FBV-MIDI
Version 1.0 Readme
7/30/18
Kevin Quann
kevin.quann@gmail.com

1. Acknowledgements / references:
This project expands upon initial protocol analysis of a Line 6 FBV longboard foot controller by Vlotech circa 2010-2011. Please find his excellent documentation below:

http://www.vlotech.nl/index.php?q=node/272
https://github.com/vlotech/FBV-tools

Please also see the Axeline/Kempline project:

https://www.youtube.com/watch?v=Ah1ph-CShgo

2. Principles of operation:
Line 6 FBV foot controllers such as the FBV3 and FBV MkII Shortboard are primarily designed to be implemented with compatible "host" Line 6 devices such as the POD modelers, Spider and Firehawk amplifiers, etc. In this arrangement, the foot controller unit transmits MIDI Sysex messages corresponding to changes in inputs (button in/out status, expression pedal positions) and in turn receives corresponding output MIDI Sysex messages from the host unit (LED on/off status, LCD display messages). This occurs via full-duplex serial (RS-485/RS-422) transmissions at MIDI baud rates (31250 bps) between the units.

For instance, depressing the "FS1" button on the FBV transmits the following from Sysex message as shown in figure 1:

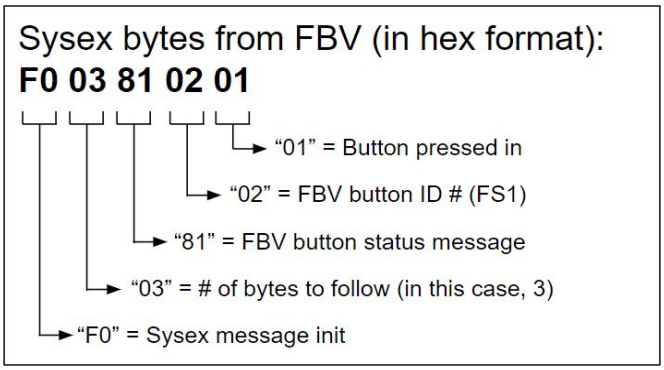

Figure 1: Example Sysex message from FBV controller

When received by a POD HD Pro, the following would be the appropriate response back to the FBV to turn on the FS1 LED (figure 2):

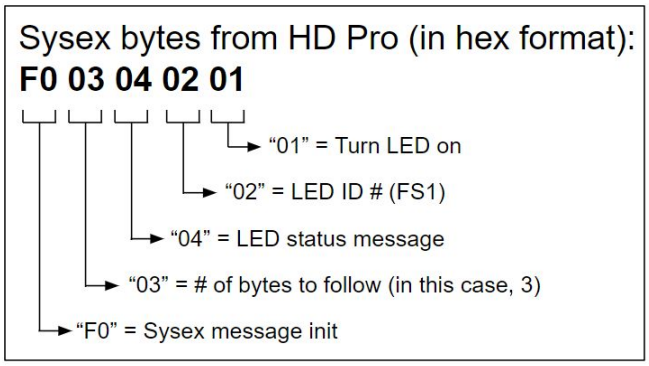

Figure 2: Example Sysex message response from POD HD Pro:

Please see the included *FBV_protocol_analysis.xlsx* for full descriptions of these messages.

These Sysex messages are not useable MIDI commands by themselves. Rather, user-defined MIDI commands must be issued from a MIDI-capable host device. Unfortunately, the FBV foot controllers do not feature traditional 5-pin DIN MIDI connectors to support stand-alone MIDI functionality and instead require some form of host device at all times to serve as an interface.

That being said, the FBV3 and FBV MkII shortboard do offer some basic MIDI-over-USB functionality. In this arrangement, the Line 6 FBV Control application assigns MIDI commands (such as PC or CC messages) to buttons on the controller. Pressing a button in will illuminate its LED and send an assigned MIDI message over USB. However, these units do not respond to MIDI-over-USB inputs and therefore dynamic changes to LED status or LCD display by the host device (such as a computer or other MIDI class-compliant device) is not currently possible. Furthermore, LED color cannot be assigned to the FBV3, nor can multiple MIDI commands (ie presets) be issued from a single button.

This project makes use of an Arduino Mega and Raspberry Pi to effectively serve as a low-cost, highly-customizable host for the FBV3 and MkII shortboard series of foot controllers and expand their functionality. In this configuration (figure 3), Line 6 Sysex messages are sent and received over Cat-5 between the FBV and an Arduino Mega with a Max488 full-duplex serial interface chip. Sysex data is then passed over USB to a Raspberry Pi where it is interpreted by a Python script (*FBV_MIDI.py*) and corresponding LED/LCD changes are made. User-defined MIDI commands are also issued as outlined in a user-configurable TSV file (ie, *Config.txt*). The Arduino Mega was chosen as it has a total of 4 UARTs and could serve as a serial multiplexer/demultiplexer between FBV Sysex data, MIDI in/out, and (optional) a 2nd Line 6 device for purposes of passing tuner Sysex data (which is only provided over RJ45). Power to the foot controller is supplied via a separate 9V DC adapter over Cat-5 patch cable. Please see *FBV_MIDI.py* and *FBV_MIDI.ino* for details regarding multiplexing/demultiplexing.

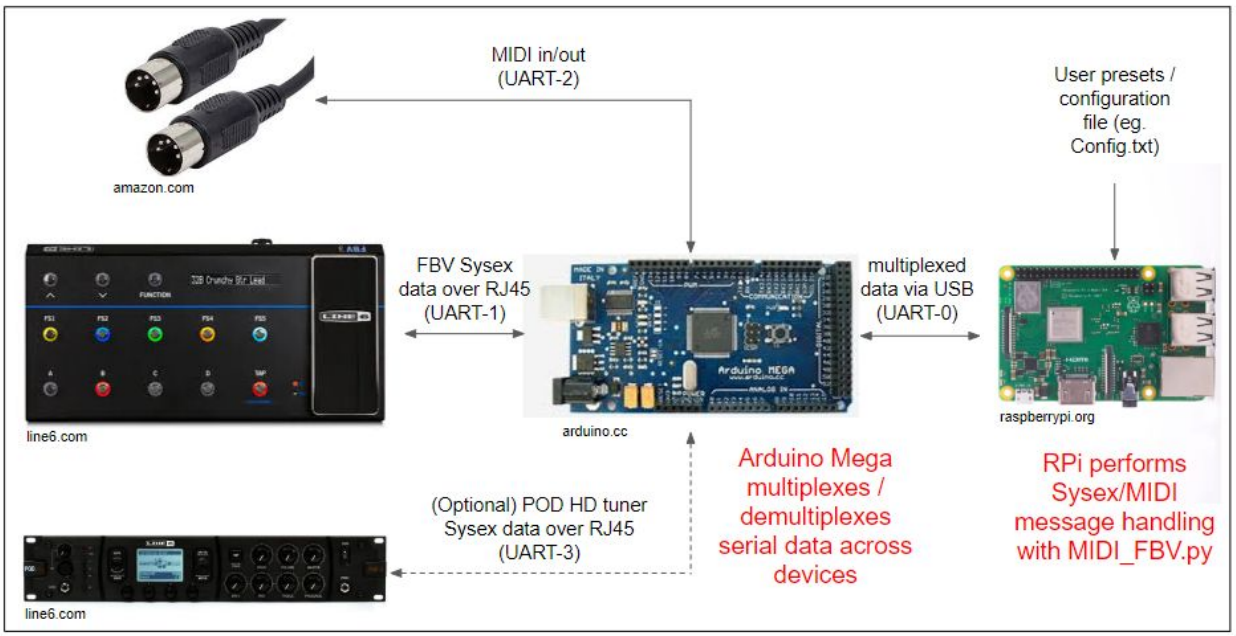

Figure 3: FBV-MIDI operating scheme

3. Setup:

Constructing the Arduino interface:

Below is the complete schematic for the Arduino interface (figure 4), this is also included as separate PDF and Fritzing files. A parts list is also included (*Parts_list.xlsx*).

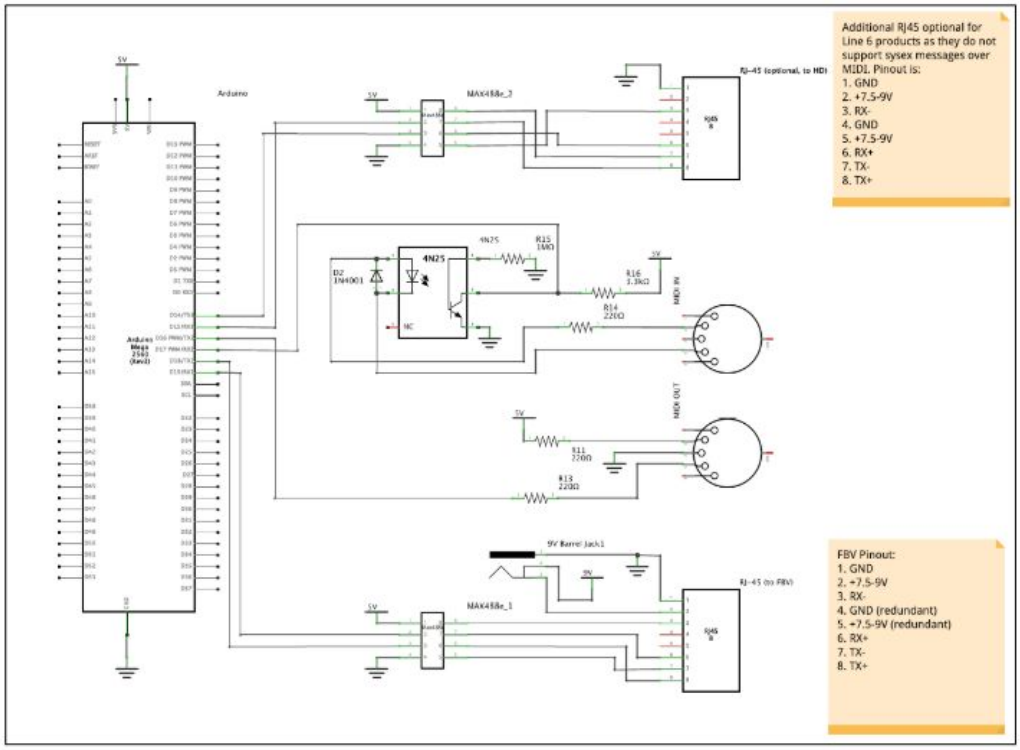

Figure 4: Arduino Mega interface schematic

ICs, resistors and diodes were soldered onto strip-board with female headers. This was then mounted in a Hammond 1591 enclosure with an RJ-45 jack (to FBV), 9VDC barrel jack, 5-pin MIDI in/out jacks, and a USB feed-through to the Arduino. An additional RJ-45 jack with Max488 chip may be placed (as shown) if using a Line 6 device for tuner information, though this is not necessary if using other devices capable of sending tuner information over a traditional MIDI-out jack. The internal components of the Arduino interface enclosure are shown below in figure 5. Additional images are included for reference.

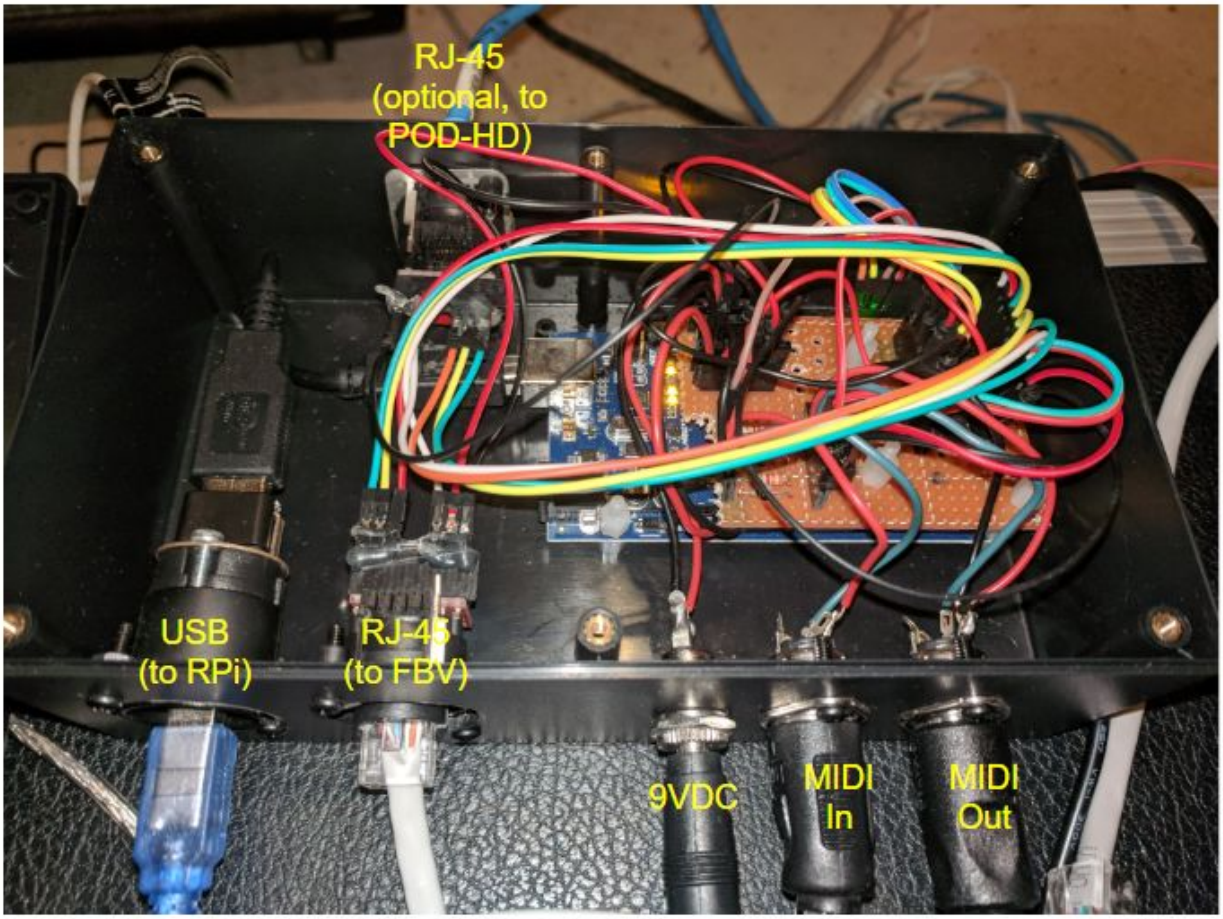

Figure 5: Arduino Mega interface

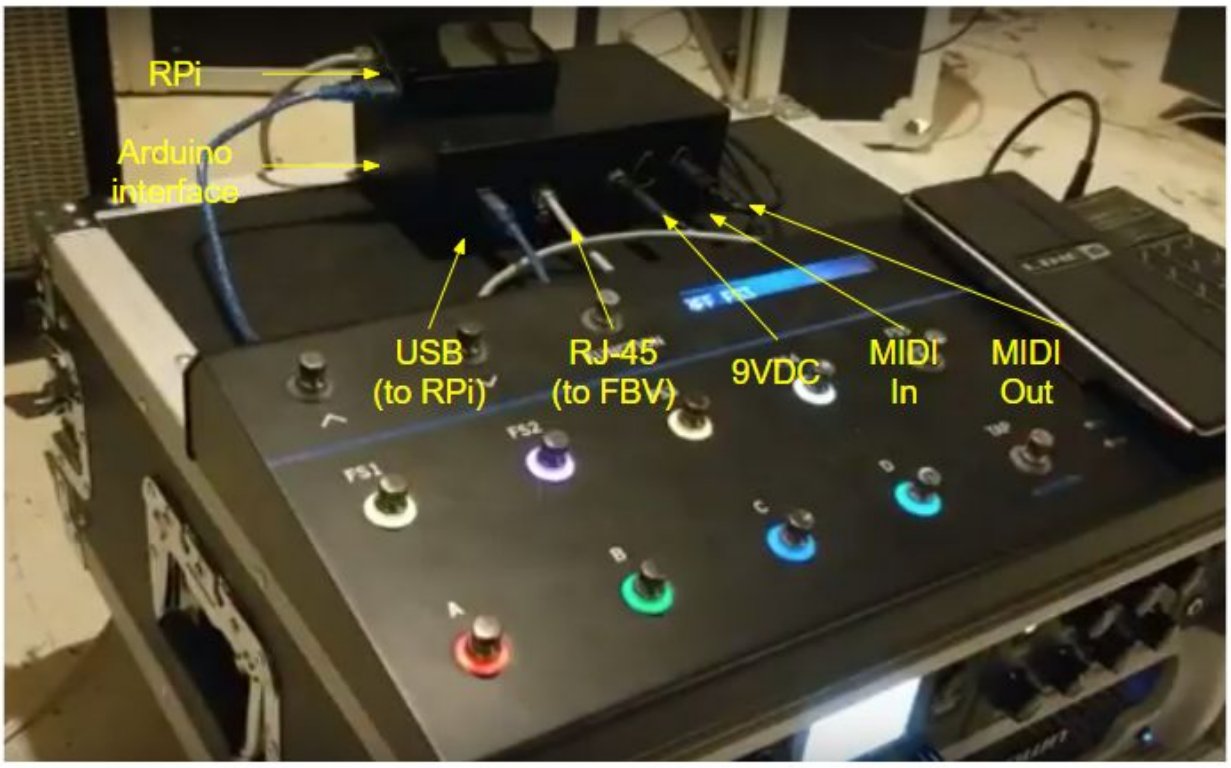

Figure 6: Completed configuration with Arduino interface and Raspberry Pi

Setting up Arduino Mega:
Download and install the Arduino IDE (may be done on Raspberry Pi):
https://www.arduino.cc/en/Main/Software
Upload the *FBV_MIDI.ino* sketch to the board (must be Arduino Mega or compatible).

Setting up Raspberry Pi:
Initialize a Raspberry Pi with Raspbian (includes Python 2.7) as follows:
https://www.raspberrypi.org/documentation/setup/
Install the following Python dependencies from command:

```
$sudo pip install mido #MIDI library used in script
```

Setting up user configuration file:
By default, this is named *Config.txt*. This tab-separated value (TSV) file contains all of the configurable parameters available to the user which determines how many "pages" of buttons are available, if buttons are assigned to behave as “instant access” type or “preset”, etc. This may be modified as needed. Please see the comments included in the Config.txt file for further documentation.

Running Python script:
Make script executable:

```
$chmod +x FBV_MIDI.py
```

Execute as follows:

```
./FBV_MIDI.py [-c / --config-file=Config.txt] [-d / --debug_mode=off]
[-f / --FBV_pass_through=off] [-m / --model=FBV3] [-s /
--serial-port=/dev/ttyAMC0] [-t / --thru=off]
```

Description of flags:

`-c / --config-file=<path/to/filename.txt>`

Default="`./Config.txt`". This is the path to the configuration file with user settings.

`-d / --debug_mode=<on|off>`

Default="`off`". When enabled this prints serial in/out data across all serial ports with timestamps to stdout

`-f / --FBV_pass_through=<on|off>`

Default="`off`". When enabled, FBV Sysex messages are forwarded to optional 2nd RJ-45 port (for other Line 6 devices), bypassing the Python processing script. Used with `--debug_mode=on`, allows for protocol analysis between FBV and Line 6 host units.

`-m / --model=<FBV3|MkII>`

Default="`FBV3`". If selecting "`MkII`", will used the appropriate button/LED layout for use as per figures 7 and 8.

`-s / --serial-port=<path/to/arduino/serial>`

Default="/dev/ttyACM0". This may vary if multiple Arduinos are connected simultaneously.

-t / --thru=<on|off>

Default="off". By default, data received at MIDI-in port is compared to configuration file to determine if a listed effect was toggled. If so, bypass/LED status is updated accordingly and a MIDI command is issued to MIDI-out to turn effect on/off. If this option is turned "`on`", all data received at MIDI-in is echoed at MIDI-out (though still compared against configuration file).

4. Usage:

On execution of the FBV_MIDI.py script, the user configuration file is loaded and the FBV controller becomes operational. Pressing the "page-up" and "page-down" buttons cycles through banks of 9 multi-function buttons (physically labeled 0-8). For instance page #0 displays the LED states of buttons #0-8, however after scrolling up to page #1 this updates to buttons #9-17. Scrolling up further to page #2 reveals buttons #18-26 and so on. The configuration file may be edited to provide the desired number of assignable buttons. For instance if 10 pages are listed,

90 total buttons will be available ((10 pages) x (9 buttons / page) = 90 total buttons). Buttons may be assigned either “instant access” type, whereby they act as simple on/off switches, or “presets” and linked to multiple other buttons to turn on or off multiple effects simultaneously. Scrolling through pages does not affect the bypass status of an effect. For instance, if the user switches on an “instant access” effect assigned to button #5 on page #0, they may scroll through other pages and come back to page #0 to find button #5 is still on. However, if for instance the user scrolls to page #3 and activates a “preset” effect assigned to button #28 which turns off multiple effects simultaneously (including button #5 in this example), when scrolling back to page #0, button #5 will be off. MIDI commands will be issued as assigned. Setting the “`-m / --model=<FBV3|MkII>`” option as described above determines which foot controller will be used. Physical button assignments for FBV3 and FBV Shortboard MkII are demonstrated in figures 7 and 8, respectively.

Figure 7: FBV3 physical button assignments

Figure 8: FBV Shortboard MkII physical button assignments. Note all buttons have an associated red, non-dimmable LED unless otherwise specified. Also note “button 4” and “page down” do not reflect the unit labeling. This was done for consistency with FBV3 units.

5. Working / tested functions:

-Page up/down
-Presets
-MIDI CC messages
-Basic MIDI in
-Multi-color LEDs
-Tuner pass-through (from Line 6 POD HD Pro)
-Expression pedals 1 & 2
-See video demo of project at:
https://www.youtube.com/watch?v=3a9S2EJr6ko&feature=youtu.be

6. To do:

-Test with Fractal / Kemper units (will require addition of sysex via MIDI-in)
-Implement function button to some extent
-Test program change MIDI signals
-Much more…..