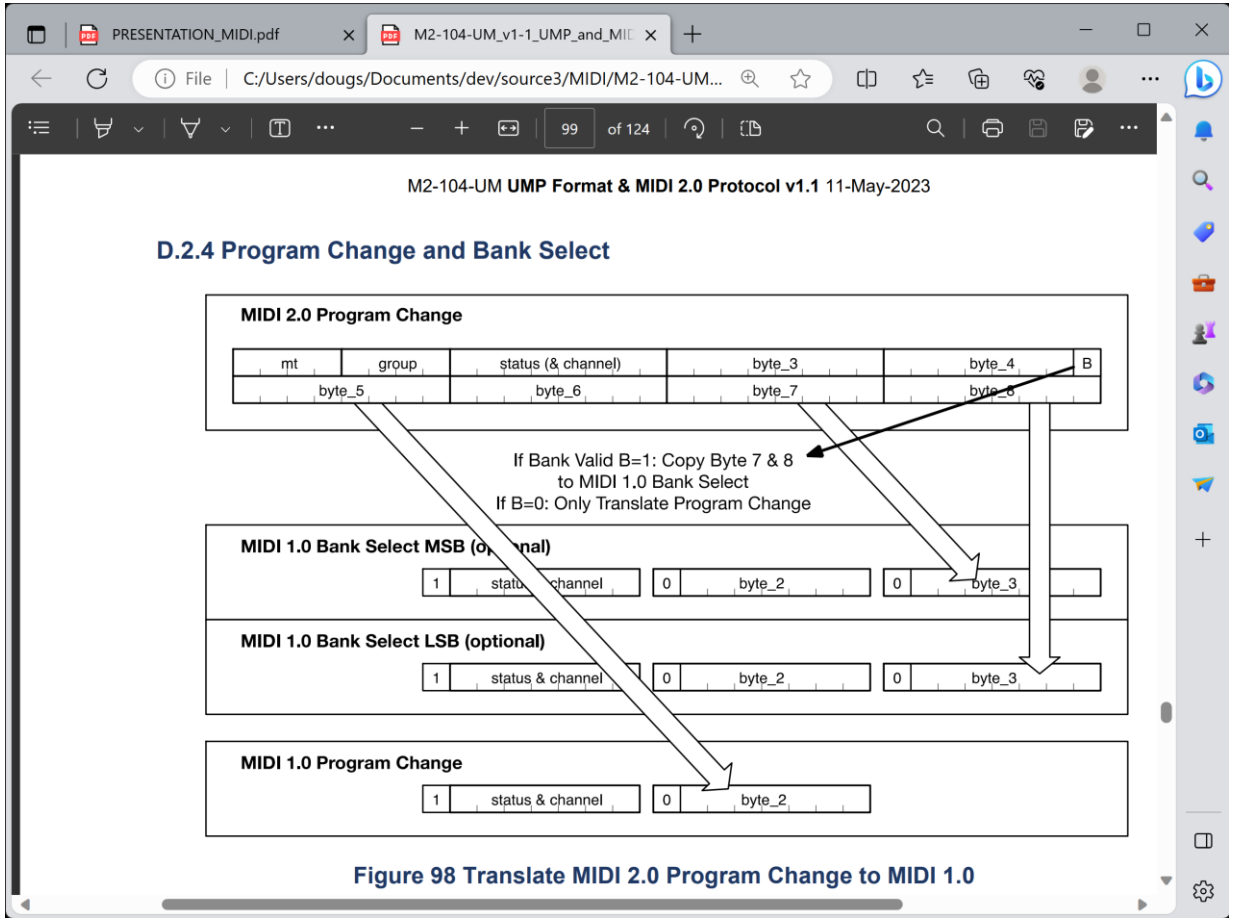
references

- https://freewrl.sourceforge.io/tests/43_MIDI
- https://en.wikipedia.org/wiki/MIDI
- https://www.midi.org/specifications
- https://www.web3d.org/documents/specifications/19775-1/V4.0/Part01/Architecture.html
- https://www.w3.org/TR/webmidi/
- https://developer.mozilla.org/en-US/docs/Web/API/Web_MIDI_API
- https://www.music.mcgill.ca/~gary/rtmidi/

MIDI2 vs MIDI1

- MIDI 1.0
  - 1990s
  - Message = String of bytes, usually 2 or 3 but can be many
  - Ports: input and output ports
- MIDI 2.0
  - 2023
  - UMP universal midi packet – 64 bits, with continuation packets when needed
  - Bi-directional endpoints
- Converting:
  - See midi.org MIDI 2 Protocol standard, appendix D

midi.org MIDI2 Protocol specs, Appendix D



FRAME SPEED ROUTING vs DIRECT MIDI THREAD CONNECTIONS

- Direct - dedicated thread for each MIDI source/input
  - Output port <= Input port
- ROUTEd – once per draw thread
  - Output port <= processing node <= Input port
- Strategy: combination – direct insofar as practical

PROPOSED NODE SET

- https://freewrl.sourceforge.io/tests/43_MIDI/NOTES_midi.txt

- Level 1
  - MIDIPortSource : MIDISource
  - MIDIFileSource: MIDISource
  - MIDIPortDestination : MIDIDestination
  - MIDIPrintDestination : MIDIDestination
  - (MIDIFileDestination : MIDIDestination)

PROPOSED NODE SET

- Level 2
  - MIDIOut : MIDIDestination
  - MIDIIn : MIDISource
  - MIDIToneSplitter : MIDINode
  - MIDIToneMerger: MIDINode
- Level 3
  - MIDIProgram : MIDINode
  - MIDIDelay : MIDINode
- Level 4
  - MIDIAudioSynth

Ecmascript MIDI packet parsing

```
function parseUmp(packet){
        var buf = new ArrayBuffer(8);
        var view = new DataView(buf);
        //using SFDouble as packet container
        view.setFloat64(0,packet,true);
        var bytes = new Uint8Array(buf);
        var res = {};
        res.channel = (bytes[1] & 0xF) + 1;
        res.command = bytes[1] - (res.channel-1);
        res.note = bytes[2];
        res.velocity = view.getUint16(4,true);
        return res;
      }
```

ECMAScript MIDI packet constructing

```
function makeUmp(command,channel,note,velocity){
        var buf = new ArrayBuffer(8);
        var view = new DataView(buf);
        view.setUint8(1,command|(channel-1),true);
        view.setUint8(2,note,true);
        view.setUint16(4,velocity,true);
        var res = view.getFloat64(0,true);
        return res;
    }
```

Demo scenes

- https://freewrl.sourceforge.io/tests/43_MIDI/88key/midi_port2key88.x3d.txt
- https://freewrl.sourceforge.io/tests/43_MIDI/88key/player_piano.mp4
- https://freewrl.sourceforge.io/tests/43_MIDI/88key/M-AUDIO_KEYSTATION32_MINI.jpg
  - player keyboard

- https://freewrl.sourceforge.io/tests/43_MIDI/88key/kb.x3d.txt
- https://freewrl.sourceforge.io/tests/43_MIDI/88key/playable_piano.mp4
- https://freewrl.sourceforge.io/tests/43_MIDI/88key/kbgen.py.txt
  - playable keyboard

- https://freewrl.sourceforge.io/tests/43_MIDI/ps_fireworks.x3d.txt
- https://freewrl.sourceforge.io/tests/43_MIDI/ps_fireworks.mp4
- midi input used to control ParticlePhysics fireworks
- MIDIProgram + MIDIDelay node use
- https://freewrl.sourceforge.io/tests/43_MIDI/Instruments.txt
  - program-change instrument numbers