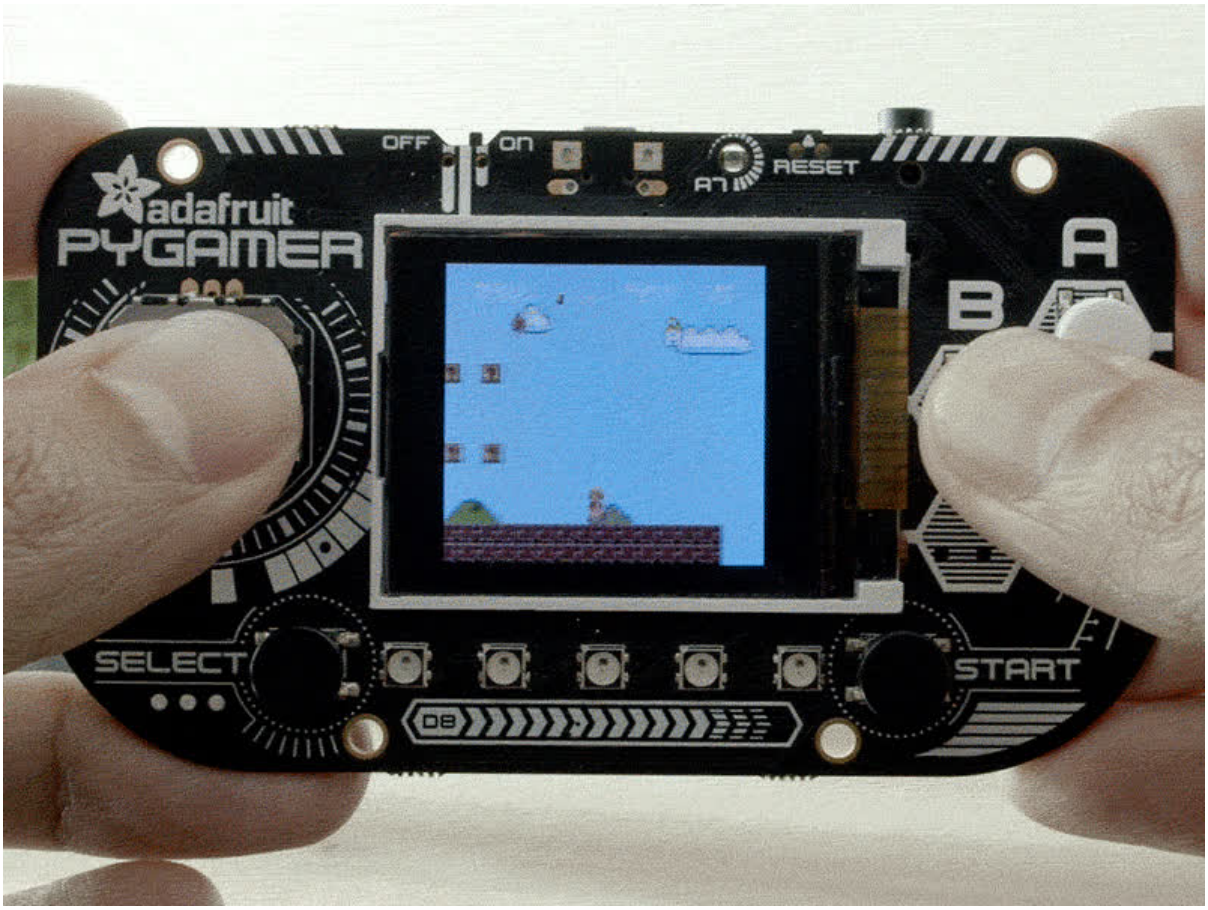
NES Emulator for Arcada

Created by lady ada

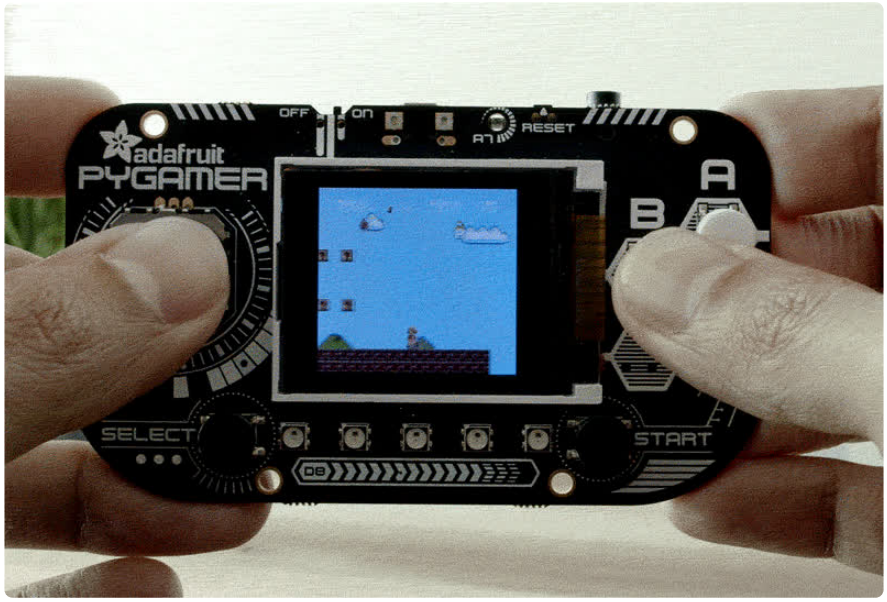


https://learn.adafruit.com/nes-emulator-for-arcada

Last updated on 2024-06-03 02:48:05 PM EDT

Table of Contents

Overview



That Arcada player is great for writing your own games in MakeCode Arcade or CircuitPython - but if you'd like to try homebrew Nintendo design, you can play NES ROMs as well thanks to a port of nofrendo (https://adafru.it/E-m) to the ATSAMD51.

This emulator plays games from off the built in QSPI storage, at full speed and with sound as well! Play any ROM that's 256KB or smaller and save/restore game state so you can take a break whenever you like.

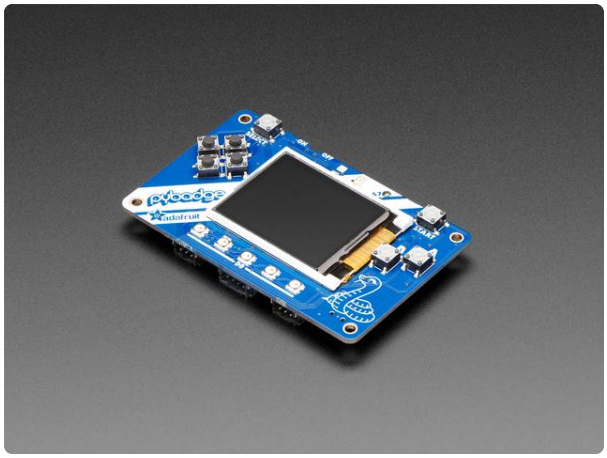
Try out some classic ROMs, or check out the amazing NES homebrew (https://adafru.it/E-k) scene to try out the classic 8-bit gaming platform that defined a generation

Supported Hardware

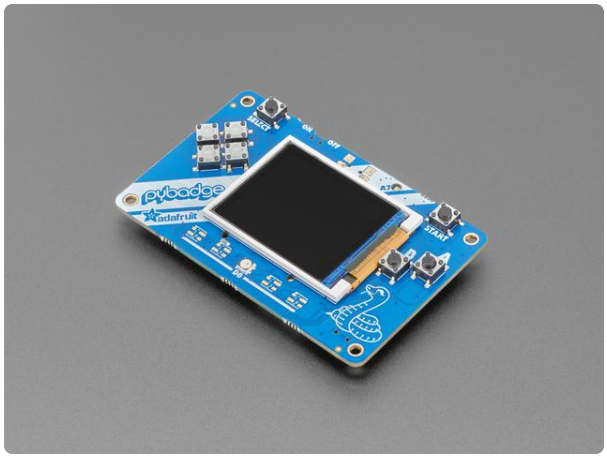
In theory any board with Arcada support (https://adafru.it/EF5) - but we recommend ones that have a gamepad + buttons:

[Adafruit PyGamer Starter Kit]
Please note: you may get a royal blue or purple case with your starter kit (they're both lovely colors)What fits in your pocket, is fully Open...
https://www.adafruit.com/product/4277

[Adafruit PyGamer for MakeCode Arcade, CircuitPython or Arduino]
What fits in your pocket, is fully Open Source, and can run CircuitPython, MakeCode Arcade or Arduino games you write yourself? That's right, it's the Adafruit...
https://www.adafruit.com/product/4242



[Adafruit PyBadge for MakeCode Arcade, CircuitPython, or Arduino]
What's the size of a credit card and can run CircuitPython, MakeCode Arcade or Arduino? That's right, its the Adafruit PyBadge! We wanted to see how much we...
https://www.adafruit.com/product/4200



[Adafruit PyBadge LC - MakeCode Arcade, CircuitPython, or Arduino]
What's the size of a credit card and can run CircuitPython, MakeCode Arcade or Arduino even when you're on a budget? That's right, it's the Adafruit...
https://www.adafruit.com/product/3939

Things it does:

- Emulate a large number of ROMs/games using the nofrendo core
- Play at full speed most of the time (if there's a lot of sprites from various parts of the ROM there may be a little slowdown)
- Play game audio through headphones/speaker
- Downsample for 160x128 displays (4-to-1 pixels), or 1:1 pixels if a 320x240 display
- Save and restore game state
- Support Adafruit Arcada boards like PyGamer and PyBadge
- Overclock the chip quite a bit

Things it does not do:

- Play ROMs larger than 256KB
- Play non-NES ROMs
- Some ROMs don't work - e.g. Dragon Warrior
- Support non-SAMD51 chips (for Teensy support, check out TeensyCEC (https://adafru.it/E-n))
- Support other hardware unless you've added support to Arcada (which you can!)

Quickstart

Unless you need/want to recompile the source code, we recommend just quick-starting by installing the following UF2s onto your Arcada board.

QSPI Filesystem

These UF2s default to using QSPI storage. So you must have a CircuitPython/FAT filesystem already in place. This is created by CircuitPython when its first installed, if you've never loaded CircuitPython, check this page for the UF2s. Install the matching one, check to verify a CIRCUITPY drive appears on your computer and then you can re-load Arcada_Nofrendo

CircuitPython Downloads

https://adafru.it/Em8

Arcada Nofrendo UF2s

Doubleclick Reset to put your board into **BOOT** mode, and drag these **UF2** files over!

pybadge or edgebadge nofrendo.UF2

https://adafru.it/E-g

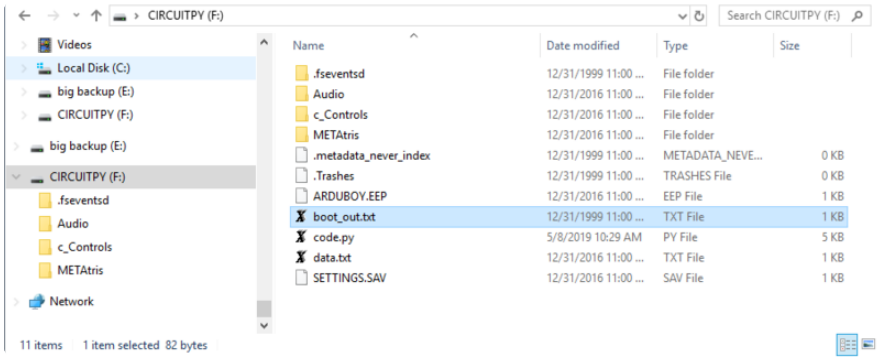
OR

pygamer_nofrendo.UF2

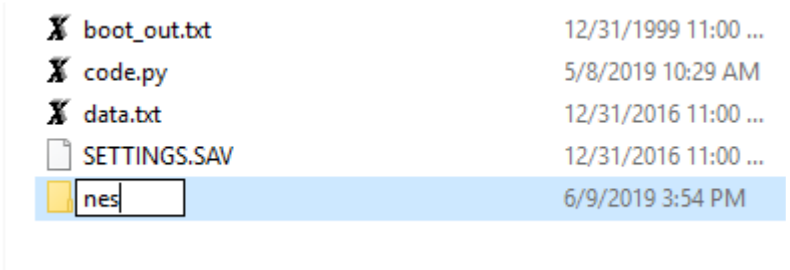
https://adafru.it/E-h

Installing ROMs

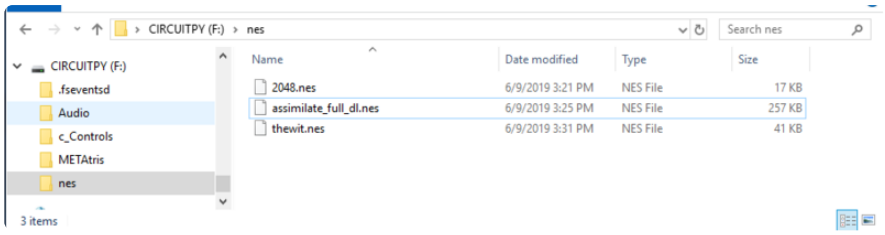
Adding ROMs is really easy. Once you've installed the UF2s above, your board will show up as a **CIRCUITPY** drive on your computer. This is the QSPI internal storage which is 2MB or 8MB. If you've run CircuitPython, or our Arduboy/Gamebuino demos, you may even have some files on there already.



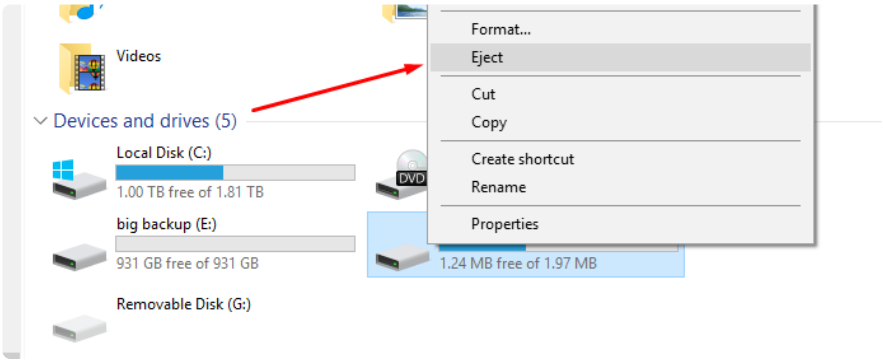
Make a new folder called **nes**



And inside of that folder, place all your **nes** files. You can create subfolders if you like.



Right-click on **CIRCUITPY** and **Eject** the drive (or drag to Trash can if its a Mac) to make sure your files were saved.



Starting ROMs

Press **Reset** to reload nofrendo. You'll see the selection menu:

You can use the joystick/D-pad to move up and down to scroll through the list of games.

If you hit the **B** button it will go up a folder level.

If you hit the **A** button on a game it will launch the game. If a folder is selected it will navigate into that folder.

Unless you recompiled to load ROMs into the SAMD51 chip's RAM, it will now load the game into the FLASH memory. This is sorta like the bootloader but it starts at the end of memory.

As it loads, if the FLASH already contains the data, it won't erase/re-write so the game will load much faster the second time. If the game won't fit, you'll get an error

That's it! Your game will be loaded and ready to play

ROMs to Try

There's hundreds of homebrew ROMs to try out. Here's a few favorites but don't let that stop you from seeking out your own fav's.

- https://www.zophar.net/pdroms/nes.html (https://adafru.it/E-j) Has public domain ROMs
- http://www.nesworld.com/article.php?system=nes&data=neshomebrew (https://adafru.it/E-k) has more NES homebrew games.
- Google for 'NES Homebrew'!

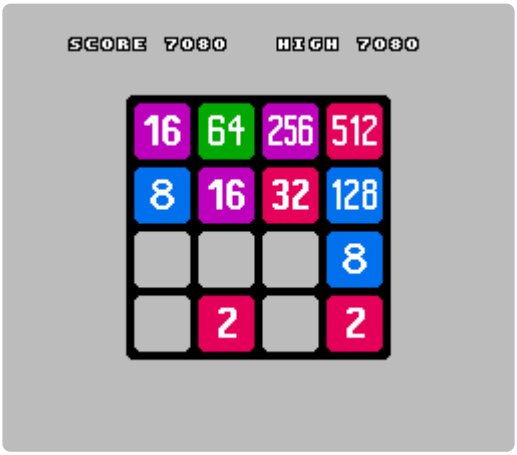
2048

This port of the popular & addictive sliding-tile game is a DIY ROM by tsone and it's fun to play & learn from.

Download from here (https://adafru.it/E-a)

2048.zip Mirror

https://adafru.it/E-b



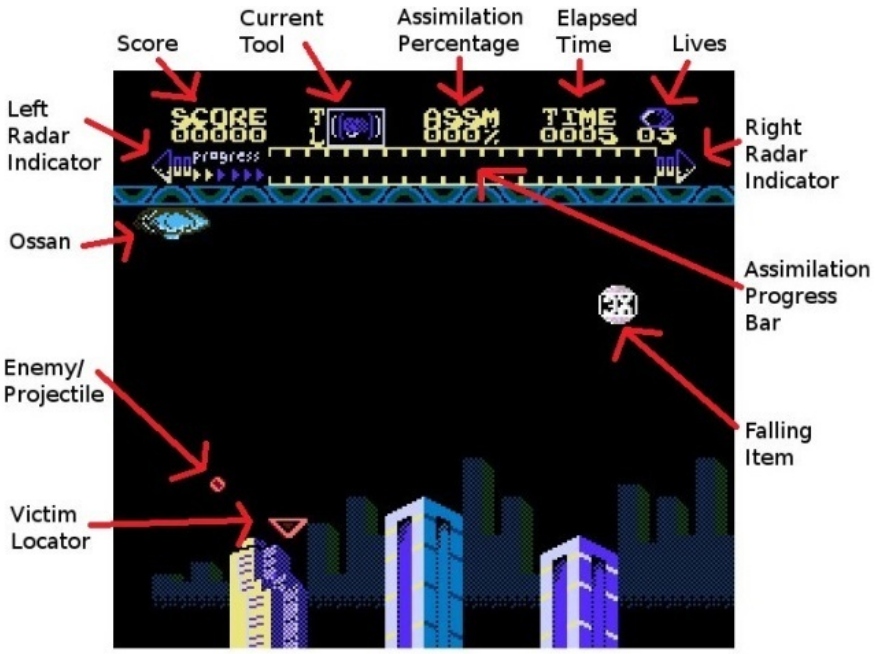
Assimilate

An alien adventure game, where you play the alien attacker!

Download from here (https://adafru.it/E-c)

assimilate.zip Mirror

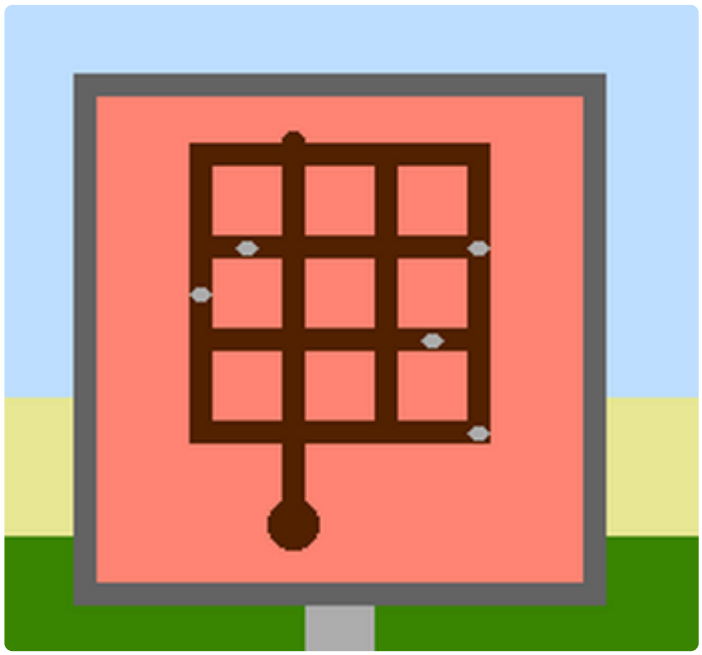
https://adafru.it/E-d



The Wit.NES

A homebrew port/tribute to The Witness - this game is a great puzzler with mysterious rules and an island with unknown history...

Download it here (https://adafru.it/E-e)



thewit.zip

https://adafru.it/E-f

Save/Restore State

Nintendo games were...surprisingly difficult. If you're from a later generation you may be shocked by the small number of lives and fact you often get kicked back to the beginning when you lose. Some games have codes that would display you could enter in that would let you restore to a middlepoint, and a few games had battery backed SRAM - Legend of Zelda and Final Fantasy were famous examples.

At this time, we don't have just-SRAM save/reload support. **However, we do have full-state save/reload support**. That means we backup the entire NES system at any exact point in gameplay. Which, I think, is a little better - just save the full game state whenever you like! You can even backup and restore your save games from a computer if you have a particularly nice game you want to keep.

Make backups of your save states! We've had one get corrupted, if you have a long game you're playing, back up the sav file on your computer.

At this time there's only one save state per game! If you'd like to add multi-file save support, we'd love to see it, but have no plans to do so (https://adafru.it/E-i)

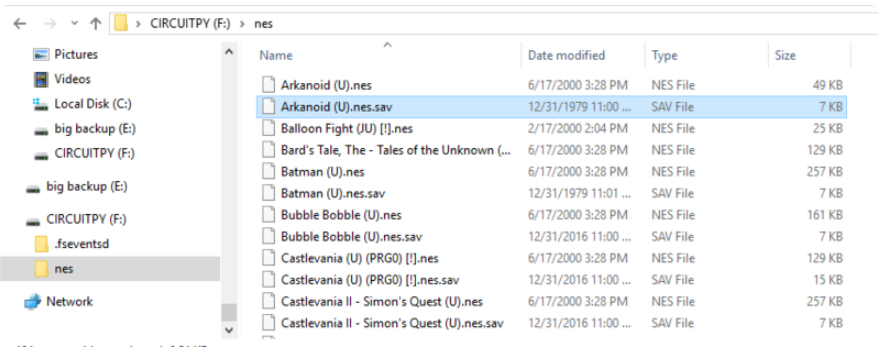
Saving State

Saving is easy - hold down **Select** + **Start** for a second, and you'll get a little popup menu:

Use the joy-pad to scroll through the menu and press **A** to select your choice:

- **Continue** - means, well, continue with the game as is, no save or restore
- **Save** - save the current game state to disk right now.
- **Reload Save** - restore the game state from disk from the last time you saved, if you want to 'do over'.
- **Save & Exit** - Save state and quit game back to main menu
- **Exit** - quit game back to main menu but don't save the game state.

If you check your **CIRCUITPY** drive, you'll now see **nes.sav** files that match each game. You can back those up if you like.



Next time you load that game you'll have a new menu pop up:

Load Save Game will load the last state saved

Start New Game will ignore the save file, and just load the game fresh

Delete Save File will remove the file from the filesystem, handy when something went wrong with the save (which happens)

Build in Arduino

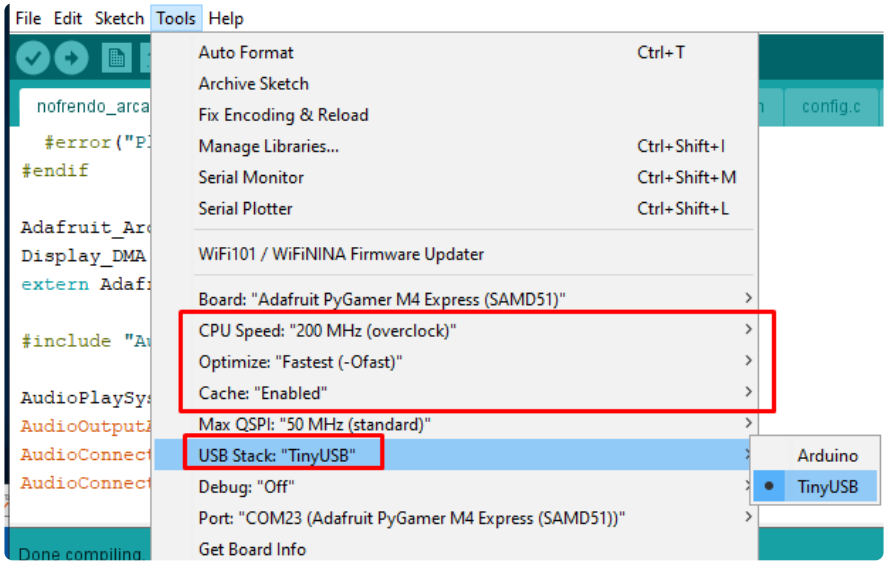
This project is in Arduino, with heavy C additions. You can build it yourself if you want to add support for a different board, or try different settings.

First up - add support for your board in Arduino so you can compile & upload code. Then load all the Arcada libraries (https://adafru.it/EUk).

Download/clone the Arcada nofrendo port from here (https://adafru.it/E-i), rename the folder `nofrendo_arcada` and open it in Arduino.

Compilation Settings

You must have a few settings when you upload:



CPU Speed must be the highest possible, we like to overclock at 200MHz. Optimizations must be **-Ofast** we don't recommend **-funroll-loops** as it doesn't seem to speed up play and it makes the build very large. **TinyUSB** is required in order to have the disk drive show up.

Configuration Settings

In emuapi.h there are a few settings you can make:

```
#elif defined(ADAFRUIT_PYGAMER_M4_EXPRESS) ||  defined(ADAFRUIT_PYBADGE_M4_EXPRESS)
  #define EMU_SCALEDOWN       2
  #define USE_FLASH_FOR_ROMSTORAGE       // slows it down, but bigger roms!
  #define DEFAULT_FLASH_ADDRESS (0x40000-2048)  // make sure this is after this
programs memory, with unrolled loops we're at 222,192! we need a little more than
256KB since roms have 10 extra bytes
  #define USE_SAVEFILES
  #define USE_SRAM
#else
```

- `EMU_SCALEDOWN` is for taking the NES output and scaling it down to 160x128 display. Set to 1 if you have a 320x240 display!
- `USE_FLASH_FOR_ROMSTORAGE` puts the ROM in FLASH, required for games over like 48KB, but if you want highest speed you can comment this out to use RAM/ malloc.
- `DEFAULT_FLASH_ADDRESS` - Where we start burning the ROM in. **This must be after the bootlader (16KB) + arduino code (~200KB)** The default is pretty good, don't mess unless you know the math.
- `USE_SAVEFILES` is the save/restore support, takes a ton of RAM up when we save because it memory maps the whole file so comment out if you are running out of RAM

- `USE_SRAM` turns on/off the SRAM implementation of the emulator. Keep on please.

If you have a board with an SD card, you can enable SD storage instead of QSPI, info on how to do that is here. (https://adafru.it/E-l)