**Guide: Setting Up a RetroPie controls using IPAC2 Controller - Extensive Tutorial + Preconfigured files.**

**Introduction**

Because button configuration is still a rather complicated topic for many new coming enthusiasts, I decided to share my own experience in a specially dedicated post rather than repeating myself on different places on the forum. Maybe what I have to share will be of benefit to some people.

My intention is to describe exactly how I configured my own arcade and share with you some preconfigured files which will allow you to replicate the design I used.

In the beginning I will talk a bit about some hardware and design choices I had to take before I even started thinking about configurations. I promise it will make sense in the end.

I will also share with you the template document which I made to keep track on the button positions, while I was configuring the controls for the different platforms. I hope it will be of use to people which plan to build an arcade with a different button scheme and can not simply copy the preconfigured files.

If you want to download all of the files shared inside this post, just go straight to the end of it and you will find a list.

I want you to keep in mind that even if you have the same button panel and you decide to follow me step by step and use exactly the same set of emulators I picked up, there is no guarantee that everything will work straight away. Mainly when it comes to individual per ROMS control configurations or specific ROMS versions chosen. Try to delve more into the official wiki pages as they are surely be more informal.

**Step 1 : Button Layout**

Before ordering all of the hardware I had to take decision which platforms I would use on the arcade cabinet, how many buttons I would need and how would I arrange them.

Out of all emulators available inside RetroPie I chose the following 8”

● Atari 2600

● FBA

● Game Boy Advance

● Mame Libretro

● Megadrive

● NES

● Playstation

● SNES

The RetroPie based arcades I have seen rarely use more than 6 action buttons per player + 2 for start and select (8 in total for each player). This would have been sufficient enough for all game platforms I planned to use except for Playstation. To be able to map all of the buttons properly I needed 2 more buttons for each player. In total 10 buttons per player.

I have also seen some people doing additional customization to their arcades by adding special buttons for starting and exiting a game. This would require additional programming work so I decided to rely on the default settings as much as I can. I didn’t want to complicate things further.

Now that I knew I need a 20 button set I started looking for a good button layout which I can use and I picked this one:

IMAGE

(You can find other button layouts available here : <http://www.slagcoin.com/joystick/layout.html> )

Before actually buying the buttons I put a lot of thought about what color buttons I want to install on the arcade. I see on internet that a lot of people are using randomly colored buttons, which I find very impractical. I wanted the mapping to be intuitive and easy to understand so I specifically decided to order buttons colored in the Super Nintendo color scheme. My reasoning is this:

A) Most people would immediately associate the button position and function as this scheme is a classic on its own.

B) It would be easier to communicate the button functions to new players in a manual.

**Step 2 : The hardware I bought**

The controller set I ended up buying included pretty much most of what I needed:

<https://www.arcadeworlduk.com/products/Arcade-Joysticks-Buttons-And-Wiring-Kit-No-2.html>

The only difference is that I contacted the seller and specifically asked for these button colors: 2x Yellow, 2x Blue,2x Red, 2x Green, 2x White, 8x Black, 2x White Player start buttons, 2 x Balltop joysticks (Red)

The rest of the hardware should be familiar to you all: Raspberry Pi 3, Power supply , SD Card, Monitor , Speakers and one peculiarity : USB Dual Extension cable (<https://goo.gl/9PvyJR> )

I will briefly tell you what was this about as I have not seen anyone else use it. This thing is totally optional, you don’t need to make it if you don't want to.

When the Arcade is finished you will end up using only one of the USB port of the PI. When you close the cabinet you won't be able to access it easily. Not without an extension cable. I made a discrete hole on the lower side of the control panel and installed the USB ports there. Through them I can add more roms to the SD card and plug in two additional xbox controllers for some sweet 4 player arcade gaming.

**Step 3 : Deciding on the button usage for each game platform**

Now that I knew how many buttons would I use and which consoles I'm going to emulate I started planning the mapping of the buttons for each console. It is crucial to make these decisions in the beginning and write them down in an easy to understand way if you want to keep track on what is going on while you are editing the config files for each console later on. If you don't do it you can easily get lost in the process.

First of all I wrote down which buttons I plan to use for each console. Later when everything was wired I had time to do some testing and rearrange some buttons. These are the control schemes I decided to program for each emulator.

IMAGE

This is what the instruction manual for my arcade looks like. It is glued on the cabinet, just over the screen. I think it's descriptive enough.

And this is how the control panel looks like.I placed additional hint icons beside each button.

I had gone through a lot of testing later on and I can recommend this mapping to anyone.

IMAGE

As you can see the controls for the MAME and FBA emulators are not specified as each individual game has a widely different control scheme. Because of this problem I had to do a custom remapping for certain games to fix the button position to either match their original layout (Street Fighter Series) or to make them match the commonly used layout for “Hit, Jump, Special” buttons (Metal Slug series , X-men). I will describe this later on.

**Step 4 : IPAC2 Button wiring**

When the cabinet was finally built it was time to wire the button switches to the IPAC2 Controller.

I assume that most of you are already familiar with the functionality of the IPAC2, but I will briefly describe what it is and how it is tied to the RetroPie.

The IPAC2 is a board which can be programmed to act as a keyboard. Keyboard buttons are assigned to each pin on the board. The pins are wired to the switches on the button panel. When the button is pressed it makes a contact and the IPAC sends a signal through USB cable to the Raspberry Pi telling him which keyboard key has been pressed. The RetroPie emulator then “looks” inside the configuration file (.cfg) “sees” which RetroArch control is assigned to that button and executes it.

If you are trying to replicate my button configuration it is important to wire the pins to the right buttons. This is how my buttons are wired to the IPAC2 board:

IMAGE

I advise you to look for more information about how exactly the cables connect to the switches here <https://www.ultimarc.com/ipac2.html> or in some youtube video tutorial.

**Step 5 : IPAC2 Configuration**

Now that the IPAC2 is connected to the buttons it is time to configure it.

To do that you will have to use a software called WinIpac

<https://www.ultimarc.com/download.html>

Through it you can assign any keyboard key to a pin.

Things are pretty straight forward. You have to connect the IPAC2 board to your PC through USB Cable. When you start the software it will recognize the board and will display it on the left. You can select pins on the image and choose a key which you want to assign to it. It is important to set your IPAC2 to work as KEY, not as a GAMEPAD.

When you are ready you press File -> Force Board Reconfigure

If you want to replicate my configuration and make it work you will have to assign specific keyboard keys to the IPAC2 pins. Fortunately WinIPAC can export .xml configuration files, which can than be imported to other boards, thus saving you all the manual work. Here is mine configuration file:

FILE

What you have to do after you download the file is to start WinIPAC, open the File -> Import menu and browse for the file. Open it. Than click on File -> Force Board Reconfigure. It is done, your board is now configured the same way as mine.

On this image you can see which Keyboard keys are now assigned to the IPAC board and respectively to the buttons.

IMAGE

Just to wrap it up, here is another image showing you all the keyboard keys used.

IMAGE

SIDE NOTE:

IPAC2 can be switched to behave as a GAMEPAD controller. I have seen many discussions about how it would be easier to make everything work if you switch to that mode. The problem as I found out is this. If you switch the IPAC2 to act as a GAMEPAD RetroArch would recognize it as a single controller. You can not map two player on it. You would technically need another IPAC2 board to do that. So don’t waste your time and forget about it.

SIDE NOTE:

I have to clarify that in the very beginning I started working on the working on this I

Stumbled on some problems with the mapping. For some reason I eventually changed the keys used for Start and Select with different keys from the keyboard. I don’t recall what exactly was the problem for me, but in the end everything worked out just fine.

**Step 6 : Planning the Button Mapping**

This is where things get really complicated for the average user.

You might already know that Retropie can use different button mapping for each emulator.

I don’t want to get too much in detail especiall because there is a very detailed wiki page about this : <https://github.com/RetroPie/RetroPie-Setup/wiki/RetroArch-Configuration>

(Read the “Hardcoded Configurations” section)

In a nutshell Retropie has 3 levels of button configurations which can be overwritten when you start a certain emulator or a certain game. The Config Hierarchy goes like this:

● Global Settings

● System Specific

● ROM Specific

While you are still inside the main menu the Global Settings apply. When you start an emulator which does not have System Specific settings written for it the Global ones keep on working. If you have a custom configuration file written for a specific system and you start the emulator, the Global Settings are overwritten and the System Specific Settings apply. If you exit the emulator the Global Settings start working again. If you start a specific ROM which has its own custom mapping both previous settings are overwritten.

This is what makes things so complicated and hard to track.

For this reason I made myself a visual map of the button layout of each individual console and specific ROMS. The previous two images I showed you are exactly that in an early stage. sIt is nothing that special really. The template is made in Adobe Illustrator. It contains a set of images, showing the button layout of my arcade and inside each button there are tree values written. These values tell you the name of the switches, the Keyboard Keys assigned to each switch and the Retroarch key values tied to the Keyboard input.

I have made a separate copy of this layout for each emulator.

Here is an example. This is the default retroarch configuration map.

These controls will be active in the Main Menu of the RetroPie. They will also be automatically assigned to each new emulator you decide to use beside the 8 I’m using.

IMAGE

Here is short description of what you should keep attention to in this image:

● In the upper left corner there is a title telling you for which Game Platform these controls Apply.

● The text in BLACK tells you how are the button switches wired to the IPAC2 board. Once set up these values never change.

● The text in BROWN tells you which Keyboard Keys has been assigned to each pin on the IPAC2 board. Once set up these values never change.

● The text in BLUE tells you which RetroArch key values are tied to these Keyboard Keys. These values can be rearranged inside the different .cfg files! By making copies of the default .cfg file and rearranging the values in BLUE you are switching the buttons functions for each emulator!!!!

● Button Labels section is basically a legend. Take notice of the blue text though, as it will tell you in which directory you should copy the edited .cfg file!

● Hotkeys Enabled section. RetroPie has a set of shortcuts for hidden functions like saving/lading a game and others. These functions can be disabled for specific gaming platforms or changed in such a way to keep being constant even if you have rearranged their position while editing some of the other buttons.

Finally, here is an archive containing my original Illustrator File + Image exports of each control scheme I mapped.

FILE

Download it and look through all of the images and try to compare them. Also don’t forget to look at the image I shared in Step 3. It will show you the same button arrangement , but without the additional information.

In the next step I will give you a quick recap of how you can manually change the configuration files for each game platform and will than share the files I’m using.

SIDE NOTE

As you can see in the archive I myself ended up with 12 different maps for my Arcade.

● 1 Global Mapping - Functioning in the main menu of the retropie.

● 8 System Specific Mappings

● 3 ROM Specific Mappings (Metal Slug series, Street Fighter series, Golden Axe, X-men)

In total I have made 28 configuration files. (.cfg)

The number of configuration files is higher than the mapping schemes , because certain game series are using a single mapping scheme, but the actual configuration file had to be duplicated several times for each individual ROM . The name of the .cfg file had to be changed to reflect the name of the ROM it should be tied to. For example all Street Fighter Games, including the Marvel fighting games (11 in total) are using the same ROM Specific mapping, but each ROM has its own copy of this very same file.

**7. RetroArch Configuration**

Now the real work begins.

Configuring the controls through RetroPie GUI system is not efficient enough, so I’m going to talk only about the manual editing of the files. I will give you an example how you can use the Mapping template to help yourself.

First you need to access the files system of the RetroPie and copy the configuration files for the systems you plan to use. It would be easier for you to edit the files on your machine rather than directly on the card. This way you can safely keep a backup copies each time you are make changes.

You can access the files structure in two ways:

● If you have a PC with a Linux OS installed on it, you can access the files structure through it.

● If you have a Windows PC you can access it through your home network. Connect the Raspberry Pi to the router. It should appear in your Network

Browse to this directory : /opt/retropie/configs/

It contains an individual folder for each emulator and one for the Global Settings. The folder name for the Global Settings is called “All”

Copy this folder along with the folders of the emulators you plan to use.

In my case I copied 8 additional folders for the emulators I listed in Step 1.

(I) GLobal Settings Configuration

First we will configure the Global Settings file according to the Controller Map I showed you before.

Here it is again:

IMAGE

Open this file with a text editor: /opt/retropie/configs/all/retroarch.cfg

You will find that there is a ton of text in it. These are all instruction which can help you understand how to configure the file in basically any way possible. But we will concentrate only on a certain part of it all.

Search for these lines of code:

---

input\_player1\_a = "x"

input\_player1\_b = "z"

input\_player1\_y = "a"

input\_player1\_x = "s"

input\_player1\_start = "enter"

input\_player1\_select = "rshift"

input\_player1\_l = "q"

input\_player1\_r = "w"

input\_player1\_left = "left"

input\_player1\_right = "right"

input\_player1\_up = "up"

input\_player1\_down = "down"

---

These are the instructions which bind the RetroArch keys to specific Keyboard Keys.

This line in specific:

input\_player1\_a = "x"

What this line is telling you is that the input for RetroArch button “A” for Player 1 is a Keyboard key “X”. What you have to do is to look at the Map image above and edit the value inside the brackets to match the Keyboard Key written above the RetroArch key for value “A”.

So in our case this same line will turn into:

input\_player1\_a = "ctrl"

IMAGE

THIS IS IMPORTANT

Before continuing forward you have to know that I made some tweeks to my files. For starters the default state of the config file will include mappings for only one player. This will change after your first configuration through the GUI system.

The default file will also be missing the Hotkeys configurations.

For those who are wondering what the Hotkeys are it is basically this:

Hot keys a re button combination which initiate hidden functions. Save/Load game, Exit Game and etc. There is always one specific Hot Key which pressed in combination with other actives a certain function.

To cut it short.

For my files I included all possible lines of codes applying to player controls and rearranged them in more convenient way. I also moved these lines of code to the very top of the cfg file.

This is how the Global Controls values are set inside my files:

---

input\_player1\_select = "y"

input\_player1\_start = "h"

input\_player1\_left = "left"

input\_player1\_right = "right"

input\_player1\_up = "up"

input\_player1\_down = "down"

input\_player1\_a = "ctrl"

input\_player1\_b = "alt"

input\_player1\_x = "space"

input\_player1\_y = "shift"

input\_player1\_l = "z"

input\_player1\_r = "x"

input\_player1\_l2 = "c"

input\_player1\_r2 = "v"

# input\_player1\_l3 =

# input\_player1\_r3 =

input\_player2\_select = "u"

input\_player2\_start = "j"

input\_player2\_left = "d"

input\_player2\_right = "g"

input\_player2\_up = "r"

input\_player2\_down = "f"

input\_player2\_a = "a"

input\_player2\_b = "s"

input\_player2\_x = "w"

input\_player2\_y = "q"

input\_player2\_l = "i"

input\_player2\_r = "k"

input\_player2\_l2 = "o"

input\_player2\_r2 = "l"

# input\_player2\_l3 =

# input\_player2\_r3 =

#hotkeys

input\_enable\_hotkey = "y"

input\_exit\_emulator = "h"

input\_save\_state = "x"

input\_load\_state = "z"

input\_state\_slot\_increase = "right"

input\_state\_slot\_decrease = "left"

input\_menu\_toggle = "nul"

input\_reset = "nul"

---

Take some time and compare them to the Map image above.

Look at the HotKeys Enabled section on the image and you will understand what value should correspond to each hotkey.

Take notice that on some of the hotkeys a value “nul” is written.

This value completely disables the button function!

Now that I have organized the lines of code in what I think is a more convenient way what I did next is to copy and paste them in this arrangement inside each of the System Specific configurations files.

SIDE NOTE

Be careful! Beside the keyboard configuration inside the config files you will find lines like this one:

input\_player1\_a\_btn =

The “btn” at the end of the line is indicating that this line of code is expecting GAMEPAD button input! If you try using it by mistake RetroArch will not detect your IPAC input, because it will expect a Game Pad button press.

(II) System Specific Configuration

I want to give you one example of System Specific Configuration, which will demonstrate for what reason and how are the Retroarch Config Values rearranged on the button panel.

For that reason I will use the Game Boy Advance mapping as an example.

Navigate to this file and open it: /opt/retropie/configs/gba/retroarch.cfg

You will see this written in it:

------

# Settings made here will only override settings in the global retroarch.cfg if placed above the #include line

input\_remapping\_directory = "/opt/retropie/configs/gba/"

#include "/opt/retropie/configs/all/retroarch.cfg"

-------

First take notice of the input remapping directory path.

For each system it is different and reflect the file location.

Second take notice that because there are no Player Input commands written above the “#include "/opt/retropie/configs/all/retroarch.cfg"” line the controls assigned to this emulator come from the Global Configuration file.

To change them you will first have to paste the Player Input values from the Global Configuration file and edit them.

But before that… why edit them in the first place?

Look again at the image from Step 3. Than look at the image from Step 6.

If I use these controls when I play Game Boy Advance game I will have to play with the RED and The YELLOW buttons for A and B respectively and the top two BLACK buttons for L and R. I would prefer to assign the L and R buttons to the GREEN and BLUE buttons.

I also want to disable all buttons I’m not suing. To “nul” them.

These will be L2 and R2, but not X and Y as when they switch position with L and R they will keep acting as HotKey buttons evoking Save/Load functions.

Look at this image:

IMAGE

The change is already reflected.

Take a look at the code:

IMAGE

Simple enough.

These are really the basics.

When you are done with the file editing copy the new files over the ones in the original directory on the SD card.

Before sharing my already preconfigured files I want to drop several lines about the ROM Specific configurations.

(IV) ROM Specific Configuration

ROM configurations work the same way.

The difference is that you will have to create your own files for each ROM . And place it in the directory where the ROM is located. It is important that both files have the same name!

The only systems for which you will find yourself doing custom Per ROM configurations are MAME and FBA basically.

If you want to create a custom Per ROM configuration for Street Fighter 3 for FBA emulator you will have to create the cfg file inside this directory:

"/opt/retropie/roms/fba/"

If the name of your ROM is “sfa3.zip”

The name of the configuration file would be: “sfa3.zip.cfg”

When you create a configuration file for “Street Fighter 3” you can duplicate it several times for all your Street Fighter ROMS. They are using the same controls after all. The only thing you will have to edit is the name of the file. It must reflect the name of the ROM !

Doing a custom configurations for ROMS is a matter of testing and fixing.

You would have to put some time and effort to clear things out.

And now finally.

Here is a zip archive containing all my configuration files.

Examine them, use them if you like.

FILE

**Conclusion**

As I was writing all this I came to realize what a complete nightmare this whole experience has been. But please don’t get discouraged it is worth it. Just try to be focused and calm and everything will eventually be resolved.

I want to express my respect to all the people involved in this project. I have no idea how are you even managing all of this.

I want to also say that the mapping template is not my idea at all. I stumbled on a Reddit post which explained things quite well.

.

<https://www.reddit.com/r/RetroPie/comments/4a8ncv/ipac2_config_on_retropie_works_with_mame_nes_snes/>

I found this to be such an ingenious way to visually represent the whole mess that I decided to replicate and expand the idea.

SIDE NOTE

I think now that someone with a good programming skills could possibly be able to create a visual .cfg file generator based on this concept. I can imagine that in such software you would at first pick your button layout. Than you will have to enter the Keyboard Keys assigned to the buttons and in a third step pick an emulator and select the buttons you want to use. Such a software would then generate the .cfg file by substituting the Keyboard Values. Sadly I’m not a programmer, but a graphical designer. Leaving you with this thought.

**List of all files included in this post:**

● IPAC2 Preconfigured .xml

● RetroArch Configuration Map Template

● RETROPIE Controller Configurations