MEGA MIDI

5

Sega Genesis / Megadrive MIDI Synthesizer Module

User Manual

Aidan Lawrence

www.aidanlawrence.com

November 2019

Contents

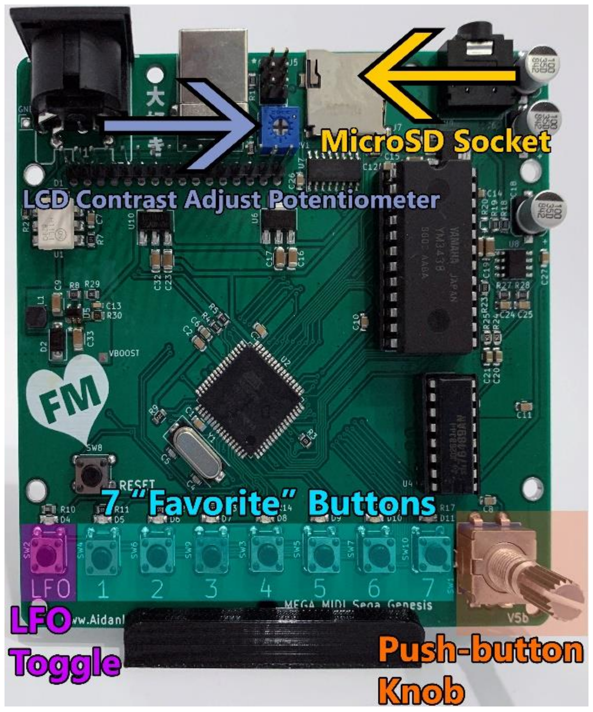
Quick Start

The Mega MIDI module is designed to be easy-to-setup and use compared to it's contemporaries. In order to use the Mega MIDI module, you must have:

- A microSD card (plus an SD card reader for your PC)
- Either a standard USB-B cable and/or a classic 5-pin MIDI DIN cable
- 3.5mm Audio cable
- Users on Windows 7, 8 or XP: The Teensyduino device driver. (If you plan to use USB MIDI)

Plug-in SD Card with .OPM files. Plug-in Audio Jack. Plug-in USB power. Optional: Plug in MIDI cable.

Device Guide



Mega MIDI Module without it's LCD

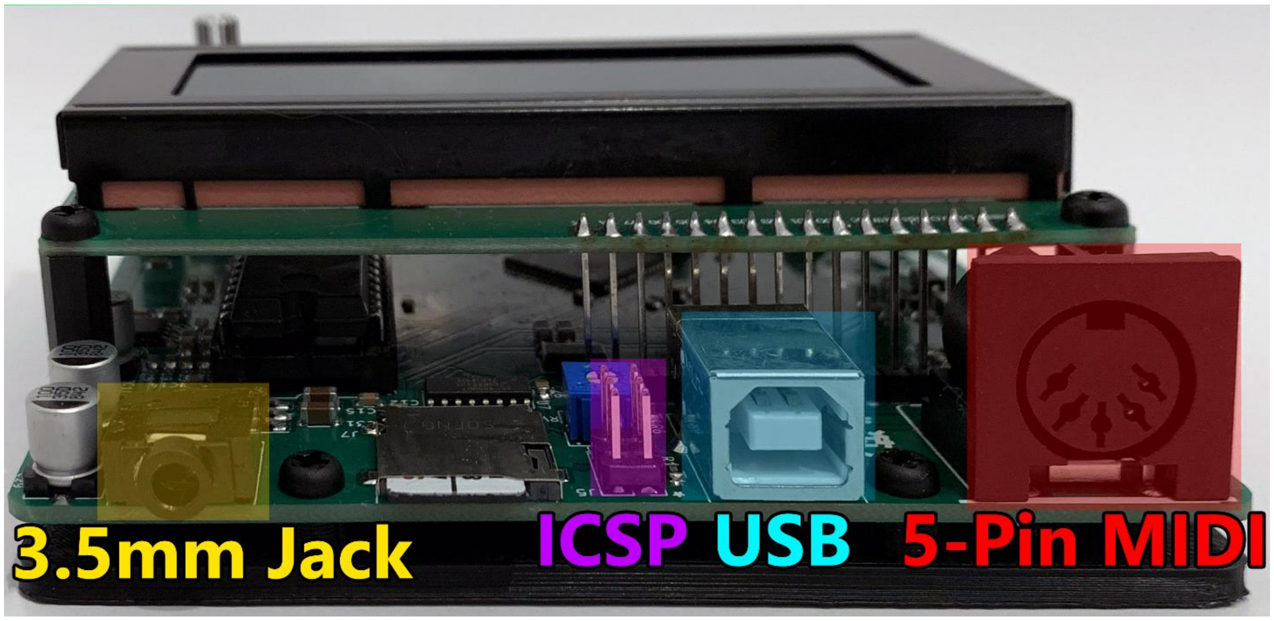
LFO Toggle: Low Frequency Oscillator Toggle. Toggle this button to activate/deactivate the Low Frequency oscillator. Control the frequency of the LFO using the MOD wheel on your MIDI controller.

Favorite Buttons: If you find a voice you'd like to have quick access to, press and hold one of the seven "favorite" buttons. After 2 seconds, your currently loaded voice will be saved as a favorite. Press your selected favorite button at any time to instantly load your saved patch. Hit the same favorite button again to go back to the current SD card patch.

Push-button Knob: AKA Rotary Encoder. Use this knob to navigate the LCD menu. Press the knob to move the cursor. Rotate the knob left and right to adjust the values where the cursor is pointing.

LCD Contrast Adjust: Use a screwdriver to rotate this potentiometer to adjust the LCD contrast.

MicroSD Socket: Insert your microSD card here.



3.5mm Audio-Out Jack: The main line-level audio output jack.

ICSP: In-Circuit Serial Programming header. Use this to reflash your firmware.

USB: Standard USB Type-B full-sized port. Use this to communicate with the Mega MIDI.

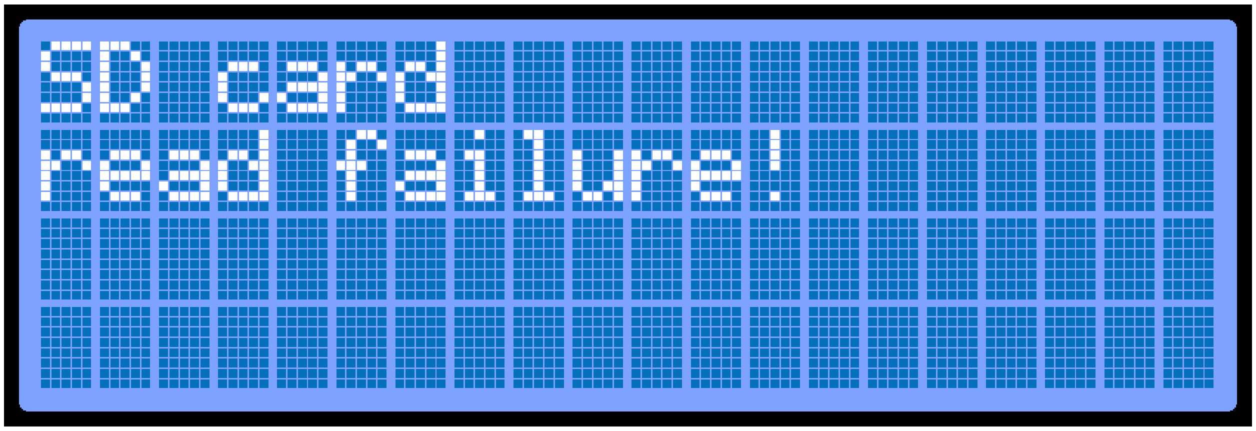
5-Pin MIDI DIN: Standard traditional MIDI DIN input port. Connect directly to a MIDI controller that also features a traditional MIDI port for instant "plug-and-play."

SD Card Setup

You must format your card as a **FAT32 device**. You can simply use your built-in operating system tools to format your card or the official formatting tool from the SD Association.

All files placed on the SD card must be in the root directory (no folders). Multiple partitions are not supported.

If your SD card fails to read, the Mega MIDI's face buttons will light-up in an alternating pattern and the LCD will read the following:


SD card
read failure!

If this SD read error occurs, simply remove and reinsert the SD card and press the **RESET button** under the LCD near the rotary encoder knob. If this issue persists, ensure that your card is not physically damaged and that it has a single partition formatted as FAT32.

OPM Voice Files

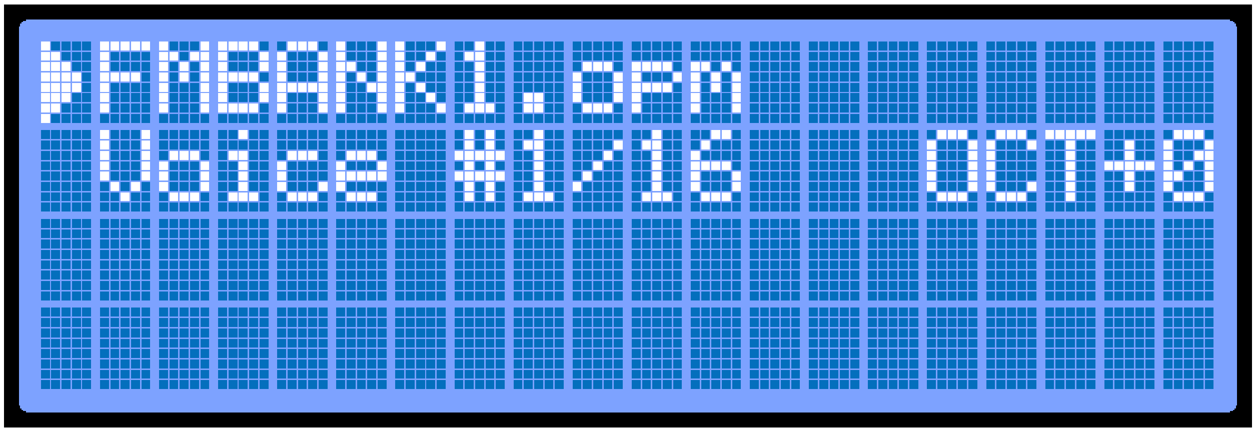
".OPM" files are used by the Mega MIDI to read-in premade voice patches. The OPM file format was created for use with the VOPM virtual synthesizer. OPM files are "plain-text" register dumps for Yamaha OPNx and OPM sound chips and contain the information necessary to program the voices for said chips.

There are several ways to obtain these OPM files.

1) Download the OPM file pack here. This contains thousands of OPM files from hundreds of Genesis games. I don't recommend putting them all on the SD card at once though, as performance will really take a hit. Instead, it is recommended to limit yourself to 200 files maximum. Here is a great FM bank collection converted to OPM from the Yamaha FB01.
2) Generate OPM files from VGM/VGZ files. Within the **tools** folder of the Mega MIDI repository, I have a few scripts and tools to help you convert VGM (Video Game Music) files to OPM files. Vgm2Opm is provided courtesy of shiru. If you'd like to easily use this tool, I recommend cloning the entire repository on a Windows computer and using the custom batch scripts I wrote.
   a. Simply place your Genesis/OPN VGM files into the "VGM_IN" folder
   b. Double-click the "CONVERT_VGM_OPM.bat" file (windows only)
   c. Retrieve your new OPM files from the OPM_OUT folder
3) Create your own OPM files in VOPM. This requires installing the VOPM virtual synthesizer into a DAW of your choice. Using VOPM, you can set all of the individual FM registers to make custom patches and then export them using the "Export" button in the top left corner. This method is only recommended for advanced users as creating FM patches is frankly a bit of a nightmare.

Whichever way you chose to go about obtaining OPM files, simply place them onto the root directory of your micro SD card. Eject the card from your PC and insert the card into the MEGA MIDI. Power on the system and if everything was successful, you should see the OPM file load automatically after the title screen.

The Main Menu

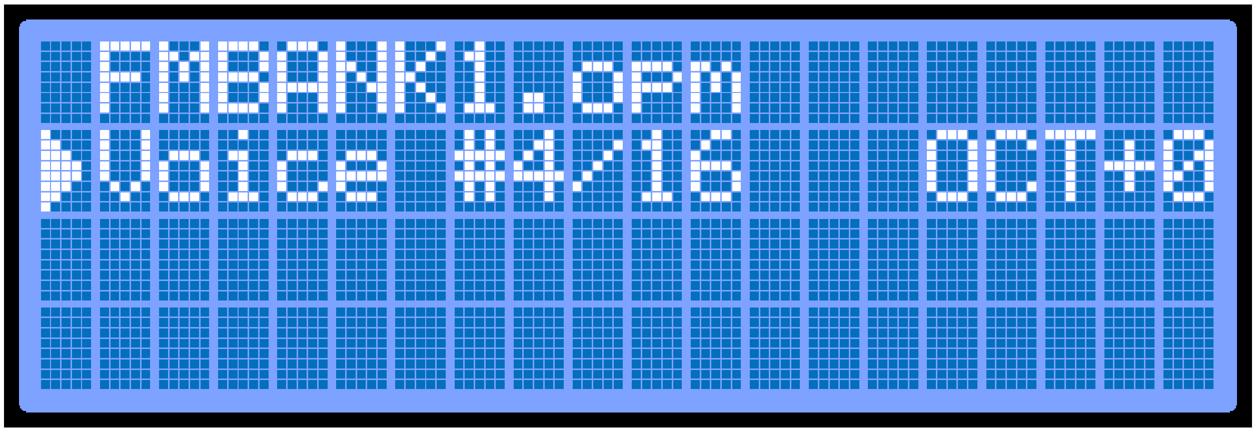


```
▶FMBANK1.opm
 Voice #1/16   OCT+0
```

On the default main menu, you will see three options and a triangle-shaped cursor. **Press the knob to move the cursor to the next option.**

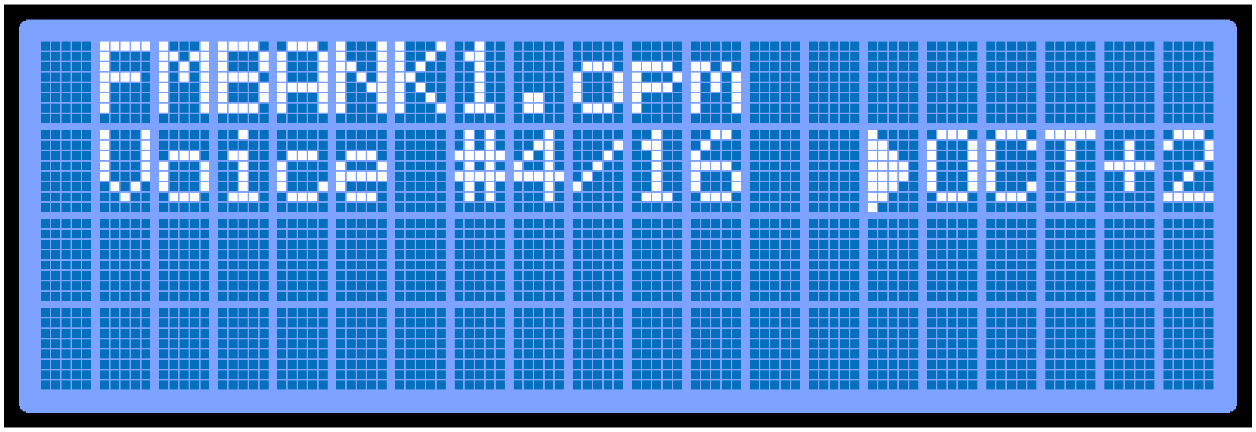
The first line shows you which OPM file is currently loaded. **Rotate the knob** to load the next or previous OPM files from the SD card. (Files do not have any particular order)

The second line shows both the Voice number and the Octave Shift. Rotate the knob while the cursor is on the “Voice” option to change between the various voices on each OPM file.



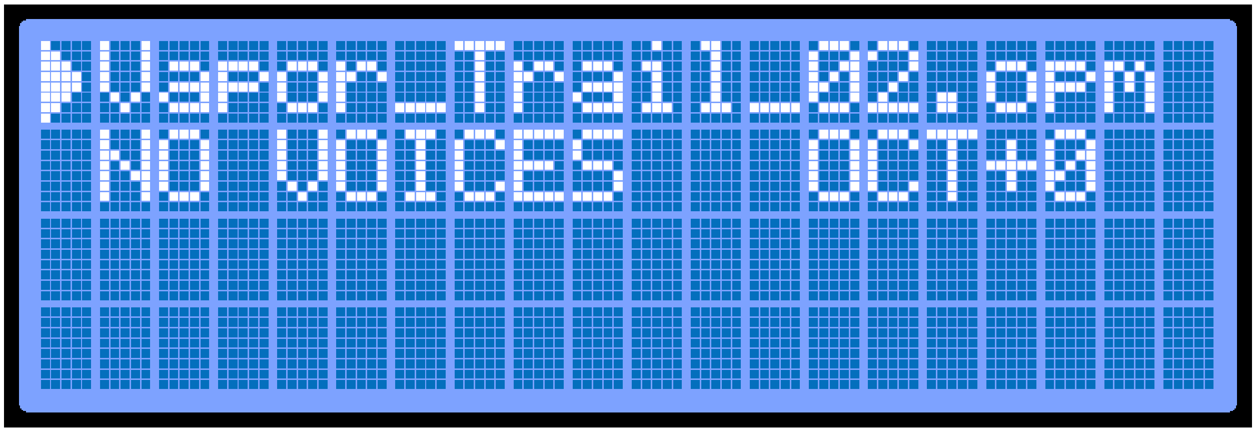
```
 FMBANK1.opm
▶Voice #4/16   OCT+0
```

Finally, the “OCT” option controls the octave shift adjust. If the voice you are using sounds too low or too high for your particular MIDI controller, you can adjust the OCT value by once again rotating the knob when the cursor is on the OCT option. You can adjust this OCT value between -3 and +3.



```
 FMBANK1.opm
 Voice #4/16  ▶OCT+2
```

If an OPM file does not contain any valid voices, the FM sound-source will be disabled. Sometimes, Sega Genesis tracks will *only* use the PCM channel of the YM2612 and will not use any FM patches, therefore, some OPM files may show the “NO VOICES” warning. The Mega MIDI does not support PCM samples, only FM patches.

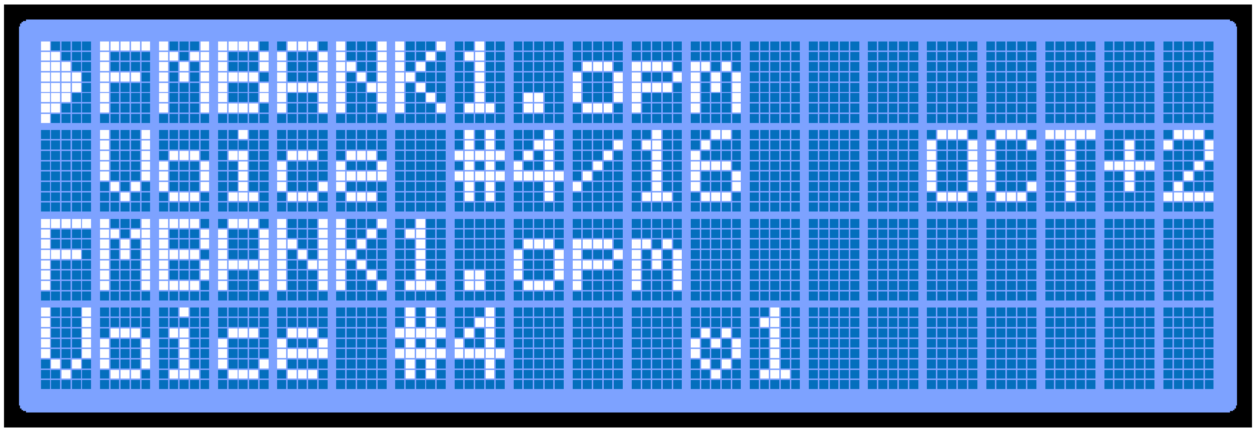


```
▶Vapor_Trail_02.opm
 NO VOICES    OCT+0
```

Saving Favorite Patches

If you’ve loaded a particular patch that you’d like to have quick access to, you can save it as a favorite. On the bottom of the Mega MIDI board, you will see seven buttons (labeled 1-7) that represent a single favorite patch each.

To save a new favorite patch, simply hold down the favorite button you’d like it assigned to. After about two seconds, the favorite patch will be saved into the microcontroller’s EEPROM. You will then notice the favorite LED blink to indicate a new favorite has been saved and the bottom two LCD rows update to show that the new favorite patch is currently loaded. The octave setting of the favorite patch is also saved and is automatically applied when the patch is loaded.

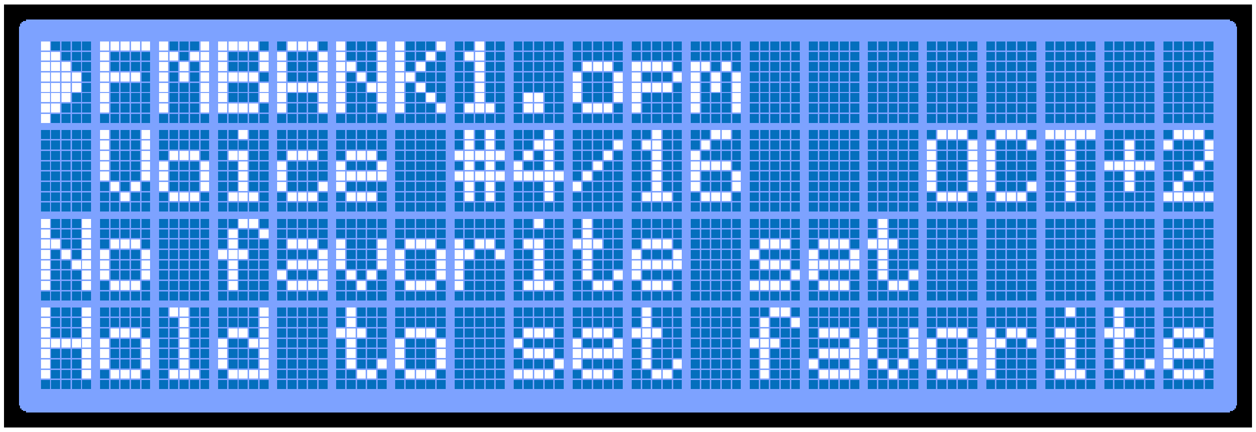


```
▶FMBANK1.opm
 Voice #4/16    OCT+2
FMBANK1.opm
Voice #4    ♡1
```

You can overwrite a favorite patch by simply holding down the same button again on a new patch.

You can load a favorite patch by tapping a favorite button that has a patch saved to it. Tapping the same favorite button again will swap back to the currently loaded SD card patch.

If the favorite button you selected does not have a patch saved to it, you will see a screen like this:


▶FMBANK1.opm
 Voice #4/16   OCT+2
No favorite set
Hold to set favorite

Note: If you reflash the firmware without disabling the EEPROM erase, your favorite patches will be reset!

MIDI Channels

By default, there are four MIDI channels

CH 1 – YM2612 Standard FM

CH 2 – SN76489 Standard PSG

CH 3 – YM2612 FM with Velocity

CH 4 – SN76489 PSG with Velocity

Change the MIDI channel in your DAW or on your MIDI controller to swap between these four options.

Note: Since the YM2612 only has six channels, enabling velocity control may cause some voices with larger ASDR cycles to sound broken or inconsistent.

YM2612 FM Synthesizer

The YM2612 FM synthesizer IC is the same exact chip found in Model 1 Sega Genesis/Megadrive consoles. This is the genuine sound chip and is not emulated. That means the sound you receive out of this device is as “real” as you can get. It also means you are constrained to the hardware limitations of the genuine IC. The YM2612 only has six channels, meaning you can only have six individual notes playing at one time. If you press more than six keys at once, the eldest key pressed will be overwritten with the new note value. You can access the YM2612 on MIDI channel 1 or MIDI channel 3 if you’d like to have velocity control.

YM3438 FM Synthesizer

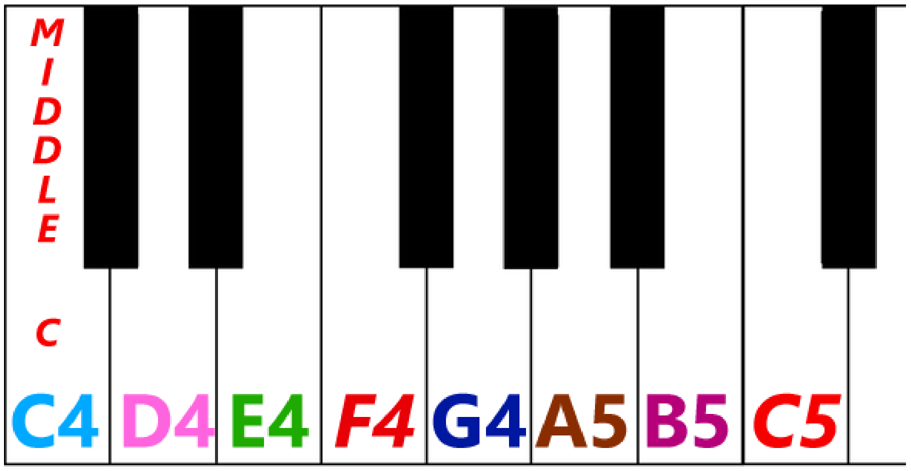
The YM3438 is the CMOS variant of the YM2612. It is pin and register compatible. This IC uses far less power than the YM2612 and is known for having a “cleaner” DAC. The YM3438 was found in ASIC form on subsequent models of the Sega Genesis. You can select a YM3438 when you buy your Mega MIDI.

SN76489 Programmable Sound Generator

The SN76489 PSG was originally present in the Genesis' predecessor, the Master System, but was ported over to the Genesis for backwards compatibility. It was still exposed to programmers of Genesis titles, however, so the PSG was often used in tandem with the YM2612. Again, since this is not an emulated IC, rather the genuine chip, you must be aware of the technical limitations. The PSG offers 3 channels of square waves. There is also a noise channel which is accessible using MIDI channel 5. Unlike the YM2612, notes are not overwritten with the PSG, so if you have pressed three keys, you will not hear any new note should you press a fourth.

You can access the PSG with MIDI channel 2 or MIDI channel 4 if you'd like to have velocity control.

PSG Noise Control (Firmware 1.2.6+)



MIDI Channel 5 gives you access to the special noise generator on the PSG. The PSG has several different noise settings, some of which are dependent on the frequency of the PSG's channel 3 tone generator.

Once you are on MIDI channel 5, use keys C4 *(Middle C)* through C5 to control the noise generator.

C4 – Periodic noise. Shift rate is set to PSG Clock/512

D4 - Periodic noise. Shift rate is set to PSG Clock/1024

E4 - Periodic noise. Shift rate is set to PSG Clock/2048

F4 - Periodic noise. Shift rate is set to PSG Clock/CH3 **(CONTROL CH3 FREQUENCY WITH MOD WHEEL)**

G4 - White noise. Shift rate is set to PSG Clock/512

A5 - White noise. Shift rate is set to PSG Clock/1024

B5 - White noise. Shift rate is set to PSG Clock/2048

C5 - White noise. Shift rate is set to PSG Clock/CH3 **(CONTROL CH3 FREQUENCY WITH MOD WHEEL)**

Tip: *If you have two MIDI controllers, you can assign one to channel 1(3) and the other to channel 2(4) to play both the PSG and the YM2612 at the same time!*

Advanced Features

Control over Serial

The Mega MIDI uses an emulated serial system over the HID protocol that can be accessed using the "teensy_gateway" tool and a properly configured terminal. Inside of the **tools** folder of the Mega MIDI repository, you will find a batch file called, "**SERIAL_CONNECT.bat**." If you are using Windows, simply double-click this batch file and two windows will open. A command line window and a PuTTY window. Keep both of these windows open and look for the PuTTY terminal window that reads "teensy_gateway"

This PuTTY window is your serial console. Simply type any of the following commands into this console and hit "enter."

Here is the command list:

COMMAND	DESCRIPTION
`+`	Move up one voice in current OPM file
`-`	Move down one voice in current OPM file
`o`	Dump current voice operator info
`l (lower-case L)`	Toggle the Low Frequency Oscillator
`>`	Shift up one octave for the YM2612
`<`	Shift down one octave for the YM2612
`?`	List the currently loaded OPM file name
`!`	Reset the sound chips
`r:FILENAME.OPM`	Request a file name to load. Case-sensitive.
`d`	Dump the YM2612's current register values

Flashing Firmware (Programming)

The Mega MIDI is open source. You can find the source repository here:

https://github.com/AidanHockey5/MegaMIDI

Precompiled HEX files can be found here: https://github.com/AidanHockey5/MegaMIDI/releases

Mega Flasher Program

By far, the easiest way to program your Mega MIDI is by using the Mega Flasher program. It is written in Python, and is therefore able to run on any operating system. Installing Python is not required for the Windows stand-alone version of this program unless you would like to run the script directly. Linux and MacOS users must install python3 and run the Python script directly.

Windows Stand-alone download

Universal Python Script

Alternatively, you can clone the repository and navigate to the tools folder to run the MegaFlasher.py file.

A video guide for firmware flashing can be found here

You will need to create an ArduinoISP programmer before running this script!

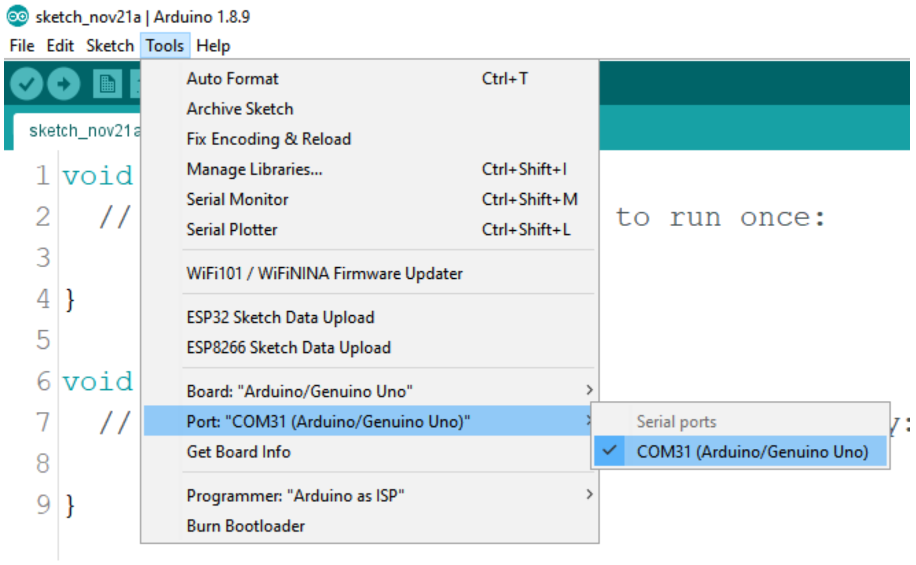
Instructions are on the next page

Warning: This flashing script does not preserve EEPROM. Your favorited patches will be lost!

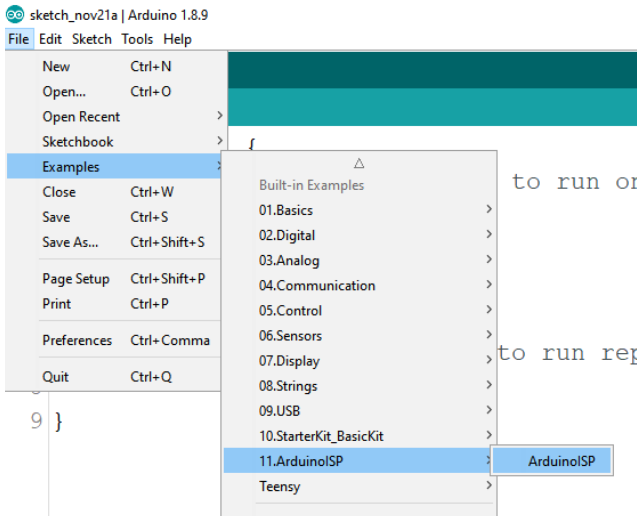
Creating your Arduino Programmer

The Mega Flasher requires that you create a programmer using any standard Arduino variant. You may use any 5V Arduino Uno, Arduino Nano, or Arduino Mega. Whichever Arduino you use, the pins used should be the same. **You will also need a 10uF or greater capacitor!**

1) Install the Arduino IDE. Installing the Arduino IDE will give you access to the ArduinoISP script as well as install all of the necessary drivers required.
2) Plug-in your Arduino with nothing else connected to it.
3) Open the Arduino IDE. Go to the *tools* menu and set your board. Then set your serial port. **Make sure to remember this port for later.**


sketch_nov21a | Arduino 1.8.9
File Edit Sketch Tools Help
Auto Format Ctrl+T
Archive Sketch
Fix Encoding & Reload
Manage Libraries... Ctrl+Shift+I
Serial Monitor Ctrl+Shift+M
Serial Plotter Ctrl+Shift+L
WiFi101 / WiFiNINA Firmware Updater
ESP32 Sketch Data Upload
ESP8266 Sketch Data Upload
Board: "Arduino/Genuino Uno"
Port: "COM31 (Arduino/Genuino Uno)"
Get Board Info
Programmer: "Arduino as ISP"
Burn Bootloader
Serial ports
COM31 (Arduino/Genuino Uno)

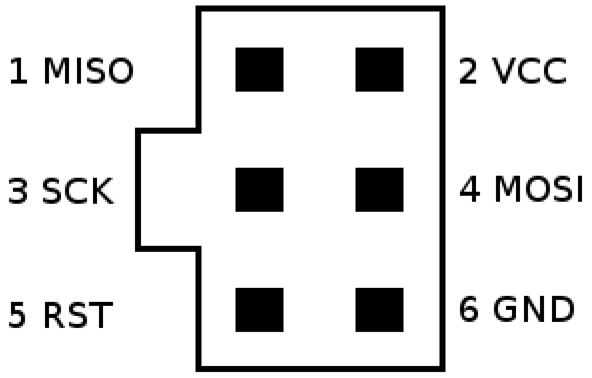
4) Next, go to File->Examples->ArduinoISP->ArduinoISP. Upload this sketch to your Arduino. Assuming everything goes well, you now have your programmer!



5) Finally, connect a 10uF or greater capacitor between the Arduino's RESET and GND pins. (Negative end of the capacitor connects to GND)

Connecting your programmer to your Mega MIDI

The Mega MIDI's ICSP header has the following pin-out. Refer to the silk-screen * on the board to orient yourself. The ICSP header is a set of six pins at the top of the board near the USB port.



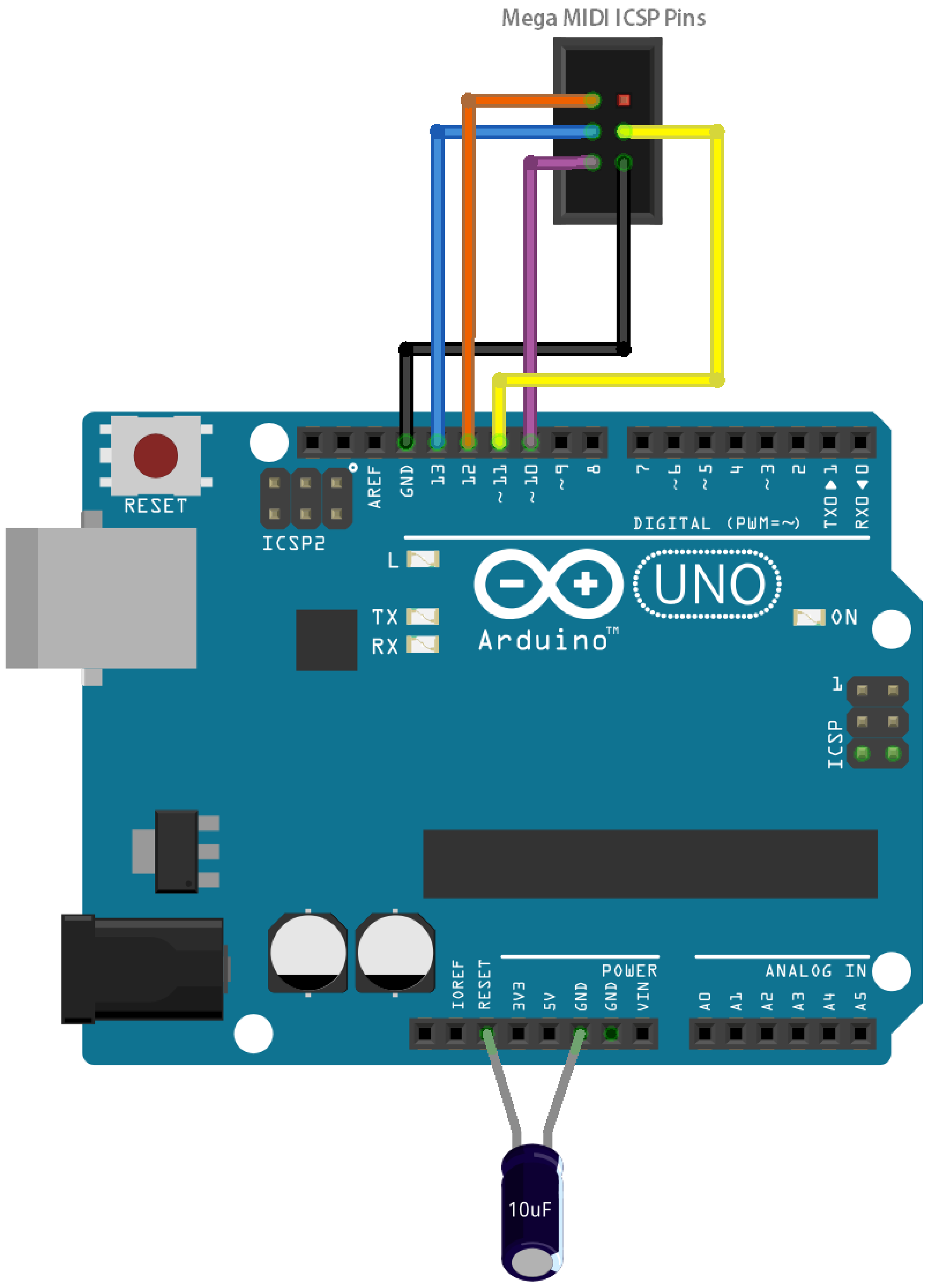
Connect the following pins from the Arduino to the Mega MIDI:

ARDUINO	MEGA MIDI
GND	6 GND
10	5 RST
11	4 MOSI
12	1 MISO
13	3 SCK
DO NOT CONNECT	2 VCC

You do not have to connect VCC. Connect both the Arduino and the Mega MIDI to USB instead. Do not use a 3.3V Arduino for this process.

Power Up your Mega MIDI and make sure your Arduino board is connected to your computer via USB.

Remember, you will need a 10uF or greater capacitor between the Arduino's RESET and GND pins, ***not* the Mega MIDI's reset and ground.**


Mega MIDI ICSP Pins
RESET
ICSP2
AREF
GND
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
TX0▶1
RX0◀0
DIGITAL (PWM=~)
L
TX
RX
Arduino™
UNO
ON
ICSP
1
IOREF
RESET
3V3
5V
GND
GND
VIN
POWER
ANALOG IN
A0
A1
A2
A3
A4
A5
10uF
fritzing

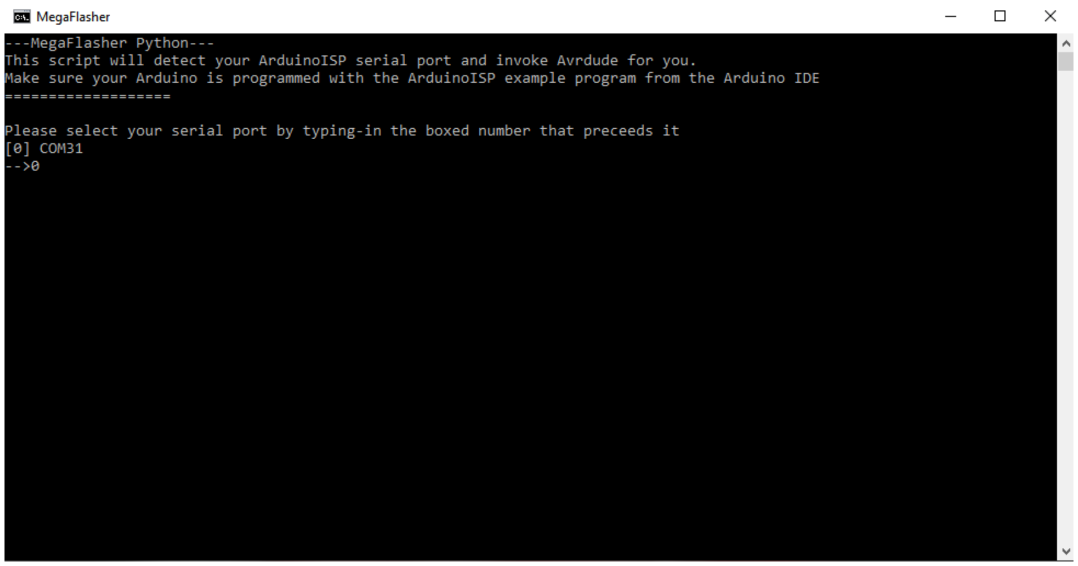
Running the Mega Flasher Program

Windows:

Windows is the easiest platform to use when flashing your MegaMIDI. A python installation is not required. Simply download the Windows Standalone version of the Mega Flasher program.

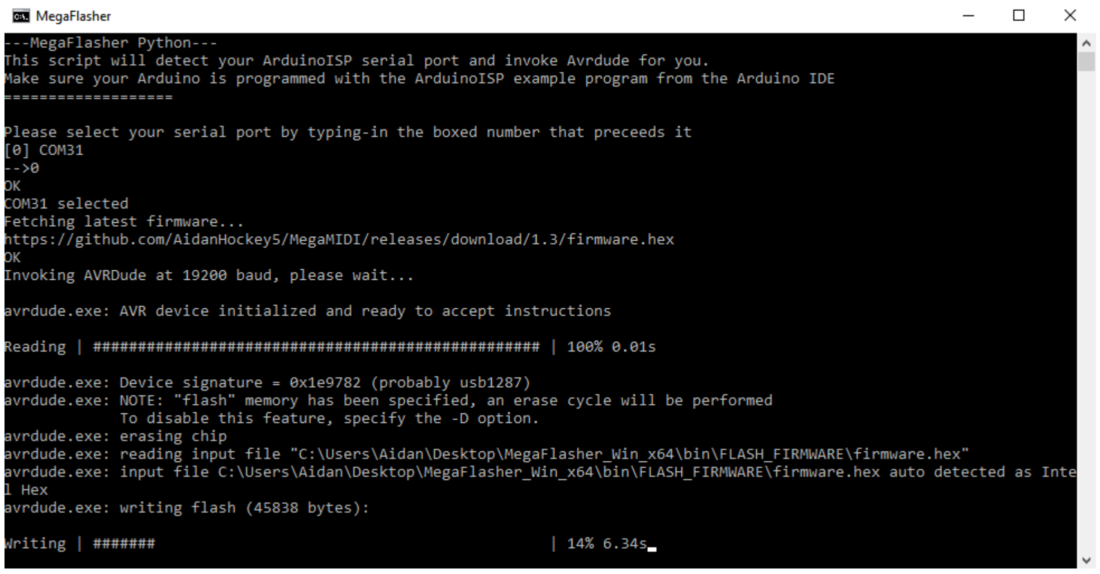
Connect your Mega MIDI up to your Arduino programmer as shown in the previous pages. Connect your Arduino and the Mega MIDI up to your computer via USB, then run Mega Flasher by running MegaFlasher.bat.

You will be asked for the serial port. If you are unsure of which serial port to use, check inside of the Arduino IDE under tools->port.



```
---MegaFlasher Python---
This script will detect your ArduinoISP serial port and invoke Avrdude for you.
Make sure your Arduino is programmed with the ArduinoISP example program from the Arduino IDE
==================

Please select your serial port by typing-in the boxed number that preceeds it
[0] COM31
-->0
```

In this example, I know that my Arduino is on COM31, which the Mega Flasher shows as option 0. Therefore, simply type ‘0’ and hit enter. If your wiring is correct, the program will automatically update you to the latest firmware. It will take about 1 minute for this process to complete.



```
---MegaFlasher Python---
This script will detect your ArduinoISP serial port and invoke Avrdude for you.
Make sure your Arduino is programmed with the ArduinoISP example program from the Arduino IDE
==================

Please select your serial port by typing-in the boxed number that preceeds it
[0] COM31
-->0
OK
COM31 selected
Fetching latest firmware...
https://github.com/AidanHockey5/MegaMIDI/releases/download/1.3/firmware.hex
OK
Invoking AVRDude at 19200 baud, please wait...

avrdude.exe: AVR device initialized and ready to accept instructions

Reading | ################################################## | 100% 0.01s

avrdude.exe: Device signature = 0x1e9782 (probably usb1287)
avrdude.exe: NOTE: "flash" memory has been specified, an erase cycle will be performed
             To disable this feature, specify the -D option.
avrdude.exe: erasing chip
avrdude.exe: reading input file "C:\Users\Aidan\Desktop\MegaFlasher_Win_x64\bin\FLASH_FIRMWARE\firmware.hex"
avrdude.exe: input file C:\Users\Aidan\Desktop\MegaFlasher_Win_x64\bin\FLASH_FIRMWARE\firmware.hex auto detected as Inte
l Hex
avrdude.exe: writing flash (45838 bytes):

Writing | #######                                            | 14% 6.34s
```

MacOS

MacOS comes with a version of python installed, however, you may need to download the full version of Python 3 from the official python website to enable pip support.

https://www.python.org/downloads/

You will also require the Arduino IDE for driver support

https://www.arduino.cc/en/main/software

After you have python3, open terminal and run:

```
python3 -m pip install pyserial
```

```
python3 -m pip install requests
```

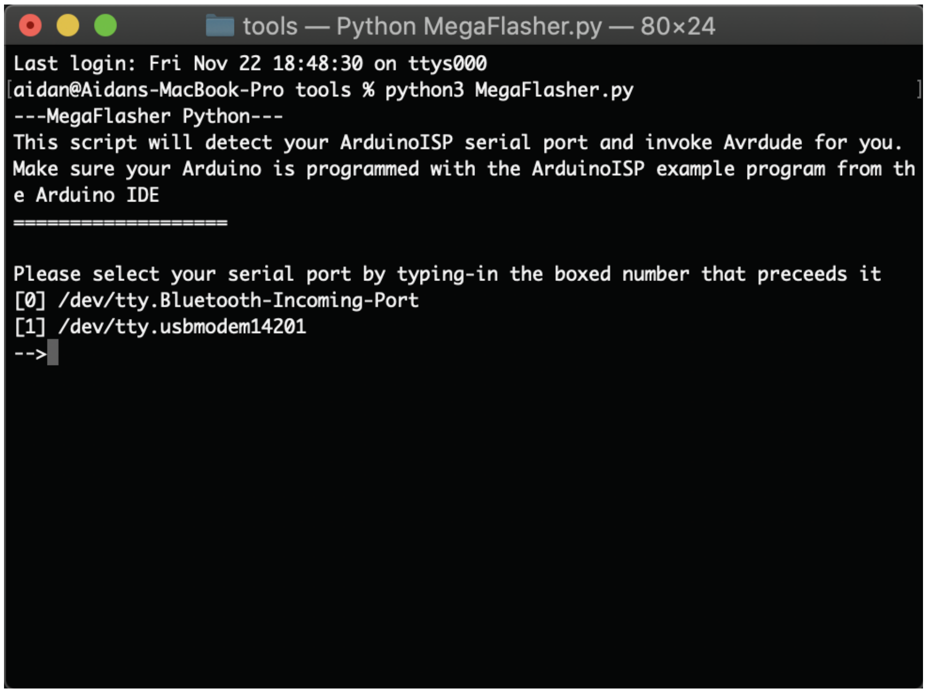
Then, cd to the Mega Flasher folder or the tools folder of the repository and run:

```
sudo python3 MegaFlasher.py
```

Connect your Mega MIDI up to your Arduino programmer as shown in the previous pages. Connect your Arduino and the Mega MIDI up to your computer via USB.

Run the script:

```
python3 MegaFlasher.py
```

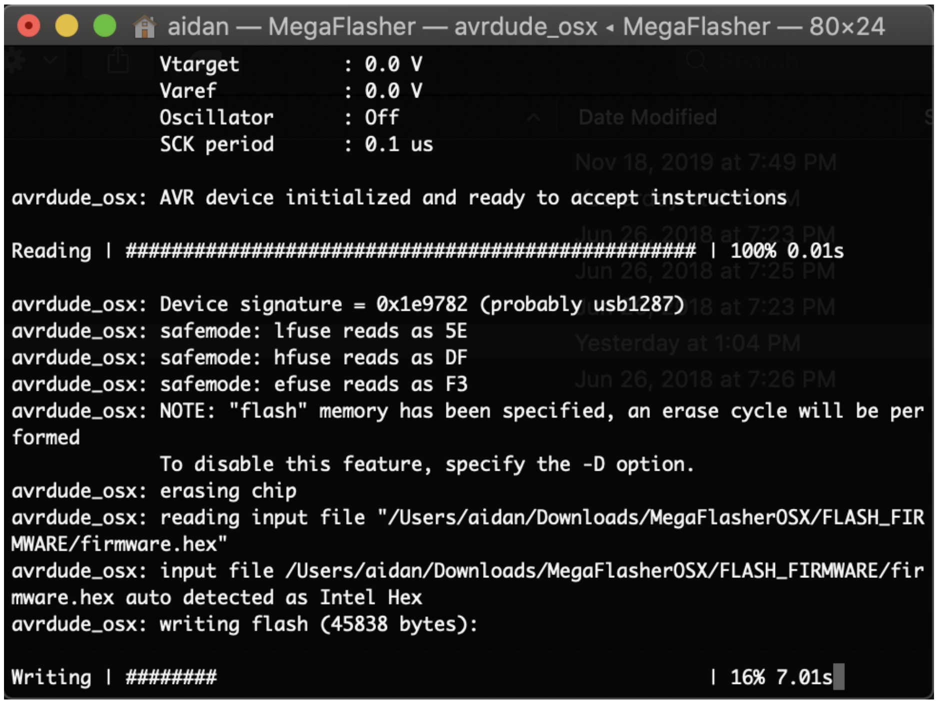


```
Last login: Fri Nov 22 18:48:30 on ttys000
aidan@Aidans-MacBook-Pro tools % python3 MegaFlasher.py
---MegaFlasher Python---
This script will detect your ArduinoISP serial port and invoke Avrdude for you.
Make sure your Arduino is programmed with the ArduinoISP example program from th
e Arduino IDE
===================

Please select your serial port by typing-in the boxed number that preceeds it
[0] /dev/tty.Bluetooth-Incoming-Port
[1] /dev/tty.usbmodem14201
-->
```

You will be asked to choose your Arduino's serial port. If you can't remember which port it's on, check in the Arduino IDE under tools->port.

In this example, there are two options. The Bluetooth port listed as option 0 and my Arduino listed as option 1. To choose the Arduino, I'll simply type '1' and hit enter.

The program should now automatically download and update your firmware.



```
         Vtarget         : 0.0 V
         Varef           : 0.0 V
         Oscillator      : Off
         SCK period      : 0.1 us

avrdude_osx: AVR device initialized and ready to accept instructions

Reading | ################################################## | 100% 0.01s

avrdude_osx: Device signature = 0x1e9782 (probably usb1287)
avrdude_osx: safemode: lfuse reads as 5E
avrdude_osx: safemode: hfuse reads as DF
avrdude_osx: safemode: efuse reads as F3
avrdude_osx: NOTE: "flash" memory has been specified, an erase cycle will be per
formed
             To disable this feature, specify the -D option.
avrdude_osx: erasing chip
avrdude_osx: reading input file "/Users/aidan/Downloads/MegaFlasherOSX/FLASH_FIR
MWARE/firmware.hex"
avrdude_osx: input file /Users/aidan/Downloads/MegaFlasherOSX/FLASH_FIRMWARE/fir
mware.hex auto detected as Intel Hex
avrdude_osx: writing flash (45838 bytes):

Writing | ########                                           | 16% 7.01s
```

Linux

Linux users must install the following dependencies if they do not already have them:

```
sudo apt install python3 python3-pip arduino
pip3 install pyserial
pip3 install requests
```

Then, cd to the Mega Flasher folder or the tools folder of the repository and run:

```
sudo python3 MegaFlasher.py
```

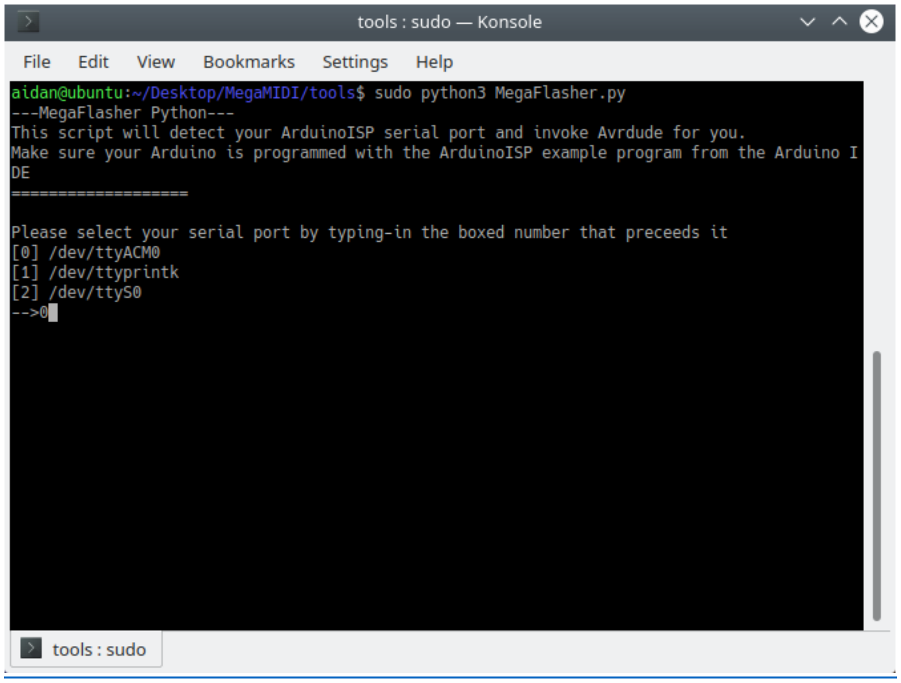
Connect your Mega MIDI up to your Arduino programmer as shown in the previous pages. Connect your Arduino and the Mega MIDI up to your computer via USB.

Run the script:

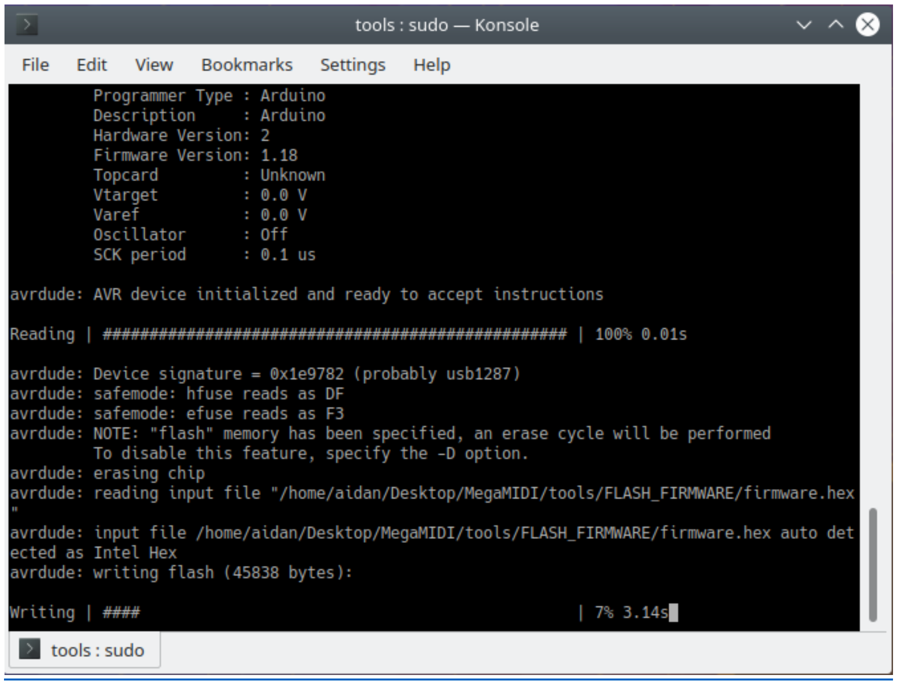
```
sudo python3 MegaFlasher.py
```

Next, you will be asked to choose your Arduino's serial port. If you can't remember which port it's on, check in the Arduino IDE under tools->port. You may need to run Arduino as sudo.

In this example, there are three options, and my Arduino is under option 0. To choose the Arduino, I'll simply type '0' and hit enter.



```
aidan@ubuntu:~/Desktop/MegaMIDI/tools$ sudo python3 MegaFlasher.py
---MegaFlasher Python---
This script will detect your ArduinoISP serial port and invoke Avrdude for you.
Make sure your Arduino is programmed with the ArduinoISP example program from the Arduino I
DE
===================

Please select your serial port by typing-in the boxed number that preceeds it
[0] /dev/ttyACM0
[1] /dev/ttyprintk
[2] /dev/ttyS0
-->0
```



```
         Programmer Type : Arduino
         Description     : Arduino
         Hardware Version: 2
         Firmware Version: 1.18
         Topcard         : Unknown
         Vtarget         : 0.0 V
         Varef           : 0.0 V
         Oscillator      : Off
         SCK period      : 0.1 us

avrdude: AVR device initialized and ready to accept instructions

Reading | ################################################## | 100% 0.01s

avrdude: Device signature = 0x1e9782 (probably usb1287)
avrdude: safemode: hfuse reads as DF
avrdude: safemode: efuse reads as F3
avrdude: NOTE: "flash" memory has been specified, an erase cycle will be performed
         To disable this feature, specify the -D option.
avrdude: erasing chip
avrdude: reading input file "/home/aidan/Desktop/MegaMIDI/tools/FLASH_FIRMWARE/firmware.hex
"
avrdude: input file /home/aidan/Desktop/MegaMIDI/tools/FLASH_FIRMWARE/firmware.hex auto det
ected as Intel Hex
avrdude: writing flash (45838 bytes):

Writing | ####                                               | 7% 3.14s
```

Finishing up

You will need to disconnect your Arduino and power-cycle your Mega MIDI by unplugging it in order for the SD card to read again.

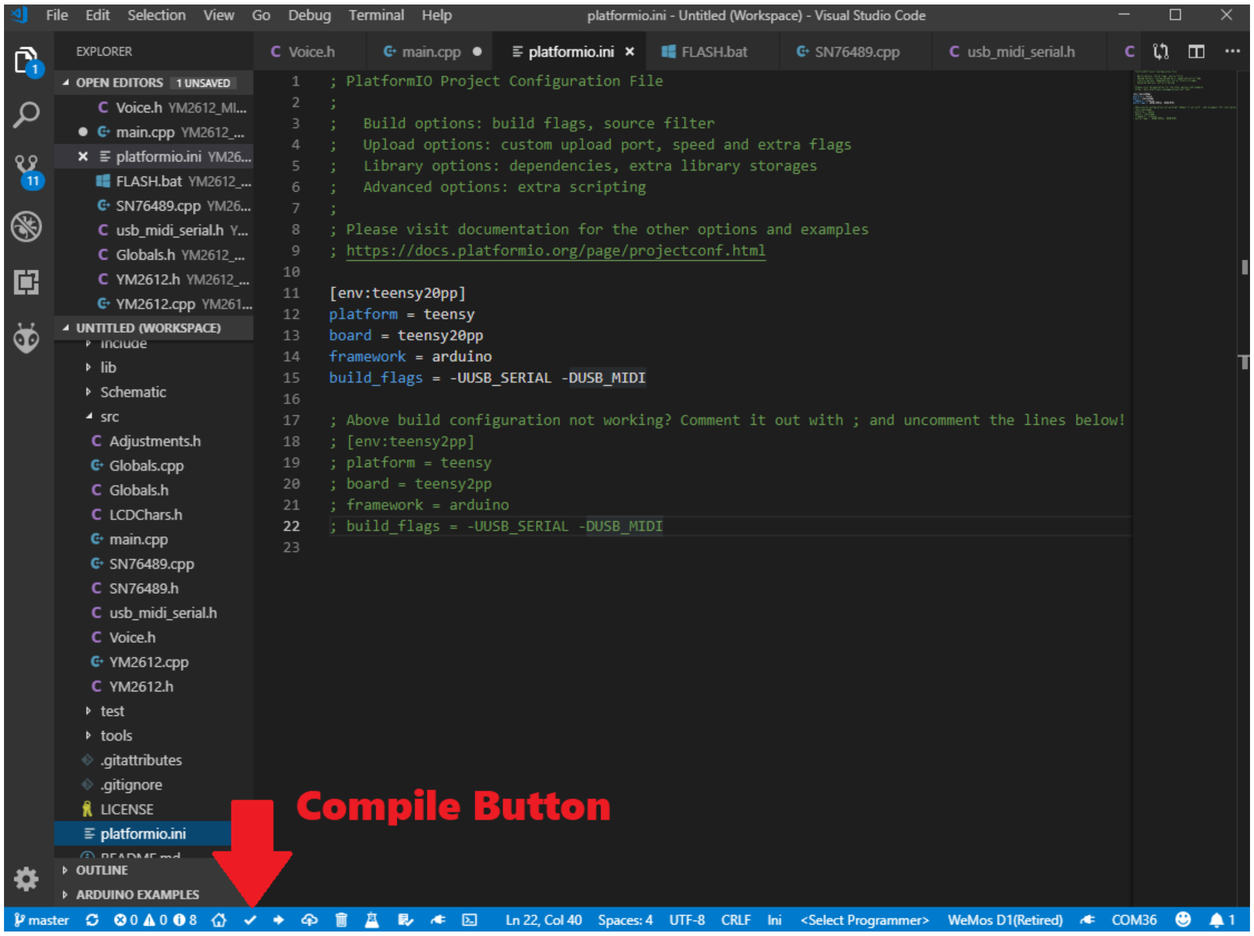
Advanced Programming and compilation

If you would like to compile the code yourself, or try an alternative method for flashing the firmware, read the following section.

This project uses Visual Studio Code with the PlatformIO plugin.

It uses the “teensyduino” toolchain. Within the repository is the *platformio.ini* file that contains all of the build flags. If the default build flags do not work, please comment them out and uncomment the alternative set of build flags below it.

After you have made your changes to the code, you can use the small checkmark button on the bottom blue bar to compile your code.


platformio.ini - Untitled (Workspace) - Visual Studio Code
File Edit Selection View Go Debug Terminal Help
EXPLORER
OPEN EDITORS 1 UNSAVED
UNTITLED (WORKSPACE)
OUTLINE
ARDUINO EXAMPLES
Compile Button
master
Ln 22, Col 40 Spaces: 4 UTF-8 CRLF Ini <Select Programmer> WeMos D1(Retired) COM36

```
; PlatformIO Project Configuration File
;
;   Build options: build flags, source filter
;   Upload options: custom upload port, speed and extra flags
;   Library options: dependencies, extra library storages
;   Advanced options: extra scripting
;
; Please visit documentation for the other options and examples
; https://docs.platformio.org/page/projectconf.html

[env:teensy20pp]
platform = teensy
board = teensy20pp
framework = arduino
build_flags = -UUSB_SERIAL -DUSB_MIDI

; Above build configuration not working? Comment it out with ; and uncomment the lines below!
; [env:teensy2pp]
; platform = teensy
; board = teensy2pp
; framework = arduino
; build_flags = -UUSB_SERIAL -DUSB_MIDI
```

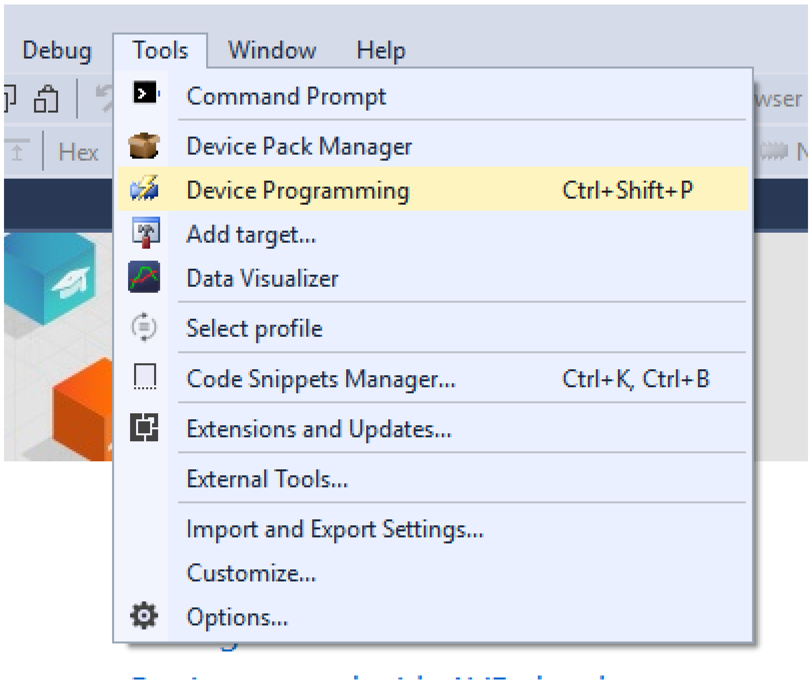
Flashing your HEX file (Legacy)

After compilation, a HEX file will be generated. You can not use the default “upload” button in VS code and must instead upload this code manually through the In-Circuit Serial Programming (ICSP) header.

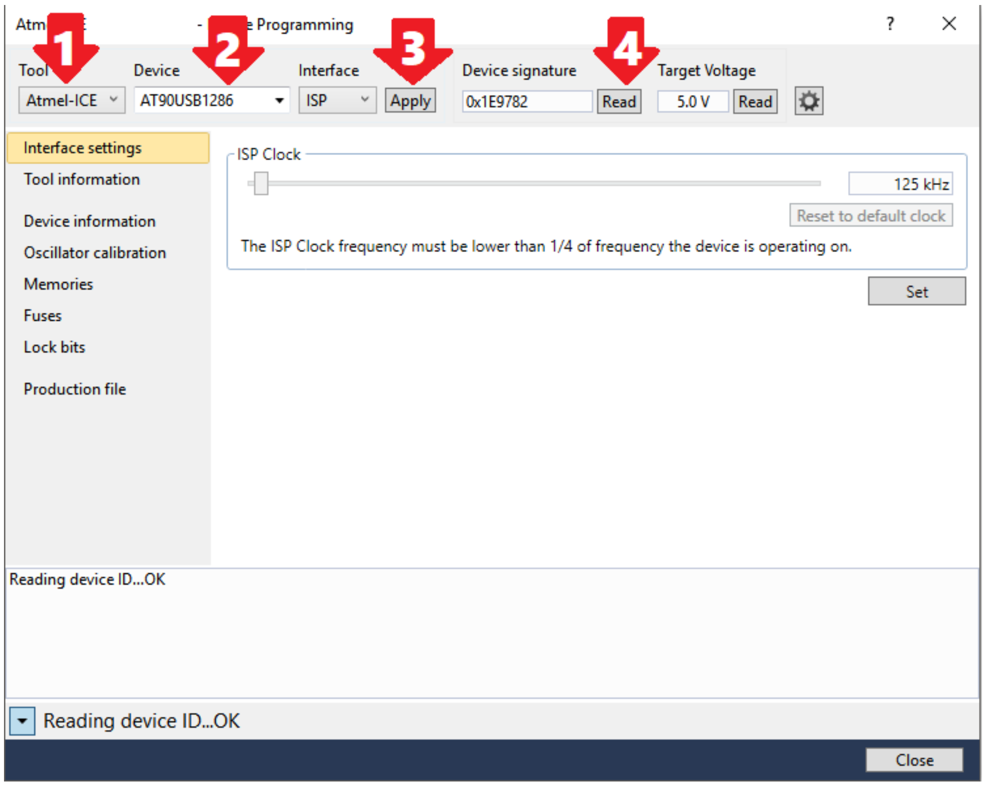
If you have an Atmel ICE, or any other standard Atmel ICSP device, you can use Atmel Studio to flash your HEX file.

Attach your ICE programmer and open Atmel Studio.

Under the **Tools** menu, select **Device Programming**.



Make sure your Mega MIDI is powered by USB! Attach the **AVR side** of the ICE to the **ISCP header** on the Mega MIDI. Set your **programming tool**, set your device to **AT90USB1286**. Set the Interface to **ISP** and hit apply. Read the device signature. It should read **0x1E9782** with a target voltage of approximately 5V and your Mega MIDI should reset.



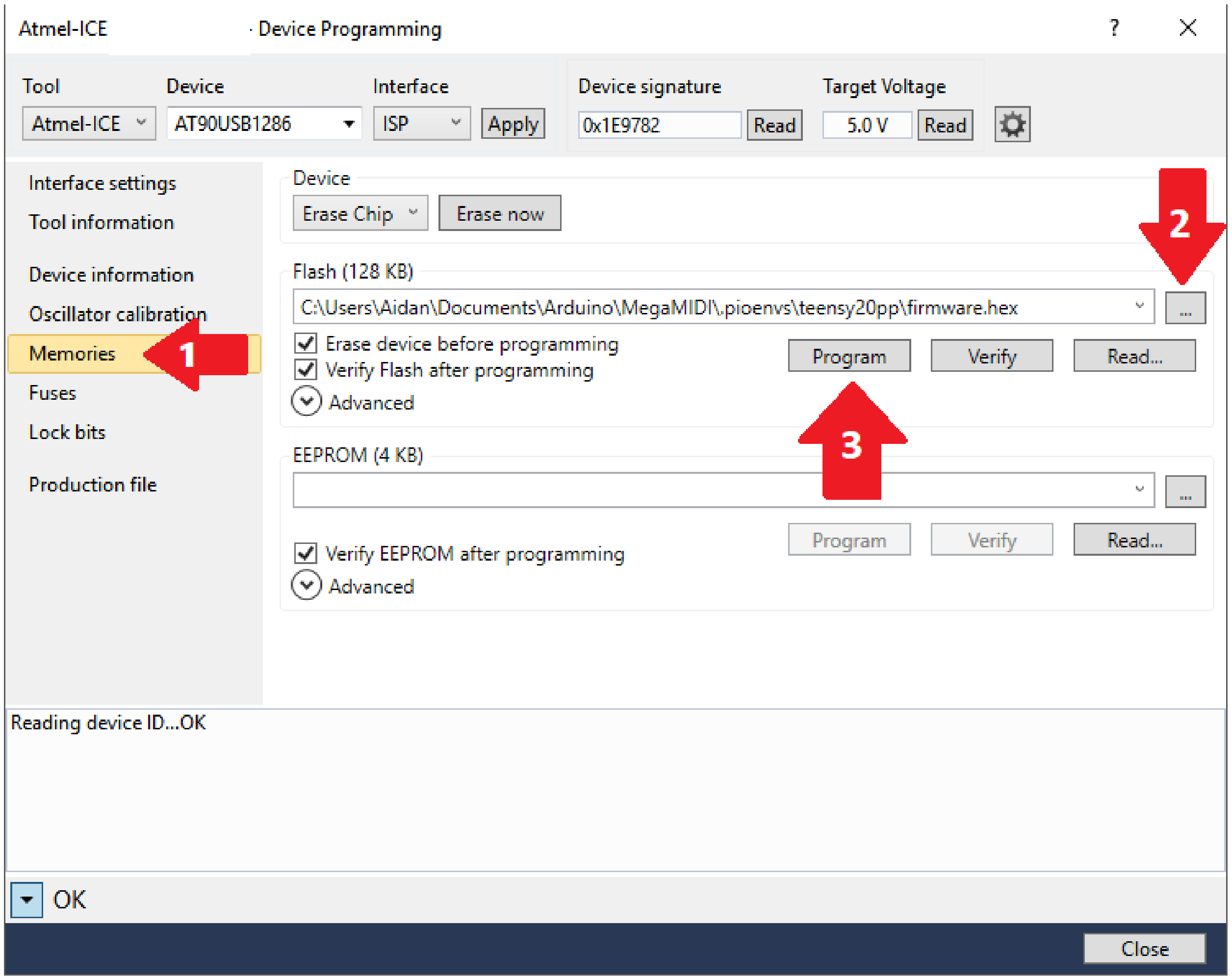
Tool: Atmel-ICE
Device: AT90USB1286
Interface: ISP
Device signature: 0x1E9782
Target Voltage: 5.0 V
ISP Clock: 125 kHz

The ISP Clock frequency must be lower than 1/4 of frequency the device is operating on.

```
Reading device ID...OK
```

Reading device ID...OK

Next, head over to the **"Memories"** tab.



Device signature: 0x1E9782
Target Voltage: 5.0 V

C:\Users\Aidan\Documents\Arduino\MegaMIDI\.pioenvs\teensy20pp\firmware.hex

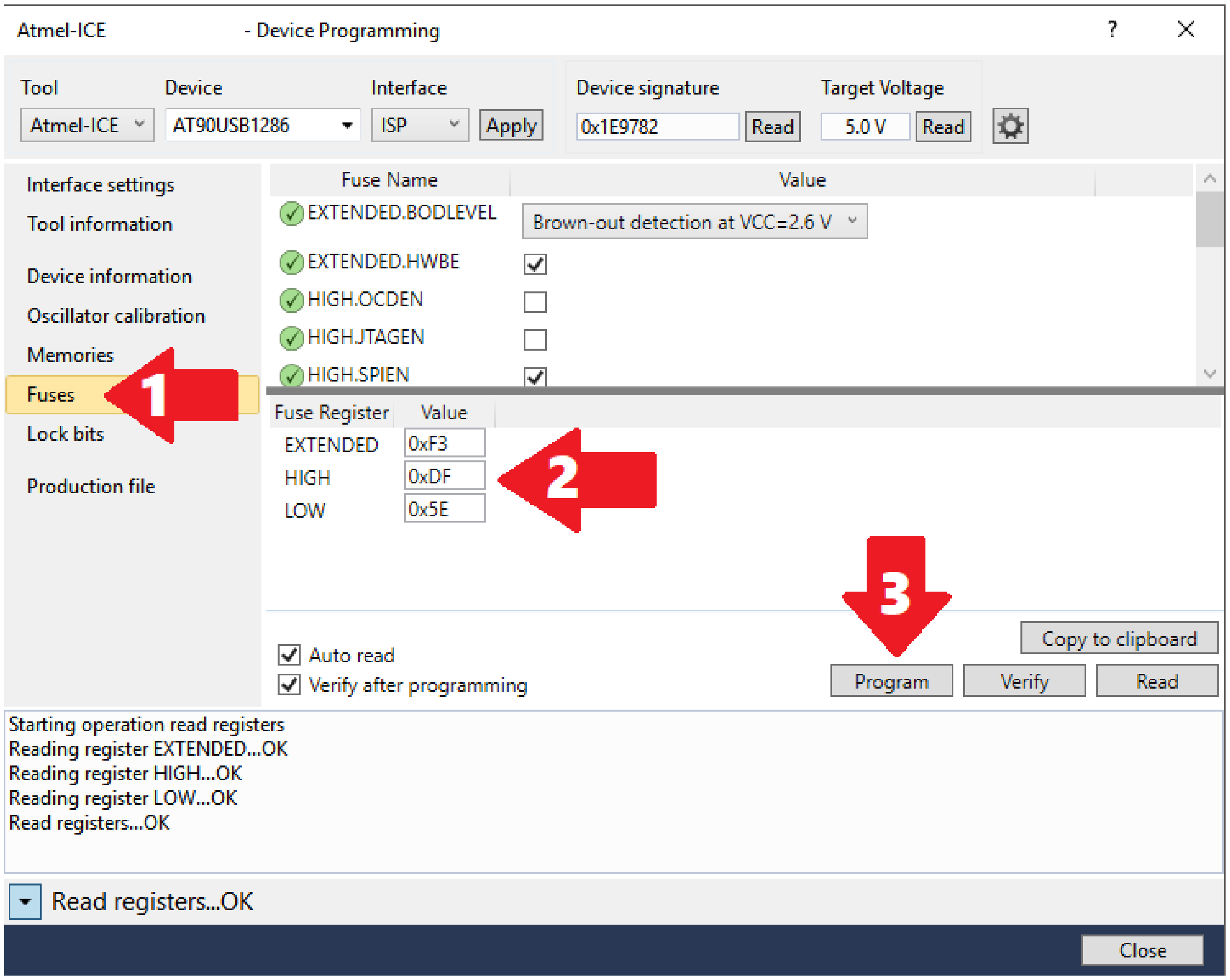
✓ Erase device before programming
✓ Verify Flash after programming
✓ Verify EEPROM after programming

```
Reading device ID...OK
```

Select your HEX file. PlatformIO will store this hex file as either ".pioenvs\teensy*\firmware.hex" OR ".pio\build\teensy2pp\firmware.hex" inside your repository. Finally, hit the **"Program"** button.

If you receive a warning from the Atmel Programming window, simply hit "Apply" and "Read" again on the tool/device/interface and Device signature buttons on the top bar.

Finally, you must make sure the fuses are set. Once you set the fuses once, you do not have to reprogram them again, even if you flash new firmware.



Fuse Name	Value
EXTENDED.BODLEVEL	Brown-out detection at VCC=2.6 V
EXTENDED.HWBE	✓
HIGH.OCDEN	☐
HIGH.JTAGEN	☐
HIGH.SPIEN	✓

Fuse Register	Value
EXTENDED	0xF3
HIGH	0xDF
LOW	0x5E

☑ Auto read
☑ Verify after programming

```
Starting operation read registers
Reading register EXTENDED...OK
Reading register HIGH...OK
Reading register LOW...OK
Read registers...OK
```

Read registers...OK

The fuse values should read:

EXTENDED	0xF3
HIGH	0xDF
LOW	0x5E

Again, once you have programmed the fuses, you do not need to program them again. If the fuses are already set to the values above, you do not need to re-flash them.

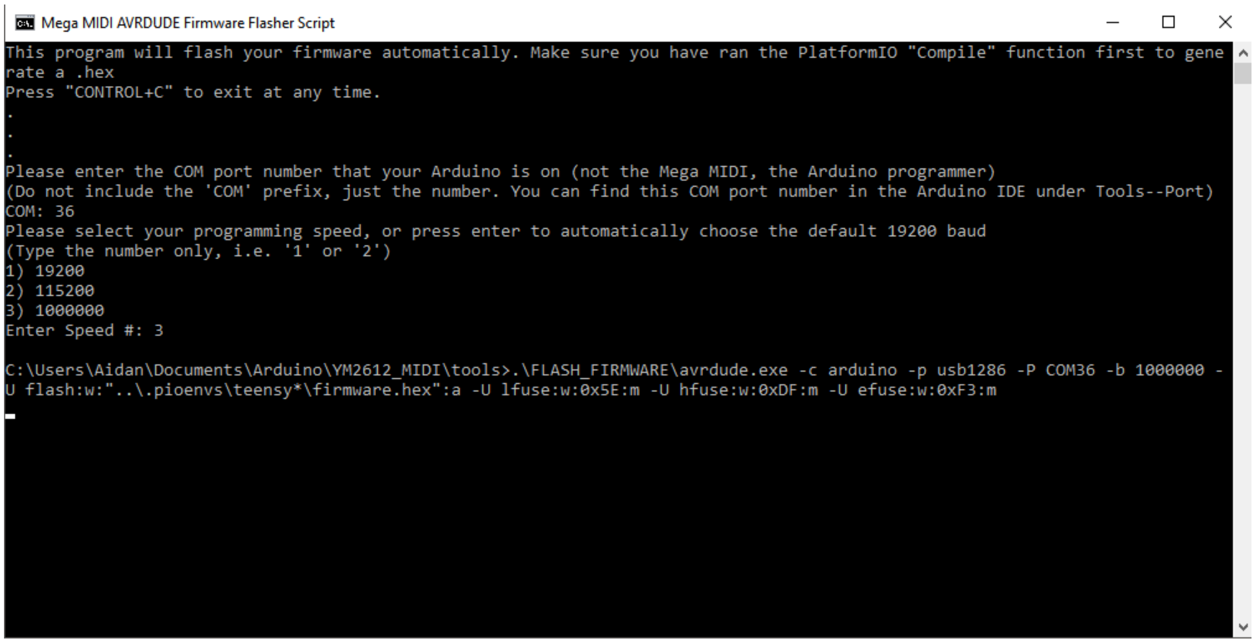
MegaMIDI AVRDude Firmware Flasher (Old, but still useful for development)

I have written another batch file to expedite the flashing process. You can find this script within the repositories **tools** folder. It is simply named **FLASH.bat**.

Double-click on **FLASH.bat** to open up a command window. It will want to know two things: What **COM port** is your **Arduino** on and what **speed** should the flasher program at?

In this example, my Arduino is on COM36. So, I would simply type “36” and hit enter. Remember, you can find the COM port number under Tools->Port in the Arduino IDE.

Next, the program will ask for a flashing speed. There are three options. Since I previously modified the ArduinoISP script to program at 1000000 baud, I will simply type the number “3” for the third option. If you chose a different programming speed, choose that speed here, or just hit enter for the default 19200 baud if you didn’t bother modifying the ArduinoISP script.

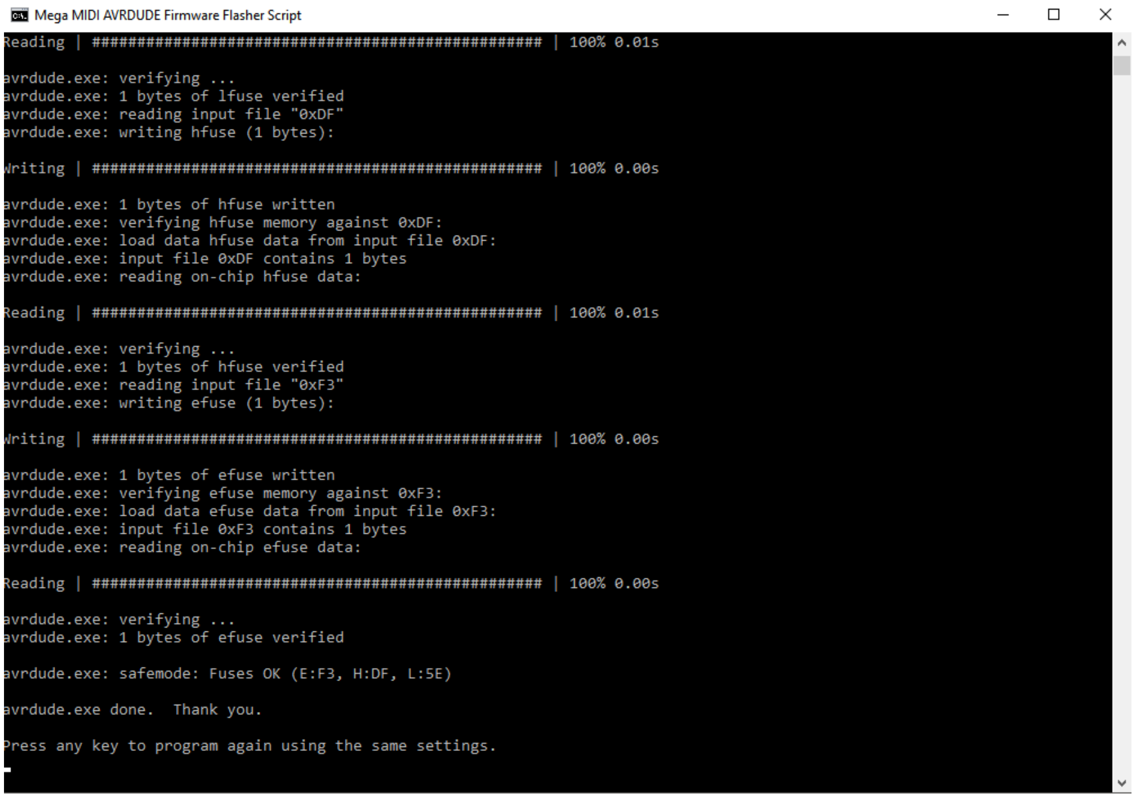


```
This program will flash your firmware automatically. Make sure you have ran the PlatformIO "Compile" function first to gene
rate a .hex
Press "CONTROL+C" to exit at any time.
.
.
.
Please enter the COM port number that your Arduino is on (not the Mega MIDI, the Arduino programmer)
(Do not include the 'COM' prefix, just the number. You can find this COM port number in the Arduino IDE under Tools--Port)
COM: 36
Please select your programming speed, or press enter to automatically choose the default 19200 baud
(Type the number only, i.e. '1' or '2')
1) 19200
2) 115200
3) 1000000
Enter Speed #: 3

C:\Users\Aidan\Documents\Arduino\YM2612_MIDI\tools>.\FLASH_FIRMWARE\avrdude.exe -c arduino -p usb1286 -P COM36 -b 1000000 -
U flash:w:"..\.pioenvs\teensy*\firmware.hex":a -U lfuse:w:0x5E:m -U hfuse:w:0xDF:m -U efuse:w:0xF3:m
```

After you hit enter, give the script a few moments to communicate with your Arduino and it should begin flashing.

Warning: This flashing script does not preserve EEPROM. Your favorited patches will be lost!

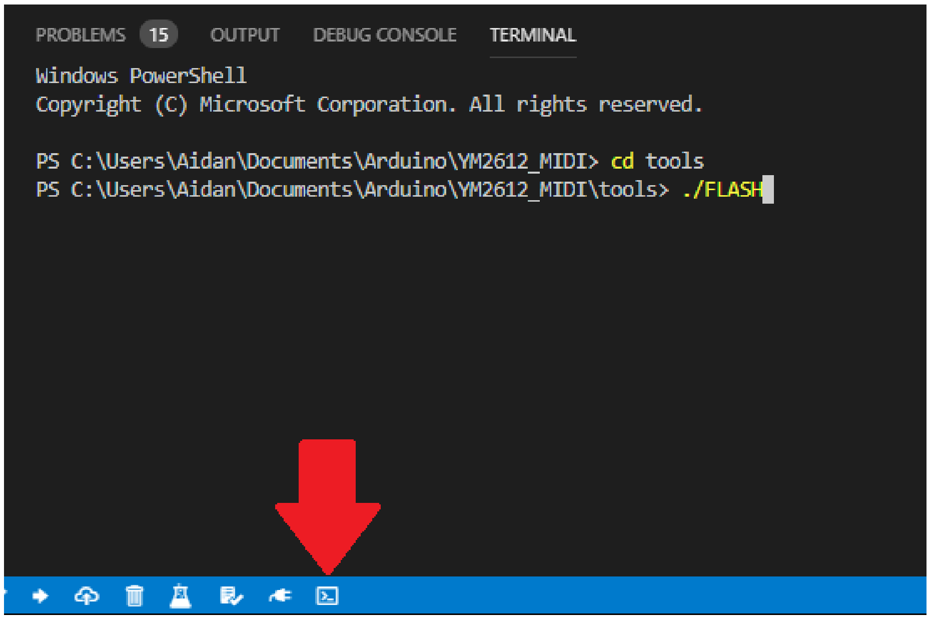
This script will also automatically handle the Fuse settings as well, so you don't need to worry about that part either.


Mega MIDI AVRDUDE Firmware Flasher Script

```
Reading | ################################################## | 100% 0.01s

avrdude.exe: verifying ...
avrdude.exe: 1 bytes of lfuse verified
avrdude.exe: reading input file "0xDF"
avrdude.exe: writing hfuse (1 bytes):

Writing | ################################################## | 100% 0.00s

avrdude.exe: 1 bytes of hfuse written
avrdude.exe: verifying hfuse memory against 0xDF:
avrdude.exe: load data hfuse data from input file 0xDF:
avrdude.exe: input file 0xDF contains 1 bytes
avrdude.exe: reading on-chip hfuse data:

Reading | ################################################## | 100% 0.01s

avrdude.exe: verifying ...
avrdude.exe: 1 bytes of hfuse verified
avrdude.exe: reading input file "0xF3"
avrdude.exe: writing efuse (1 bytes):

Writing | ################################################## | 100% 0.00s

avrdude.exe: 1 bytes of efuse written
avrdude.exe: verifying efuse memory against 0xF3:
avrdude.exe: load data efuse data from input file 0xF3:
avrdude.exe: input file 0xF3 contains 1 bytes
avrdude.exe: reading on-chip efuse data:

Reading | ################################################## | 100% 0.00s

avrdude.exe: verifying ...
avrdude.exe: 1 bytes of efuse verified

avrdude.exe: safemode: Fuses OK (E:F3, H:DF, L:5E)

avrdude.exe done.  Thank you.

Press any key to program again using the same settings.
```

You can leave this firmware flasher script open and simply press any key to quickly flash the Mega MIDI again without re-entering your settings.

Tip: You can also keep this flasher script (and any of the other batch tools) inside of your Visual Studio Code terminal window.



```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Users\Aidan\Documents\Arduino\YM2612_MIDI> cd tools
PS C:\Users\Aidan\Documents\Arduino\YM2612_MIDI\tools> ./FLASH
```

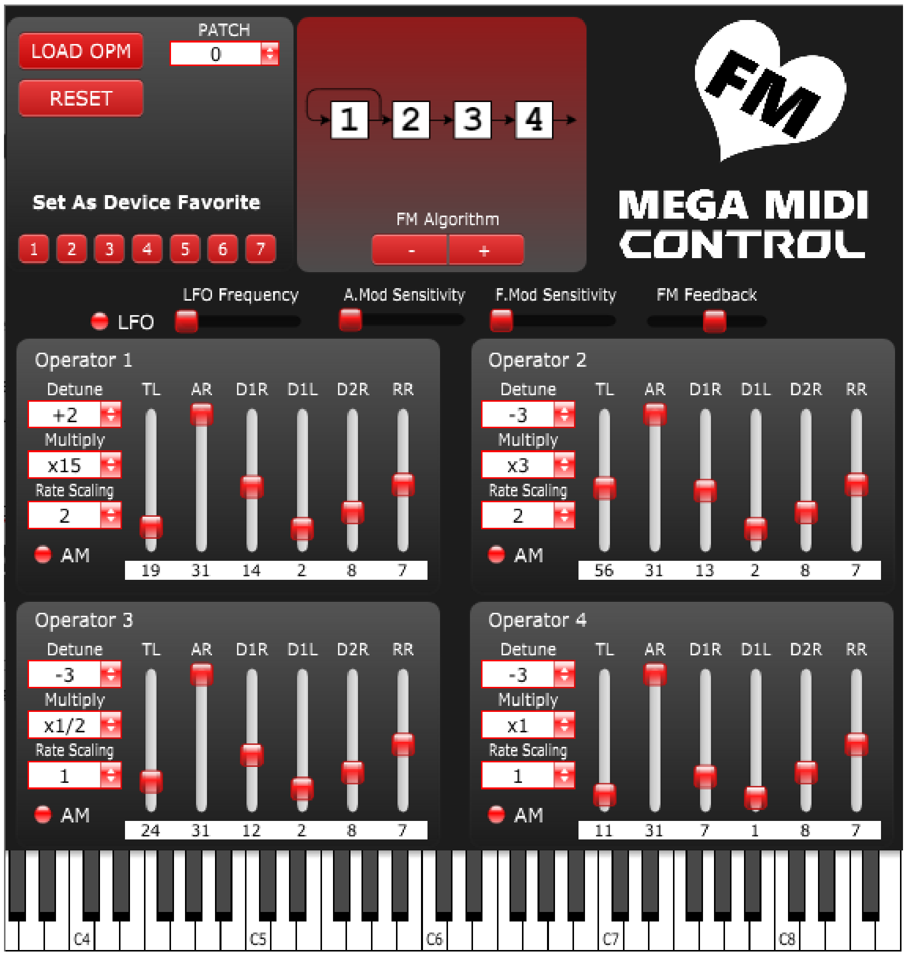
Open the terminal and type: “**cd tools**” followed by **“./FLASH**” (or any other batch script you’d like to run). When running the FLASH script in the VS Code terminal, you can compile your code, then set up your flash settings and flash from the console. For every following flash, just double-tap the ENTER button to quickly upload your code.

The AVRDude command is:

```
avrdude -c arduino -p usb1286 -P [COMPORT] -b 19200 -U flash:w:"firmware.hex":a -
U lfuse:w:0x5E:m -U hfuse:w:0xDF:m -U efuse:w:0xF3:m
```

Make sure to replace the COM port with your Arduino’s port and “firmware.hex” with the file location of your firmware.

Mega MIDI Control VST (Firmware 1.3+)



PATCH 0

	Detune	Multiply	Rate Scaling	TL	AR	D1R	D1L	D2R	RR
Operator 1	+2	x15	2	19	31	14	2	8	7
Operator 2	-3	x3	2	56	31	13	2	8	7
Operator 3	-3	x1/2	1	24	31	12	2	8	7
Operator 4	-3	x1	1	11	31	7	1	8	7

Mega MIDI Control is the DAW-compatible MIDI VST that can be used to directly control the parameters of the YM2612. It is a highly modified version of the CTRLR panel created by Thea Flowers.

Mega MIDI Control is built with CTRLR. This software is cross-compatible with Windows, Linux, and MacOS, though some operating systems perform better than others. Windows is recommended.

Downloads

Windows Only:

Stand-alone exe

x86 VST DLL

x64 VST DLL

Universal Panel Download for Linux, MacOS, and Windows (Requires CTRLR Installation):

CTRLR Panel

DAW Installation

Windows:

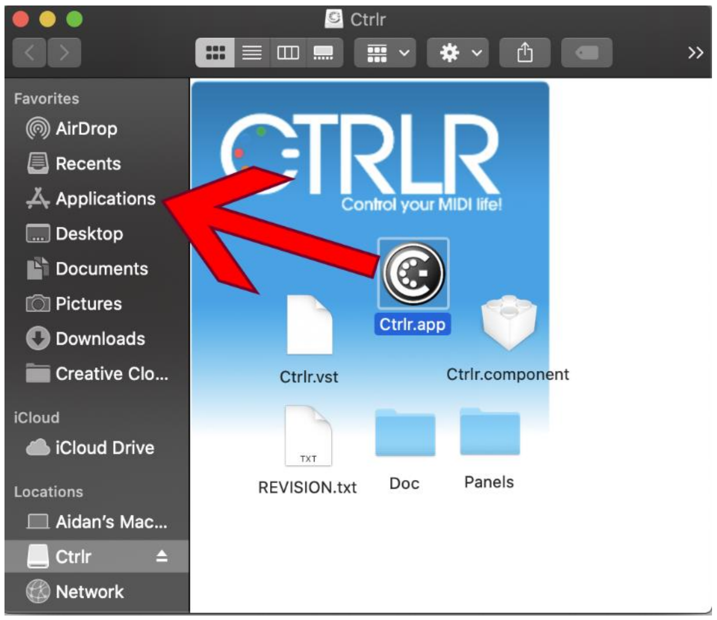
On Windows, stand-alone versions of the VST exist and do not require a separate installation of CTRLR. The easiest way to get started using Mega MIDI Control would be to simply use the exe provided in the links above. If you would like to install one of the VST DLLs, you will need to reference your DAW's user guide, as plugin installation differs between them. For example, here is the documentation for FL Studio and Ableton when it comes to installing VST plugins.

MacOS:

MacOS requires the CTRLR program in order to run panels. CTRLR also provides a VST and AU that can host panels within DAWs.

CTRLR Exists as both a stand-alone program and as a VST/AU. After downloading the CTRLR program, you will be presented with a set of files and folders.

Move the Ctrlr.app file to your Applications folder to install the stand-alone.



To install the VST, copy the Ctrlr.vst file to:

/Library/Audio/Plug-Ins/VST

To install the AU, copy the Ctrlr.component file to:

/Library/Audio/Plug-Ins/Components

Linux:

Linux requires the [CTRLR program]() in order to run panels. A .so plugin file is also provided.

The Linux version of CTRLR seems to be very buggy and unpolished, so running Mega MIDI Control on this platform is not recommended. As of writing this, the file-open dialogue appears to not work. A work-around for this is double-clicking the Mega MIDI Control .bpanelz file and opening-with the CTRLR program.

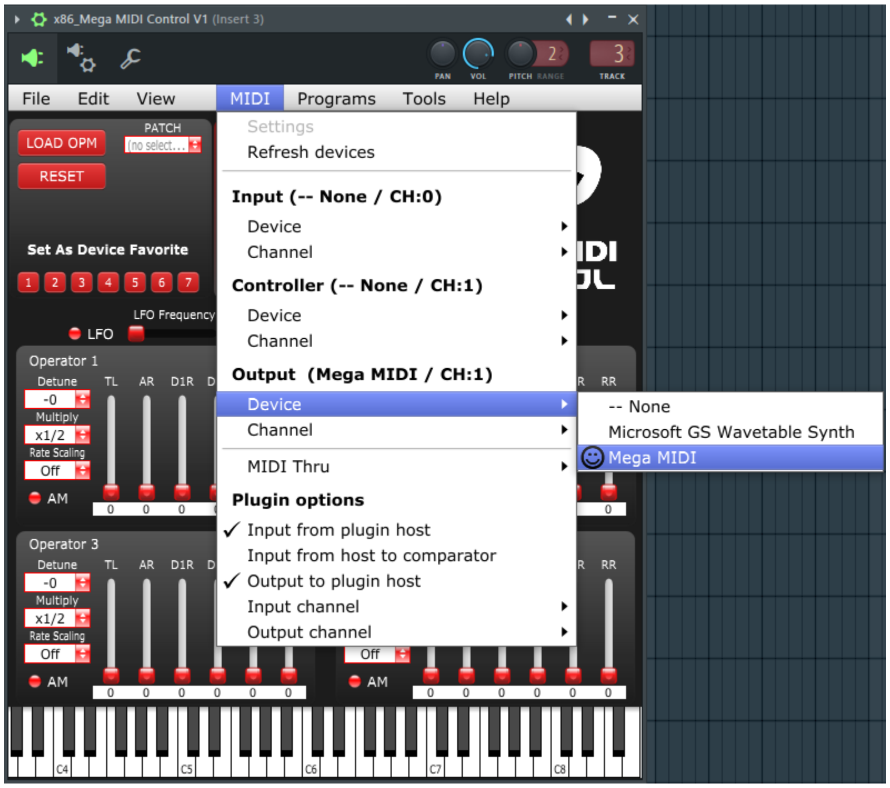
Using Mega MIDI Control

Note for all users: If you use a DAW to host CTRLR or one of the Mega MIDI Control VSTs, you MUST ensure that the Mega MIDI is not enabled in your DAW's MIDI settings. You will enable MIDI through the VST itself!

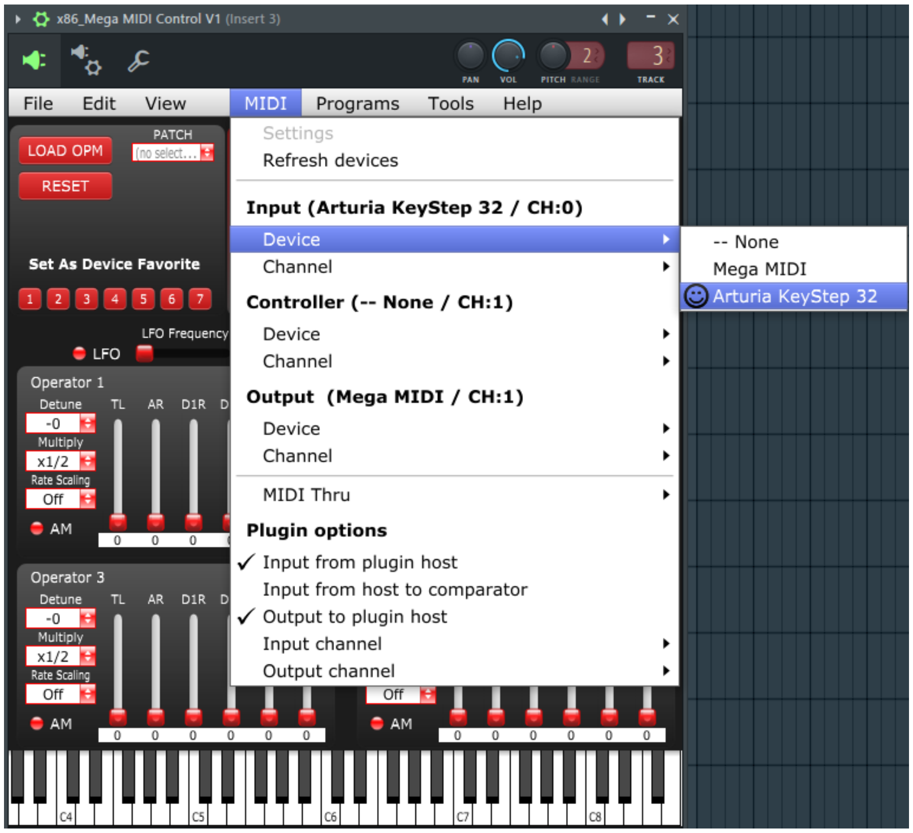
Make sure your Mega MIDI is plugged-in **before** you open the VST/Standalone.

If you find yourself in edit-mode in CTRLR, simply go to Panel->Panel Mode.

First, you need to enable the Mega MIDI as a MIDI Output device.



If you would like to control your Mega MIDI with an external keyboard, you can look under the MIDI->Input->Device menu. Again, making sure that your external keyboard is connected before you activate the VST/standalone. Alternatively, you can use the traditional DIN MIDI port on the Mega MIDI.



Voice Editing Quick Start

So, here’s the thing. FM synthesis is fairly complex. Creating a voice from absolutely nothing is quite difficult to do, and this guide will not go over how to program an FM synthesizer – you’re on your own for that one!

What you can do, however, is load-in .opm patch files from your computer and see how the modulators change. Modulators are all of the sliders, buttons, checkboxes, etc. that represent values on the synthesizer. When you change one of these modulators on the VST, you are directly programming the registers on the YM2612. Every modulator is also exposed to the DAW, meaning that they can be adjusted in real time.

You can save up to 7 unique patches from the VST to the Mega MIDI if you’ve created something you want to keep by using the “Set as Device Favorite” buttons.

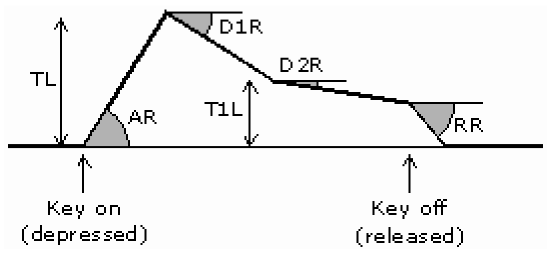
Short and sweet – FM synthesis uses programmable “operators” that create and modulate each other’s waveforms. Changing the algorithm changes how the operators interact with one another.

Definitions

Definitions courtesy of http://www.smspower.org/maxim/Documents/YM2612

On each operator, you will see the same set of modulators:

Modulator	Function
TL	Total Level – The highest amplitude of the waveform.
AR	Attack Rate – The angle of initial amplitude increase.
D1R	The angle of initial amplitude decrease.
D1L	Secondary amplitude reached after the first period of rapid decay.
D2R	The angle of secondary amplitude decrease
RR	Release Rate – The final sharp decrease in volume after the key is released.
AM	Amplitude Modulation toggle. Is this operator affected by the LFO?
Detune	Gives small variations from the overall frequency × MULTIPLY.
Multiply	Multiplies the overall frequency, with the exception that 0 results in multiplication by ½.
Rate Scaling	RS is the degree to which the envelope becomes narrower as the frequency becomes higher.

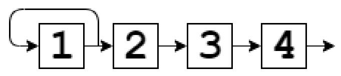


There are also a few global modulators:

Modulator	Function
LFO	Low Frequency Oscillator toggle.
LFO Frequency	Adjust Low Frequency Oscillator period.
A.Mod Sensitivity	Amplitude Modulation Sensitivity.
F.Mod Sensitivity	Frequency Modulation Sensitivity.
FM Feedback	Adjust the algorithm feedback loop.

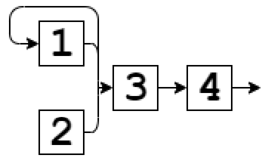
Algorithms

0



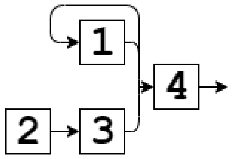
Distortion Guitar

1



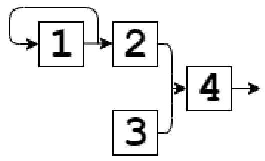
Harp, PSG-like Sounds

2



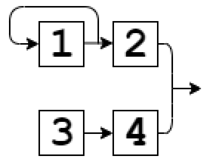
Bass, Electric Guitar,
Brass, Piano, Woodwinds

3



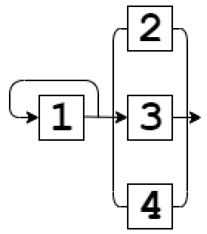
Strings, Folk Guitar, Chimes

4



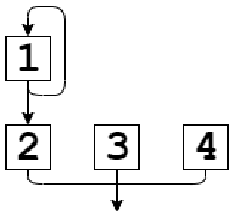
Flute, Bells, Chorus, Bass Drum,
Snare Drum, Tom Tom

5



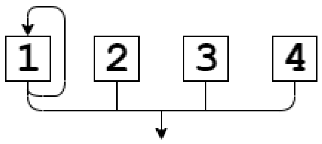
Brass, Organ

6



Xylophone, Tom Tom, Organ,
Vibraphone, Snare Drum,
Bass Drum

7



Pipe Organ