Send MIDI Data via RS-232

Overview

Most MIDI messages are sequences of 2-3 8-bit bytes. The largest hurdle when integrating the DecaBox with Crestron / AMX / Control4 equipment is properly formatting these messages. An excellent, detailed description of their syntax can be found here:

- https://www.midi.org/specifications/item/table-1-summary-of-midi-message
- https://www.midi.org/specifications-old/item/table-3-control-change-messages-data-bytes-2

> In our documentation, we use this format to describe raw byte data: `$AA $BB $CC`. The $ sign signifies hexadecimal values, which means that AA, BB, etc are in the range [00 FF] hex, which maps to [0 255] decimal. Spaces are inserted between bytes in this documentation merely for convenience in reading.

MIDI messages are zero based and *generally* bytes 2 and 3 have a maximum value of $7F, or 127 decimal. A simple online decimal <-> hex converter can be found at

https://www.binaryhexconverter.com/decimal-to-hex-converter

Here are sample MIDI messages:

```
$90 $01 $7F    Note on, MIDI channel 1, note #2, full velocity $8F $07 $40    Note off, MIDI
channel 16, note #8, 50% velocity$B1 $03 $10    MIDI CC #4 (foot controller), ~30% intensity,
MIDI channel 2
$F0 $01 $02 $03 $04 $05 $06 $F7    A short MIDI sysex message. These are framed by [F0 F7] always.
```

(The MIDI channel is the lower nibble of the first byte. That is, $BX means MIDI CC message and X can vary between [$0 $F]. Hex $0 means MIDI channel 1, $1 means MIDI channel 2, etc. $F is channel 16.)

```
\xF0\x01\x02\x03\x04\x05\x06\xF7
```

Sending MIDI to Data to the Decabox

Once the 2-3 byte MIDI messages have been correctly formatted, the next step is to send them to the DecaBox. That syntax looks like this:
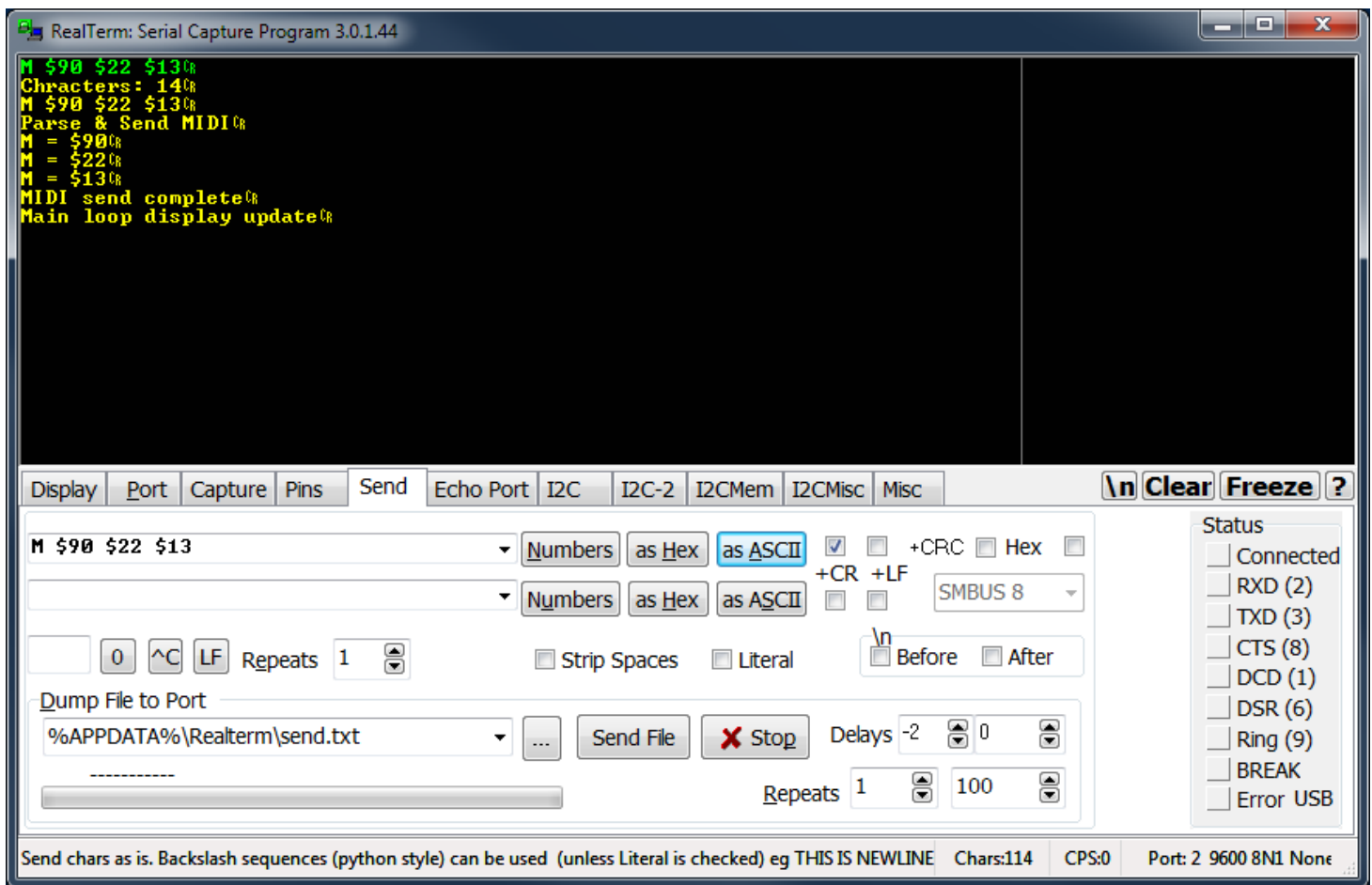
```
M $AA $BB $CC...$FF[cr]
- The command starts with a capital M.
- It is followed by single space character ' '.- Then hex values written as strings in ASCII
format.
- Command terminates with a carriage return.
Examples, using the same commands listed above:
M $90 $01 $7F[cr]   Note on, MIDI channel 1, note #2, full velocity M $8F $07 $40[cr]   Note
off, MIDI channel 16, note #8, 50% velocityM $B1 $03 $10[cr]   MIDI CC #4 (foot controller),
~30% intensity, MIDI channel 2M $F0 $01 $02 $03 $04 $05 $06 $F7[cr]   A short MIDI sysex
message. These are framed by [F0 F7] always.
Each of these commands are sent as ASCII STRINGS, whose length varies between 13 and ~30
characters.
```

Screenshot. Green is text sent to the box, yellow are replies.

RealTerm: Serial Capture Program 3.0.1.44

```
M $90 $22 $13cr
Chracters: 14cr
M $90 $22 $13cr
Parse & Send MIDIcr
M = $90cr
M = $22cr
M = $13cr
MIDI send completecr
Main loop display updatecr
```

Display | Port | Capture | Pins | Send | Echo Port | I2C | I2C-2 | I2CMem | I2CMisc | Misc

\n Clear Freeze ?

M $90 $22 $13   ▼ Numbers | as Hex | as ASCII ☑ ☐ +CRC ☐ Hex ☐

                                                +CR +LF

                 ▼ Numbers | as Hex | as ASCII ☐ ☐   SMBUS 8 ▼

[   ] 0 ^C LF Repeats 1 ⬍   ☐ Strip Spaces ☐ Literal   \n
                                                              ☐ Before ☐ After

Dump File to Port

%APPDATA%\Realterm\send.txt ▼ [...] Send File ✖ Stop Delays -2 ⬍ 0 ⬍

-----------   Repeats 1 ⬍ 100 ⬍

Status
☐ Connected
☐ RXD (2)
☐ TXD (3)
☐ CTS (8)
☐ DCD (1)
☐ DSR (6)
☐ Ring (9)
☐ BREAK
☐ Error USB

Send chars as is. Backslash sequences (python style) can be used (unless Literal is checked) eg THIS IS NEWLINE   Chars:114   CPS:0   Port: 2 9600 8N1 None

Revision #4

Created Sun, Nov 10, 2019 2:34 AM by ESINC

Updated Sun, Nov 10, 2019 2:50 AM by ESINC