Midibox CV to extend “DOUTs”

Anyone who considers 2 or 8 gates too few, or wants to trigger vintage drum synths/modules (Roland x0x style), may find the existing hardware of the Midibox CV insufficient. The solution is simple.

Hardware

A dout module is needed, which will be connected to J8 of the coremodule. A doutx4 provides 32 gates/triggers. Some advice at this point: because it will be built without any optocouplers or transistors to protect the DOUT, it is essential not to apply any external voltage to the Gates/Triggers, and also to prevent short circuit.

Software

A few changes have to be made to the sourcecode. On the one hand, the dout has to activate the gate/trigger on receiving the appropriate NoteOn: on the other hand you may want to reduce the duration of an impulse to 1ms independent of the Note duration.

Here as background is the trigger characteristic of some drum machines. The x0x-boxes (606,808,909...) trigger the sound at decrease of voltage at the gate instead of increase.
IE at the trailing edge of the +ve pulse rather than the leading edge? I'm not sure if this is correct. Wikipedia says that Roland and Sequential gear is V-Trig {normal low, set high to trigger} and Moog, Korg and Yamaha is S-Trig {normal high, set low to trigger}. The manual for the Kenton Pro Solo MIDI-CV converter says: “ ... Most synths / sequencers & drum machines will want the Positive edge pulse, but a few require the Negative edge instead. (e.g. Korg Monopoly) .. ” I've also seen written elsewhere that the 606 trigger out is V-Trig, but the 707 internal triggering for it's own voices is S-Trig
Also, if the trailing edge was the trigger, how would gates and envelopes work ie ADSR when AD happens at leading edge, D as gate is held high, and R when gate drops low? - Bunsen

Activate dout:

Download the source of Midibox CV at http://www.ucapps.de/mios_download.html and search for the following in “main.asm”:

```
USER_MPROC_NotifyReceivedEvent
    ;; process MIDI event
    call    CV_MIDI_NotifyReceivedEvent

    ;; for best latency: branch to USER_Tick so that the new CV values
    ;; will be mapped immediately
    rgoto    USER_Tick
```

Replace it with:

```
USER_MPROC_NotifyReceivedEvent

    ;; BEGIN --- control DOUT pins via Note events at channel #1
    movf    MIOS_PARAMETER1, W        ; Note Off -> Note On with velocity 0
    andlw    0xf0
    xorlw    0x80
    bnz    USER_MPROC_NRE_NoNoteOff
USER_MPROC_NRE_NoteOff
```

```
    bsf    MIOS_PARAMETER1, 4
    clrf    MIOS_PARAMETER3
USER_MPROC_NRE_NoNoteOff

    movlw    0x90                ; check for Note On at channel #1
    cpfseq MIOS_PARAMETER1, ACCESS
    rgoto USER_MPROC_NRE_NoNoteChn1
USER_MPROC_NRE_NoteChn1
    ;; MIOS_DOUT_PinSet expects pin number in WREG, value in MIOS_PARAMETER1
    movf    MIOS_PARAMETER3, W        ; velocity == 0: off, velocity != 0:
on
    skpz
    movlw    0x01
    movwf    MIOS_PARAMETER1

    movf    MIOS_PARAMETER2, W        ; pin number: note number - 0x24, we
start with C-2
    addlw    -0x24
    andlw    0x7f
    call    MIOS_DOUT_PinSet
USER_MPROC_NRE_NoNoteChn1
    ;; END --- control DOUT pins via Note events at channel #1

    ;; process MIDI event
    call    CV_MIDI_NotifyReceivedEvent

    ;; for best latency: branch to USER_Tick so that the new CV values
    ;; will be mapped immediately
    rgoto    USER_Tick
```

What happens here? Midibox CV is listening to the first channel (beginning from tune C-2) for a NoteOn and activates the corresponding dout. A NoteOff deactivates the dout.

Define the number of connected DOUT shift registers:

If you're using more than one DOUT shift register, you must also change this code in “main.inc” to reflect the number of shift registers:

```
    ;; initialize the SRIO driver
    movlw    0x01
    call    MIOS_SRIO_NumberSet
```

To set it to the maximum (16), just replace the code with this:

```
    ;; initialize the SRIO driver
    movlw    0x0F
    call    MIOS_SRIO_NumberSet
```

1ms Extension For Vintage Drum Machines:

Those who want to trigger vintage drums have to modify the sourcecode as follows: Search for the

following:

```
USER_SR_Service_Finish
    ;; ---[ handle with control surface variables (flashing cursor, etc) ]--
-
    goto    CS_MENU_TIMER
```

Replace it with:

```
USER_SR_Service_Finish
    clrf    MIOS_PARAMETER1
    movlw   0x00
    call    MIOS_DOUT_SRSet
    movlw   0x01
    call    MIOS_DOUT_SRSet
    movlw   0x02
    call    MIOS_DOUT_SRSet
    movlw   0x03
    call    MIOS_DOUT_SRSet

    ;; ---[ handle with control surface variables (flashing cursor, etc) ]--
-
    goto    CS_MENU_TIMER
```

This leads to a reset of all DOUTs once per cycle - this lasts 1ms. So the drum modules can be triggered with a 1ms latency.

MIDIbox SEQ allows up to 48 digital outputs as this type of trigger.
http://ucapps.de/midibox_seq_options.html

Changing the midi channel:

If you want the DOUT triggers to respond to MIDI events on a channel other than #1, replace instances of “0x80” and “0x90” with the following values:

- Channel 1: 0x80, 0x90
- Channel 2: 0x81, 0x91
- Channel 3: 0x82, 0x92
- Channel 4: 0x83, 0x93
- Channel 5: 0x84, 0x94
- Channel 6: 0x85, 0x95
- Channel 7: 0x86, 0x96
- Channel 8: 0x87, 0x97
- Channel 9: 0x88, 0x98
- Channel 10: 0x89, 0x99
- Channel 11: 0x8A, 0x9A
- Channel 12: 0x8B, 0x9B
- Channel 13: 0x8C, 0x9C
- Channel 14: 0x8D, 0x9D
- Channel 15: 0x8E, 0x9E
- Channel 16: 0x8F, 0x9F

Forum articles:

http://www.midibox.org/forum/index.php/topic,13478.0.html – Information on receiving on ALL channels, information on excluding specific DOUT pins from being reset with the 1ms trigger extension code

http://www.midibox.org/forum/index.php?topic=2701.0 (German)
http://www.midibox.org/forum/index.php?topic=6333.0 (German)

From:
https://wiki.midibox.org/ - **MIDIbox**

Permanent link:
https://wiki.midibox.org/doku.php?id=how_to_use_midibox_cv_with_a_dout

Last update: **2009/05/06 14:23**