RWTH Aachen University
Media Computing Group

Building the FabArcade

Anke Brocker
anke.brocker@rwth-aachen.de

Florian Busch
florian.busch@rwth-aachen.de

Contents

Abstract

Do you want to really experience playing vintage 80's video games? Got a FabLab in your neighbourhood? Ready to build something real? Then make a FabArcade! The FabArcade is the first fully open-source video arcade gaming station that can be built completely at any FabLab using digital tools.
It's a fully functioning video arcade that lets you relive those cool retro video games from the 80's. Finally play Galaga the way it was meant to be played, with a professional joystick and arcade buttons - even including a real coin slot for US quarters (or whichever currency you prefer) Plus, the FabArcade also runs additional games we developed in Java, which can guide you if you are looking for developing for this stunning device yourself.

Chapter 1

Building the FabArcade

To build your own FabArcade you can follow this guide. We will show you how to prepare the hardware, the woodwork and also the software. We will guide you through this process in small steps, such that you can understand and customize the way to build not a FabArcade but your own and unique one!

1.1 Goals and limitations

The goal was to create a visually attractive and structurally stable arcade entertainment system, taking advantage of the technological progress made ever since the golden age of arcade video games in the 70s and 80s without being unfaithful to the principles on which the original arcade machines were based. In other words, we respected and extrapolated the general structure into our own design, but chose a different geometrical form and visual concept, both more adequate to the current technological stand, to bring it to life.

Each FabArcade must be completely reproducible in a FabLab, which means our limitations were none other than those imposed by the Lab itself, i.e. mainly those of the laser cutter. In our case, this had a strong impact on the

materials we were able to use and its dimensions as well as the main construction principle.

1.1.1 Laser cutter

The laser cutter limitations defined the dimensions of the basic module which would compose the FabArcade – a 5 mm thick, 600 mm wide and 300 mm high wooden plate. The size of the material to our disposal automatically made us think of a division of the FabArcade into smaller pieces which would be cut separately and, when combined, would result in a fully functional, stable arcade machine.

Chapter 2

Setting up the Hardware

First, we will prepare the hardware. The hardware part covers the woodwork, but also the electrical wiring. For us hardware is not only the basis of the software but also the basis of the parts which holds the hardware the software runs on. So you can really say there are only two parts - Software (the programs running on the Raspberry Pi) and Hardware, which covers all that is not software.
Last but not least you will get to know how everything fits together and where to embed which piece into the FabArcade.

2.1 Design process

2.1.1 The form

We decided to give the classical arcade machine design a twist without losing sight of its roots. We analyzed the structure of the original arcade and split its foundations into two groups: the anatomical and the technological arguments.

Of course, the measures and design decisions connected with human anatomy remained intact – the height of the controller, the dimensions of the machine itself, the position and angle of the screen, etc..

The technological limitations, on the other hand, are now different than they were in the 70s and 80s, and we decided to develop a geometry that took advantage of the current technological stand regarding audiovisuals – mainly the dimensions of the screen. A monitor today is flat enough that a depth of 500 – 600 mm throughout the whole height was unnecessary and would effectively turn the FabArcade into an empty box. Nevertheless, the challenge was to find a form that, although slim, would be structurally stable enough to withstand the rage of a frustrated player, shaking the arcade back and forth and slamming their angry fist against its wooden panels.

We decided that a 5 mm thick wood layer was not only too thin to be really stable, but was also not heavy enough. No matter how much stability the geometry would provide, if the FabArcade turned out to be too light it could be drifted off or even lifted or tilted, causing potential damage to the structure or the electronics. Thus we agreed upon using a triple wooden layer with an adequate 15 mm thickness, which turned out to be thick and heavy enough.

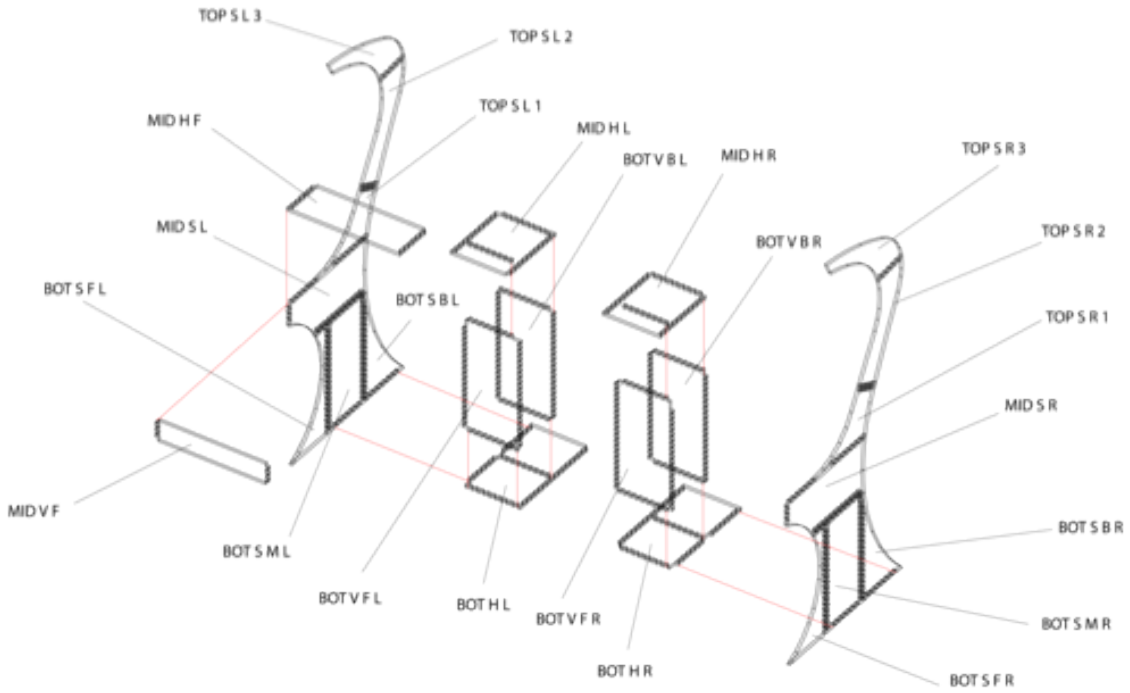

Figure 2.1: You can download this assembling overview to see more details if the above picture is too small.

2.2 The labelling of the pieces

Every piece has a unique name that describes its position. It is structured as follows:

BOT / MID / TOP: Whether the piece belongs to the lower (BOT), middle (MID) or upper (TOP) part.

S: A piece name contains an S if it's a side piece (Its orientation is parallel to the player's sight)

H / V: Whether it's a horizontal (H) or vertical (V) piece.

F / B: Whether the piece is in the front (F) or in the back (B) from the player's perspective.

L / R: Whether the piece is to the left (L) or to the right (R) from the player's perspective.

2.3 Instructions

Now it is time to start building your very own FabArcade. Start to cut all the pieces with the lasercutter. Make sure while cutting that all pieces of the same name have the same proportions such that they fit together later on.

HINT!: After finishing the FabArcade we realized that placing the coin slot and the door through which the user is able to get coins back into BOT V F R was not a perfect idea. That spot is situated quite low and hard to reach because of the gamepad controller. In case you know you have got space on a side of the arcade you should place coin slot and door there (i.e. in BOT S M R or BOT S M L). Therefore you need to change the file of the BOT V F R in the following way: Copy the two boxes which represent the holes for coin slot and door and insert them into BOT S M R or BOT S M L. Of course you also have to check if the measurements of your coin slot fit the ones in the file. If you put the arcade into a corner with no room on the sides, stick with the first possibility.

2.3.1 General Instructions

Each piece has to be 15 mm thick in order for the puzzle piece system to work. In our case, we had to cut each 5 mm plate three times and glued them together.

Keep a file at hand, as perfect precision is impossible to achieve when gluing or even cutting the pieces.

Do not start assembling the FabArcade until each piece is completely dry.

Use glue to assemble the FabArcade. The best glue might be one specialised on wooden surfaces.

Important! When cutting the pieces in which the coin slot and the door to get the coins back will be inserted, do not throw away the cutouts of the hole for the door. The best

looking one will function as the door to open access to the coins later on.

Exceptions

The gamepad is placed on the MID H F board. There are actually 2 different MID H F boards; one (MID H F 1) which effectively becomes the surface which the player sees and touches (it is the "upper" board) and another one (MID H F 2) which will come underneath and have enough space for the gamepad's body and electronics.

We cut MID H F 2 twice and MID H F 1 once in order to achieve the desired thickness of 15 mm.

Do not glue the MID H F 1 board and the MID H F 2 board(s) together until the gamepad has been mounted!

2.3.2 Assembling

1. Assemble BOT S M L and BOT V F L.

2. Assemble BOT S M R and BOT V F R.

3. Assemble BOT V B R to BOT S M R.

4. Assemble the pieces resulting between BOT V F L and BOT V F R.

5. Add BOT V B L – you've already assembled half of the lower part of the arcade's body.

6. Add BOT H L and BOT H R.

7. Add the coin slot into the right hole (Figure 2.3) which size depends on the size of the coin slot. Now add the little door to retrieve the money (left hole) into BOT V F R (or wherever you decided to place it) and place some kind of container inside of the arcade's body where the coins will fall into (Figure 2.4).

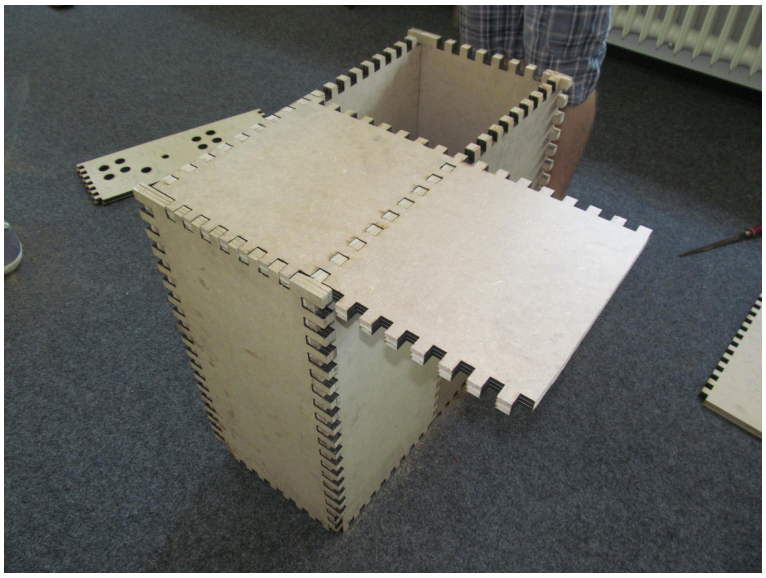

Figure 2.2: The Bottom part of the FabArcade. View from below, i.e. BOT H R is facing the top.

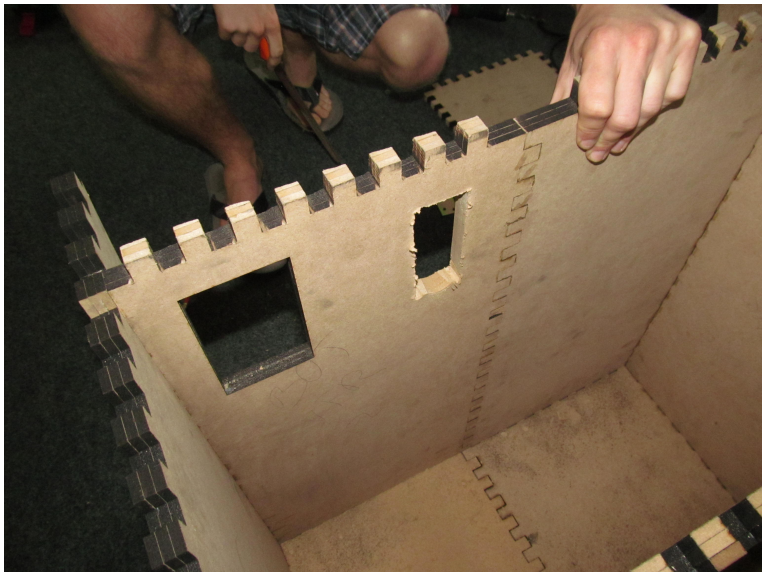

Figure 2.3: Holes for coin slot and for retrieving coins.

8. Saw or file a little hole to pass the coin slot's cables and let them out. Just in case, tape or fix the wires to the inside of the box to prevent them from being ripped off if someone accidentally yanks at them.

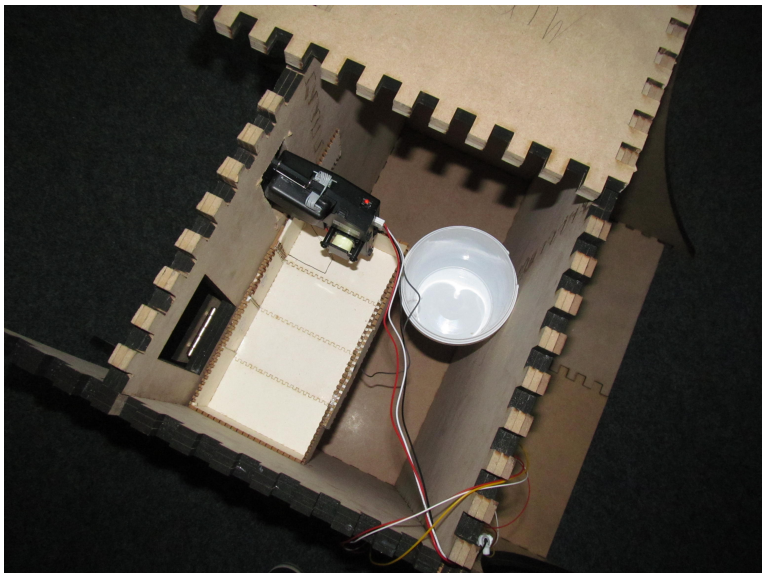

Figure 2.4: Bottom of the FabArcade from the inside. You can see the container for the money and the integrated coinslot. The wooden coloured, square box on the left is the container for the coins (white little bucket was just used while the glue dried to ensure the box does not fall down).

It is important that you check everything is correct up until this point before going on to step 9.

9. Add BOT S F L, BOT S B L, BOT S F R and BOT S B R.

10. Add MID H L and MID H R. Now the lower part of the arcade is closed.

11. Add MID S R and MID S L, and then MID V F and MID H F.

The FabArcade is composed of two halves which can be disassembled for easy transport. You have already successfully built one of them (Figure 2.5).

You can now decorate the lower half of the arcade as you wish. Be sure to use material-friendly decoration means.

12. Assemble TOP S L 2 to TOP S L 3 and TOP S R 2 to TOP S R 3.

Figure 2.5: Finished bottom viewed from the front.

13. TOP S R 1 and TOP S L 1 are connected by a board (240 x 550 mm) which improves stability. Fix it to the TOP boards using four brackets at an approximately height of 20 mm above the bottom margin.

14. Add a board (200 x 550 mm) that connects TOP S R 1, TOP S L 1, TOP S R 2 and TOP S L 2 vertically. Look at Figure 2.6 to understand in which way to place this board between the pieces (the middle board in the picture). This board also delivers more stability and the panels back will be attached to it to carry its weight.

15. Assemble all TOP parts together.

16. Add a 80 x 550 mm board between TOP S R 2 and TOP S L 2 to improve stability.

17. Add a third and a fourth reinforcement board, 150 x 550

Figure 2.6: Finished top part of the FabArcade.

mm, each between TOP S L 3 and TOP S R 3. One should be placed on the end of one side and the other one near the end of the other side (view Figure 2.6, the two topmost boards.)

You can now decorate the upper half of the arcade.

18. Fixate the gamepad from underneath MID H F and place all electronics on the bottom half of the arcade.

19. Now prepare for the revetment and the display: First of all you have to decide how many layers the revetment

Figure 2.7: Finished bottom with gamepad.

should have and how thick each layer should be. Afterwards engrave a few paperboards with the special laser patterns which allow it to bend smoothly. Also prepare wood that will be used as a frame for the display. The wood should have a size of 400 x 585 mm (because our lasercutter can only cut to a limit of 600 x 300mm we took two wood sheets which had a size of 200 x 585 mm each to form the frame). Of course make sure to measure your display dimensions and adjust the size of the wood for the frame. You need the following sizes and quantity of paperboards for one layer of the revetment:

1. 585 x 300 mm: twice
2. 585 x 200 mm: twice
3. 585 x 150 mm: once

Of course you have to adjust the special pattern to the size of each paperboard when you cut it with the laser.

20. You have to dismantle the monitor panel out of the display. Now measure the panel's dimensions and cut a hole into the wood panels that fits the panel as a frame. Fix the

panel to the frame and now place the whole construction into the FabArcade. The panels back has to be attached to the board mentioned in point 14. You need to leave space for one 150 x 585 mm and one 200 x 585 mm paperboard underneath the display frame.

21. Measure the size of the *reset/home button* and cut a hole according to the size in the middle of the 150 x 585 mm paperboard to insert the button thereabouts. To assure that the paperboard is solid enough to last a push of the *reset/home button* fix a fortification underneath. Leftovers from the cutting process can be used as a fortification. When sticking the fortification underneath to fix it to the revetment, take care not to seal the home button hole.

22. Now it is time to fix the speaker onto one 300 x 585 mm paperboard. Before attaching decide which fancy shape the window for the sound should have. Cut the shape into the board where you want to situate it. We chose the pacman shape to refer to the famous arcade game.

Figure 2.8: Wholes for sound in pacman shape.

23. Adhere the paperboards to the top, front half of the arcade. Leave the back side partly open to ensure that the electronic is still accessible later on. Thus close a part of the back (roughly from the top until the board, where the panel is affixed; but you can decide how much to close. Just make sure you can reach all electronic parts including the volume

control for the speaker) and the front side of the bottom half by one piece of 1mm thin paperboard each.

24. Now decorate a head piece as you find suitable and add it to the top.

Figure 2.9: Example for the design of a head piece.

25. Assemble the bottom and the upper halves of the arcade using hinges. Obviously, don't glue them together if you want to take them apart later, as doing so would make this task obviously quite difficult. Use two hinges to join TOP S R 1 and MID S R, and another two for the other side.

26. Now lock the little door next to the coin slot to prevent thieves stealing your fortune.

27. The last step is to assure that your FabArcade will be user friendly. Therefore it is necessary to label some parts of the FabArcade to ensure the user knows where e.g. to interject the coins or which button is the start button.

In order to indicate the named examples we drew the following icons:

- **Home button icon:** Of course the FabArcade needs a button where the user can jump back to the main

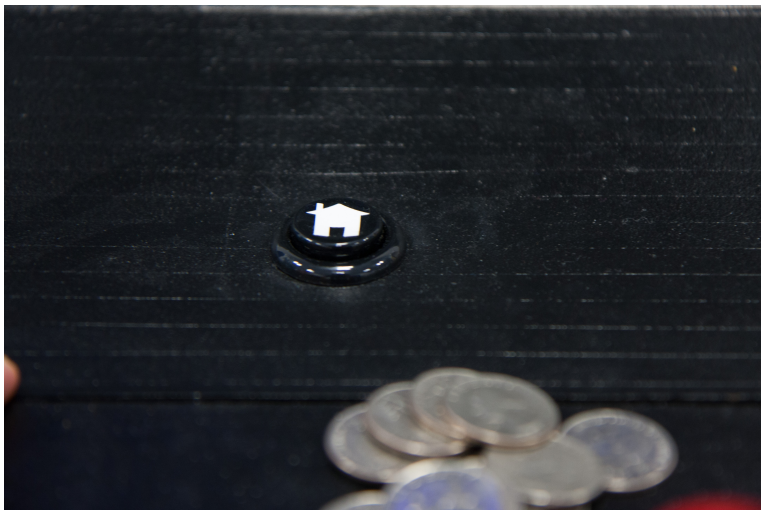

Figure 2.10: Home button to reset the FabArcade and come back into the main menu.

Figure 2.11: Start buttons one labelled as single player and one as multiplayer.

menu and get the chance to choose another game. That is the reason why we integrated the mentioned*Reset* button in the lowest paperboard. To assure user identifies this button as the *Reset* or *Home* button, it is labelled with a home button icon. Use the lasercutter to cut the icon out of adhesive foil.

- **Player icon:** It is possible for some games to either

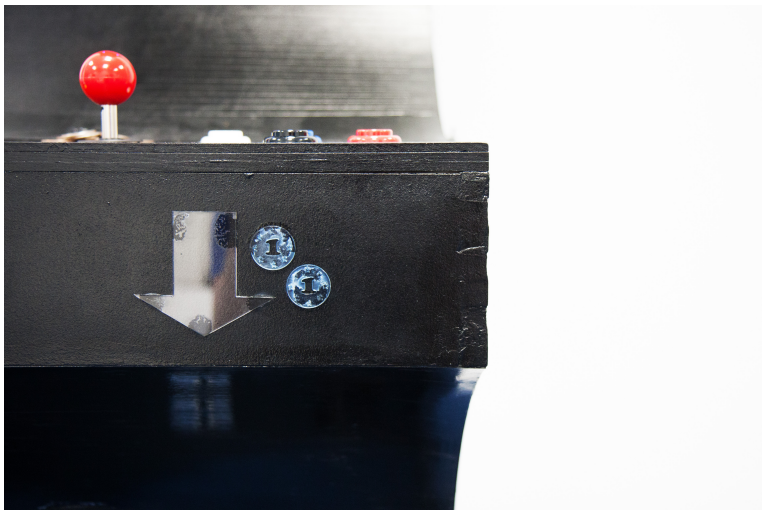

Figure 2.12: Coins and arrow indicate the direction where to insert the coins.

play as a single player or a multiplayer. Therefore the FabArcade has two start buttons (the white ones on the gamepad), one for single- and one for multiplayer. Pay attention that you label the two buttons properly. Simply use the single player design twice for the multiplayer button. Use the lasercutter to cut the icons out of adhesive foil.

- **Coin icon:** We designed a coin icon to indicate where the user has to insert the coins to start a game. Therefore the coins are placed together with an **Arrow** on the FabArcade to give directions where to put the coins.

Congratulations! You are now the proud owner of a FabArcade - housing. Let's continue on and get the software ready.

2.4 Controller

Here we will walk you through connecting the controller peripherals (i.e. Buttons, Joysticks, Coin-Slots) to a PC as an USB keyboard.
Therefore we are using an Arduino UNO, which is running software, that converts its inputs to a USB keyboard key.
Because of this, you can easily debug games on the Raspberry Pi with a common USB keyboard. You will find the mapping at 2.4.2 in figure 2.15.

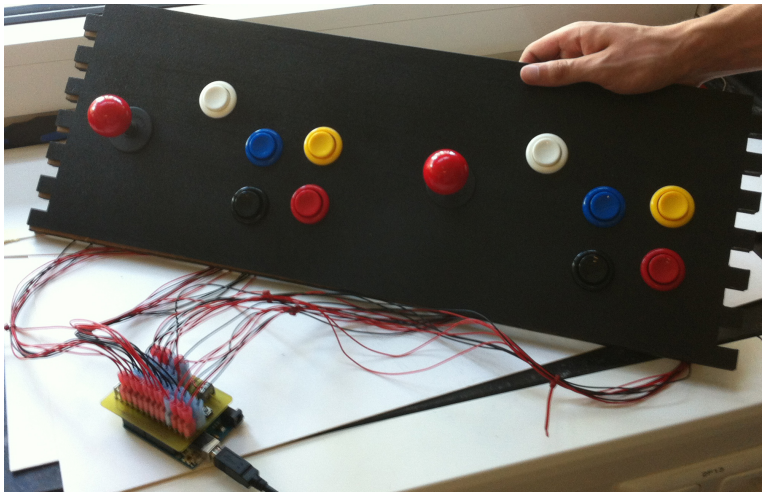

Figure 2.13: *Arduino UNO* attached to the game controller

2.4.1 Preparing the Hardware

Setting up the Arduino's software requires two steps.
First, we will need to program it as usual using the Arduino IDE.
Then flash the USB-controllers' firmware in order to emulate a USB keyboard. All required files are provided by us. You will find them in the download section.

Programming First, download the *Controller software package* from the download section. In the archive you will find a software sketch, that has to be uploaded to the arduino.
The software also requires the *Bounce library* to work correctly. This library is included in the archive's library directory and can be installed by putting it in the library directory of the Arduino IDE.

Windows: `My Documents/Arduino/libraries/`
Linux and Mac: `Documents/Arduino/libraries/`

For a more detailed explanation and a tutorial for automatic installation under Arduino 1.0.5, visit the website *http://arduino.cc/en/Guide/Libraries*.

After installing the library open up the sketch in your Arduino IDE, connect your UNO and upload the sketch.

Customizing the Software If you want the keyboard to emulate different (or a different number of) keys (button mapping see 2.4.2), you can simply adjust the variable *buttonRage* and edit the *pinChar* array at the top. The values are key IDs as specified in the HID usage table page 53, which you can find in the download section.

Please note that PINS are always used sequentially from PIN0 upwards but excluding PIN1 because it is needed for serial communication.
Keep this and the maximum number of keys of the Arduino UNO in mind if you change the number of keys

being used.

Flashing the firmware In order to use the Arduino UNO as an USB keyboard a new firmware has to be flashed to the Arduino's USB controller.
In order to do so you will need a tool called *dfu-programmer*. Installation depends on your OS, on most GNU/Linux systems this program should be available in a software repository.
On Windows you will need a special program called *Atmel Flip*. You can find the necessary firmware for our purpose in the already downloaded *Controller software package*.

To flash the USB-controller we first need to put the Arduino into dfu-mode
This is done by connecting it to the PC, then shorting pin 1 and 2 of the ICSP as shown in 2.14.
Doing so the indicator LED on the UNO should flash 3 times very quickly. You can verify that your Arduino is

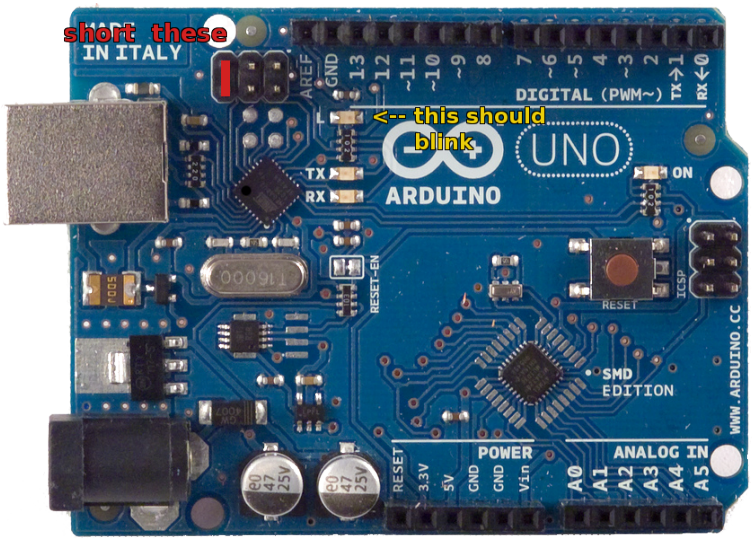

Figure 2.14: Short the shown PINs on the *Arduino UNO*

in dfu-mode by running `/susb` and looking for an *Atmel* entry.

Please note that the USB-controller will always be accessible using dfu-mode.

It is therefore not possible to brick the Arduino using this method.

After you have put the UNO into dfu-mode flashing should be as easy as running `./keyboard.sh` in the firmware directory.
It is not possible to upload sketches to the UNO while the keyboard firmware is active.
If it is necessary to upload new sketches you have to flash the old firmware by repeating the just explained process and run `./usbserial.sh` instead of the keyboard firmware.

To verify that the keyboard firmware is correctly flashed on the UNO plug-cycle check whether the Arduino is recognized as a USB keyboard.
A more in-depth explanation and some troubleshooting in the comments can be found at *http://mitchtech.net/arduino-usb-hid-keyboard*.

Programming the coin slot To set up the coin slot please follow the instructions in the manual that comes with it. We used the following settings for our slot:

- Number of coin types (E) = 1
- Number of samples per coin (H1) = 15.
  You need to choose at least 15 here or your setup might not work at all, even if you just feed it the same coin 15 times.
- Number of Pulses (P1) = 1
- Accuracy (F1) = 5

Please change the two switches on the coin slot, such that they are set to *SLOW* and *NO*.

With these settings you are ready to set up your coin(s). Just follow the procedure outlined by the manual.

After you have set up your coin(s), it is important to configure the AP mode as well. Contrary to what the manual suggests, this is not an optional step and the default value might not be 1 (it wasn't for us).
Therefore you need to manually check and set it to 1 if necessary.

2.4.2 Connecting the Hardware

Now that we have finished preparing the Arduino, the peripherals can be connected. In order to achieve the button-to-key mapping below, the peripherals also have to be connected to the specified pins as seen in 2.15.
Since pull-ups are active for all PINs just connect the peripherals so they will be tied to ground when the switch is enabled.

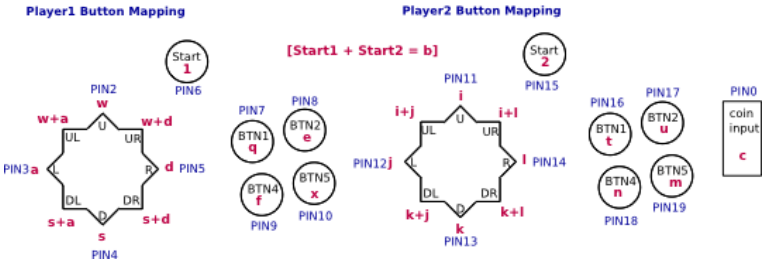

Figure 2.15: Key-to-button mappings for the game controller. The red letter denotes the key(s), which is reported to the host, while the blue PIN is connected to ground.

The FastOn Shield For more stability and efficiency, and because many connections to ground are needed, we also designed a shield for the Arduino. You can find the files needed to build it in the download section.

To build the shield you will need to use the attached schematics and board files to mill or etch yourself a pcb. Then you need to solder on the components, which are almost exclusively flat connectors.
To use the *Vin* connector on the shield you also need to solder on a cable and a screw terminal. For details you should refer to the eagle files.

To use the shield with the buttons and the Arduino itself follow these steps:

1. Put the shield on the Arduino

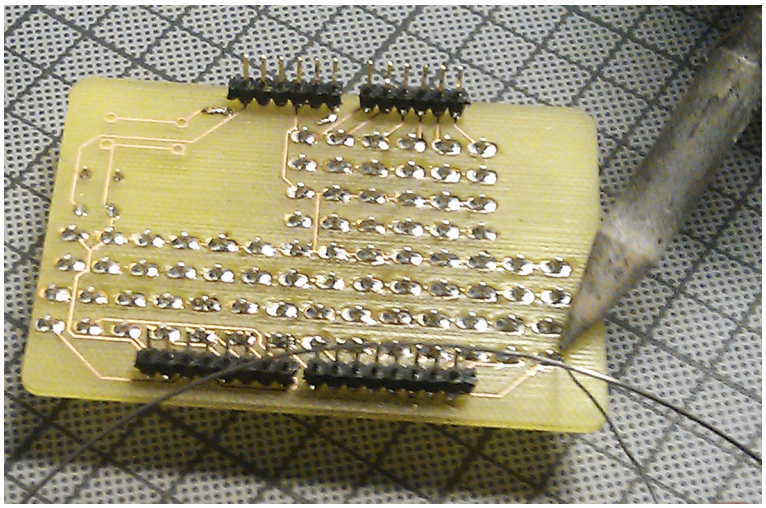

Figure 2.16: Our Arduino shield

2. Connect the 12 V power supply unit to the Arduino (this belongs to the coin slot)

3. Now connect all the buttons to their respective pins and ground as shown in 2.15. Buttons should be connected so that they close the circuit when pressed. The pins on the shield appear in the same order as they do on the Arduino and are always on the outside. The two middle rows are exclusively ground.

4. At last connect the VCC pin of the coin slot to the *Vin* pin on the Arduino using the screw terminal on the shield. Connect the wire to the coin slot and ground wire of the coin slot the same way you connected the microswitches.

Chapter 3

Setting up the Software

After we constructed the hardware of the FabArcade, we have to get the operating system working. By following the guide you will obtain a system for your arcade, based on the power of linux.

3.1 General

First of all, we do not provide roms or bioses in any way! Roms and bioses for advmame / gngeo have to be put in the corresponding directory, if you want to use them. (example: NeoGeo roms are stored in */home/arcade/gngeo_roms/*).
If you try to start a game in the menu and the rom for this game is not in the corresponding rom folder, the Pi will only show the loading screen and than fall back to the menu (advMame only puts out some info on tty1 and quits - leaving tty7 with a loading screen). We have not deleted the files for the games (see */home/arcade/games*) because the creation of the images (*background.jpg* and *loading.jpg*) took us very long and we wanted to avoid that you have to go through all this again.
At the following two configuration methods overclocking of the Pi is not considered in detail. If you are having heavy *Processing* / arcade games, you can simply overclock the Pi

by selecting the option in the `raspi-config`.
Our Raspberry Pi in the FabArcade runs at 900Mhz. We did this to be able to play any game out there, which runs on the Pi. Make sure to check the core temperature (via `/home/arcade/scripts/core_temp`), such that your Pi does not overheat, when doing overclocking!

About connectivity
The Pi is trying to get an ip address on its ethernet interface via DHCP. If you plug in a wireless adapter, the wireless interface will bind to *10.0.0.1* and will serve clients in its wireless network with ip addresses from *10.0.0.50* to *10.0.0.100*. The routing capabilities (wireless to ethernet) are not enabled by default, but can be activated (see the end of 3.5).

3.2 Structure of the home folder

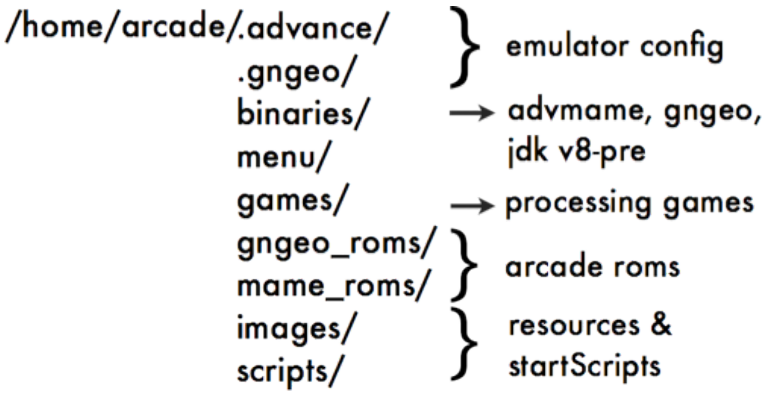

Figure 3.1: Structure of arcade's home folder

3.3 Installing the operating system

Now we present two options to configure the Raspberry Pi for the arcade. First, the novice method, which takes only very small effort and and it requires no great knowledge. The second method is the standard method, where you can customize things yourself and learn how you can deal with the Raspberry Pi.

3.3.1 Novice Method

In this method, it is only necessary to download the provided image and mirror it to the SD card (via dd). In this way the Raspberry Pi is ready for arcade gaming and ready to use. First you will mirror the image, then enlarge the root partition. After that you will be able to copy games to the FabArcade and start playing.

Mirroring the image

First, the SD card should be inserted into the SD slot of the laptop or PC.
The following commands are to be run in the console (on Linux/OSX - for Windows check below for how to setup the sd under windows). Type:
`sudo su` to obtain root privileges. Continue typing:
`diskutil` list will let you identify where the SD card is mounted (might be under */dev/disk1*). Type:
`diskutil unmountDisk [device node]` to unmount the SD card (example: `diskutil unmountDisk /dev/disk1`).

Now type:
`dd if=[path of the image] of=[device node]`
This will mirror the image file to the SD card (on Linux and OSX) (example: `dd if=/Users/user/Downloads/FabArcade_NOVICE_4GB.img of=/dev/disk1`)
If a different operating system is used, like Windows, or

further information is required, you can visit the following page:
http://elinux.org/RPi_Easy_SD_Card_Setup
After some time your sd card is a copy of our image (Might take up to hours). Type:
`diskutil unmountDisk [device node]` to unmount the sd card (repeat until successful).

IMPORTANT: Never remove the SD card without having run the unmount command first. The SD card can get corrupted and you will have to go back to mirroring the image to the sd card.

If everything went well you can now plug your sd card into your Raspberry Pi and hook it up to a monitor (e.g. via HDMI), connect it to your network, plug in the power adapter and boot the os. The menu for the arcade should appear now.

Resizing the SD card

If you followed all the steps correctly you should now have the Pi running the os. Maybe you noticed that the Pi system partition is smaller than the sd card you put it on. If you are running the os on a large SD card you might consider using many and huge roms or write big java games yourself. You can resize the partition the os is on to fill your entire SD card.

First you need to get access to the console. You can either press alt+ctrl+F2 to get to tty2 and login with root (password: *pi*) or you can access your Pi via ssh (See the standard method on how to connect). If you have access you type and run `raspi-config` in the console which will take you to a menu where the first selected row allows you to „Expand“ the root partition.
Press enter and follow the instructions. After one reboots the partition of the os should be enlarged to you entire SD card, such that you have plenty of space for your games and roms.
You can now skip to 3.5 to start copying games.

3.3.2 Standard Method

In the standard method you will configure your FabArcade from the very ground up.
We will start with Raspbian and install all required software, edit the system configuration and add our default user. In this way you can customise your FabArcade in the way you like.
After everything is setup you can start copying games to your FabArcade (see 3.4)

Install Raspbian

1. Download the latest version of the Linux distribution Raspbian from *www.raspberrypi.org/downloads*

2. To write the distribution onto the SD card, the following commands must be run in the terminal:
`sudo su` to obtain root privileges. Continue typing:
`diskutil list` will let you identify where the SD card is mounted (might be under `/dev/disk1`). Type:
`diskutil unmountDisk [device node]` to unmount the SD card (example: `diskutil unmountDisk /dev/disk1`)
Go on by typing:
`dd if=[path of the raspbian image] of=[device node]`
This will mirror the image file to the SD card (on Linux and OSX) (example: `dd if=/Users/user/Downloads/2013-09-25-wheezy-raspbian.img of=/dev/disk1`)
If a different operating system is used, like Windows, or further information is required, you can visit the following page:
http://elinux.org/RPi_Easy_SD_Card_Setup
After some time your sd card is loaded with Raspbian (Might take up to hours). Type:
`diskutil unmountDisk [device node]` to unmount the sd card (repeat until successful).

IMPORTANT: Never remove the SD card without

having run the unmount command first. The SD card can get corrupted and you will have to go back to mirroring the image to the sd card.

3. You have to hook up the Raspberry Pi to screen, connect to the Internet, plug in the power adapter and boot with the SD card.

IMPORTANT: If only the red light is on, the Pi is not booting properly and it should be started at step 2.

First login and system adjustments

Don't worry if there is no image displayed on the screen. You should log in via ssh on the Raspberry Pi. If you will see an image on the monitor already, you will be able to grab the ip address from the screen. Type in your local terminal:
`ssh pi@[the ip address of the Pi]`
example: `ssh pi@192.168.55.56` When asked for adding the Pi's key, type `yes`. Raspbian's default password for the user *pi* is *raspberry*

5. When logged in run the command:
`sudo su` and
`raspi-config`
Make sure you expand your root partition such that it fits your sd card (first option in *raspi-config*).
After that you can change system preferences the way you want. You may want to consider overclocking your Pi since games will run better then. We used the *Medium* setting there, which will set the cpu clock to 900MHz.
When done adjusting settings, you can exit the *raspi-config* tool by choosing *Finish* at the bottom, but don't reboot now, choose *No* when asked.

6. Now we will alter the display settings, type:
`nano /boot/config.txt` and make the following adjustments: Set the display resolution of the arcade: Change
`#framebuffer_width=1280` and
`#framebuffer_height=720`

to
`framebuffer_width=320` and
`framebuffer_height=240`
Remove any „#“ infront of these two lines and in front of the line
`disable_overscan=1` and
`hdmi_force_hotplug=1`
Compare with the image 3.2
Save with the key combination ctrl+o and press ctrl+x to exit the file.

7. In the following we will change the hostname and copy over the files provided by us. Change your Pi's hostname (you can change *FabArcade* to whatever you like), type:
`echo "FabArcade" > /etc/hostname`

8. You should now reboot your Pi once. Type:
`sudo reboot`

Note that the boot procedure will take longer, since the sd card will be resized. This only occurs once. You should also see a image on your monitor connected to your Pi.

User management

We will now enable the root user and delete the default user *pi*.

9. Once the system is back up, login via ssh as before. Type:
`sudo su` and
`passwd`
You will be prompted to enter a new password for the user root. We chose *pi*, but it is really up to you. Once done you should exit this session. Type:
`exit` and
`exit`

10. Login to your pi via ssh, but this time not with

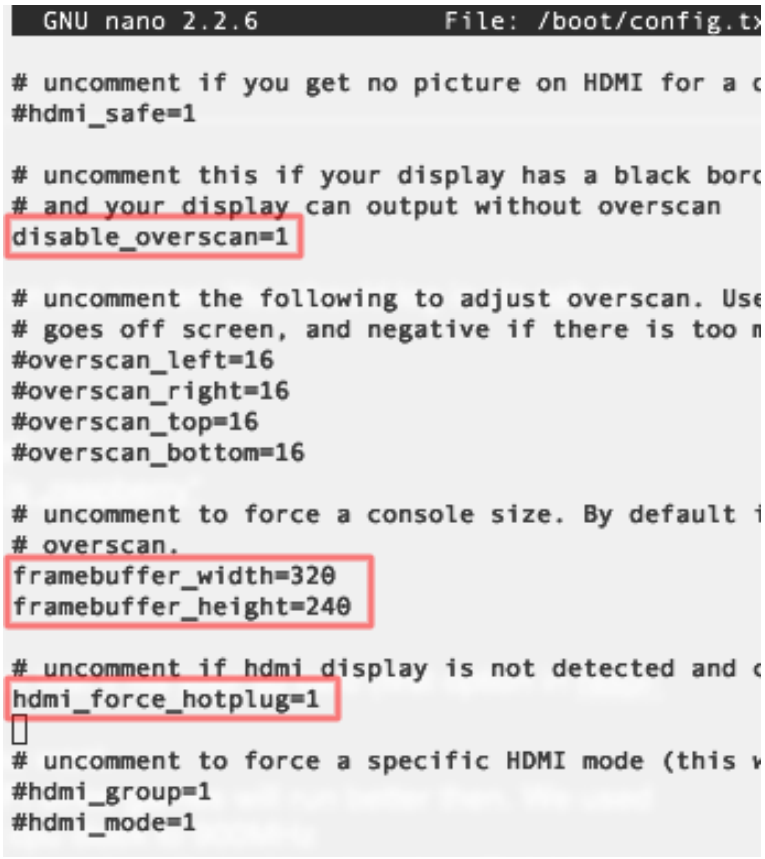

Figure 3.2: How the *config.txt* should look like after editing

the user `pi` but with the user `root` instead. Type:
`ssh root@[ip address of the pi]`
example: `ssh root@192.168.55.56`
Note that you will have to enter the password you chose in the previous step (we chose *pi*).

11. Go on by deleting the user *pi*. Type:
`deluser pi` followed by
`delgroup pi`
Finally delete `pi`'s profile folder under */home* type:
`rm -R /home/pi`
Compare with 3.3.

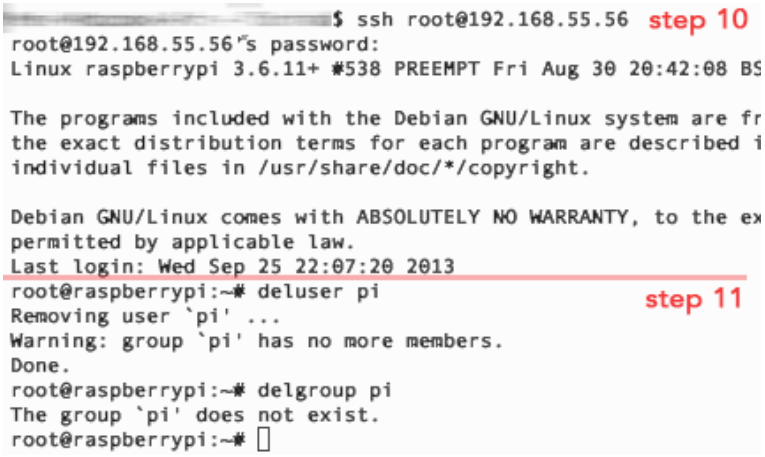

Figure 3.3: Deleting the user *pi*

Install Software

Next we will install all the software required by the arcade to work flawless.

12. Please note, that this will require an internet connection. Make sure you have plugged in an ethernet cable. You can check by trying to contact google (we assume google's servers are up), type:
`ping www.google.de`
Once you receive a response (*bytes from*) you know that you are connected. If not please ensure the cable is fine and your internet connection is enabled.

13. First we will update the Pi's package library, type:
`apt-get update` and
`apt-get upgrade` (enter `Y` if required - the upgrade can take some time) and
`apt-get dist-upgrade` followed by
`rpi-update`

14. Your Pi is now up to date. Let's go on by installing the software, type:
`apt-get install xorg` (enter `Y` when asked)
The X server is executed from *xinit* and it is necessary for the graphical output. This will take some time to install.
`apt-get install openbox`

Openbox is a window manager and will ensure the focus of java applications. Note that it is usually installed by default on Raspbian, such that the process might be aborted.
`apt-get install xli`
xli can draw an image on the root windows of the X Server.
`apt-get install xbindkeys` (enter `Y` when asked)
We will use *xbindkeys* to kill java games when *b*/both start buttons is/are pressed.
`apt-get install libsdl1.2-dev` (enter `Y` when asked)
This lib is used by gngeo and advmame. The installation will take some time.
`apt-get install samba samba-common-bin` (enter `Y` when asked)
Samba will make your Pi available to receive and serve files in the local network. It is very useful if you want to transfer roms to you FabArcade. We will link our configuration later.
`apt-get install alsaplayer-common alsaplayer-text` (enter `Y` when asked)
alsaplayer is used by some java games to play back audio files. The installation will take some time.
We will now adjust some files from *alsaplayer* such that it will fall back in text mode when launched with an active x server running. Type (one command):
`mv /usr/lib/alsaplayer/interface /libgtk2_interface.so /usr/lib /alsaplayer/interface/libgtk2_interface.so_backup`
Next link *alsaplayer* to the *aplay* file. Type:
`mv /usr/bin/aplay /usr/bin/aplay_original`
and
`ln -s /usr/bin/alsaplayer /usr/bin/aplay`
After this, everytime a game tries to play back audio with *aplay*, it will be redirected to *alsaplayer*.
Optional: Install *htop* to see processes and how they perform, type:
`apt-get install htop`

You have now installed all the required software.

A new user and our bundle

15. If you are logged in to your Pi, please close the connection now or open a new terminal on your pc.
We will now copy the files provided by us to your FabArcade to */home/arcade*.
Make sure you have extracted the folder *arcade* from the zip file. Please note that you will have to change the first path of the command to point at the newly extracted folder *arcade* on your local hard drive.
To copy the files, you need to type into your local terminal - so not on the Pi - the scp command (one command):
`scp -r [path of the folder provided by us] root@[ip address of the Raspberry Pi]:/home/`
example: `scp -r /Users/user/Desktop/arcade root@192.168.2.4:/home`
Take a look at 3.4, if you got problems with the command.

```
[redacted]$ scp -r /Users/f[redacted]n/Desktop/arcade root@192.168.2.4:/home
root@192.168.2.4's password:
advmame.rc                                    100% 8789     8.6KB/s   00:00
.bash_history                                 100% 5613     5.5KB/s   00:00
.bash_logout                                  100%  220     0.2KB/s   00:00
.bash_profile                                 100%  197     0.2KB/s   00:00
.bashrc                                       100% 3243     3.2KB/s   00:00
openbox.log                                   100%   61     0.1KB/s   00:00
.DS_Store                                     100% 6148     6.0KB/s   00:00
```

Figure 3.4: Copying our bundle to *arcade*'s home folder

16. Next you need to set the permissions. To do this you need to login to your Pi as root via ssh.
Once logged in type:
`chmod -R +x /home/arcade`
This will make all files be executable
`chmod -R 777 /home/arcade`
This will make all file readable/writeable to everyone. It is now time to create the arcades default user.

17. After that you want to add a user called *arcade*.

This user will later be logged in automatically and also start the menu and all the needed scripts. To create the user type:

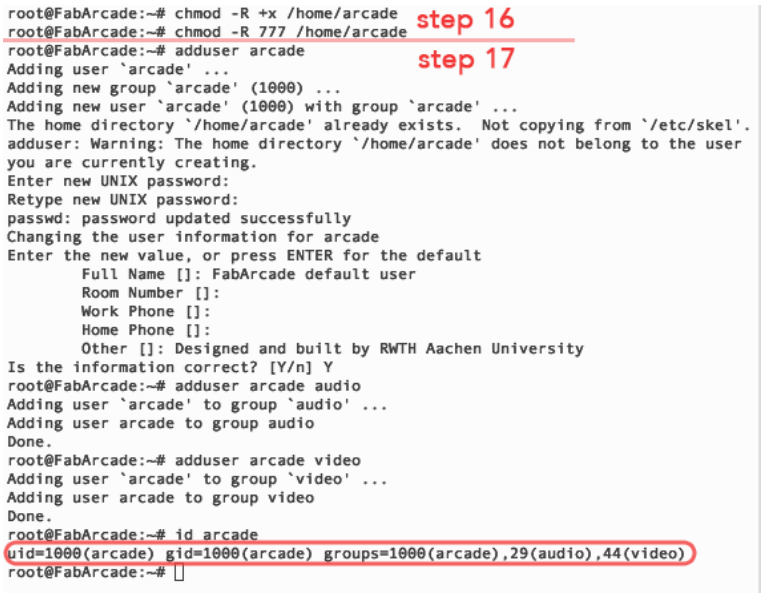

Figure 3.5: Changing the permissions and adding the user *arcade*

`adduser arcade`
Enter a password and account details (up to you - we used *arcade* as the password). To provide rights for the new user to access video and audio devices, type:
`adduser arcade audio`
`adduser arcade video`
You will now set the password for the samba login, type:
`smbpasswd -a arcade`
You will be asked to enter a password, we used *arcade* again. If you want to check if the user has been added successful type:
`id arcade`
This should then return the groups the user *arcade* currently is in. Please look for the group *video* and *audio*.
Check 3.5 to ensure success. If the correct groups were returned, you can continue. Otherwise redo the steps of 17.

Linking our bundle to the os

We will now link the os to the files provided by us.

18. Start by letting the Pi know where the java binaries are.
We include *jdk-8-ea-b116-linux-arm-vfp-hflt-13_nov_2013*.
Note: paths for *javac* and *javaws* were not changed by these steps. If you want to use their features, make sure that you keep in mind that their call will launch binaries of the old jdk (in our case version 1.7.0)
To register the new binary type:
```
update-alternatives --install
"/usr/bin/java" "java"
"/home/arcade/binaries/jdk1.8.0/bin/java" 1
```
To activate the new binary type:
```
update-alternatives --config java
```
Chose the version which path points to the new binary, in our case *1*.
You may want to take a look at 3.6 if you are unsure about your selection.
Verify that all went right by looking up the version of java, type:
```
java -version
```

If you get a version number (greater or equal to 1.8.0) in return, everything went fine.
Compare with 3.6 if unsure.

19. Now let's link the samba configuration file to our *smb.conf*. First remove the old *smb.conf*. Type:
```
rm /etc/samba/smb.conf
```
After that you should link the new file to this place, type (one command):
```
ln -s /home/arcade/scripts/smb.conf
/etc/samba/smb.conf
```

```
root@FabArcade:~# update-alternatives --install "/usr/bin/java" "java" "/home/arcad
e/binaries/jdk1.8.0/bin/java" 1
root@FabArcade:~# update-alternatives --config java
There are 2 choices for the alternative java (providing /usr/bin/java).

  Selection    Path                                          Priority   Status
------------------------------------------------------------
* 0            /usr/lib/jvm/jdk-7-oracle-armhf/jre/bin/java   317       auto mode
  1            /home/arcade/binaries/jdk1.8.0/bin/java        1         manual mode
  2            /usr/lib/jvm/jdk-7-oracle-armhf/jre/bin/java   317       manual mode

Press enter to keep the current choice[*], or type selection number: 1
update-alternatives: using /home/arcade/binaries/jdk1.8.0/bin/java to provide /usr/
bin/java (java) in manual mode
root@FabArcade:~# java -version
java version "1.8.0-ea"
Java(TM) SE Runtime Environment (build 1.8.0-ea-b116)
Java HotSpot(TM) Client VM (build 25.0-b58, mixed mode)
root@FabArcade:~#
```

Figure 3.6: Linking java and printing the version number

Autologin and permission adjustments

20. We will make the user *arcade* login automatically when your Pi boots by the following steps.
The file */etc/inittab* is to be altered.
Make sure you bind *tty1* to autologin (*-f arcade*) your Pi.
Configure the *inittab*, type:
`nano /etc/inittab`
Scroll down in the file (ctrl/strg+v) edit this line (one line):
1:2345:respawn:/sbin/getty –noclear 38400 tty1
to this (one line):
`1:2345:respawn:/bin/login -f arcade tty1 </dev/tty1 >/dev/tty1 2>&1`
The *inittab* should now look alike 3.7.
Save the file with ctrl+o and exit *nano* with ctrl+x.

```
#
# Note that on most Debian systems tty7 is used by the X Window Syste
# so if you want to add more getty's go ahead but skip tty7 if you ru
#
1:2345:respawn:/bin/login -f arcade tty1 </dev/tty1 >/dev/tty1 2>&1
2:23:respawn:/sbin/getty 38400 tty2
3:23:respawn:/sbin/getty 38400 tty3
4:23:respawn:/sbin/getty 38400 tty4
5:23:respawn:/sbin/getty 38400 tty5
6:23:respawn:/sbin/getty 38400 tty6


# Example how to put a getty on a serial line (for a terminal)
#
#T0:23:respawn:/sbin/getty -L ttyS0 9600 vt100
```

Figure 3.7: How the *inittab* should look like after editing

21. Next we will edit the *sudoers* file to allow the user arcade to execute privileged commands without the password. Type:
`nano /etc/sudoers`
Change the last line:
pi ALL=(ALL) NOPASSWD: ALL
to this:
`arcade ALL=(ALL) NOPASSWD: ALL`
Note we only changed *pi* to *arcade*, which is sufficient.
The *sudoers* file should now look like 3.8.
Again, save the file with ctrl+o and exit *nano* with ctrl+x.

22. Now we will change the permissions of the framebuffer, which is required by gngeo to run flawless. Type:
`chmod 777 /dev/fb0`

```
# Allow members of group sudo to execute any command
%sudo   ALL=(ALL:ALL) ALL

# See sudoers(5) for more information on "#include" directives:

#includedir /etc/sudoers.d
arcade ALL=(ALL) NOPASSWD: ALL
```

Figure 3.8: How the *sudeoers* file should look like after editing

You have now setup your Pi, your FabArcade! Reboot and start copying games to your Pi (see 3.4).

3.4 How to copy games to your FabArcade

3.4.1 Connect to your FabArcade

Once the Pi is up again you should be able to see a new device on your network: *FabArcade*.
You can now connect to your FabArcade with a *Samba client*, which you should usually already have installed.

1. On a Mac open a Finder window and press *cmd+k* (on Windows you will see the *FabArcade* in your network neighbourhood), type:
`smb://[ip address of the Pi]`
example: `smb://192.168.55.56`

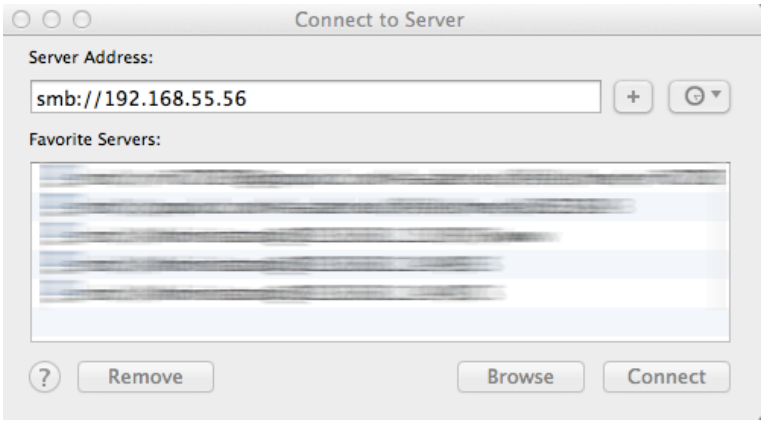

Figure 3.9: Connecting to the FabArcade via *samba*

After logging in, you should have access to the folders as seen in 3.10

2. If you want to access the folder *Arcade root* you will have to be logged in as the user *arcade* (via your samba client). All the other folders are read/writeable as *guest* by default.
Please make sure that no one gets access to these shares by accident. You can adjust the shares in the *smb.conf* file by typing (logged in via ssh):
`nano /home/arcade/scripts/smb.conf`

Scroll down to the very bottom. To make a share password protected, remove the # in front of *valid users = arcade*.
Note: the password of the user *arcade* is the one you entered earlier (*smbpasswd* - we used *arcade*).

3.4.2 Copy the games you want to play

Put your gngeo roms in */home/arcade/gngeo_roms/*. This folder is called *gnGeo roms* when connected via *samba*. Note that you will need to put a bios file (*neogeo.zip*) there as well. In */home/arcade/mame_roms/* (called *MAME roms* on *samba*) you should place the mame roms and a bios for the desired consoles (They have to be zip files).

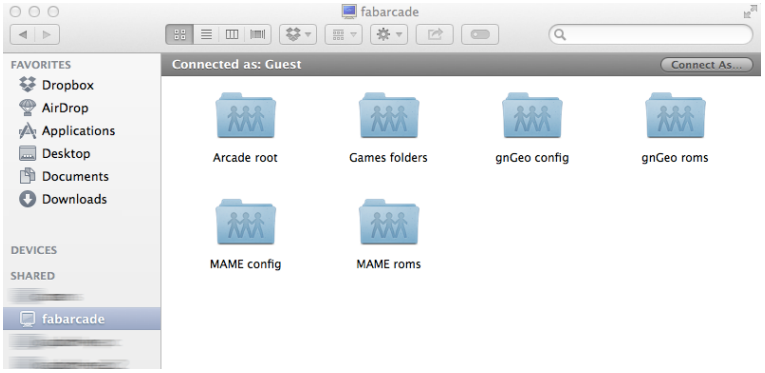

Figure 3.10: Folder overview

3.5 Optional: Setting up a wireless connection

To copy games to your FabArcade you can plug a network cable into the Pi. It will try to get an ip address in your LAN via DHCP. Since you might not have the ability to plug in a cable, but want to manage you FabArcade anyway, we present you a way of setting up a wireless connection to your FabArcade.

If you used the novice image, the wireless configuration is already included. Just plug in the *TP-Link wireless adapter (tl-wn823n)* and connect via the network *FabArcade* after a reboot of the Pi.
The password is "*FabArcade?AtRWTH!*" - without the "

If you have a USB wireless adapter laying around that supports *Access Point mode*, you can plug it in your Pi and do the following steps to create a wireless access point to which you can connect to.
Note: We set this up to be working with a *TP-Link wireless adapter (tl-wn823n)*. This is the only wireless adapter we tested.
There will be a DHCP-Server running on the Pi, meaning you can connect to the Pi with any common wireless-enabled laptop. You will receive an IP from the Pi.
Please note: If you have to pull something from the internet to the Pi, you need to connect an ethernet cable. The following steps will only enable you to connect to your Pi wirelessly and manage it.
Note that setting up a wireless connection might not be complicated, but if you want to customize your configuration you should have some general linux knowledge.

1. First you need to login to your Pi via ssh as *root*. Make sure it has a working internet connection.

2. You need to pull some software from the internet, type:
```
apt-get install hostapd udhcpd
```

3. We provide the most configuration files for you in

/home/arcade/scripts. All you need to do is adjust them such that they fit your needs. Type
`nano /home/arcade/scripts/hostapd.conf` to alter the settings of the wireless network which will be created. The *ssid* is the name of the network and *wpa_passphrase* its passphrase. Type
`rm /etc/hostapd/hostapd.conf` to delete the old configuration (if it existed) followed by (one command)
`ln -s /home/arcade/scripts/hostapd.conf /etc/hostapd/hostapd.conf`
This will link the configuration in the scripts folder to your os.

4. The next file you may alter is the configuration of the DHCP-Server. Type:
`nano /home/arcade/scripts/udhcpd.conf`
Basically there is not much to change. To delete the old configuration type:
`rm /etc/udhcpd.conf` followed by
`ln -s /home/arcade/scripts/udhcpd.conf /etc/udhcpd.conf` to link the configuration like the one for the wireless settings.

5. After this you should copy over the file which tells the os where the configs are, type:
`cp /home/arcade/scripts/hostapd /etc/default/hostapd`

6. Replace the hosted binary with the one we provide (it contains a fix for the wireless adapter we used - may be optional if a different adapter is used), type:
`cp /home/arcade/binaries/hostapd /usr/sbin/hostapd`

7. Make it executable, type:
`chmod +x /usr/sbin/hostapd`

8. Enable the hosted and dhcp server service to be loaded at boot, type:
`update-rc.d hostapd enable` and
`update-rc.d udhcpd enable`

9. If you want the Pi to share its ethernet connection

with its wireless interface, **which is disabled by default**, type:
`nano /home/arcade/scripts/setupRouting` and change
ENABLEROUTING=FALSE to
`ENABLEROUTING=TRUE`
Save with the key combination ctrl+o and press ctrl+x to exit the file.
Note that every time you change this, you should reboot the Pi.

10. Everything is now setup. Reboot the Pi and make sure you have the wireless adapter plugged in. You can then connect to the wireless network as described in 3.5.

Typeset February 26, 2014