Retro Gaming with Raspberry Pi

Created by Phillip Burgess

https://learn.adafruit.com/retro-gaming-with-raspberry-pi

Last updated on 2024-03-08 02:29:45 PM EST

Table of Contents

Overview

I'm a child of the 1980s. Miami Vice! Skinny ties! Big hair! Honest, I had hair then...and every town had at least one good video game arcade.

Thanks to the super affordable Raspberry Pi and some clever software, anyone can re-create the classic arcade experience at home. Adafruit brings the genuine "clicky" arcade controls, you bring the game files and a little crafting skill to build it.

Classic game emulation used to require a well-spec'd PC and specialized adapters for the controls, so it's exciting to see this trickle down to a $40 computer.

Picking a Game Emulator

Before you begin with adding an arcade control you'll need some games to play! In particular we're going to be talking about Retro Gaming here

Retro gaming is one of the most popular uses for the Raspberry Pi, and there are now a multitude of **ready-to-go SD card** images packed with emulator software (though most require sourcing your own ROM files). The days of installing emulators onesy-twosey are behind us; and the Pi Store (an online app store for Raspberry Pi, recommended in earlier versions of this guide) was shuttered in 2015.

Which emulator is for me?

Before committing to a big arcade project build, we recommend **testing** one or more of these packages with a **keyboard** connected and confirm you can set up and run all the games you're most interested in...then move ahead with more interesting controls.

These are all free downloads, so there's no harm in downloading them all and seeing what fits your tastes. Also, as you work with each one and tweak and tune, it's not uncommon to have to wipe and start over. **Keep careful notes** of your setup process and any configuration changes you make!

Once you've downloaded an emulator image, burn it onto your SD card using these instructions (https://adafru.it/pHe)

Help - this emulator doesn't work!

If you encounter difficulty with these packages or just need tips on setting them up, **please visit the FAQ and/or support forum on the corresponding project's web site, n ot the Adafruit Forums.** They'll be better equipped to answer questions about their own software.

(We can only help out if you have questions specifically about **our arcade controls or the retrogame software** we'll introduce on the last page. Please ask for such help in the Adafruit Customer Support Forums (https://adafru.it/jlf)...do not submit support questions as bug reports on GitHub, they'll be ignored there...and be as specific as possible in describing both the variant of Raspberry Pi hardware you're using and which software package and release, exact version number or date, etc. Thanks!)

So! Here are a few gaming OSes we're currently aware of...

Once you've picked it out and installed it on your Raspberry Pi, you can continue to the next step

RetroPie

This is our recommended emulator package!

From the RetroPie web site **(https://adafru.it/qoa)**:

"RetroPie allows you to turn your Raspberry Pi or PC into a retro-gaming machine. It builds upon Raspbian, EmulationStation, RetroArch and many other projects to enable you to play your favourite Arcade, home-console, and classic PC games with the minimum set-up. For power users it also provides a large variety of configuration tools to customise the system as you want.

"RetroPie sits on top of a full OS, you can install it on an existing Raspbian, or start with the RetroPie image and add additional software later. It's up to you."

From the site's Download page (https://adafru.it/rA3), there are **separate versions** of RetroPie optimized for **single-** and **multi-core** Raspberry Pi boards (i.e. Pi Zero, original Model A or B, A+ or B+ versus the multi-core Pi 2 and Pi 3); these are not cross-compatible; be sure to start with the right version for your model of Pi!

In particular, we like the 3.8.1 release, as it includes some ready-to-run games such as Doom (see all releases here (https://adafru.it/sNB))

Click here to download the v3.8.1 release for Raspberry Pi 2 and 3 **(https://adafru.it/sNC)**

Click here to download the v3.8.1 release for Raspberry Pi Zero and 1 **(https://adafru.it/sND)**

Breaking down some of the jargon above...

- Raspbian is the standard and most popular Linux operating system distribution for the Raspberry Pi; it's what most people start with when setting up a Raspberry Pi for "normal computer stuff."
- EmulationStation is a graphical front-end that lets you select among different emulators or games installed on the system, as well as configure gaming controls and other options.
- RetroArch is among the broadest and most popular multi-platform emulators... this is the code that runs the actual games. But it's not the only one...RetroPie (and most other gaming OSes) collect several others for different situations.

PiPlay

From the PiPlay web site **(https://adafru.it/rA4)**:

"PiPlay, formerly called PiMAME, [is a] pre built Raspberry Pi OS made for gaming and emulation.

"Also included is a suite of software designed to reduce the complexity and time needed to setup a fully working system. An updater is included with the distribution."

PiPlay lacks the breadth and active development of RetroPie; it's **not currently Pi 3 or Zero compatible**, hasn't been updated since 2015 and the forums are falling victim to spambots. Still, some users report an easier time configuring things here than other distributions...as long as you're running on the right hardware.

Recalbox

From the Recalbox web site **(https://adafru.it/rA5)**:

"Recalbox offers a wide selection of consoles and game systems. From the very first arcade systems to the NES, the MEGADRIVE and even 32-bit platforms, such as the Playstation.

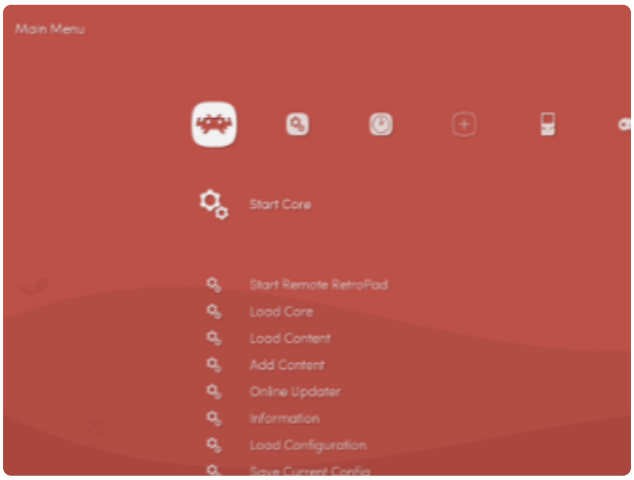
"With Kodi already included, Recalbox also serves as a Media Center. By connecting it to your home network, you will be able to stream videos from any compatible devices (NAS, PC, External HDD, etc.)."

A relative newcomer with an impressive variety of emulated systems. **One SD card image** (from their **"DIY recalbox"** link (https://adafru.it/rA6)) is **compatible with all current Raspberry Pi boards** (no separate single- or multi-core downloads).

It appears that recalbox has its own **support for interfacing arcade buttons and joysticks** to the Pi's GPIO header, **obviating the need for our own software** explained later in this guide. Have not experimented with this yet!

Lakka

From the Lakka web site **(https://adafru.it/rA7)**:



"Lakka is a lightweight Linux distribution that transforms a small computer into a full blown game console.

"Lakka is the official Linux distribution of RetroArch and the libretro ecosystem.

"Each game system is implemented as a libretro core, while the frontend RetroArch takes care of inputs and display. This clear separation ensures modularity and centralized configuration."

Lakka is not as all-inclusive as other packages, but as a result it's **extremely compact**. There are separate downloads for single- and multi-core Raspberry Pi boards.

Batocera

Batocera.linux (https://adafru.it/UBY) is another gaming-focused distribution that I've personally not had time to experiment with, but was highly recommended.

Since these are all free to download, might as well try some different options and see what feels best.

Adding Controls: Hardware

At this point you should have one (or more) games playable from the keyboard. The next step is adding nifty controls...

Console-Style Controls with USB

If you're aiming for a **home TV gaming console** feel, things are super easy: a variety of **USB-compatible gaming controllers** exist that'll plug straight into the Pi and you're done!

If aiming to simulate a particularly nostalgic console, chances are you can find an equivalent controller in USB format. Look around a site like NewEgg or just Google search for some options...I've seen classic-style controllers resembling the NES, Genesis, Playstation and more, as well as USB adapters if you'd prefer using actual original controls.

Pictured: Logitech F310

Arcade and Handheld Gaming Controls

Let's look at arcade-style controls first, since they're physically larger and thus easier to work with...

Just as there are USB console-style controllers, so too are there USB arcade-style setups. They're often a bit costly though...also, nearly all of them are laid out for 1990s fighting games. If the ready-mades meet your needs, fantastic. But maybe you want something simpler, or may have ergonomic preferences (I've always prefered right-handed joysticking, for example). Making your own control box (or even a whole cabinet) is a satisfying DIY project!

This is where things take a creative turn. There's no One Right Way™ to arrange controls that can cover every game. Instead, pick a few favorites and devise a layout that handles the most frequently-used inputs well. For everything else, you can still use a keyboard.

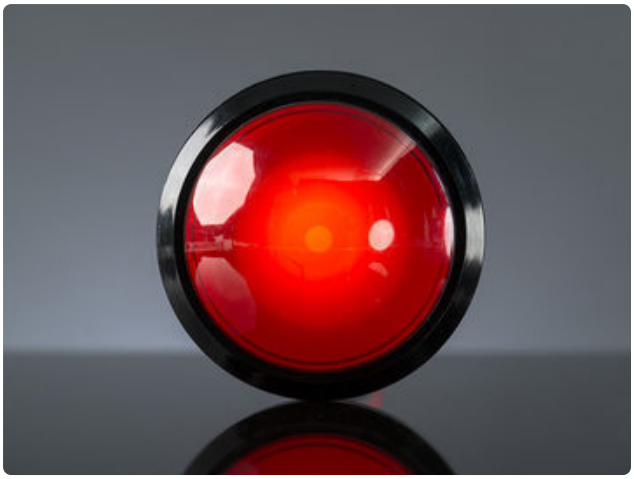
You'll also need to build your control panel using the **materials and tools best suited to your own skills**. I'm fortunate to have access to a laser cutter that can work with acrylic, but that's a tall order for most. Scrap plywood or a metal project box are viable materials (a cigar box works great too!), while a drill, hole saw, Dremel tool or wood rasp are all reasonable tools for making holes. Improvise!

We have a nice assortment of arcade-style controls in the shop...

Our Small Arcade Joystick (http://adafru.it/480) is the gamer's equivalent of the IBM Model M keyboard — clicky and built like a tank!

This is an 8-way "digital" joystick. Most classic games are designed for this type of stick, and it's easy to interface...the Raspberry Pi can't read proportional "analog" joysticks directly.

Our 30mm Arcade Button (http://adafru.it/471) is available in six different colors and is similarly industrial-grade. Any momentary push-type button ("normally open" contacts) will also work.

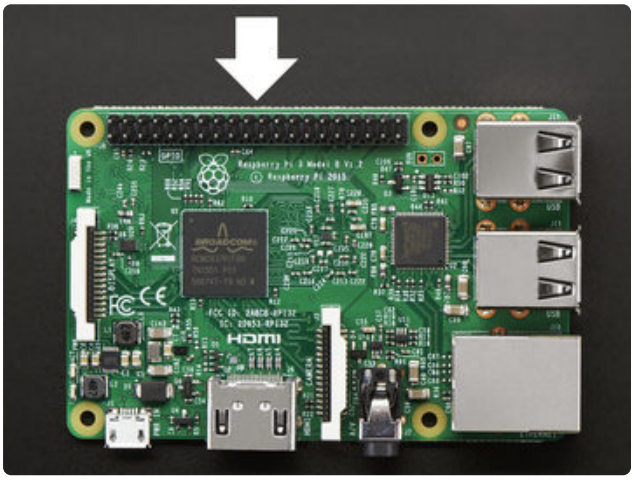


Large (http://adafru.it/1192) (60mm) and Massive (http://adafru.it/1187) (100mm) arcade buttons come in five colors and are irresistible! These bigger buttons are typically seen on quiz games rather than fast-twitch shooters.

A classic Atari or Commodore joystick can also be used, since these are just passive switches internally. Most later gaming consoles used serial protocols for their controls...unfortnately those won't work here...but as mentioned previously, USB clones exist for most.

We offer a variety of arcade controls, but we don't sell COMMON SENSE. When drilling and cutting, take precautions such as wearing SAFETY GLASSES, always cut AWAY from yourself, etc.

Connecting to the Raspberry Pi

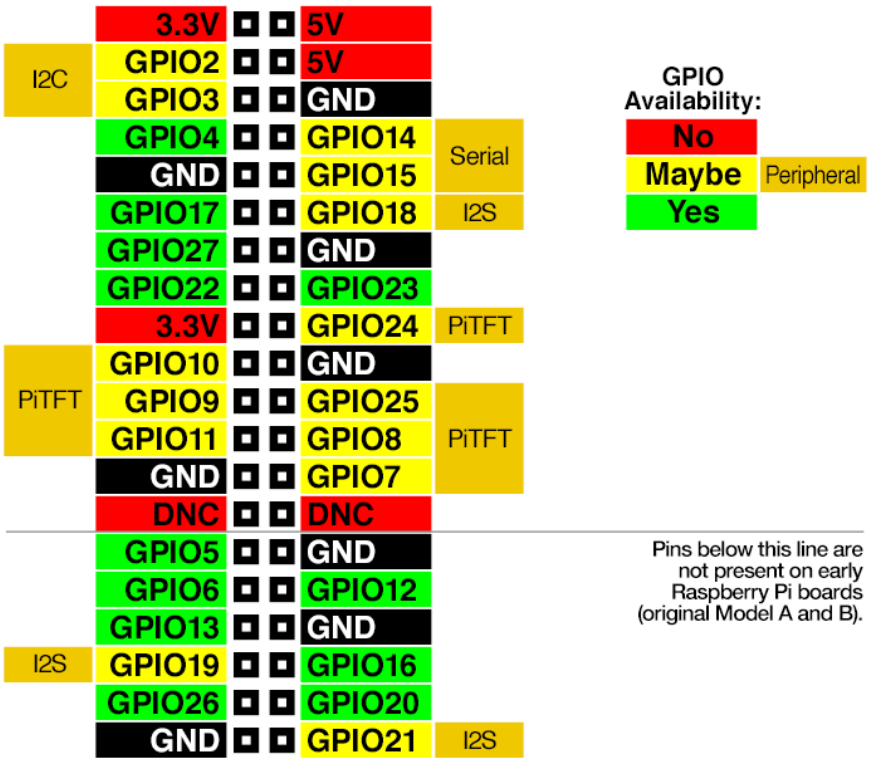


The controls will be wired to the 40-pin GPIO (general-purpose input/output) header on the Raspberry Pi board.

Early model Raspberry Pi boards had a 26-pin header...same idea, just with fewer spots to connect things.

Each pin on this header has a unique "GPIO number" ...not in-order, but we provide a map below for translating. Buttons and a joystick (each of 4 directions) will connect between any available GPIO pin and a ground (GND) pin.

This map is turned 90- degrees from the photo above...so the 5V pins are nearest the corner of the board, while GPIO21 is nearest the USB connectors:



- **All** of the **green pins** are fair game for connecting buttons.
- **Yellow pins may or may not be available** for controls, depending what hardware features are enabled on the system:
    - GPIO2 and GPIO3 are off-limits if using any I2C peripherals (for example, most real-time clock boards).
    - GPIO14 and GPIO15 are off-limits if using the TTL serial port (for example, with a serial console cable, or thermal printer).
    - If using one of our PiTFT displays (as most of our portable gaming projects do), another seven GPIO pins are out of commission: GPIO7-GPIO11 and GPIO24-25.
    - Our I2S audio amplifier board (https://adafru.it/qpB) needs GPIO pins 18, 19 and 21.
- **Avoid red pins**, these carry power and aren't suitable for controls.
- **Black pins** are **ground** points. The other leg of each button (and the "common" pin from a joystick) will need to connect to one of these. There are several scattered around...if you need more, one of our small Perma-Proto boards (http://adafru.it/1171) can be used to provide a single large "ground rail" where one side of all the buttons can be connected. Alternately, if you have extra unused GPIO pins and just need a couple extra ground connections, we'll show a software work-around for this.

Most any "passive" joystick or buttons should work — that is, they're internally just mechanical switches. This is the most common type. There are some higher-end

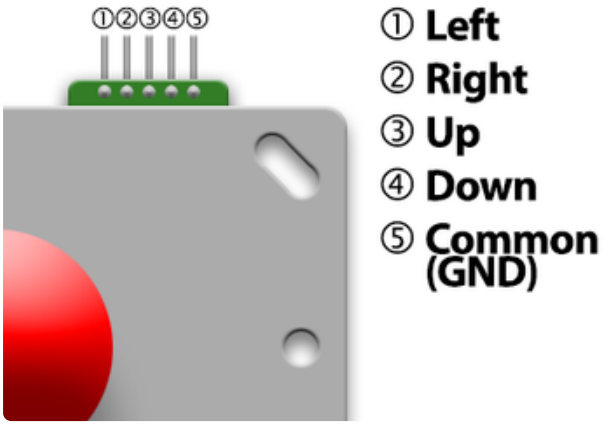
sticks and buttons using optical sensors that require power and output inverted logic, but our Retrogame software doesn't handle these.

Do I need Pull-up or Pull-down resistors?

You do not need external pull-up resistors for the buttons or arcade controller...they can wire direct to the GPIO pins and grounds. The Raspberry Pi has its own "internal" pull-ups.

Just as the layout and build technique of the control panel requires creative interpretation, so too will you need to decide on your own wiring methodology. We have some parts that can help. Aside from the aforementioned Perma-Proto board, there are quick-connect wires (http://adafru.it/1152) that work with the buttons and jumper wires (http://adafru.it/793) in various lengths that can be used with a joystick or plug directly into the GPIO pins (without a Perma-Proto board). Options abound! You'll likely need some combination of these, and may need to solder some connections.

Example: Connecting an Arcade Joystick & 2 Buttons



Here's a pinout diagram for our arcade stick. Only one wire needs to go to GND, then each of the other four goes to a different GPIO pin.

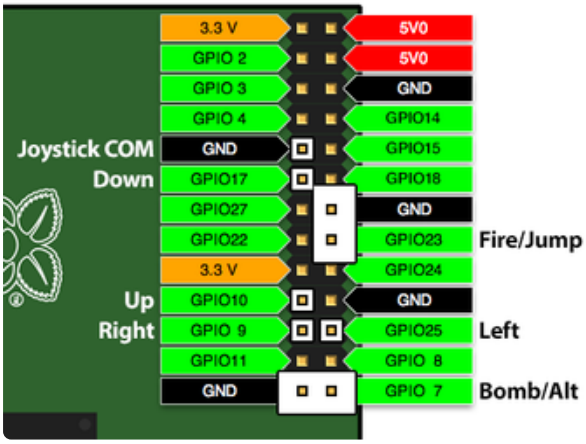
These directions apply when the stick is oriented **with the header at the top**. It's fine to install the stick in a different orientation, you'll just need to adapt the wiring connections to match.

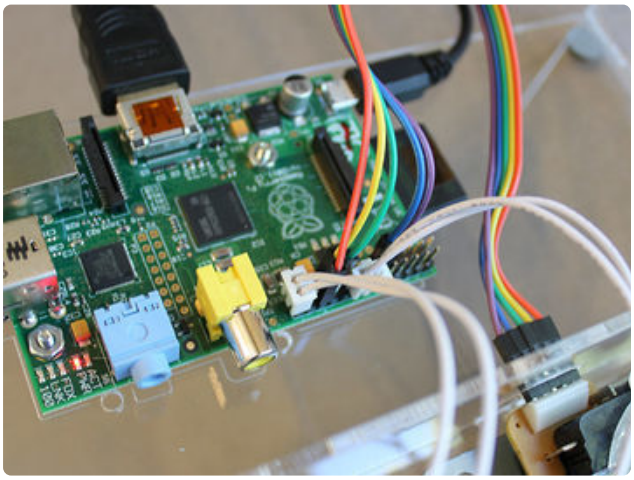
If using quick-connect wires for buttons: the end plug tends to block adjacent pins on the GPIO header. You can trim it down a bit using an X-Acto knife or Dremel tool, or cut the plugs off and solder the wires to a Perma Proto board, or simply plan out your wiring to avoid nearby pins...

- Joystick UP: **GPIO10**
- Joystick DOWN: **GPIO17**
- Joystick LEFT: **GPIO25**

- Joystick RIGHT: **GPIO9**
- Button A (one contact): **GPIO23**
- Button B (one contact): **GPIO 7**
- Joystick Ground, Button A & B second contacts: **GND** (theres a bunch of GND pins available)

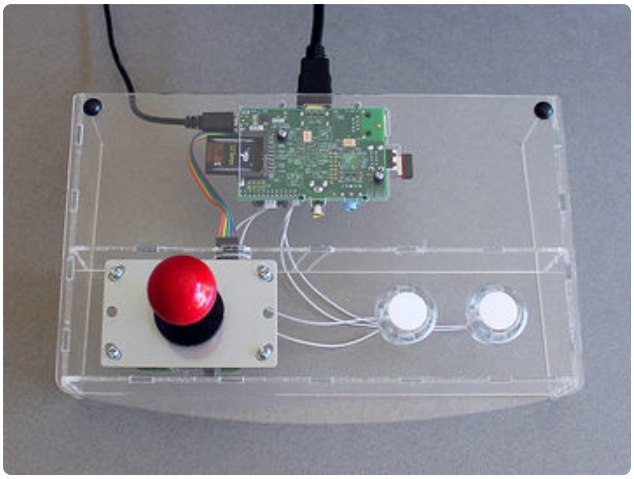
(This works fine with just HDMI TV output, but not with PiTFT displays which require several pins for their own use...you'll need to work out a different pin mapping in that case.)





Here's a basic **no-soldering** wiring setup we use for TV gaming with one joystick and two buttons, using 5 female-to-female jumpers (small white squares) for the joystick and two unmodified quick connects (larger white rectangles) for the buttons. Notice how the quick connects each span a GPIO and adjacent ground pin.

It's a pretty basic layout, but sufficient to accommodate quite a few classic games. For the remaining seldom-used functions (coin insert, start game), a regular USB keyboard is kept nearby.



An example case I made from laser-cut acrylic, but any workable material will do. **A cigar box makes a great fit!**

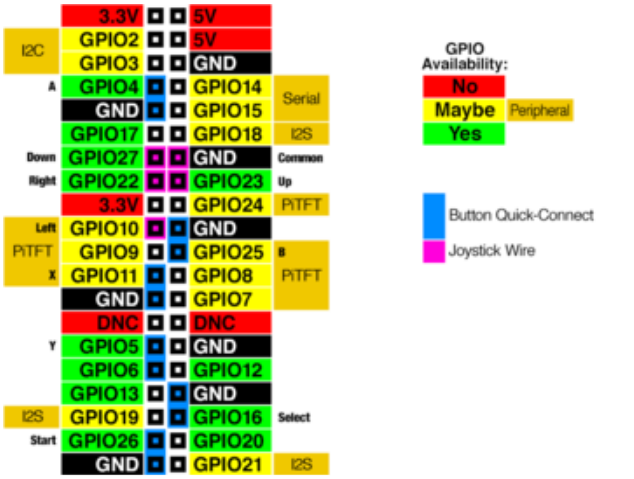
The case should be at least 50mm (2 inches) deep to accommodate the buttons and solderless connectors.

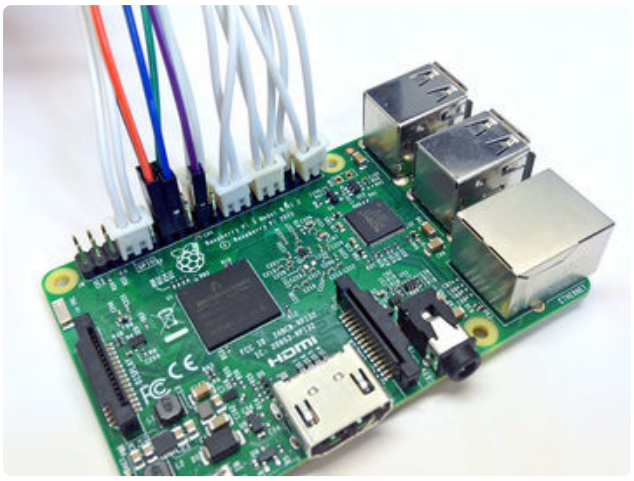
The shelf at the top holds a small USB keyboard. This is a catch-all for seldom-used functions (e.g. coin or game start). Only the essential controls were assigned to arcade buttons...but keep adding others if you like!

I no longer have the vector files for this laser-cut case. To be honest, it's pretty crude and minimal, and I would encourage you to design your own with all the controls you like, in whatever materials you're most comfortable working with.

Adding More Buttons

Here's an example layout if we want to add six buttons plus a joystick, using the same button quick-connects:





This layout avoids most of the peripheral pins (e.g. I2C), but does use pins that would be required for a PiTFT display...so this is only viable if using HDMI or composite video out.

This is mostly due to the size of the button quick-connect plugs. Trimming the flanges off with an X-Acto knife, you can probably create a more compact arrangement.

“A”, “B”, “X” and “Y” here refer to the controller button names used in EmulationStation, not necessarily the keys to which they’re assigned.

Sharp-eyed readers will notice the “Y” button is connected between GPIO 5 and 6, rather than 5 and GND. On the “Configuring Retrogame” page, we’ll explain how one or more GPIO pins can work as makeshift GND connections.

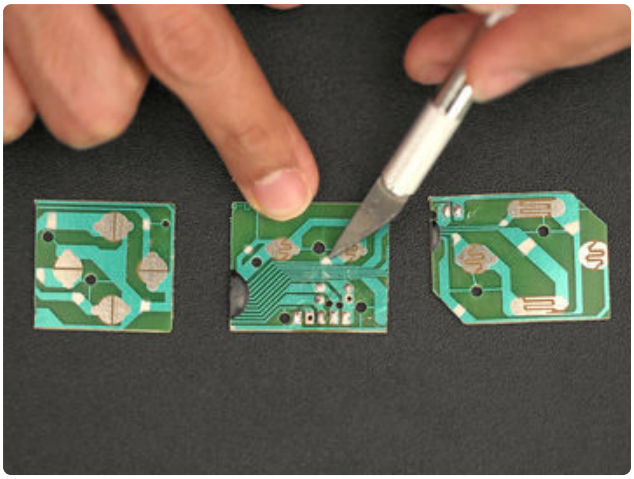
Example: A Mini Portable Gaming Handheld

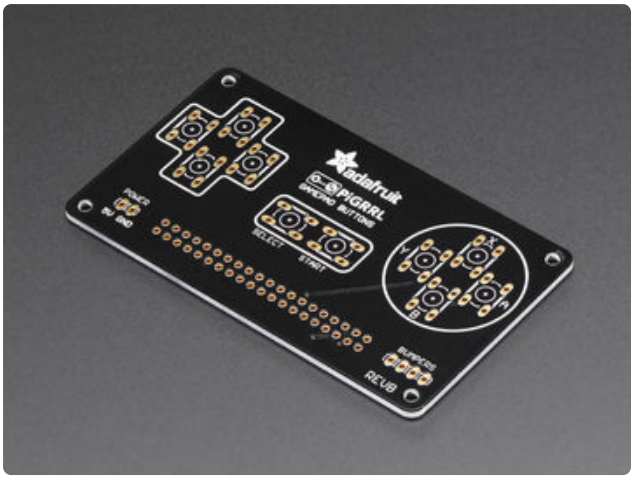
For handheld projects, the principles are the same as above, just squeezed into a very constrained space. Smaller components are used and the connections are a bit more challenging.

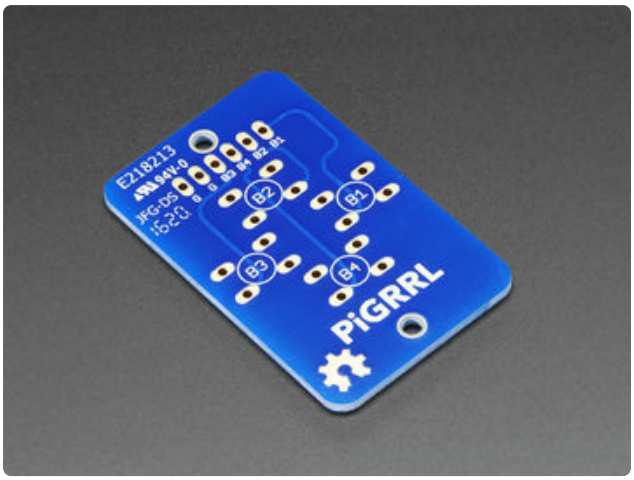
You'll find examples in our PiGRRL (https://adafru.it/kTD), PiGRRL 2 (https://adafru.it/rBm), Pocket PiGRRL (https://adafru.it/rBn), PiGRRL Zero (https://adafru.it/rBo) and Super Game Pi (https://adafru.it/rBp) projects.

Most of these use a PiTFT display, so there's a set of GPIO pins that are off-limits. Otherwise, it's all the same idea... buttons wired to GPIO and ground pins.

The Raspberry Gear (https://adafru.it/rBq) project uses a different approach...a microcontroller acts as an intermediary to the Pi's USB port.







In earlier projects, we'd hack apart an actual game controller to repurpose the buttons and circuit boards. More recently, we've started producing custom gamepad PCBs to which buttons and wires can be soldered:

PiGRRL 2.0 Custom Gamepad PCB (http://adafru.it/3015)
PiGRRL Zero Custom Gamepad PCB (http://adafru.it/2934)
"Soft" tactile buttons (http://adafru.it/3101)
"Clicky" tactile buttons (http://adafru.it/367)

Installing Retrogame

If using a USB game controller, there's nothing to see here...you're done, go ahead and play! These directions apply if you're using custom controls as covered on the prior page.

Our **retrogame** utility is the software glue that links our GPIO-connected controls to "virtual" keyboard input.

Some of our projects (eg PiGRRL Zero) provide a **ready-made SD card image**...if you're building with the same hardware combo, it's good to go with no changes, **retrogame is already installed.** Otherwise...

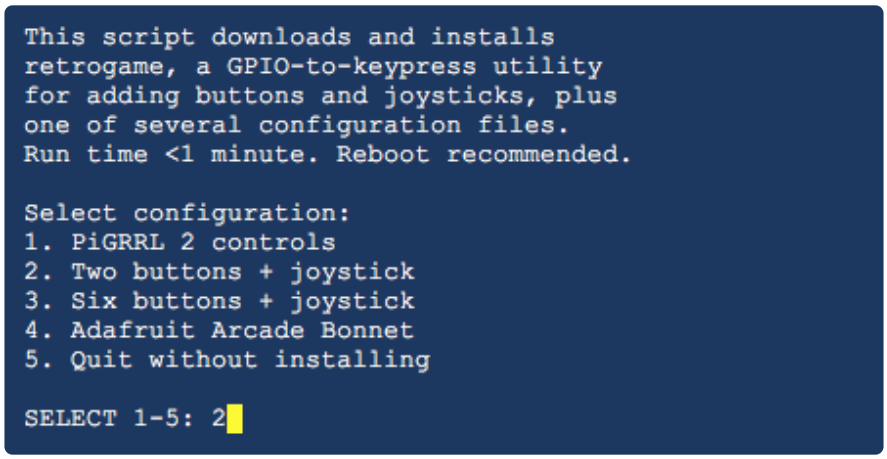
Download and Install

We have a script to help set this up. You'll need access to the Pi's command line... either log in over the network using ssh, or the emulation package you're using should have some option to access a command line (e.g. with RetroPie, press "F4").

Then type (or copy & paste, if using ssh):

```
cd
curl https://raw.githubusercontent.com/adafruit/Raspberry-Pi-Installer-Scripts/
master/retrogame.sh &gt;retrogame.sh
sudo bash retrogame.sh
```

When run, you'll be asked to pick one of several control configurations. For example, if using the simple joystick-and-two-buttons arrangement from the prior page, pick #2. If you're planning on making your own custom setup, just pick any of these, and we'll explain on the next page how to customize this.

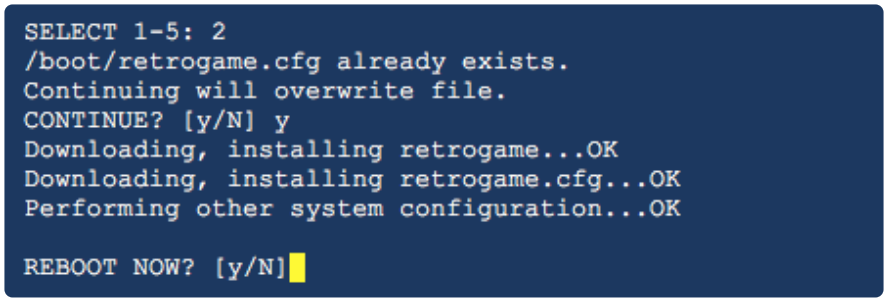


If you've installed retrogame previously, you'll be warned that its configuration file will be rewritten. Press "y" if that's okay. Otherwise, exit and make a backup of that file, so you can restore it later.

The script then downloads and installs the retrogame executable (/usr/local/bin/retrogame) and configuration file (/boot/retrogame.cfg), then performs some administration tasks.../etc/rc.local is amended to make retrogame launch on startup, and a "udev rule" file (/etc/udev/rules.d/10-retrogame.rules) is generated to help it simulate a keyboard.

Once finished (should only take a moment), you'll be prompted to reboot.

If you have other setup tasks to perform (installing other packages, or changing overscan settings using raspi-config), you don't need to reboot just yet...answer "n" in that case.



Configuring Retrogame

New retrogame: Settings File

The latest version of retrogame reads the file **/boot/retrogame.cfg** on startup. A sample retrogame.cfg file was placed there by our installer on the prior page.

Below is an example from the PiGRRL 2 configuration file: There's a lot of stuff in there but really it's pretty simple...most of it is comments, which start with a '#' character.

```
# Sample configuration file for retrogame.
# Really minimal syntax, typically two elements per line w/space delimiter:
# 1) a key name (from keyTable.h; shortened from /usr/include/linux/input.h).
# 2) a GPIO pin number; when grounded, will simulate corresponding keypress.
# Uses Broadcom pin numbers for GPIO.
# If first element is GND, the corresponding pin (or pins, multiple can be
# given) is a LOW-level output; an extra ground pin for connecting buttons.
# A '#' character indicates a comment to end-of-line.
# File can be edited "live," no need to restart retrogame!

# Here's a pin configuration for the PiGRRL 2 project:

LEFT       4  # Joypad left
RIGHT     19  # Joypad right
UP        16  # Joypad up
DOWN      26  # Joypad down
LEFTCTRL  14  # 'A' button
LEFTALT   15  # 'B' button
Z         20  # 'X' button
X         18  # 'Y' button
SPACE      5  # 'Select' button
ENTER      6  # 'Start' button
A         12  # Left shoulder button
S         13  # Right shoulder button
ESC       17  # Exit ROM; PiTFT Button 1
1         22  # PiTFT Button 2
```

```
2         23  # PiTFT Button 3
3         27  # PiTFT Button 4

# For configurations with few buttons (e.g. Cupcade), a key can be followed
# by multiple pin numbers.  When those pins are all held for a few seconds,
# this will generate the corresponding keypress (e.g. ESC to exit ROM).
# Only ONE such combo is supported within the file though; later entries
# will override earlier.
```

The important lines, in the middle, consist of a key name followed by a GPIO pin number. That's really all there is to it.

Your basic letter and number key names are straightforward...for function keys and such, you can look in the file keyTable.h (included alongside the retrogame executable) for valid names, or in `/usr/include/linux/input.x` (remove the initial "KEY_" from the name).

If you want to use a GPIO pin as a spare ground for buttons: use "**GND**" as the key name, then list one or more pin numbers separated by spaces.

If you need just one more key...not as a gaming control, but for accessing menus or similar...follow a key name with multiple pin numbers (these can already be assigned to other keys themselves, that's fine). When those buttons are all held for a few seconds, the corresponding keypress will be generated. We used this in our Cupcade project to access the Escape key. Currently only **one** such multi-button combo can be assigned.

If you want to put retrogame.cfg somewhere other than /boot, or assign a different name: in /etc/rc.local, where retrogame is being run, insert the absolute pathname to your configuration file as an argument (i.e. just before the "&"):

```
/usr/local/bin/retrogame /boot/myConfig.cfg &amp;
```

Locating the configuration file in **/boot** makes it easier to edit this file on a non-Pi system, since the SD card can be inserted in a USB reader and accessed from most any computer.

If editing in **/boot** directly on the Pi, you'll need to do this as root, i.e. `sudo nano /boot/retrogame.cfg`

If you change the name or location of the configuration file, you'll need to reboot or restart retrogame. But if you're just editing the pin assignments in the existing file, there's no need...**retrogame will pick up on these changes automatically**, while its running, once changes are written to the file.

Configuring Older Retrogame

This is for the old version of retrogame that required recompilation. You probably don't need this page but its here for historical purposes!

Classic retrogame: Edit Source Code

Assigning keys and buttons in the old retrogame is quite a bit more involved...this is why we recommend just downloading and installing the newer version. But if this is what you have around, here's the procedure...

Edit the retrogame.c file in the unZIPped Adafruit-Retrogame directory...

```
cd Adafruit-Retrogame
nano retrogame.c
```

About a hundred lines down, you'll see a table that looks like this:

```
ioStandard[] = {
        // This pin/key table is used when the PiTFT isn't found
        // (using HDMI or composite instead), as with our original
        // retro gaming guide.
        // Input   Output (from /usr/include/linux/input.h)
        {  25,     KEY_LEFT     },   // Joystick (4 pins)
        {   9,     KEY_RIGHT    },
        {  10,     KEY_UP       },
        {  17,     KEY_DOWN     },
        {  23,     KEY_LEFTCTRL },   // A/Fire/jump/primary
        {   7,     KEY_LEFTALT  },   // B/Bomb/secondary
        // For credit/start/etc., use USB keyboard or add more buttons.
        {  -1,     -1           } }; // END OF LIST, DO NOT CHANGE
```

Careful now...make sure you're editing the **ioStandard**[] table, not the similar-looking ioTFT[] table. The latter applied specifically to the Cupcade project and nothing else.

Within each set of braces is a GPIO pin number and a corresponding key name (from /usr/include/linux/input.h). Because we're editing C source code, it's vitally important to **maintain the syntax exactly.** Open brace, number, comma, key name, close brace, comma. The last item must be {-1, -1}.

If you need an extra ground pin (and have extra GPIO pins available that you're not using for controls), set the key code to **GND** instead.

Write your changes to the file and exit the editor, then type:

```
make retrogame
```

This should build the retrogame executable. If you instead get an error message, there's a problem in the edited table — most likely a missing curly brace, comma or semicolon.

Depending what's in /etc/rc.local, you might need to move retrogame to a different location (such as /usr/local/bin). Reboot to use the new settings. Test it out...if anything needs changing, you'll have to repeat the whole edit-compile-move-reboot sequence.

Arcade Cabinet Pack Assembly

[Raspberry Pi Arcade Cabinet Pack](https://www.adafruit.com/product/3272)
Couldn't get one of those Nintendo Mini Classic consoles? No problem, we've got you covered with this no-solder no-tools required kit! Video games are one of...
https://www.adafruit.com/product/3272

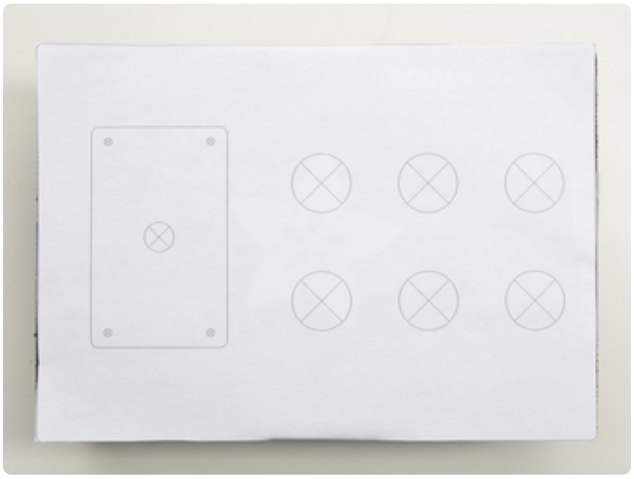
With the Arrow Presents: Raspberry Pi Arcade Cabinet Pack (http://adafru.it/3272) you can use every piece of the kit (including the box!) to make an arcade controller.

All thats needed is 4 #6 screws and bolts (available at your local hardware store), a Raspberry Pi (Pi 3 suggested) and an HDMI Monitor.

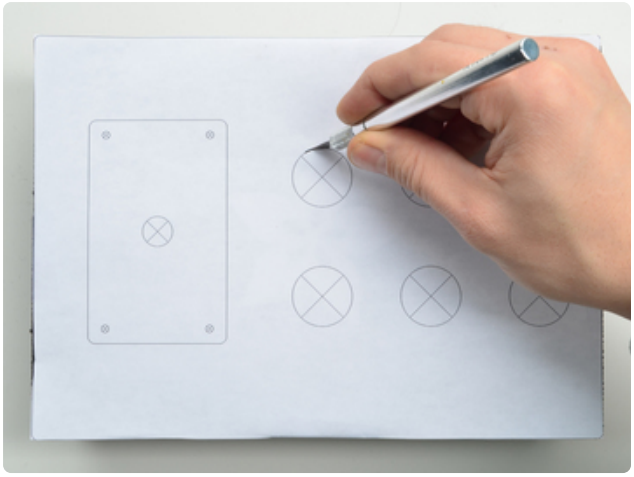
Start by downloading the template PDF below to get started!

Arcade_Template.pdf

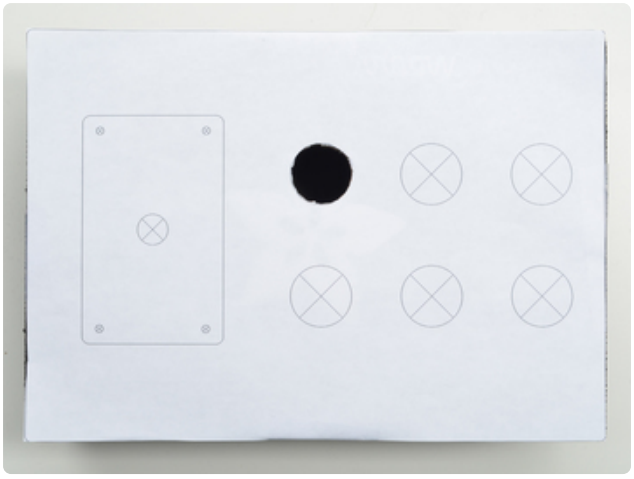
https://adafru.it/tjA



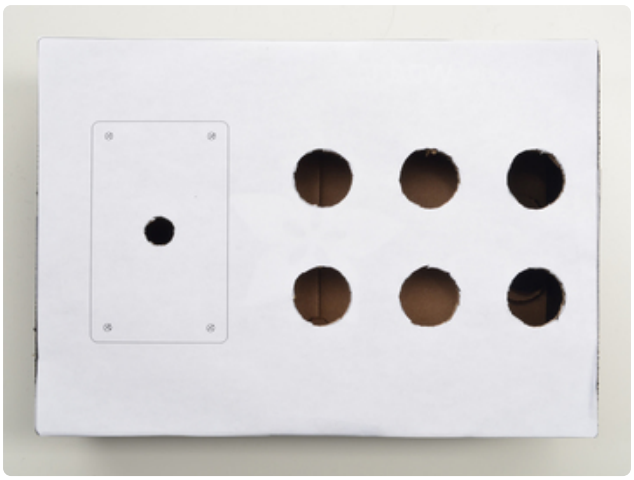
Cut along the outline of the PDF, and scotch tape it to the box.



Using an Xacto knife, cut along the 6 button holes, as well as the center hole on the left for the joystick.



Do not cut along the rectangle, that is used as a guide to make sure the holes line up properly.



For the 4 small holes around the joystick, pierce the box in an X shape, which will be enough to press the bolts through later

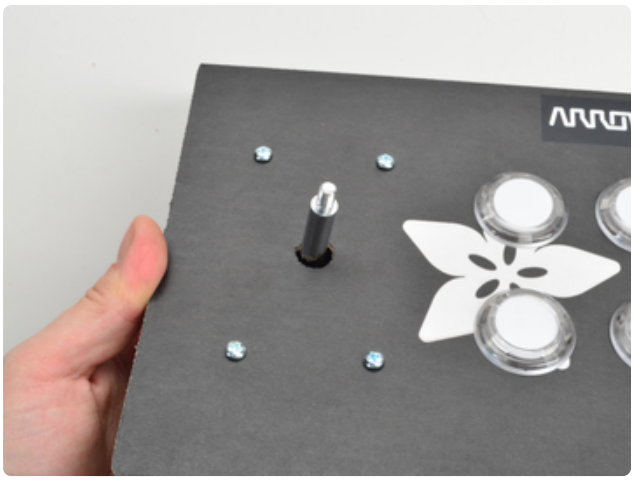
Gently push the arcade buttons in. They should be fairly snug, so support the cardboard from the back to prevent it from buckling.

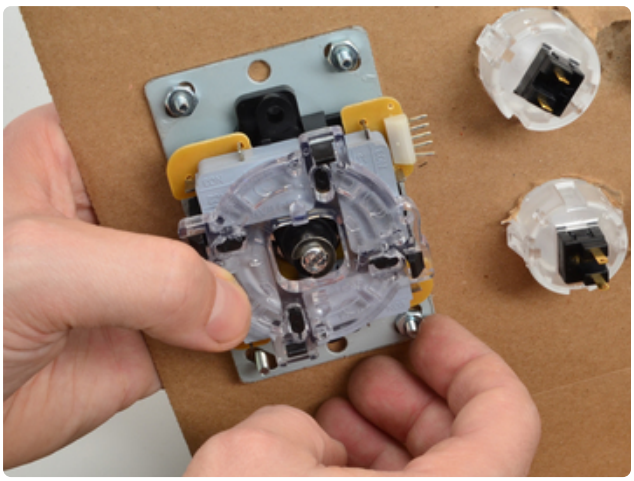
Once they're in, the box holds up surprisingly well.

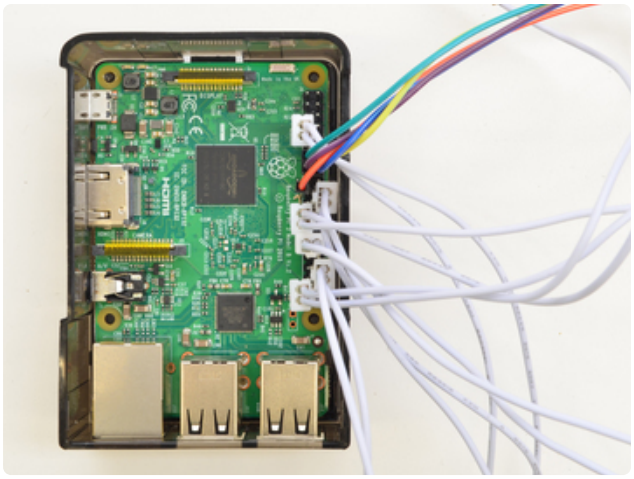
Press each bolt into the slotted holes.

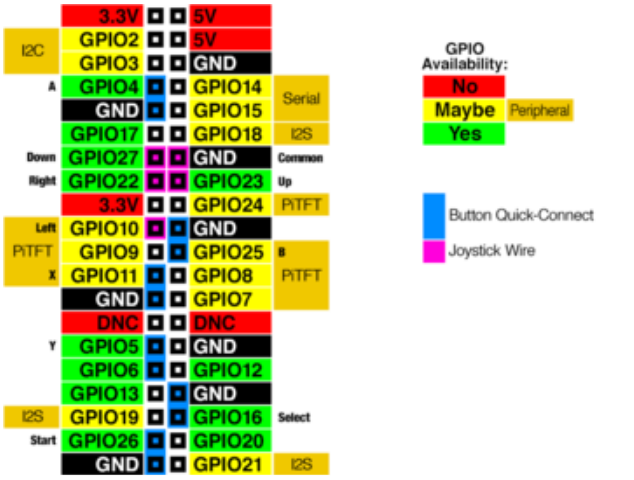
Unscrew the red ball from the joystick and fit it in the box from the back.

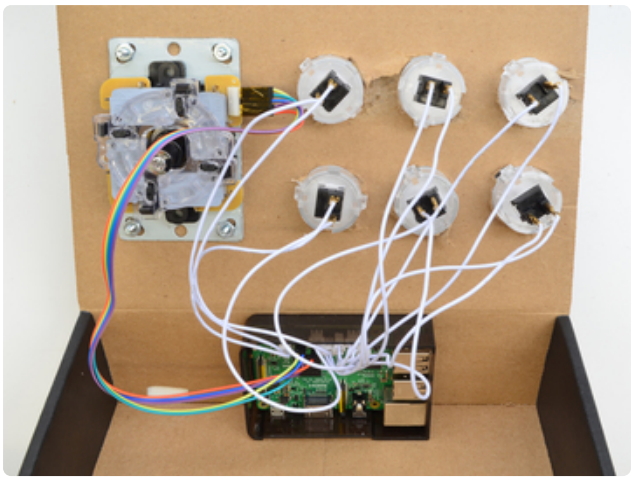
Line up the screws and bolt it in!











Plug in the buttons and arcade as shown here. You can refer back to this earlier page (https://adafru.it/sPA) for specifics.

You can also cut a small hole in the back to run HDMI & power cables :)

Finish the setup by following the rest of this guide to:

1. Installing RetroPie onto your Raspberry Pi (https://adafru.it/tjB)
2. Installing Retrogame and selecting "6-button + Joystick" (https://adafru.it/sct)

Configure your keyboard keys according to the RetroPie input configuration screen so you can use the joystick + buttons to move around RetroPie and play games!

RetroPie: Improving Emulator Performance

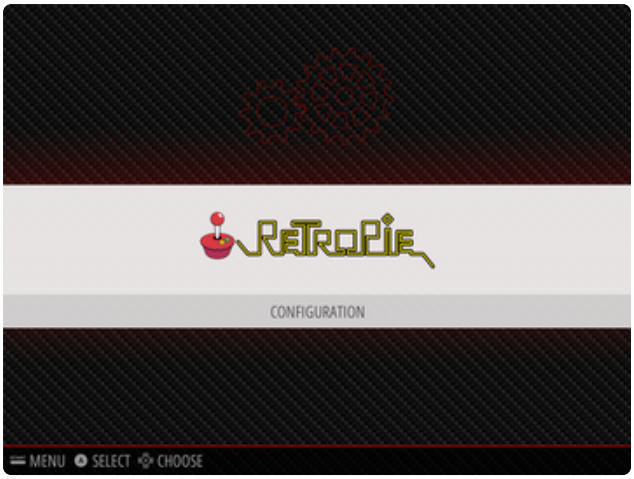
On **single-core** Raspberry Pi boards like the **Model A+** and **Pi Zero**, emulation speed is sometimes less than stellar...the screen may update like molasses, or sound may be choppy. Let's look at some ideas to fight back...

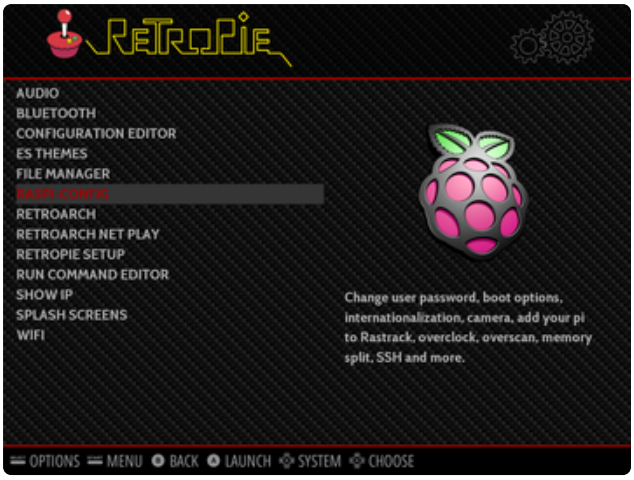
All of these methods require a USB keyboard attached.

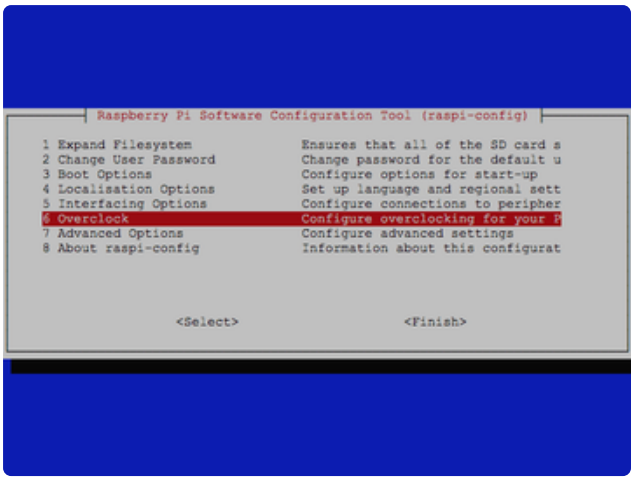
The directions shown here are for **RetroPie 4.1**, but the same general ideas apply to other versions and possibly even other emulation packages, the steps will just be different.

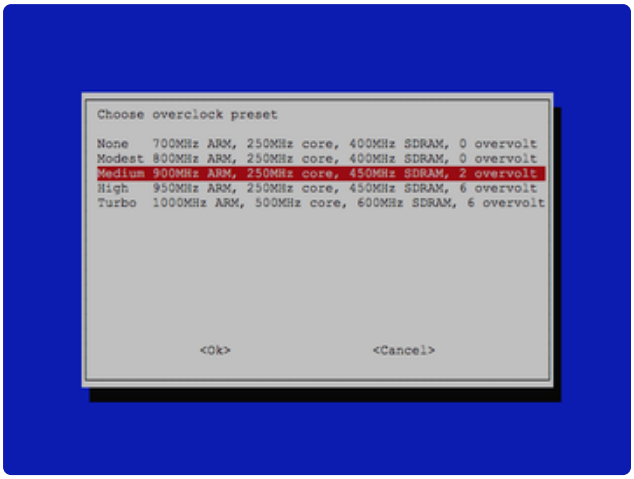
Overclocking

Most Raspberry Pi boards can handle some overclocking — running the processor slightly faster than its "official" speed. Too fast though, and the system can become unstable and crash. Due to manufacturing variances, every single board is slightly different in this regard...you'll need to do some experimentation to find the fastest reliable setting for your specific board. Restart after making any changes and **try running some games for several minutes** to really put the system through its paces.









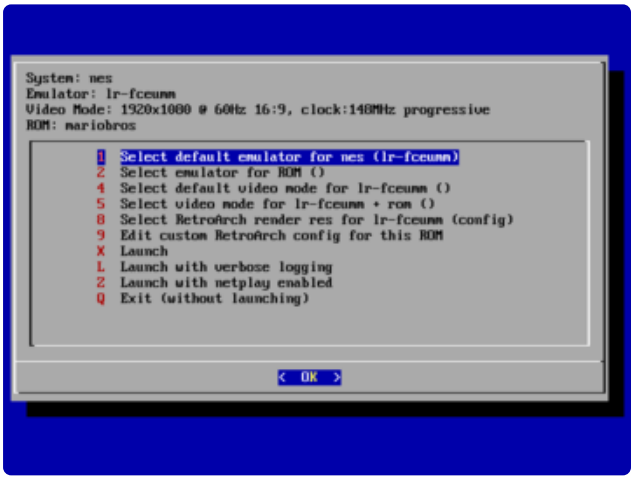
You can access the **raspi-config** utility from the RetroPie configuration screen, or press “F4” to exit to a command line and run “sudo raspi-config” manually.

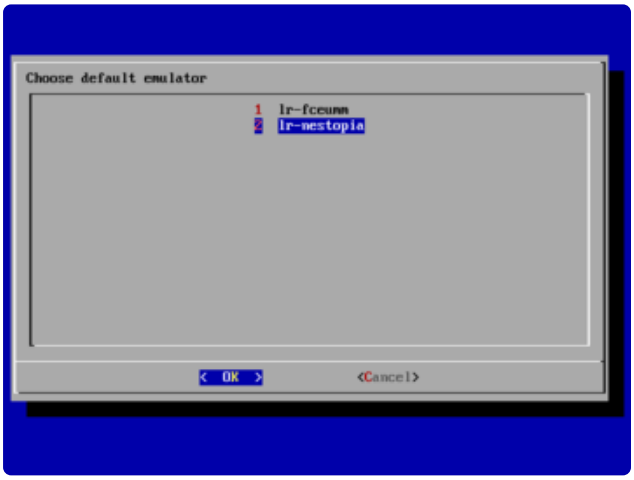
raspi-config is keyboard-based, mostly using the **arrows** and **enter** keys; arcade controls, if connected, probably won’t have the intended effect.

Different Pi models offer different overlock settings. On the Model B+ being tested here, the “Medium” setting seems quite reliable. Other board types may overclock automatically and there’s nothing to adjust here.

If you're using a **PiTFT display**, the "core" frequency setting should **not exceed 300 MHz**; the display will glitch and may not work at all. So avoid the higher overclock settings if using one of these screens, or manually edit the file /boot/config.txt to fine-tune various system frequencies individually.

Choosing an Alternate Emulator





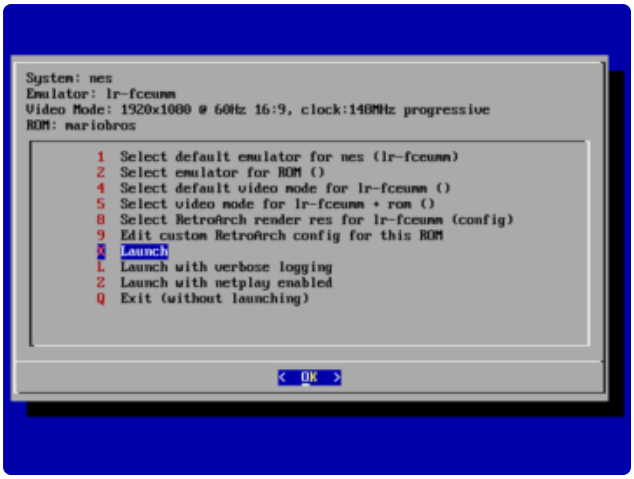
When launching a game, there's a brief moment when it's possible to access some launch options...

When you see this "launching" screen, press the space bar (or any other key that generates an actual keycode...not a "meta" key like shift or control).

This brings up a menu...similar to raspi-config, it operates with the keyboard, not arcade buttons.

The first option lets you select an alternate emulator program (if available). For example, RetroPie 4.1 includes two different NES emulators: lr-fceumm (the default) and lr-nestopia. Different emulators make tradeoffs with regard to performance and "accuracy" of the emulation. Try an alternative and see how it performs (use the "Launch" option).

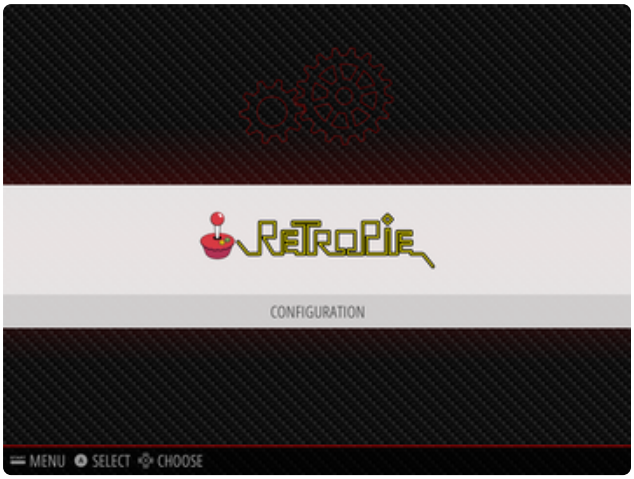
The selection you make will "stick" — it is **not necessary to use this menu every time**, unless you want to switch back to the original emulator choice. There's also an option to use an alternate emulator only for specific ROMs, if you have different needs for different games.

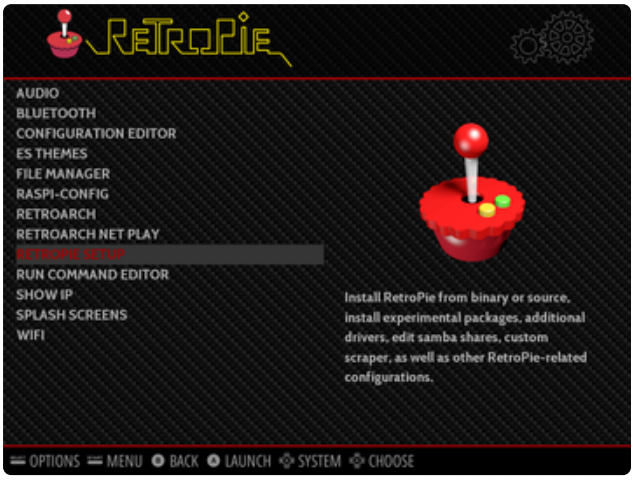


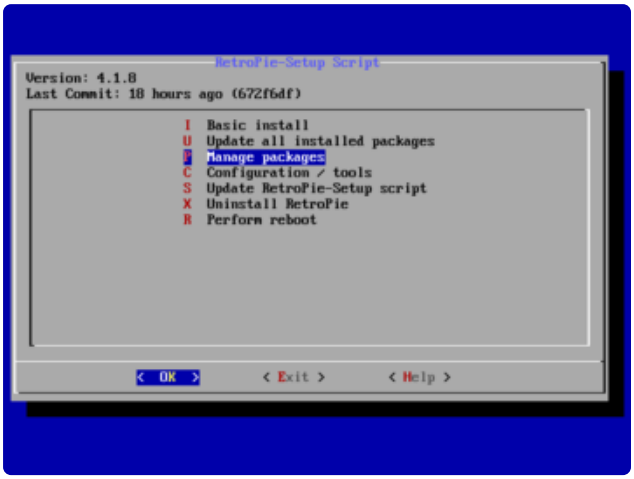
The above steps are fairly quick and easy. If they meet your needs, consider the job done! If it's still not quite the performance you need, the situation gets more involved...

These next steps require a network connection. On a Model A+ or Pi Zero, that usually requires a USB hub and an Ethernet or WiFi adapter, along with some configuration (the RetroPie 4.1 configuration menu includes WiFi setup). Alternately, if you have a spare Model B+ around, that's easy to get on an Ethernet network...move the SD card over there for setup, then move it back to the target system when done.

Installing Additional Emulators





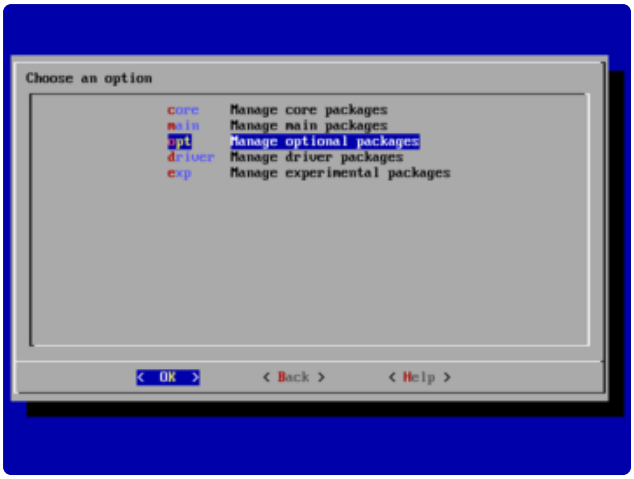


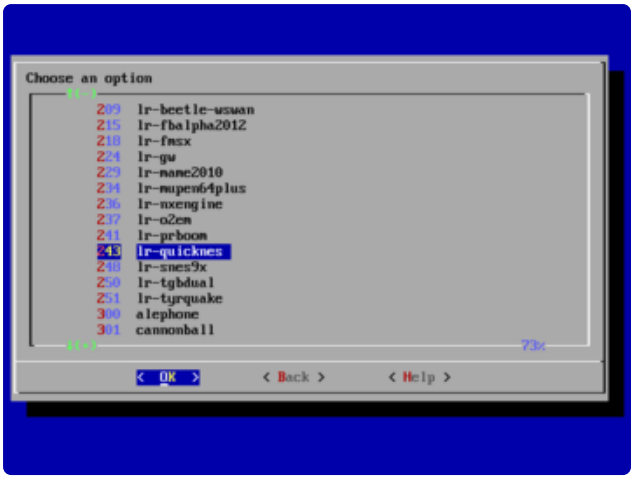
Some “bleeding edge” emulators might offer better performance than the stock options provided by RetroPie. As with the “Alternate Emulators” above, it sometimes comes with a tradeoff in accuracy (hence their non-inclusion), so you’ll need to experiment and decide if it’s worth it.

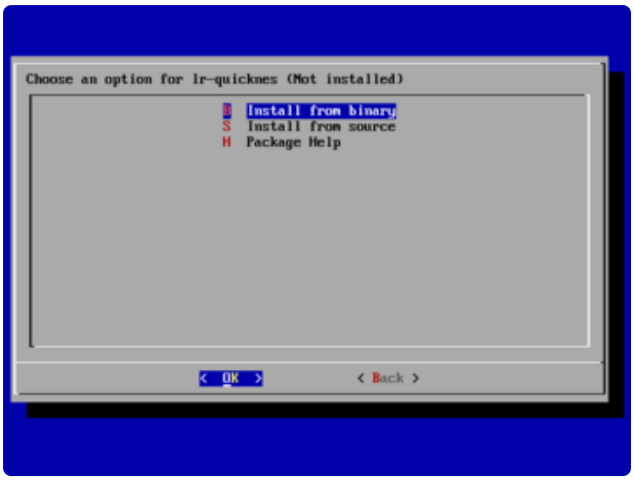
From the RetroPie configuration screen, select “RetroPie Setup,” which switches to another of these keyboard-based menus.

Select “Manage packages,” then either “Manage optional packages” or “Manage experimental packages.” You’ll get a list of various emulators and/or games that aren’t normally installed by default. lr-quicknes was an optional package for NES emulation...it’s faster than the others...too fast, in fact, so I ended up not using it...but that’s all part of the experimentation process.

Choose the “**Install from binary**” option to install the selected package. Don’t bother with the “source” option...it’s very time consuming and usually provides no benefit.

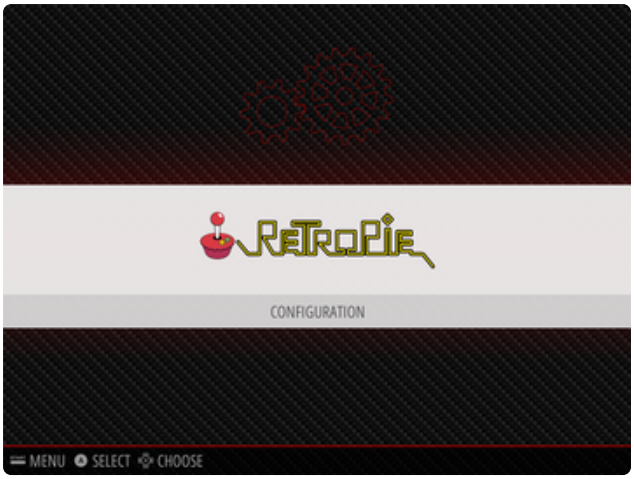


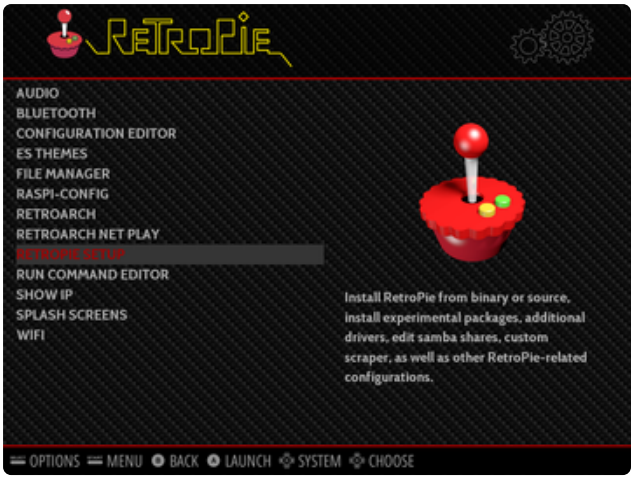


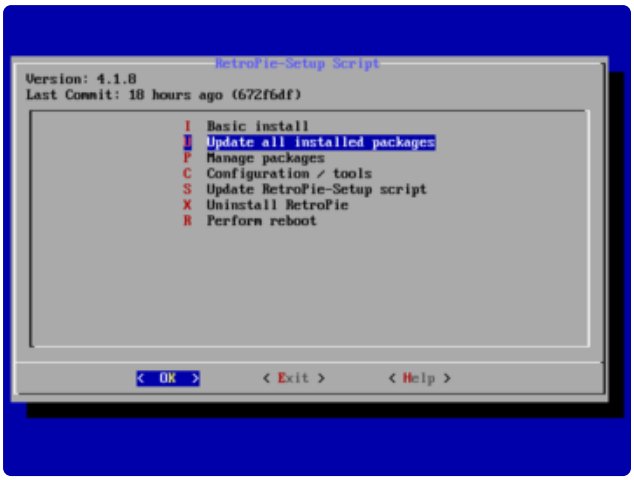


Updating Emulators to Latest Versions

Sometimes you just need the newest-and-shiniest version of an installed emulator to provide the best available performance...







From the RetroPie configuration screen, select “RetroPie Setup.” This brings up the keyboard-driven menu again. Choose the “Update all installed packages” option to make a sweep through the system that’ll update everything to the latest release, including the kernel.

This system-wide update can be extremely time-consuming…possibly an hour or so, depending on processor and network speeds.

Even the default lr-fceumm on a non-overclocked Pi ran great after updating!

If using a PiTFT display, think twice before trying this option. Perhaps make a **backup** of your SD card first. Kernel updates have been known to break PiTFT support in some circumstances, and you’d have to start all over.

Troubleshooting RetroPie and retrogame

This page documents the most common pitfalls encountered when making retro gaming projects, offering some solutions and where to turn for further help.

There are some things we can fix and some we can’t…we’re not involved in RetroPie’s development, for example…but we’ll try to point you in the right direction.

Most of the following troubleshooting steps will require a **USB keyboard attached**. Some require a **network connection**.

General Troubleshooting

There's a lightning bolt icon in the upper right part of the screen!

- That icon means the Raspberry Pi isn't receiving adequate power; most likely, you need a higher current power supply. Very occasionally this is caused by a flimsy USB cable with thin wires...try swapping out for something heftier.

retrogame Related Troubleshooting

retrogame is our software that converts GPIO button actions into keyboard events. **These are the problems we're best equipped to fix.**

Some common things to check for any retrogame installation:

- Confirm button/joystick wires go to the correct pins and to ground. A multimeter with a continuity beep function is helpful for testing.
- The retrogame configuration file (**/boot/retrogame.cfg**) uses Broadcom pin numbers...these are **not** sequential along the GPIO header pins. This site has a nice reference chart (https://adafru.it/uFH) (use the "BCM" numbers).
- Earlier versions of retrogame didn't use a configuration file...you had to edit and compile the source code. That's just horrible. If you're running an early version like that, this would be a good time to upgrade. See the Installing Retrogame (https://adafru.it/sct) page.

Some of my buttons/controls aren't working!

- Check connections and configuration file pin numbers as explained above. (If some controls are responding, that's an encouraging start...it at least means the code is running.)
- The key codes generated by retrogame might not be assigned to EmulationStation's keyboard inputs; one or the other will need to be changed. Either edit /boot/retrogame.cfg, or, from the EmulationStation main screen, press Start to access the main menu, then select "Configure Input" and proceed through each of the controls.

NONE of my buttons/controls are working!

- Confirm that retrogame is actually running...either exit to the command line (F4) or log in using ssh, then use “ps -ef | grep retrogame” to check. If you used our installer script or one of our ready-made SD card images, it should be started automatically on boot (added to /etc/rc.local).
- Confirm that the file “/etc/udev/rules.d/10-retrogame.rules” exists. Our installer script creates this file, but if you installed retrogame manually or from source, it may have been overlooked.

My controls only work if there’s also a USB keyboard plugged in!

Confirm that the file “/etc/udev/rules.d/10-retrogame.rules” exists. Our installer script creates this file, but if you installed retrogame manually or from source, it may have been overlooked.

Retrogame doesn’t work with my optical buttons!

Unfortunately, yes. retrogame only works with “passive” switches between a GPIO pin and ground (logic low=pressed). It won’t work with switches that have the opposite logic level (high=pressed).

I ran Adafruit’s retrogame installer script and rebooted, and now the keyboard and network are unresponsive!

This can happen if you’re running an early Raspberry Pi (Model A or B) with the 26-pin GPIO header and select the “Six buttons + joystick” option in the retrogame installer. That particular configuration is set up for our Arcade Pack and newer (40 pin) Raspberry Pi boards. Some of the pin numbers referenced don’t exist on the older 26-pin header and lead to trouble.

If this happens to you, not to worry. Power off the Pi and insert the SD card in a reader on a PC or Mac. Look for a file called retrogame.cfg...edit this file and change the pin numbers to match your specific controller wiring.

RetroPie Related Troubleshooting

RetroPie provides the actual emulation software and a nice user interface. As this is third-party code, the “depth” of problems we can troubleshoot is more limited, but here are some of the common issues we’ve seen and how to resolve them...

My controls work in the EmulationStation UI, but not in one or more specific emulators!

This can happen with certain emulators (usually older or more esoteric ones) that don't use the libretro library. Among other things, libretro allows the global controller configuration to be used everywhere. There are a couple of workarounds that might help, but no guarantees...

- You can rummage around in /opt/retropie/configs and look for a configuration file specific to the problem emulator, then edit its keyboard layout to match your controls. The format of this file, if one even exists, is likely specific to that one emulator, so you'll need to do some research (Google search, etc.), it's not something we can help out with.
- You can try hunting for an alternate emulator based on libretro, if there's one available (see "Installing RetroPie Packages" below).

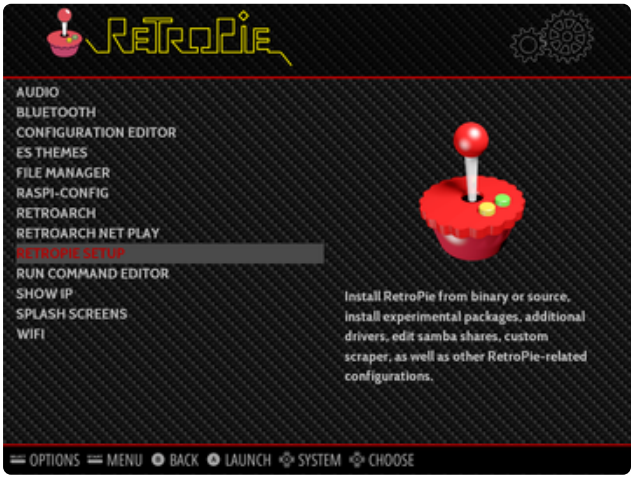
The ROM files I have worked in a different emulator before, but aren't working in RetroPie!

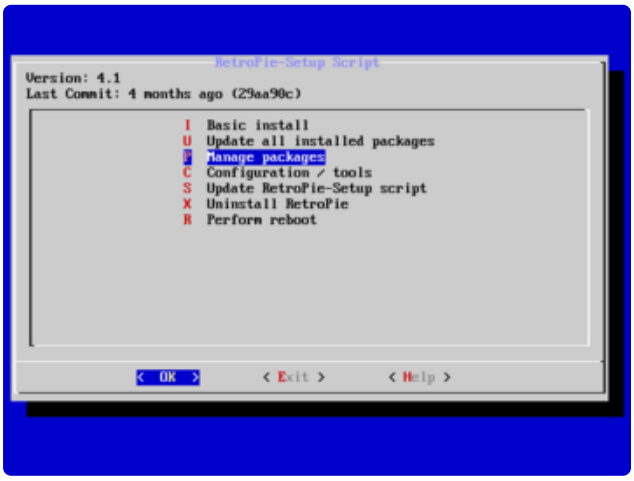
This can happen if the ROM file format changes between versions of an emulator, or if two emulators for the same system use different formats.

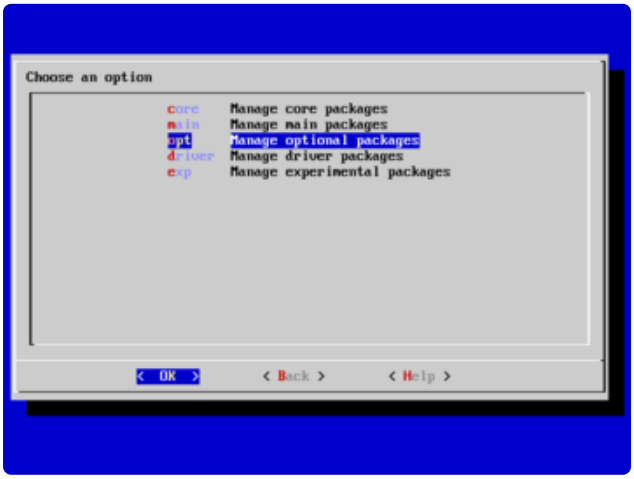
- Do some research (Google search, etc.) to see if this is the case. It's possible there may be utilities to convert among different ROM formats.
- Try installing an alternate emulator, if there's one available (see "Installing RetroPie Packages" below).

A few emulators may require a "BIOS file" in order to function, but it's not included with the software for legal reasons. This is something you'll have to research and track down.

Installing RetroPie Packages







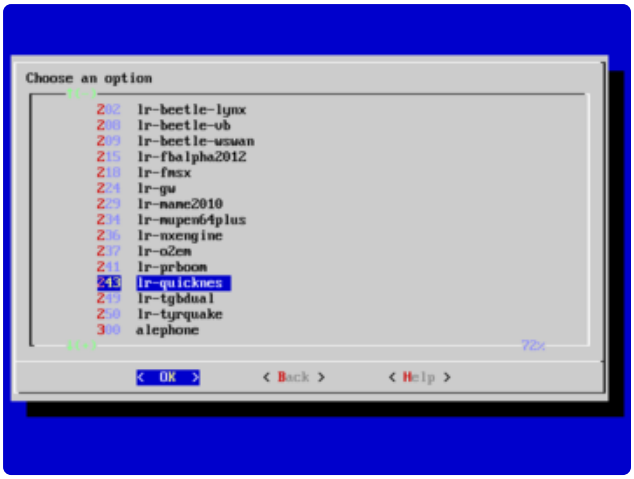
To add support for a system not present in RetroPie by default, or to add an alternate emulator program for an existing system, select “RetroPie Setup” from the RetroPie menu. **This brings up a text-menu-based interface and will require a USB keyboard to navigate.**

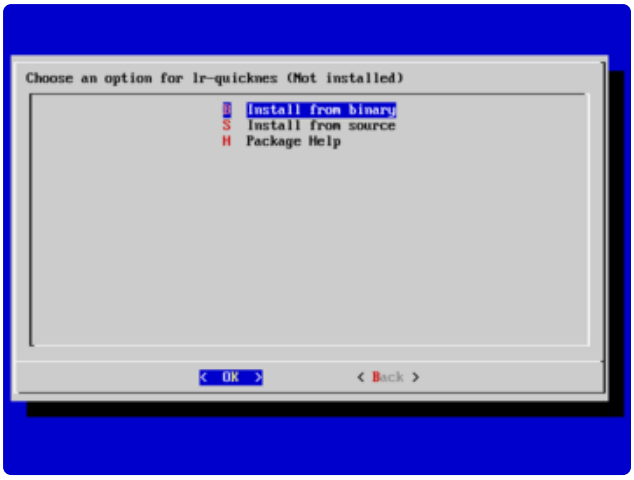
Select “Manage packages” and then one of the core, main, optional or experimental selections...you’ll probably want to navigate through each of them to see what’s available, keeping in mind that each successive category might be a little rougher around the edges.

Your best bet are packages whose names begin with “lr-”. This means they’re built using **libretro** and the control inputs should already work with what you have! Other packages may require their own manual controller setup, which can be a real nuisance.

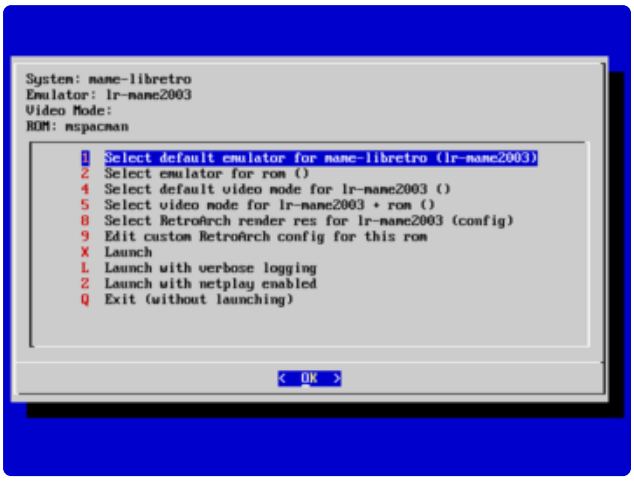
When asked, select “**Install from binary**.” The source option takes much longer and won’t provide any benefit for the average user...only attempt that if you know you need absolutely bleeding-edge code.

It’s totally valid to install multiple emulator packages that handle the same type of system. Each might have different performance or compatibility benefits, so it’s worth testing your options. See “Accessing Alternate Emulators” below.


Choose an option
202 lr-beetle-lynx
208 lr-beetle-vb
209 lr-beetle-wswan
215 lr-fbalpha2012
218 lr-fmsx
224 lr-gw
229 lr-mame2010
234 lr-mupen64plus
236 lr-nxengine
237 lr-o2em
241 lr-prboom
243 lr-quicknes
249 lr-tgbdual
250 lr-tyrquake
300 alephone
72x
< OK >
< Back >
< Help >


Choose an option for lr-quicknes (Not installed)
B Install from binary
S Install from source
H Package Help
< OK >
< Back >

Accessing Alternate Emulators



When you launch a game, you'll briefly see this nondescript launching message. **You have a couple of seconds to hit a button or key...**

The first option in this configuration menu lets you select a different emulator package for a given system type...or even on an individual ROM-by-ROM basis, if different games benefit from different emulation software.

Test it out with. If you don't like the results, next time you launch that game you can access the configuration menu again and restore the original selection.

More RetroPie Help

For issues not covered above, the best sources for RetroPie-specific help are the official documentation and forum on the RetroPie web site:

RetroPie Documentation **(https://adafru.it/uFI)**

RetroPie Forum **(https://adafru.it/uFJ)**