

MEGA MIDI

Sega Genesis / Megadrive MIDI Synthesizer Module

User Manual

Aidan Lawrence

www.aidanlawrence.com

May 2019

Contents

Quick Start	3
Device Guide	3
SD Card Setup.....	4
OPM Voice Files.....	5
The Main Menu	6
Saving Favorite Patches	7
MIDI Channels.....	8
YM2612 FM Synthesizer	8
SN76489 Programmable Sound Generator	9
Advanced Features	9
Control over Serial.....	9
Flashing Firmware (Programming).....	10
Environment.....	10
Flashing your HEX file	10
Flash Firmware With an Arduino	13
MegaMIDI AVR Dude Firmware Flasher	15

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 12V, center-positive power supply that can supply at least 500mA (Standard 5.5mm Jack)
- 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 12V power. Plug-in MIDI/USB cable.

Device Guide



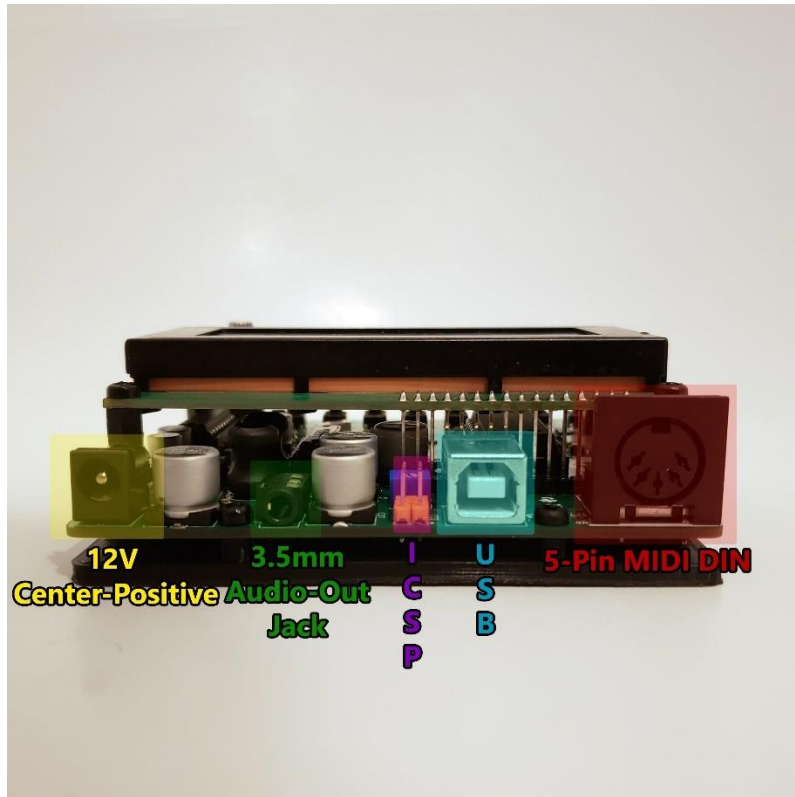
LFO Button: Low Frequency Oscillator Toggle. Toggle this button to activate/deactivate the Low Frequency oscillator. Control the frequency of the LFO using the MOD wheel on your MIDI controller.

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.



12V Center-Positive: The master power jack. 12 Volt, 2x5.5mm standard barrel jack. Center-positive. This jack must be connected at all times as the Mega MIDI cannot be powered by USB.

3.5mm Audio-Out Jack: The main amplified 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. Does not supply power.

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.”

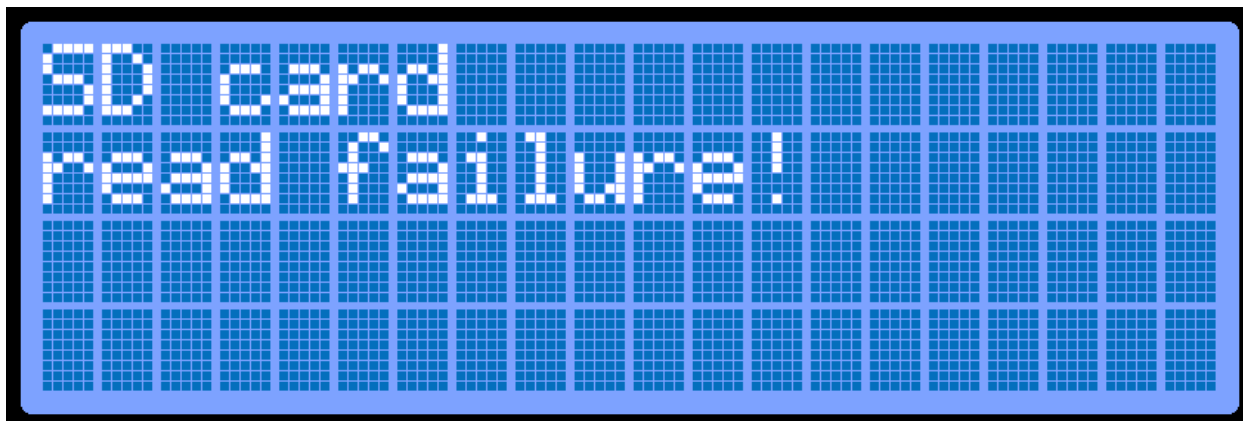
Experiencing a lot of static noise? Inserting the power plug slowly or at an off-angle may cause the amplifier to oscillate. Simply remove your plug, wait a moment, then plug your Mega MIDI back in.

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:



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

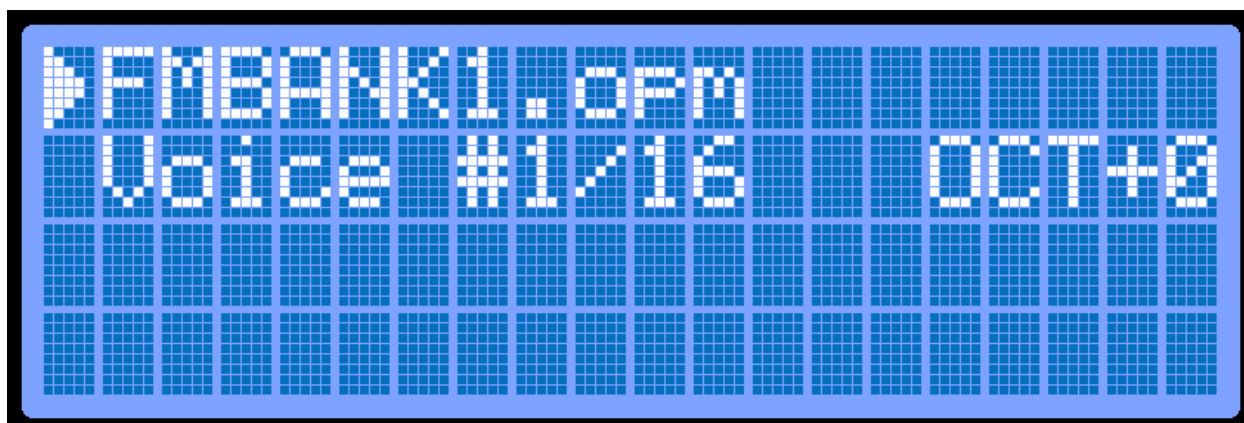
“.OPM” files are used by the Mega MIDI to read-in premade voice patches. The OPM file format was created for use with the [VOPM virtual synthesizer](#). OPM files are “plain-text” register dumps for Yamaha OPNx and OPM sound chips and contain the information necessary to program the voices for said chips.

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.

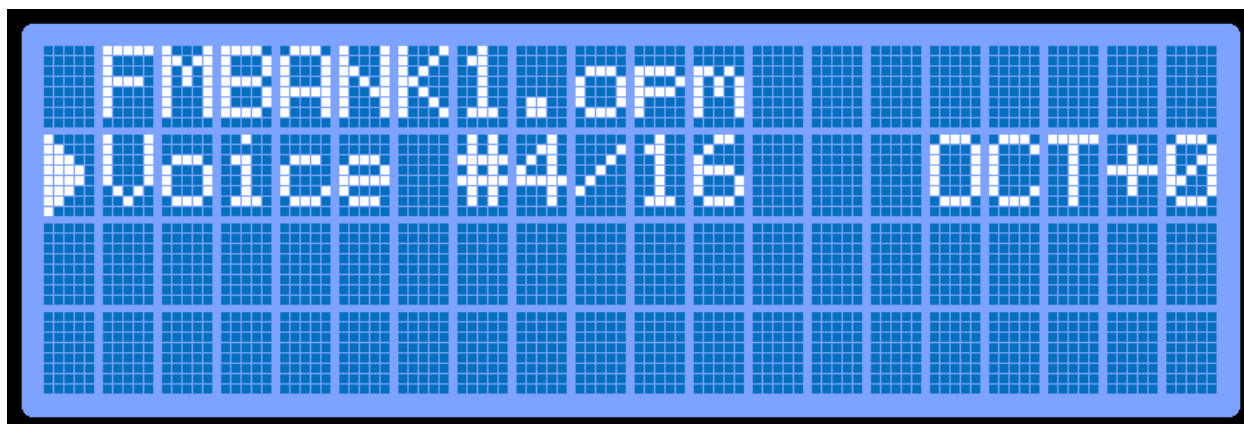
The Main Menu



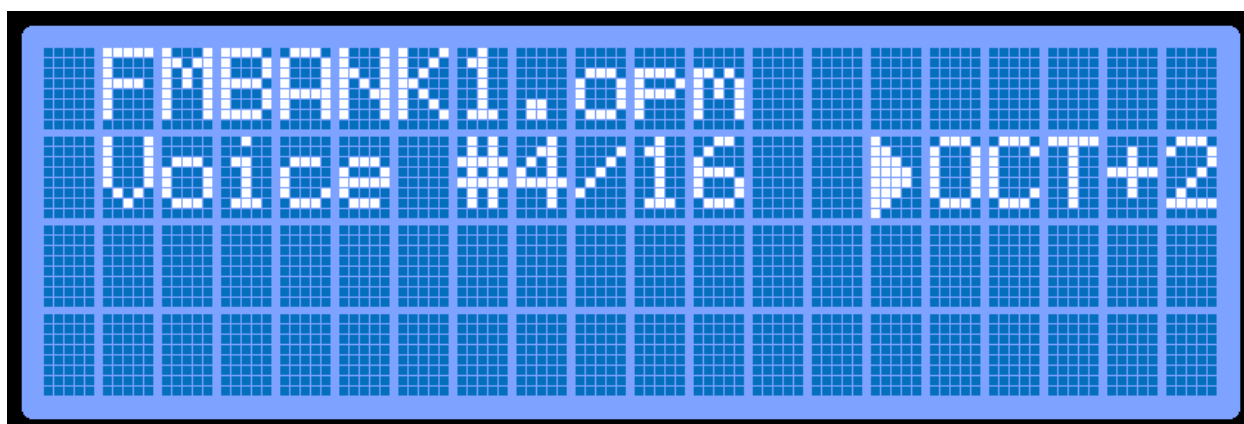
On the default main menu, you will see three options and a triangle-shaped cursor. **Press the knob to move the cursor to the next option.**

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)

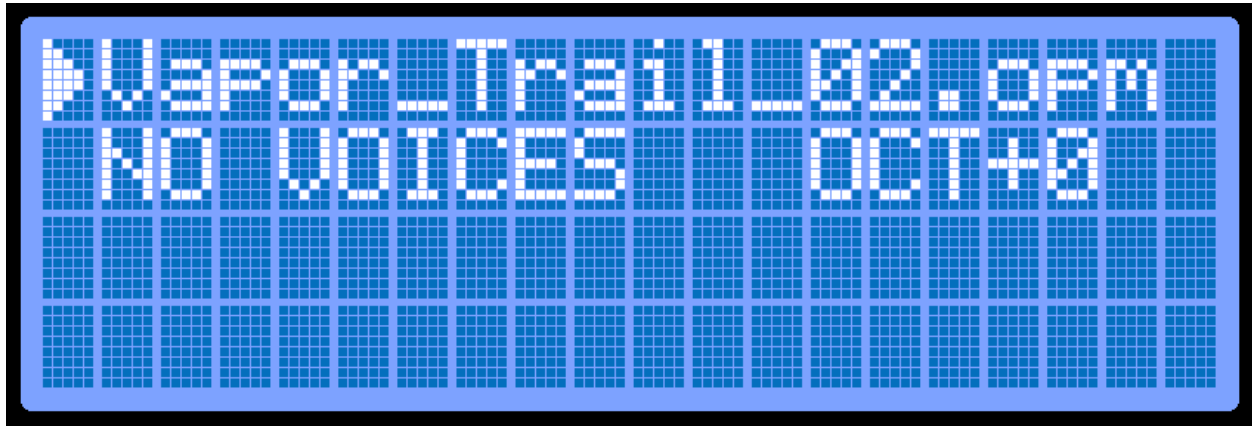
The second line shows both the Voice number and the Octave Shift. Rotate the knob while the cursor is on the “Voice” option to change between the various voices on each OPM file.



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.



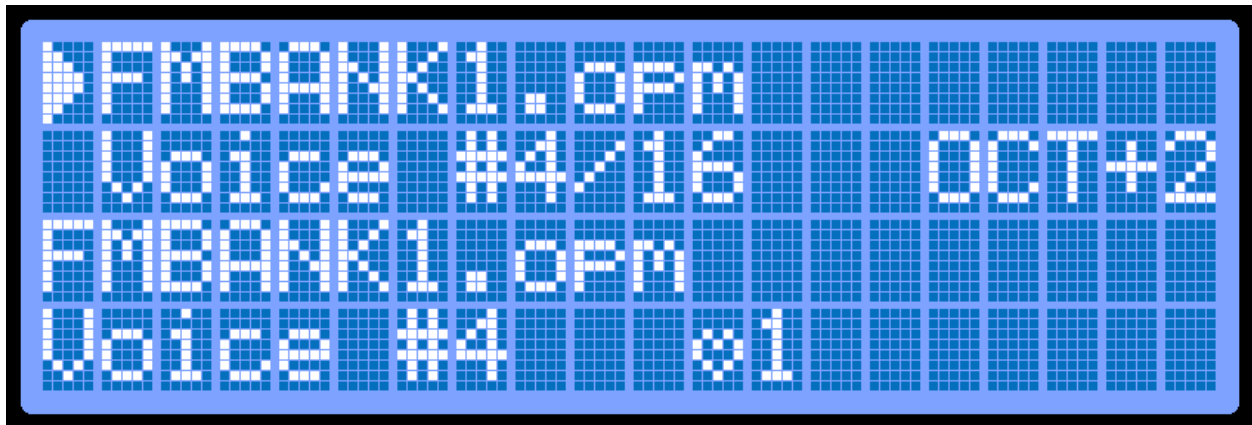
If an OPM file does not contain any valid voices, the FM sound-source will be disabled. Sometimes, Sega Genesis tracks will *only* use the PCM channel of the YM2612 and will not use any FM patches, therefore, some OPM files may show the “NO VOICES” warning. The Mega MIDI does not support PCM samples,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



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.

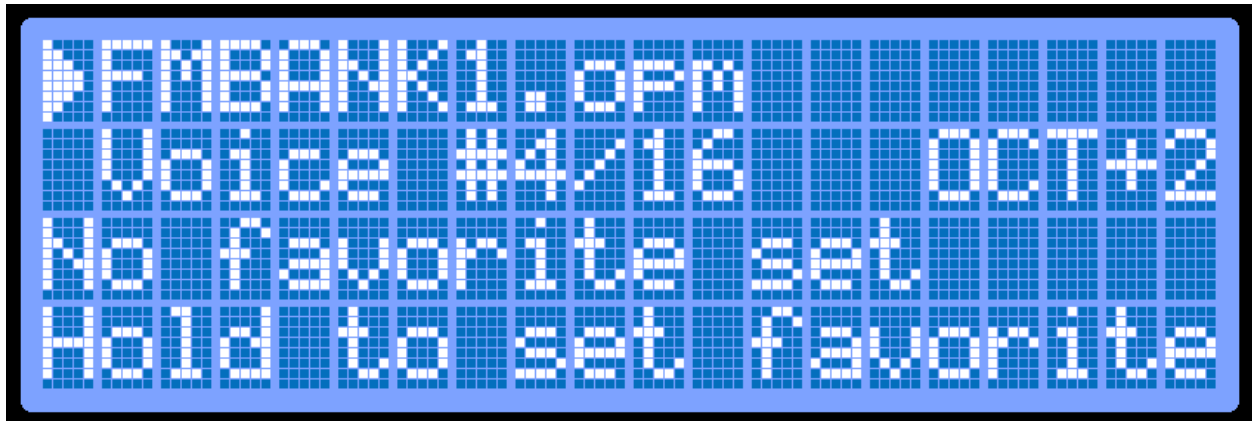
To save a new favorite patch, simply hold down the favorite button you’d like it assigned to. After about two seconds, the favorite patch will be saved into the microcontroller’s EEPROM. You will then notice the favorite LED blink to indicate a new favorite has been saved and the bottom two LCD rows 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.



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:



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.

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, but this has been disabled for the Mega MIDI. 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.

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>

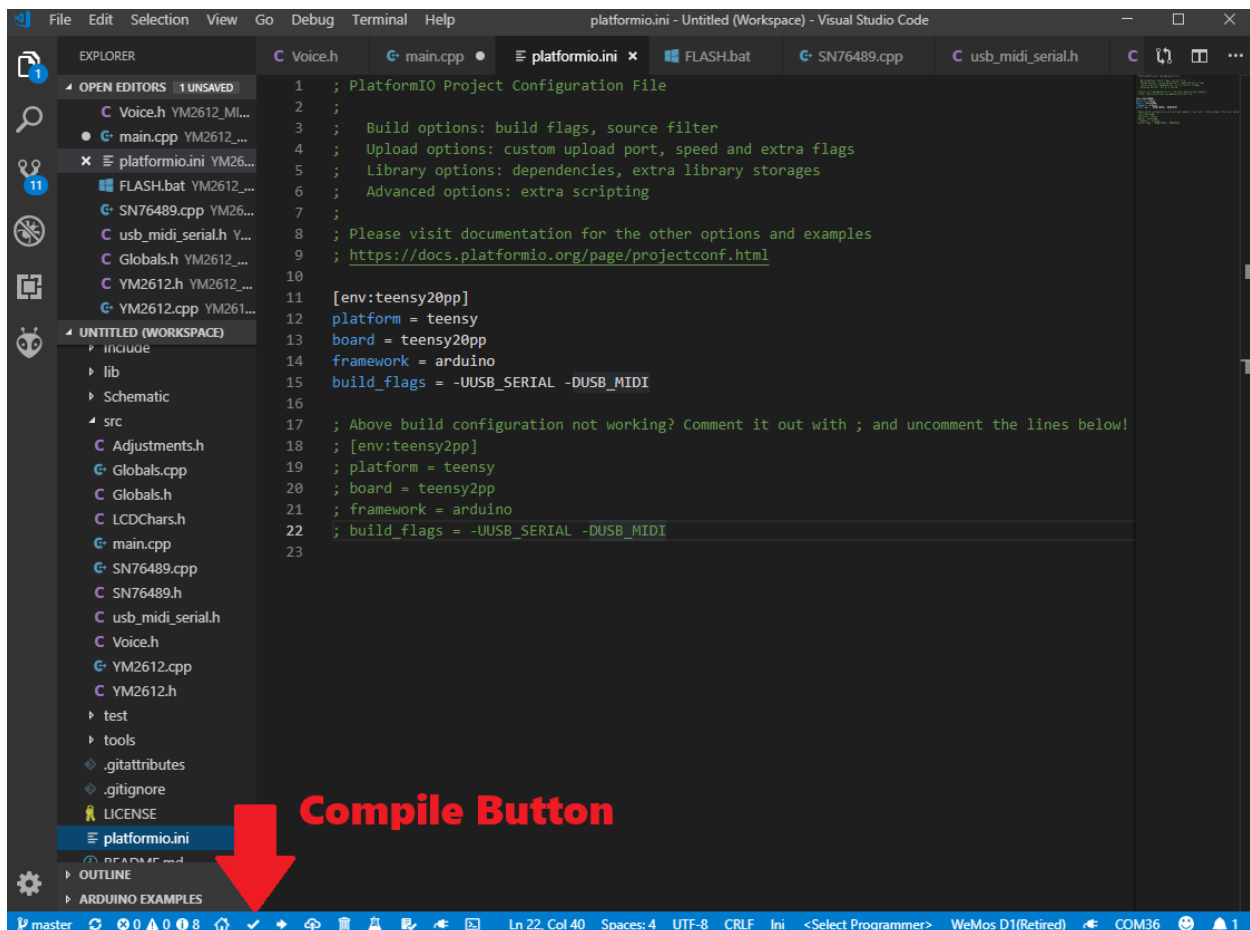
Precompiled HEX files can be found here: <https://github.com/AidanHockey5/MegaMIDI/releases>

Environment

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.



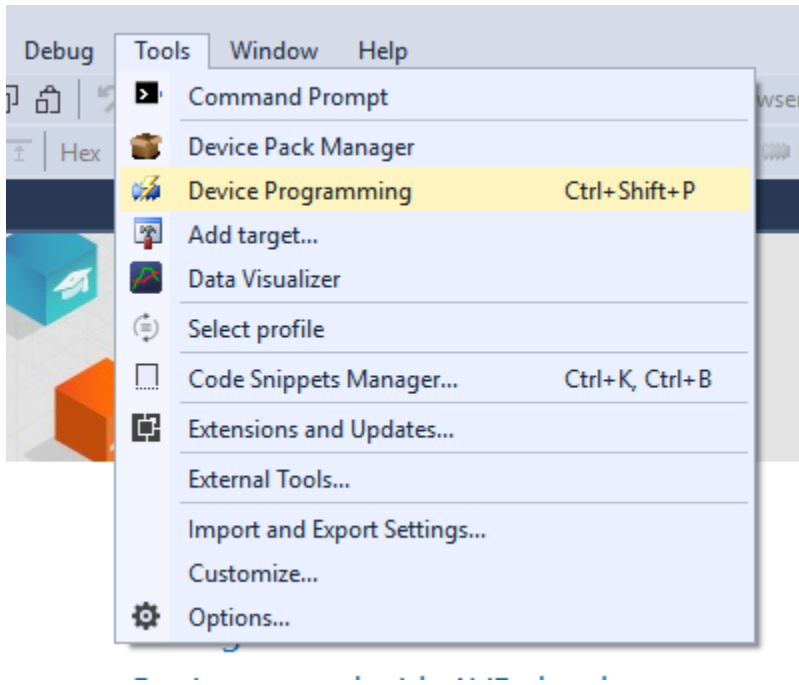
Flashing your HEX file

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.

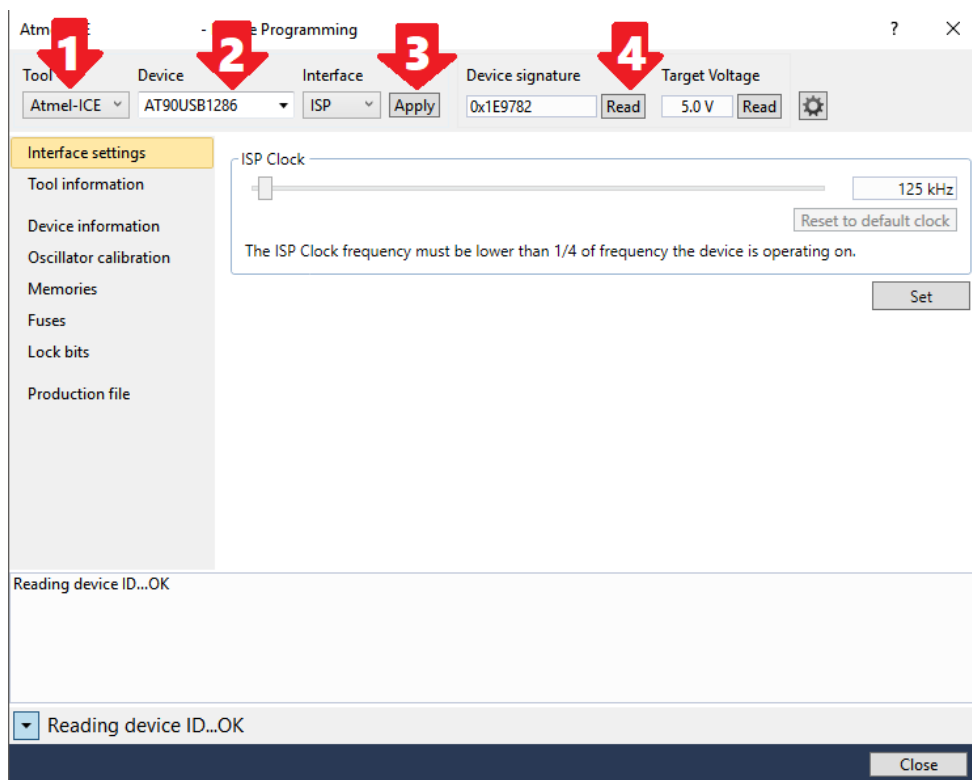
If you have an Atmel ICE, or any other standard Atmel ICSP device, you can use Atmel Studio to flash your HEX file.

Attach your ICE programmer and open Atmel Studio.

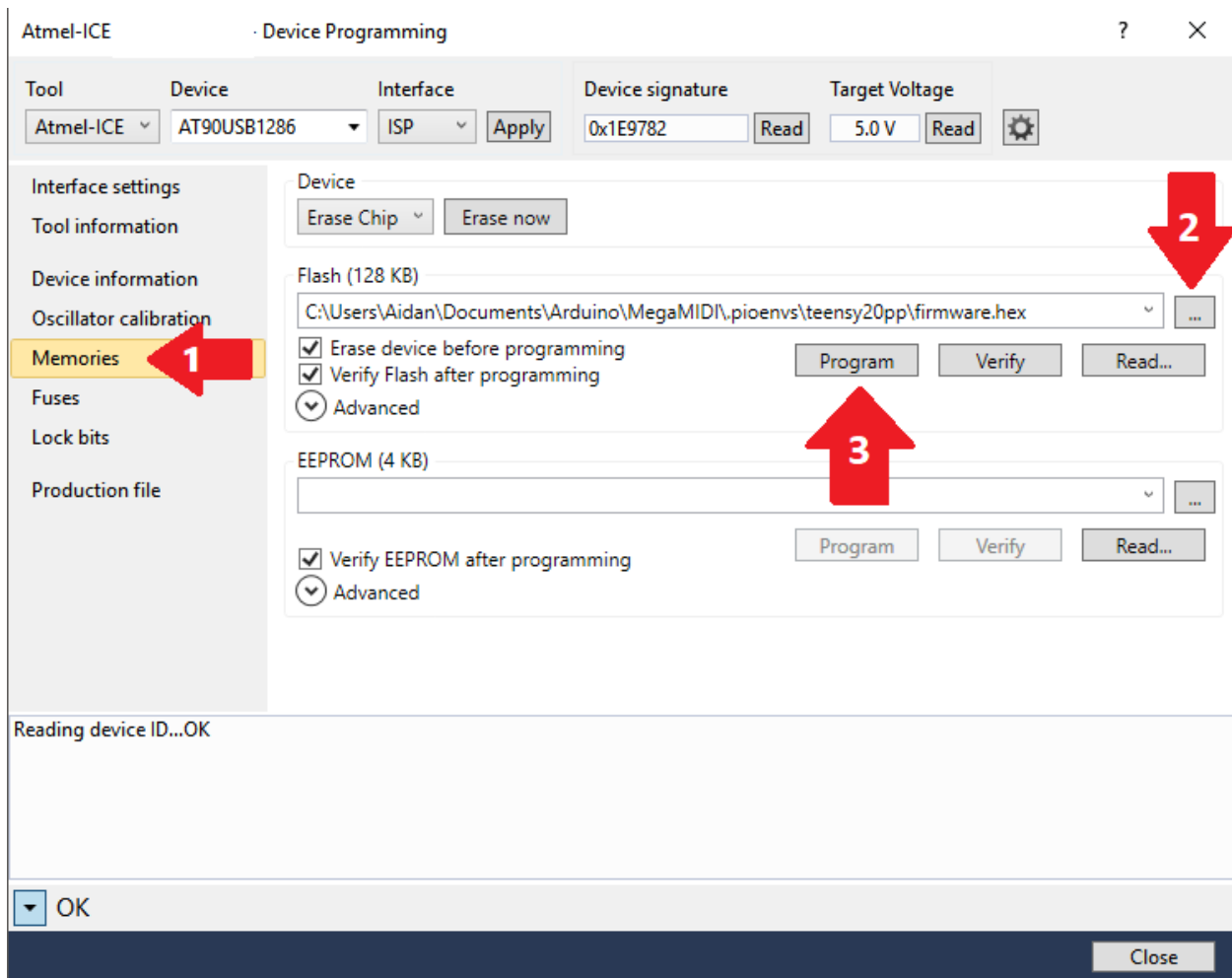
Under the **Tools** menu, select **Device Programming**.



Make sure your Mega MIDI is plugged into 12V! Attach the **AVR side** of the ICE to the **ICSP 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.



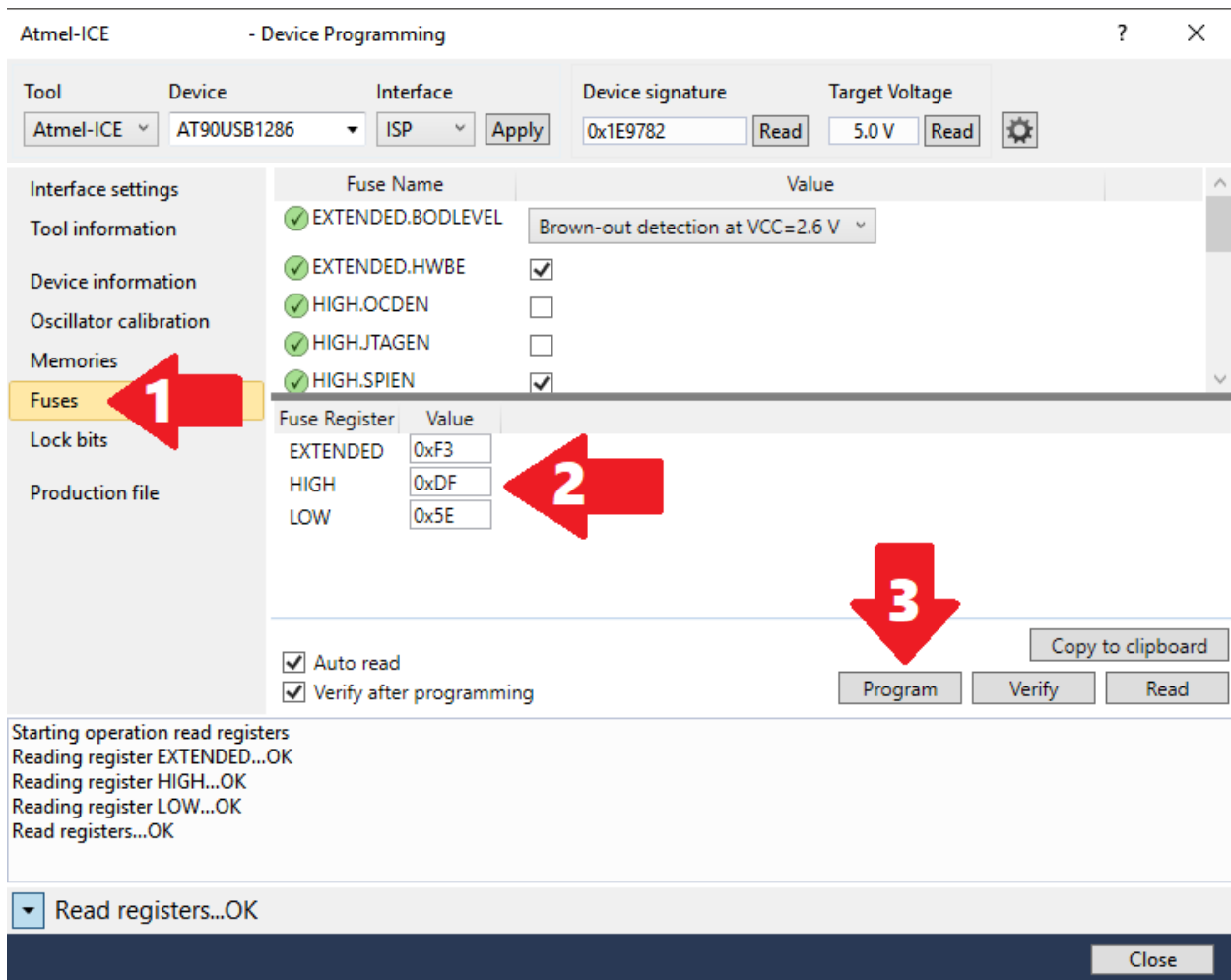
Next, head over to the “Memories” tab.



Select your HEX file. PlatformIO will store this hex file within the “.pioenvs\teensy*\firmware.hex” folder within 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.



The fuse values should read:

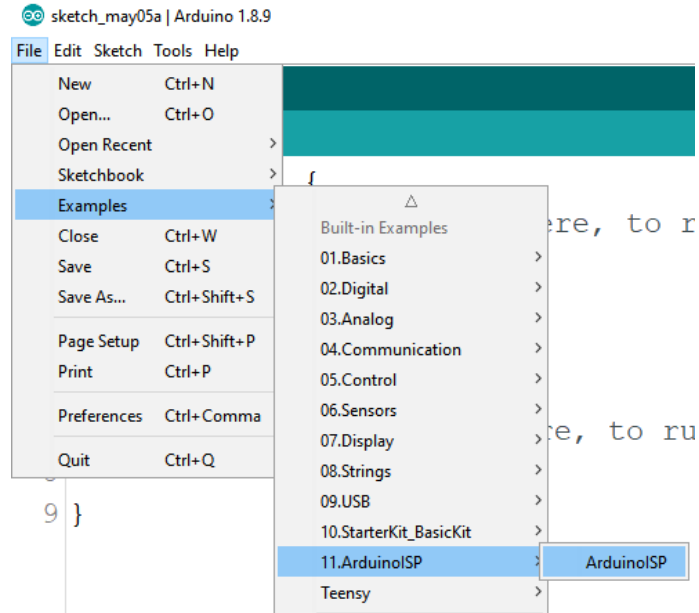
EXTENDED	0xF3
HIGH	0xDF
LOW	0x5E

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.

Flash Firmware With an Arduino

You do not need a fancy in-circuit serial programmer like the Atmel ICE to reprogram your Mega MIDI. In fact, you can just use a plain Arduino Uno/Nano board.

Open the Arduino IDE and load up the ArduinoISP Example Sketch.



It is strongly recommended to go to line 142 and comment out:

```
#define BAUDRATE 19200
```

And **uncomment**:

```
#define BAUDRATE 1000000
```

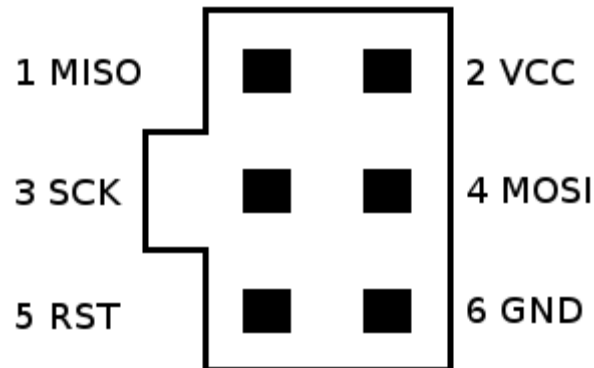
This will make your upload speeds significantly faster. You may choose any of these speeds, but make sure to keep your chosen upload speed in mind for the following steps.

Upload this script to your Arduino Uno/Nano.

While you're here, take a peek at the **Tools->Port** menu in the Arduino IDE and jot down **which port your Uno/Nano is on**. This will also be important later.

Once the ArduinoISP sketch is uploaded, it's time to connect your pins.

The Mega MIDI's ICSP header has the following pin-out. Refer to the silk-screen on the board to orient yourself.



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. 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.

MegaMIDI AVRdude Firmware Flasher

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.

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

```
Mega MIDI AVRDUDE Firmware Flasher Script
This program will flash your firmware automatically. Make sure you have ran the PlatformIO "Compile" function first to generate 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
```

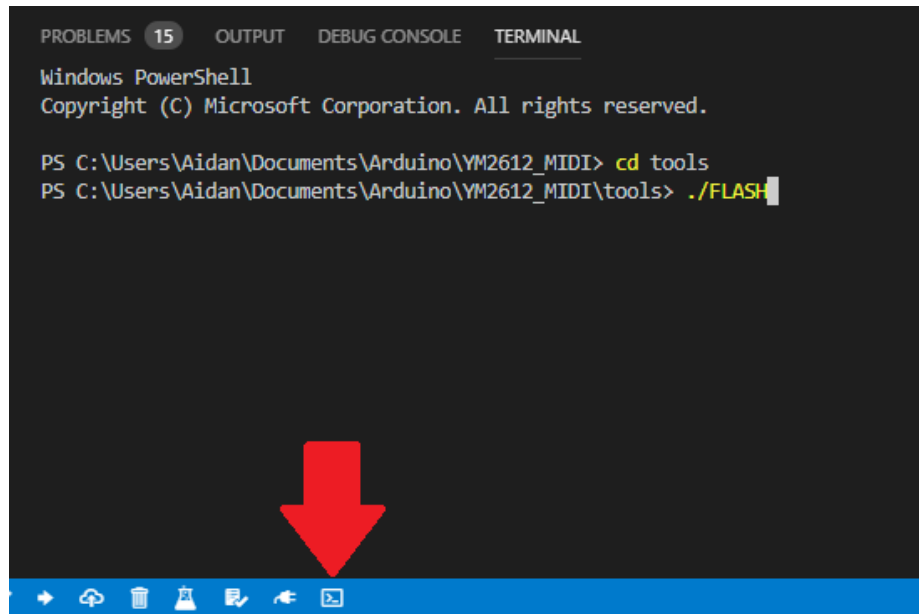
After you hit enter, give the script a few moments to communicate with your Arduino and it should begin flashing.

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.



```
PROBLEMS 15 OUTPUT DEBUG CONSOLE TERMINAL
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 115200 -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.