“Building and using the FreeDeck-deej-combo”

a complete guide by

Florian Klugmann

GitHub

https://github.com/FKlugmann/FreeDeck-deej-combo

Introduction

This Paper is a complete guide to source, build, setup and use the FreeDeck-deej-combo. The Device is a configurable keyboard with OLED screens as buttons and audio-faders which let you control your “Windows Volume Mixer”. A build in rotary encoder lets you easily switch between your pages of your FreeDeck configuration.

Special thanks goes out to Kilian Gosewisch, the inventor of the FreeDeck project, and Omri Harel, the inventor of the deej-project. Please check out their GitHubs for the latest updates of their project.

FreeDeck by Kilian Gosewisch

https://github.com/FreeYourStream/

deej by Omri Harel

https://github.com/omriharel/deej

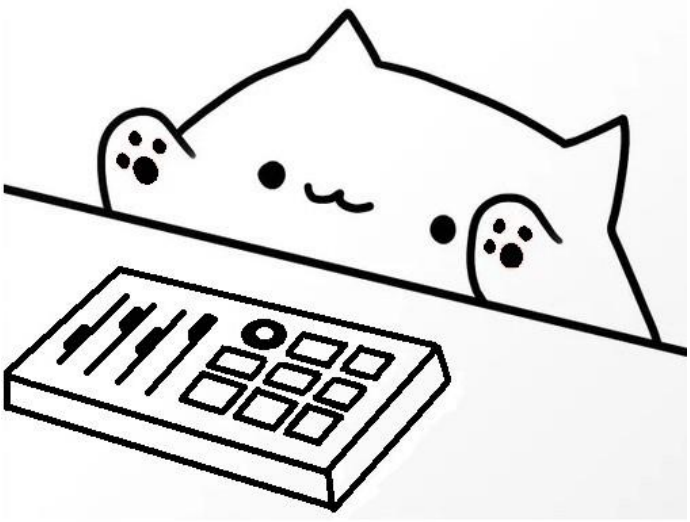

Table of Contents

Components and Sourcing Guide

This chapter lists all the required components and explains where you need to pay attention while buying.

1.1 List of Components

Quantity	Part	Modell
1x	Microcontroller Board	Arduino Pro Micro 32U4 5V 16MHz
1x	Card Reader	SPI Micro SD Card Reader Module
1x	SD-Card	Micro SD-Card < 16 GB
8x	Screens	SSD1306 128x64 OLED Display Module 0.96”
8x	Push Buttons	8x8x5mm 4 Pin Silicone Tactile Push Buttons
2x	Multiplexer	74HC4051 DIP-16 Multiplexer
1x	Rotary Encoder	EC11 Rotary Encoder 9.5mm shaft with push button
2x	Resistor	47k Ohm THT Resistor <12mm length
1x	LED	3mm LED THT
1x	Resistor	680 Ohm THT Resistor <12mm length
4x	Faders	75mm 10k Potentiometer B103
1x	PCB	Custom Freedeck-Deej PCB
4x	Screw	M3x5mm screw
4x	Rubber Feet	Anti-Slip Pads ≥14x14x3mm

1.2 Compatible Components

1.2.1 Microcontroller Board

The core component of the Freedeck and Deej projects is the microcontroller. For this build you need an “Arduino Pro Micro 32U4 5V 16MHz”, like the one shown in Figure 1.

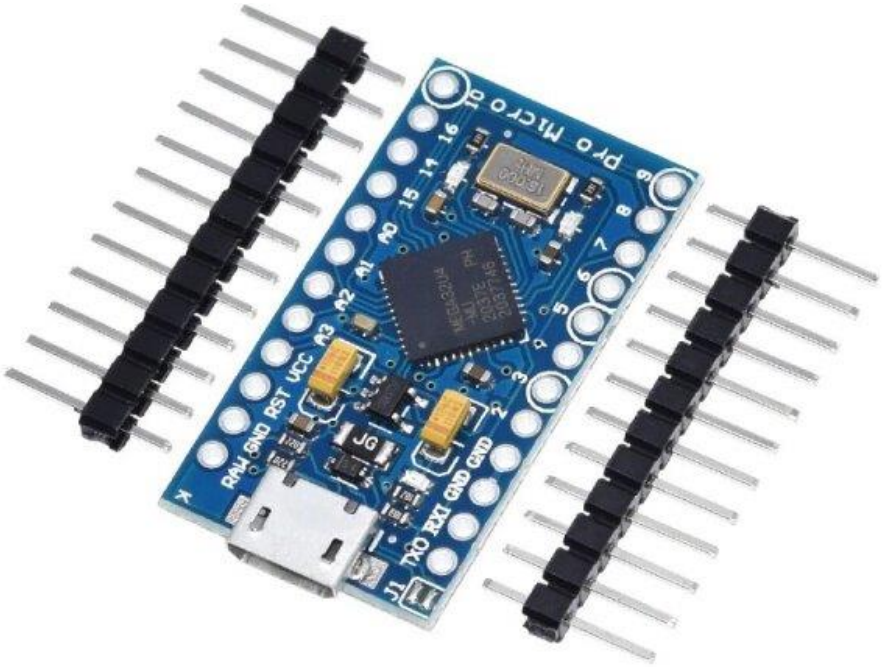

Figure 1: Arduino Pro Micro 32U4 5V 16MHz

When buying this, pay attention that you get the version with the Atmega32U4 microcontroller. Also, there are versions with different bootloaders. Make sure you get the one with the Arduino Leonardo bootloader. The Arduino Pro Micro should not have pre-soldered pin headers, because you need to mount the Arduino upside down for this build. I recommend buying the basic ones with the blue PCB, because I can’t ensure that different versions fit the footprint on the custom PCB.

Example Source

1.2.2 Card Reader & SD-Card

For the SD-card reader chose a “Micro SD-Card Reader SPI Module” with a PCB which looks like the one shown in Figure 2. Make sure you chose one with a single row of pin headers, and not the one with 2.

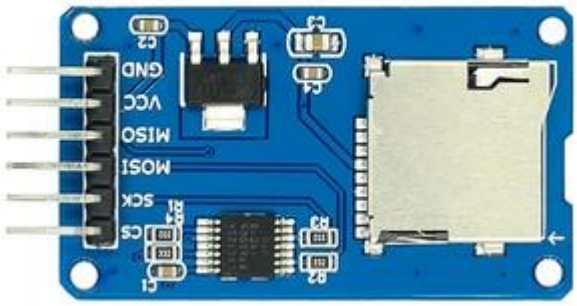

Figure 2: SPI SD-card reader

For the SD-Card chose any SD-Card that has a capacity of 16GB or less. I had Problems once with a 32GB one. Since the SD-card only needs to store the config-file of the Free-Deck, not much space or speed is required.

Example Source Reader

Example Source SD-Card

1.2.3 Multiplexer

You need “4051 Multiplexers” in a “DIP-16” package. I recommend the high-speed-cmos variant “74HC4051” like the one shown in Figure 3. Watch out not to buy the SMD ones.

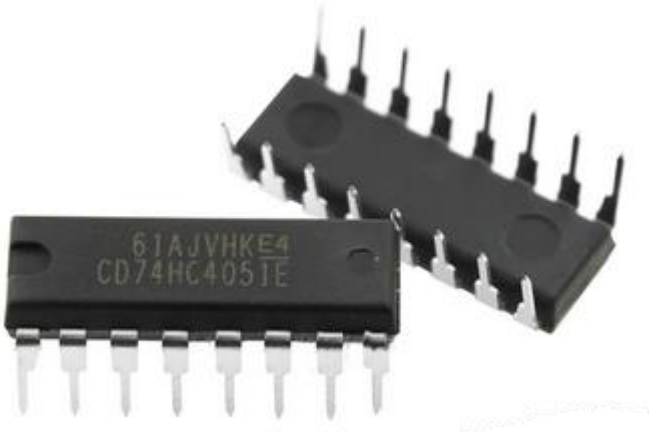

Figure 3: Multiplexer 74HC4051 DIP-16

Example Source

1.2.4 Screens

The screen is the hardest component to choose because the variety of versions is big. You need to look for “SSD1306 OLED Display Modules” with a 128x64 pixel resolution. Mainly there are 2 types of these screens, ones with a black and ones with a blue PCB. The blue ones have slightly bigger screens. For this build you need to look for the ones with the **black** PCB, like the one in shown in Figure 4.

Figure 4: SSD1306 0.96" OLED screens

The Custom PCB and Case is only compatible with this kind of screens. Make sure that you get screens with 4 pins with following pinout (left to right):

[GND] , [VDD/VCC] , [SCK] , [SDA]

There are versions with switched GND and VCC pins, so watch out! The quality of the screens and the intensity of the coil whine is varying, so I recommend to not cheap out on them.

Example Source

1.2.5 Push Buttons

The needed push buttons are “8x8x5mm silicone tactile 4-pin push buttons” like the ones shown in Figure 5. These buttons are completely silent but have the downside, that the trigger-point isn’t as distinctive. Alternatively, you could buy normal 6x6x5 4-pin switches and bend the pins to fit the board. But beware, the clicky-noise is hearable on microphones.

Figure 5: 8x8x5mm silicone tactile push buttons

Example Source

1.2.6 LED and Resistor

For the Status-LED we need a 3mm THT LED, like the one shown in Figure 6. Since the LED is only showing up while transferring the FreeDeck-config, you can pick any colour you want.

Figure 6: 3mm LED THT

To limit the current, we need a wired resistor. If you got any laying around from 220 Ω to 1kΩ you can use them. I recommend 680Ω. The resistor should not be longer than 12mm to fit the PCB.

Example Source LED

Example Source Resistor

1.2.7 Rotary Encoder and Resistors

The needed rotary encoder has a push button and a 9mm or 9,5mm shaft. You can easily install other encoders, since the footprints are similar. I recommend buying one like shown in Figure 7 to fit the knob of the case. Encoders with longer shafts stand out way too much.

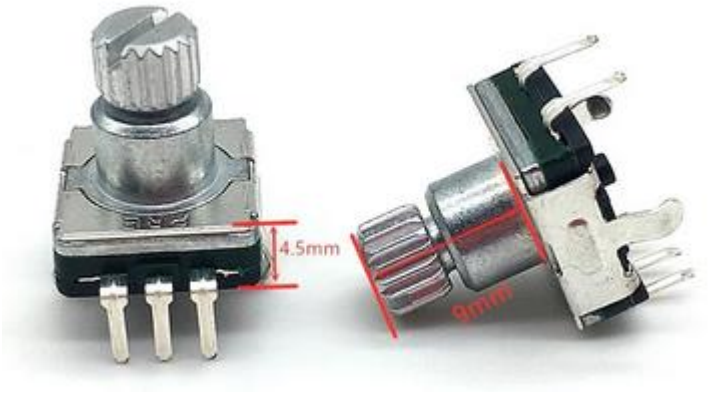

Figure 7: EC11 rotary encoder 9.5mm shaft

For the encoder you need 2 wired pull-up resistors. If you have anything from 10kΩ to 100kΩ laying around, use them. Otherwise, I recommend buying some wired 47kΩ resistors. The Resistors should not be longer than 12mm to fit the PCB.

Example Source Encoder

Example Source Resistor

1.2.8 Faders

To control the audio with deej, we need 75mm B103 10k potentiometer faders. You need that typical one with the green base, like shown in Figure 8.

Figure 8: 75mm B103 10k potentiometer fader

Example Source

1.2.9 Screws

To secure the custom PCB into the case you need 4 M3x5mm Bolts. The head of the bolt shouldn’t be higher than 6mm and not wider than 8mm. A normal M3x5mm Allen-screw fits perfectly.

Example Source

1.2.10 Rubber Feet

To stop the case from moving on desk, you need some rubber pads. Any self-sticking pads that are smaller than 15 by 15mm and that are higher than 1.5mm should fit the case.

Example Source

1.3 Ordering a Custom PCB

To get the manufacturing data for the custom PCB you need to go to the GitHub of this project and download “Gerber_files_Fredeck8_deej4.zip”. You don’t need to unzip it, because most of the manufacturers require the zipped folder.

For this guide I will explain the ordering process of the custom PCB using the example of “JLCPCB.com”. I can recommend this shop because of the low price and fast delivery.

JLCPCB.com

First you need to create an account on JLCPCB.com. When you have done that and you are locked in, you need to click on the “Order now” button on the upper right corner. Right in the middle of the screen should be a blue “Add gerber file” button. Click it and upload the zipped gerber-files folder. When you successfully uploaded the gerber-files, a picture of the front and backside of the PCB, like shown in Figure 9, should appear.

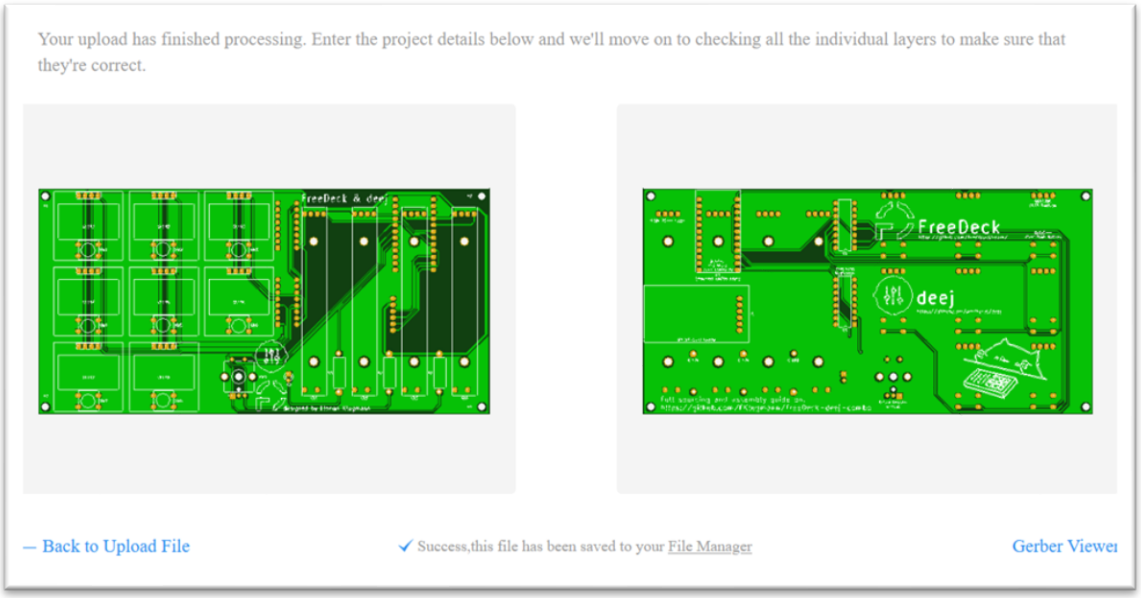

Figure 9: PCB on JLCPCB order screen

You can check the PCB on a separate tab by clicking on “PCB Viewer”. I recommend using the options as shown in Figure 10.

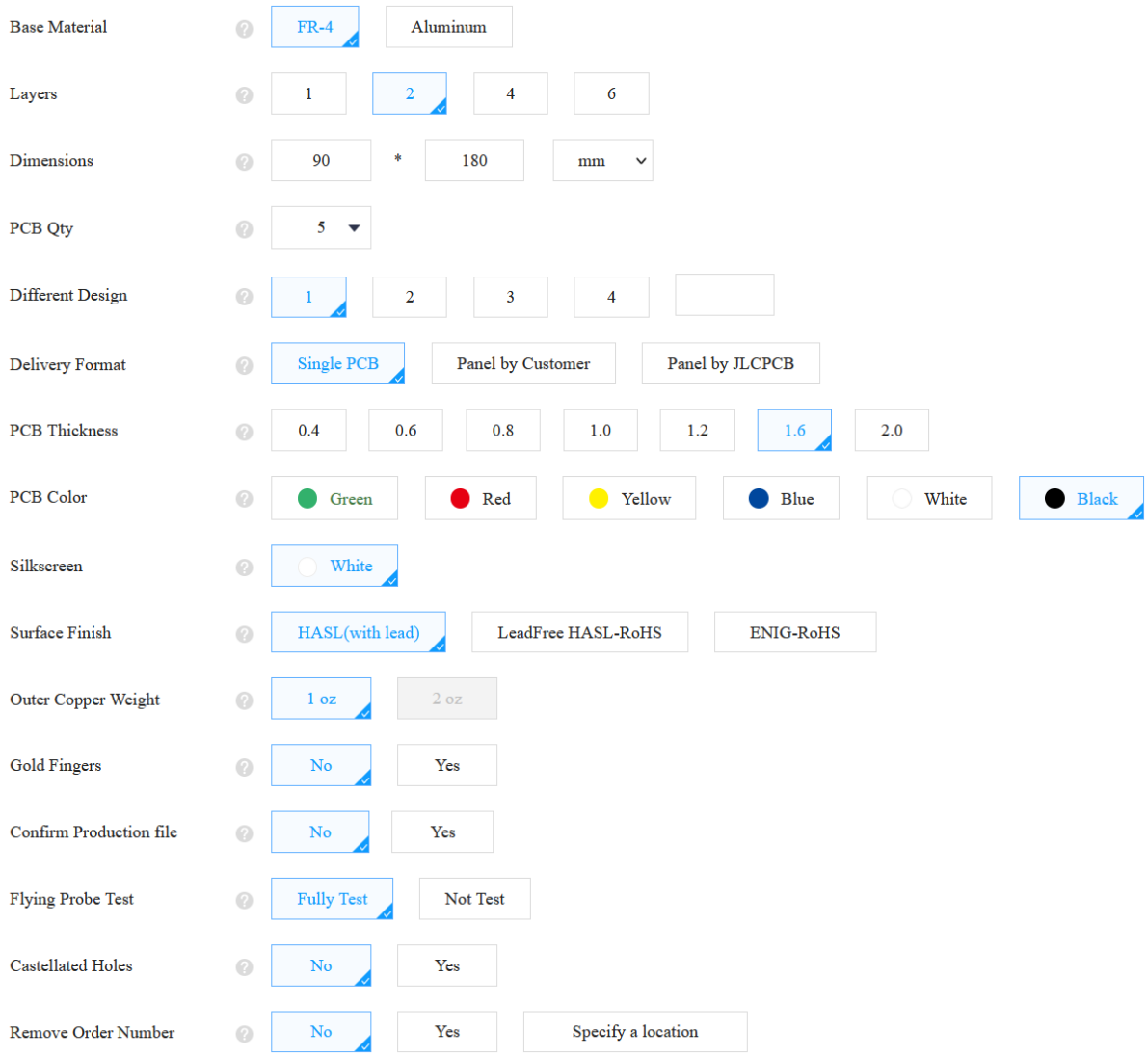

Figure 10: Order settings JLCPCB

When all settings are done, you can save it to your cart. Then choose your shipping options and you are ready to checkout and place your order.

2 PCB Assembly and Soldering Guide

2.1 Preparation

2.1.1 3D Printed Solder Guide

On the GitHub you will find a “Freedeck_Screen_Solderguide.STL” file. I recommend 3D-printing this part, because it makes the screen soldering progress a whole lot easier.

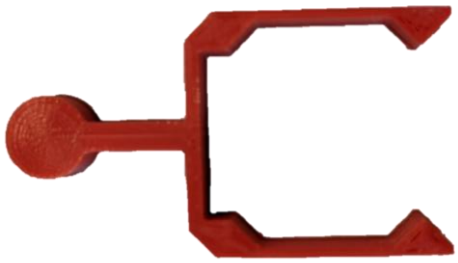

Figure 11: 3D-printed screen solder-guide

2.1.2 SD-Card Reader

The Pin-header of the SPI SD-card reader are normally bend 90°. To mount it on the PCB, you need to bend the pins straight like shown in Figure 12. When the pins are straightened, you need to clip them to about half of the full length. Nose pliers are a perfect tool for this job. Due to the force exerted by bending, the solder joints on the back can break. I recommend a quick reflow of these joints.

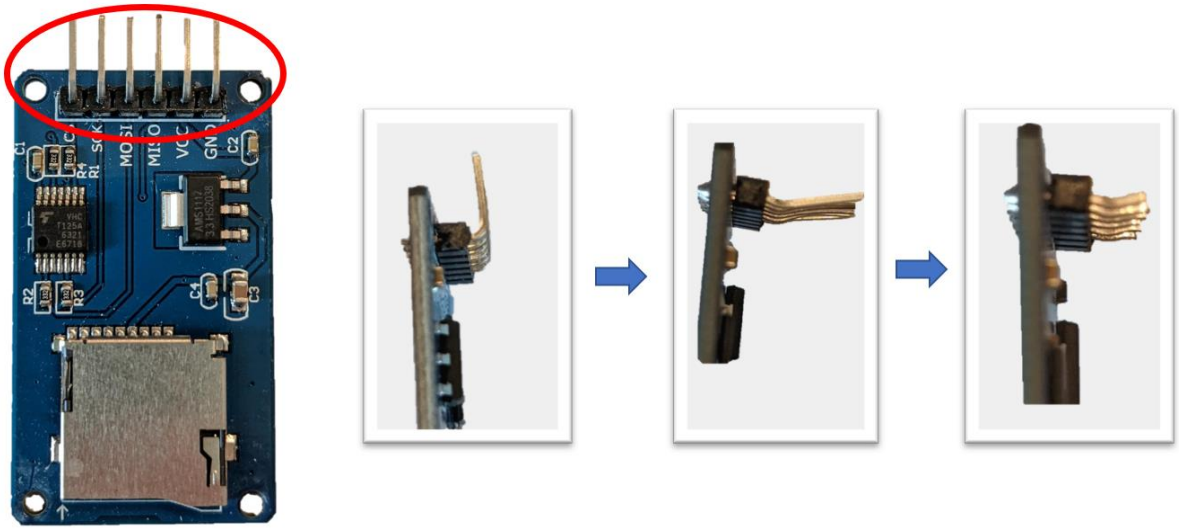

Figure 12: SD-card reader pin bending

2.1.3 Arduino Pin-Headers

To mount the Arduino on the board, soldering the pin-headers is required. Since we are going to mount the Arduino backwards, we need to mount the pin-headers like shown in Figure 13. Make sure the pin-headers pointing **upwards**, when looking on the front of the Arduino. Once they are soldered on, it's hard to get them off without destroying the Arduino. To ensure that the pins are straight, you can stick the pin-headers into a breadboard, while soldering. You can also use the custom PCB as a guide, but don't solder the pin-headers to it! I recommend to put solder on the Jumper ***J1*** (right next to the USB port on the Arduino). It bypasses the voltage regulator of the Arduino and the screens are getting direct power via USB. I recommend testing your Arduino before soldering it to the custom PCB. For that, connect it to the PC and look for it in the Arduino IDE (Chapter 3.2).

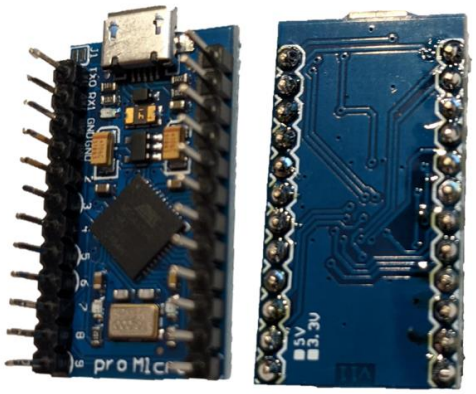

Figure 13: Arduino pin-header position

2.1.4 Cutting the Fader shaft

For better looks and useability, I recommend cutting the fader shafts to a length of approximately 7mm before soldering them to the board. A suited tool for this job is a bolt cutter, since you get a clean cut. You could also use a small metal saw, but get rid of the metal chips before soldering to prevent shorts.

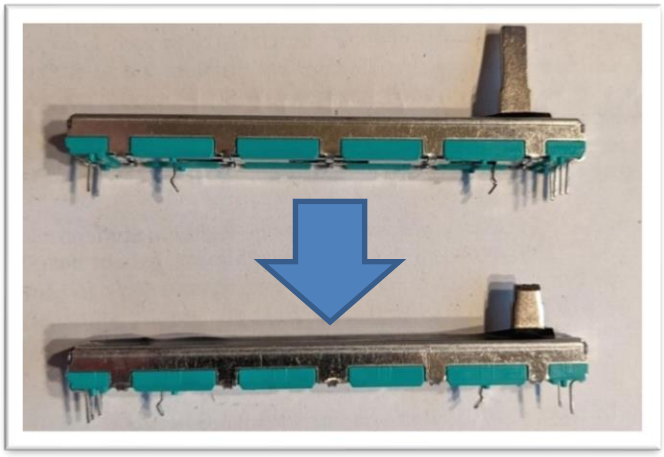

Figure 14: Cutting length of the fader shaft

2.2 Soldering

When all the steps of the preparation are done, you can begin to solder the parts to the custom PCB. You need to follow the Soldering order step by step in the following order! Do **not** skip a step, because some parts can only be soldered on in the right order!

2.2.1 Multiplexer

First you need to solder the two Multiplexers onto the **backside** of the PCB. The designated places are “U2” and “U3”. Make sure you align the marking of the footprint with the one on the IC, like shown in Figure 15.

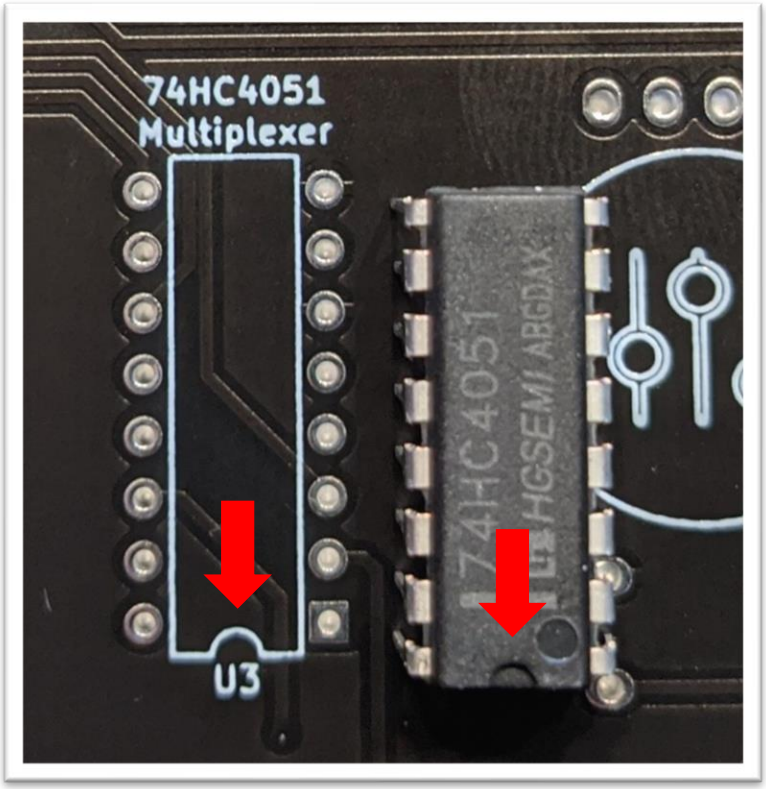

Figure 15: Multiplexer alignment

2.2.2 Buttons and Screens

The Buttons and screens have to be soldered in pairs and in a strict order. I recommend starting from button “SW8” and screen “OLED8” and going down to one. In every pair you need to first solder on the button on the front side of the PCB. When the button is on, you can place the screen with the solder-guide above it, as shown in picture 2 on Figure 16. Hold the screen in place by pressing it down with a finger, turn around the PCB and solder one of the screen pins. Once a pin soldered to a pad you can release the pressure of your finger and the screen should stay in place. You can now solder the other pins to the PCB and simply pull out the solder-guide. When soldered on correctly it should look like picture 3 in Figure 16.

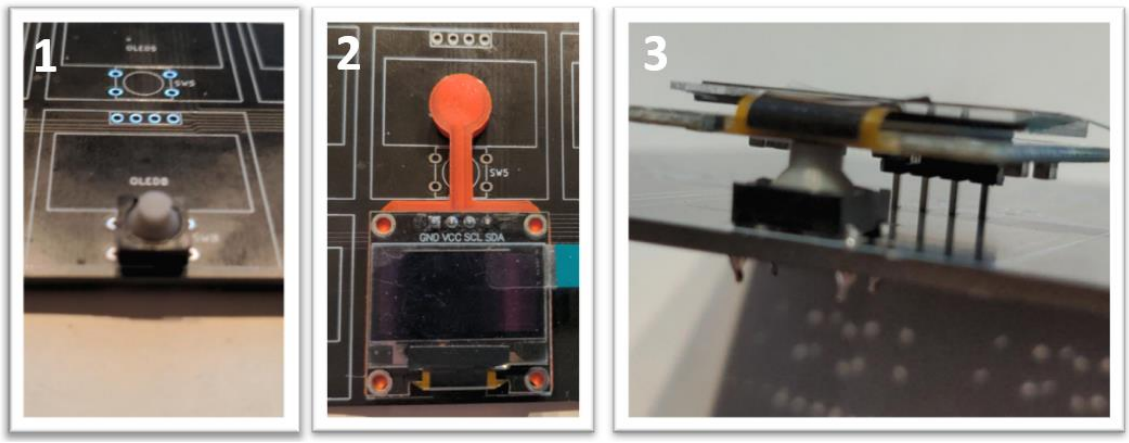

Figure 16: Soldering the buttons and screens

When all buttons and screens are soldered, your custom PCB should now look like Figure 17.

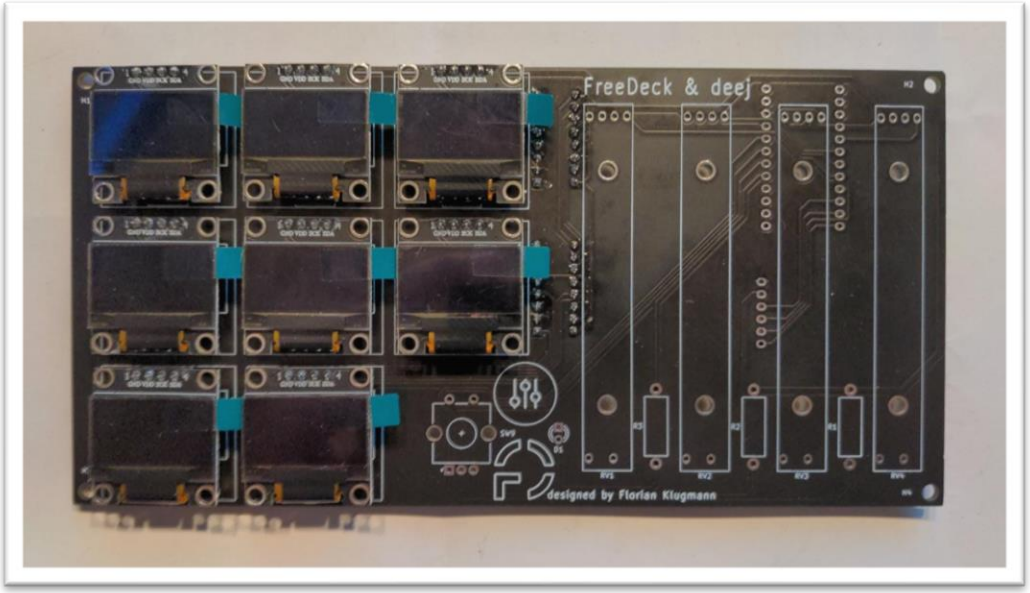

Figure 17: Custom PCB with screens

2.2.3 SD-Card Reader

The SPI SD-card reader needs to be soldered on the backside of the PCB as shown in Figure 18. While soldering, you need to push the SD-card reader against the custom PCB, so it's flush with the surface.

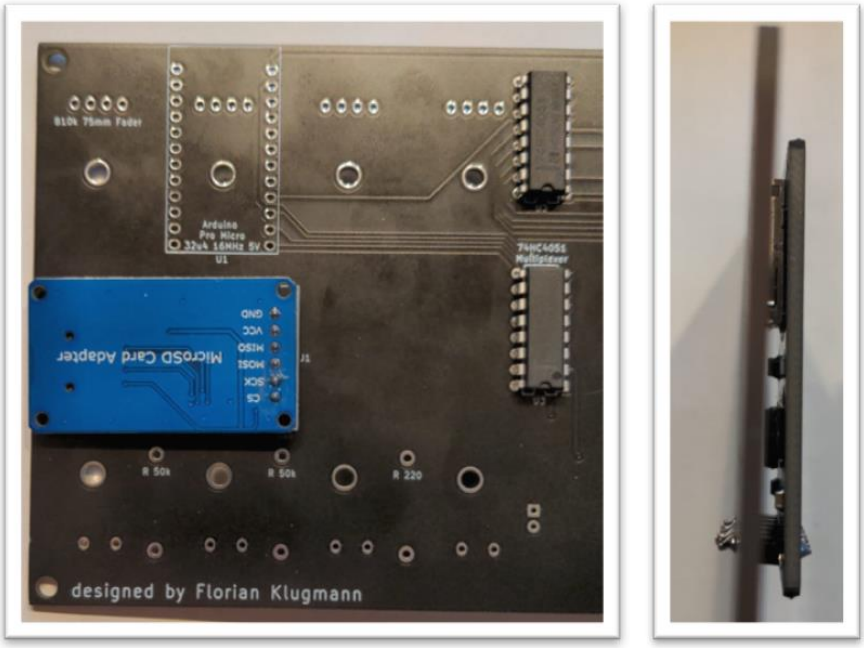

Figure 18: SD-card reader position

2.2.4 Rotary Encoder, Resistors and LED

Now you need to solder on the resistors as shown in Figure 19. When soldering resistors, you don't need to pay attention to polarity.

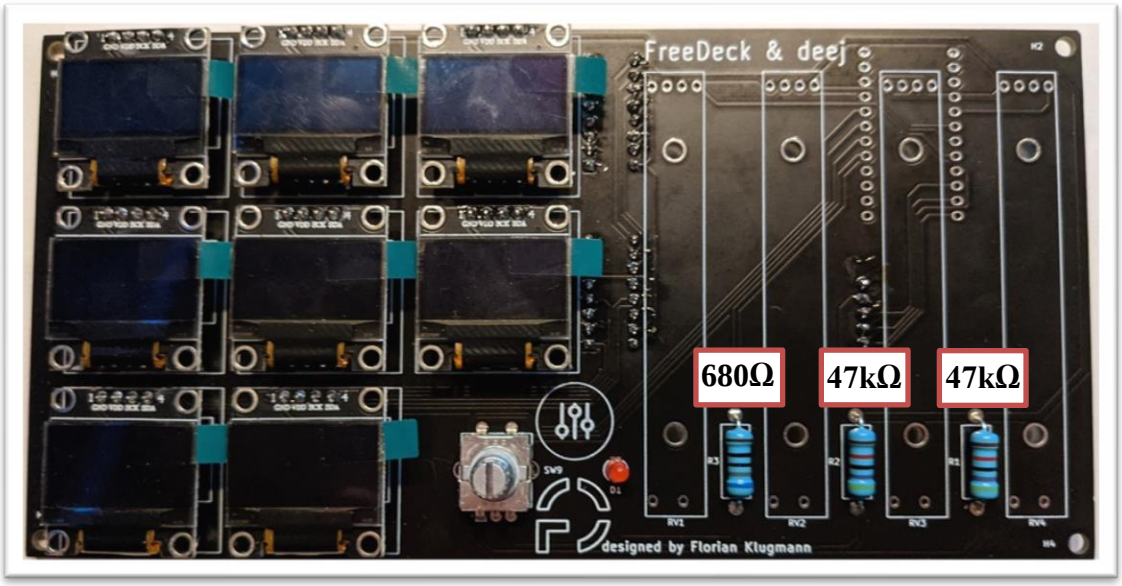

Figure 19: Position of Resistors, LED and rotary Encoder

For the LED you need to consider the polarity. Like shown in Figure 20, the short pin of the LED needs to be soldered to the square pad (GND) of the PCB.

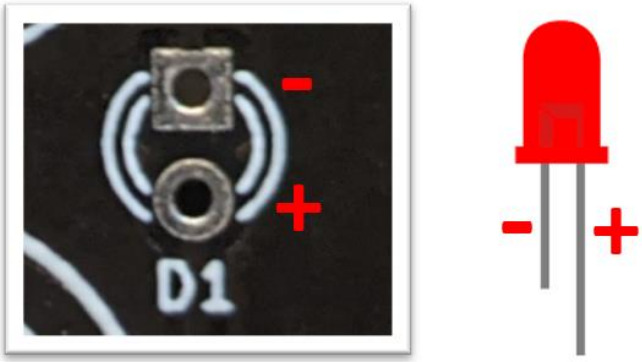

Figure 20: LED polarity

The encoder only fits in one direction. Solder it flush with the frontside of the PCB. If there are plastic pins on the underside of the rotary encoder, that don't fit the PCB holes, cut them off. When you are finished soldering, just cut the excess wires of the LED and resistors.

2.2.5 Arduino and Faders

Before we are going to solder on the prepared Arduino, we need to solder on the Fader ***RV3***, because the upper pins are laying below the Arduino. To prevent the fader pins from touching the Arduino PCB we need to bend the pins before soldering, as shown in Figure 21. They need to be flush with the surface of the custom PCB!

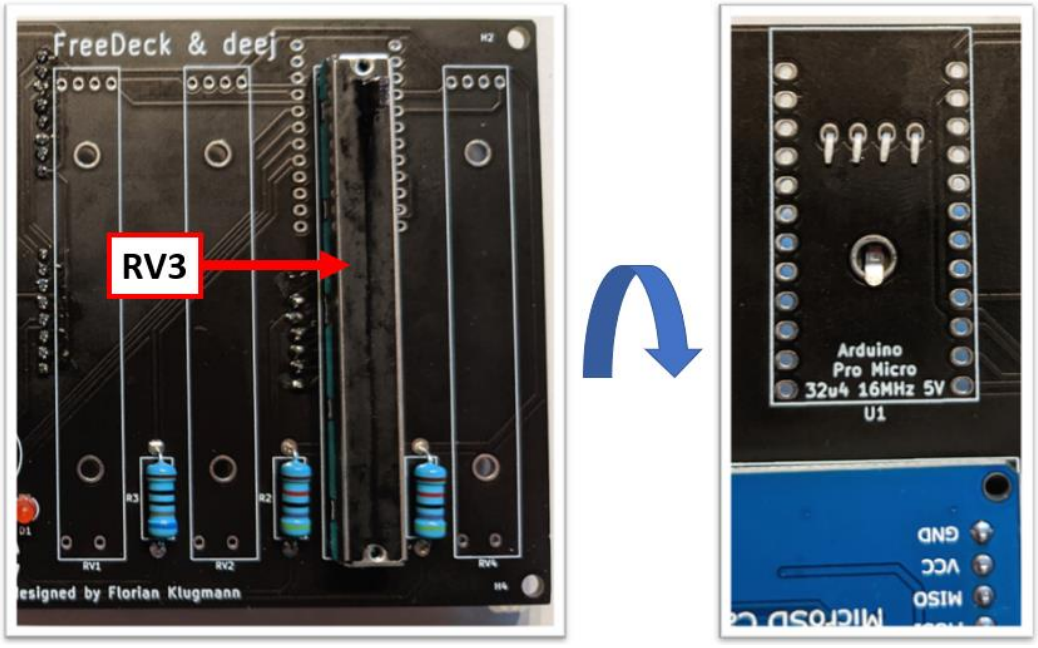

Figure 21: Bend pins fader RV3

When the fader ***RV3*** is soldered on, I recommend soldering on the prepared Arduino, since the pins are now easily accessible. Push the Arduino through the holes on the back-side, so that the plastic base of the pin headers is flush with the surface. Then solder on the pins on the frontside as shown in Figure 22.

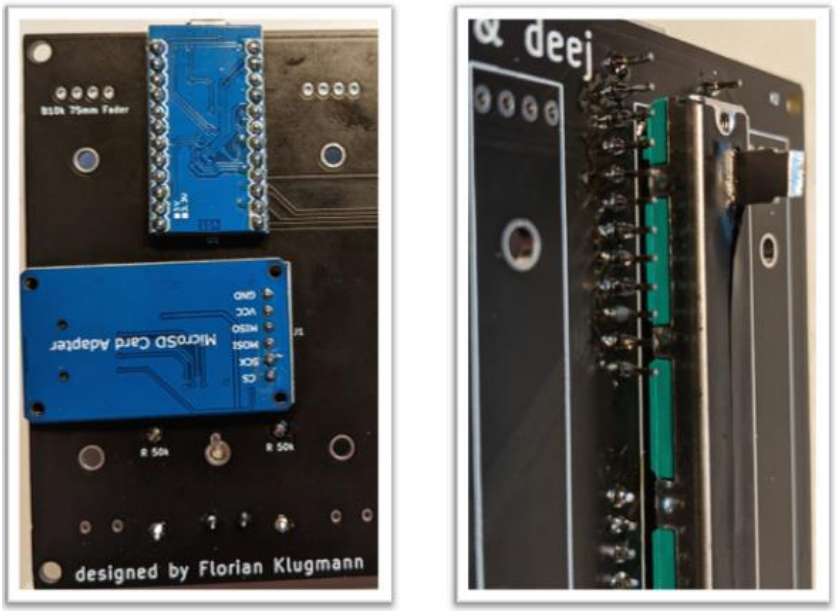

Figure 22: Position of the Arduino

After soldering the Arduino onto the PCB, you can continue with the remaining faders. You don’t need to bend the pins of the other faders since there is no collision with other components.

When the last fader is soldered, your PCB is finished and ready for flashing the firmware! Your PCB should now look like Figure 23.

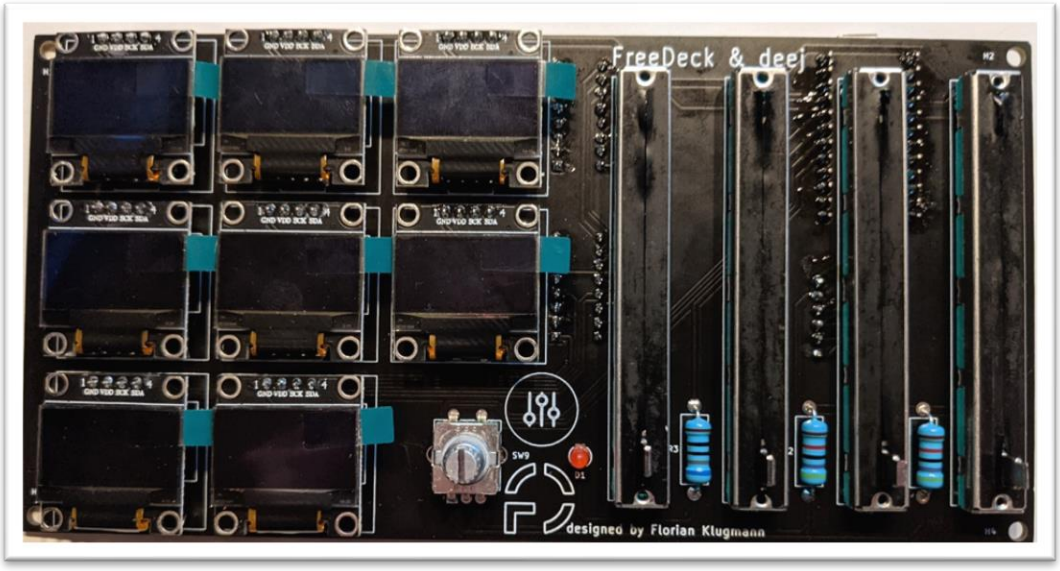

Figure 23: Finished FreeDeck & deej PCB

3 Configuring and Flashing the Firmware

3.1 Arduino IDE and Packages

To flash the Firmware onto the FreeDeck-deej-combo you need to download the Arduino IDE from arduino.cc . Make sure you download the Windows installer and not the app, or the version you need for your specific system. Install and start the Arduino IDE. Before we get to flashing the firmware, we need to install some packages. For that, you need to click “Tools” and then on “Manage Libraries…” like shown in Figure 24.

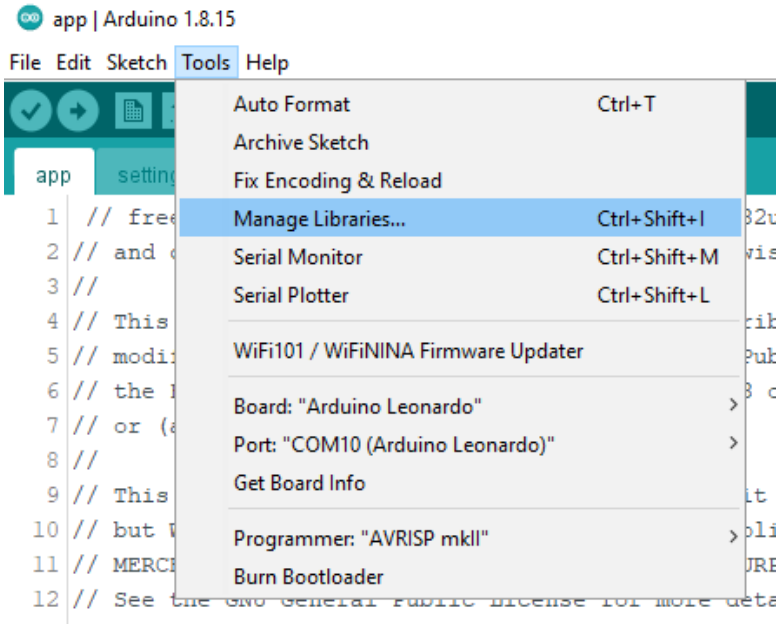

Figure 24: Arduino Library Manager

Now the “Library Manager” should be open. First, we need to download the library “SdFat”. Just type the name into the search bar and the package should appear, as shown in Figure 25. You need to install a Version “**1.1.x**”, a Version “2.x.x” won’t work with the code!

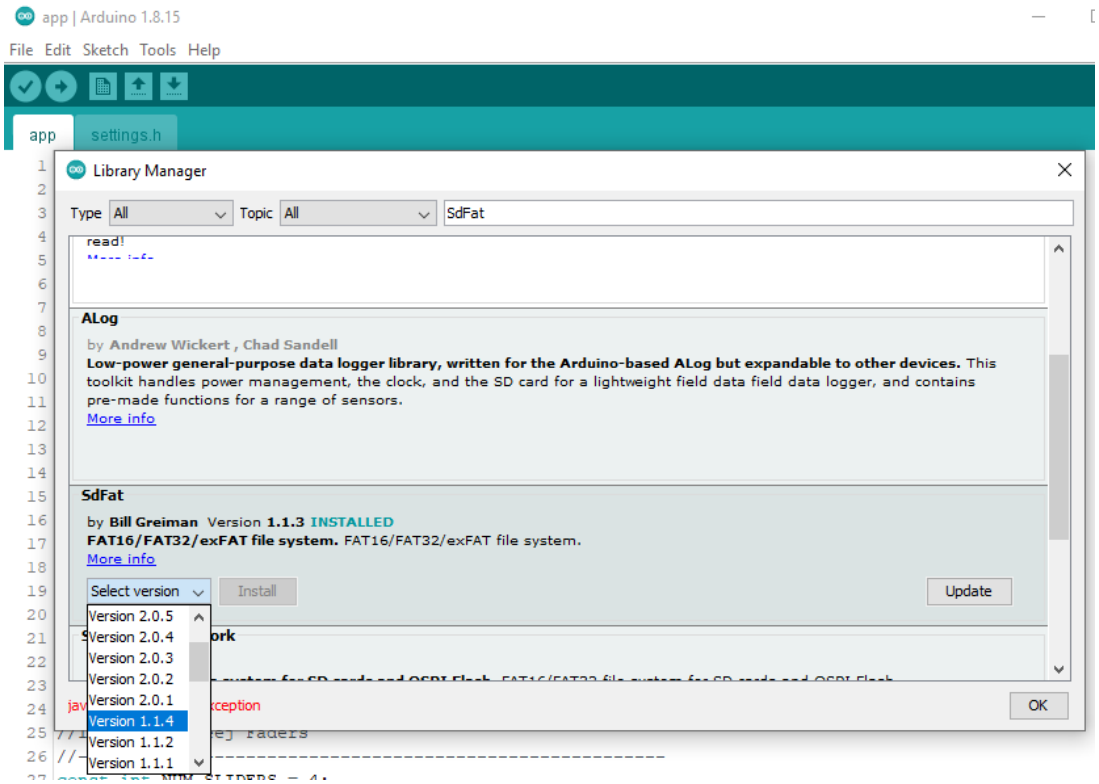

Figure 25: Installing SdFat library

The next package needed is the “HID-Project”. Just search for it in the Library Browser and it should appear as shown in Figure 26. You need to install a Version “**2.x.x**”.

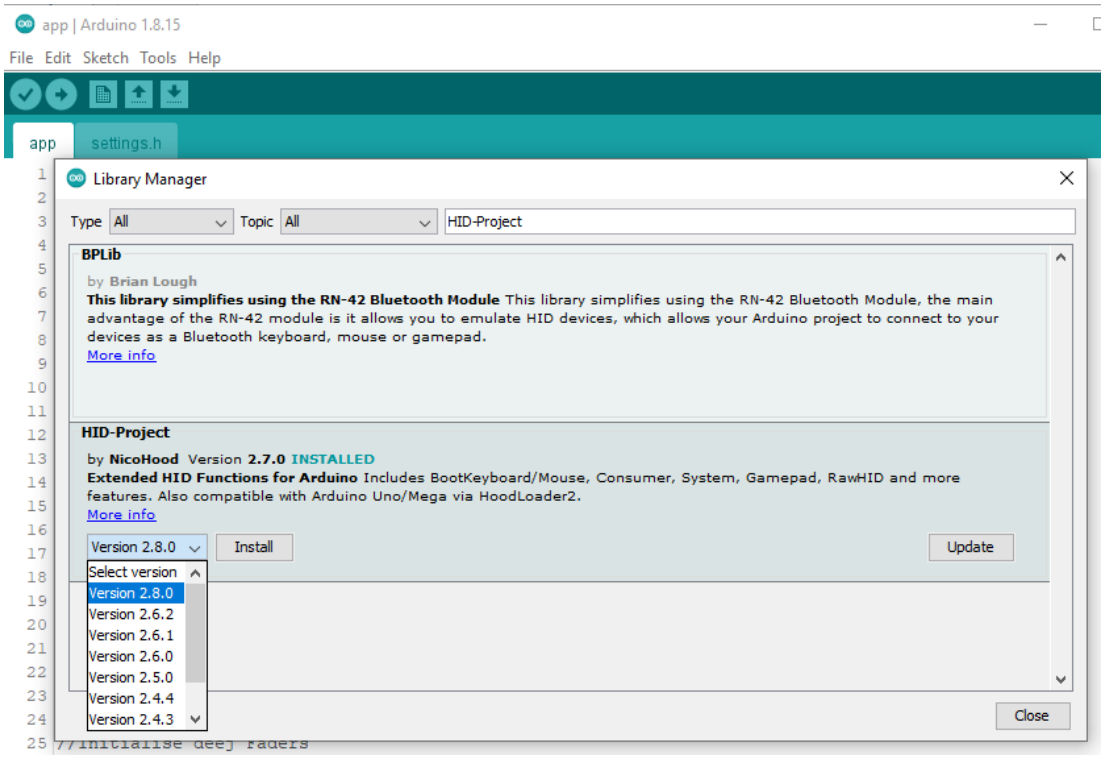

Figure 26: Installing HID-Project library

The last package you need to install is the “Encoder” library by “Paul Stoffregen” as shown in Figure 27. You need to search for “Rotary Encoder” to find it, just “Encoder” won’t work. The Version “**1.4.2**” works fine with the code, so I recommend downloading it.

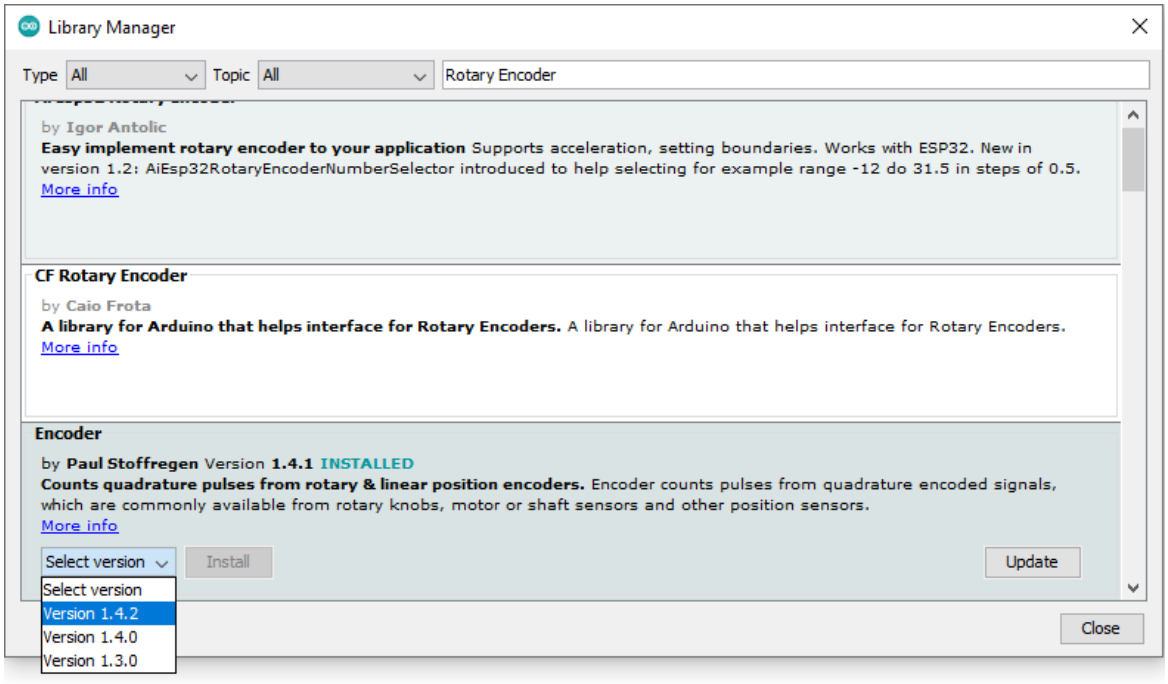

Figure 27: Installing Encoder library

3.2 Compiling and Flashing the Code

For this section you need the “Modified_Freedeck_Firmware” folder from the GitHub. Open “Modified_Freedeck_Firmware/app/app.ino” with Arduino IDE and the tabs “app” and “settings.h” should appear in the IDE.

To set the right board the code is compiled for, you need to click on “Tools”. Inside the dropdown menu you need to set the “Board:” to “Arduino Leonardo”, like shown in Figure 28.

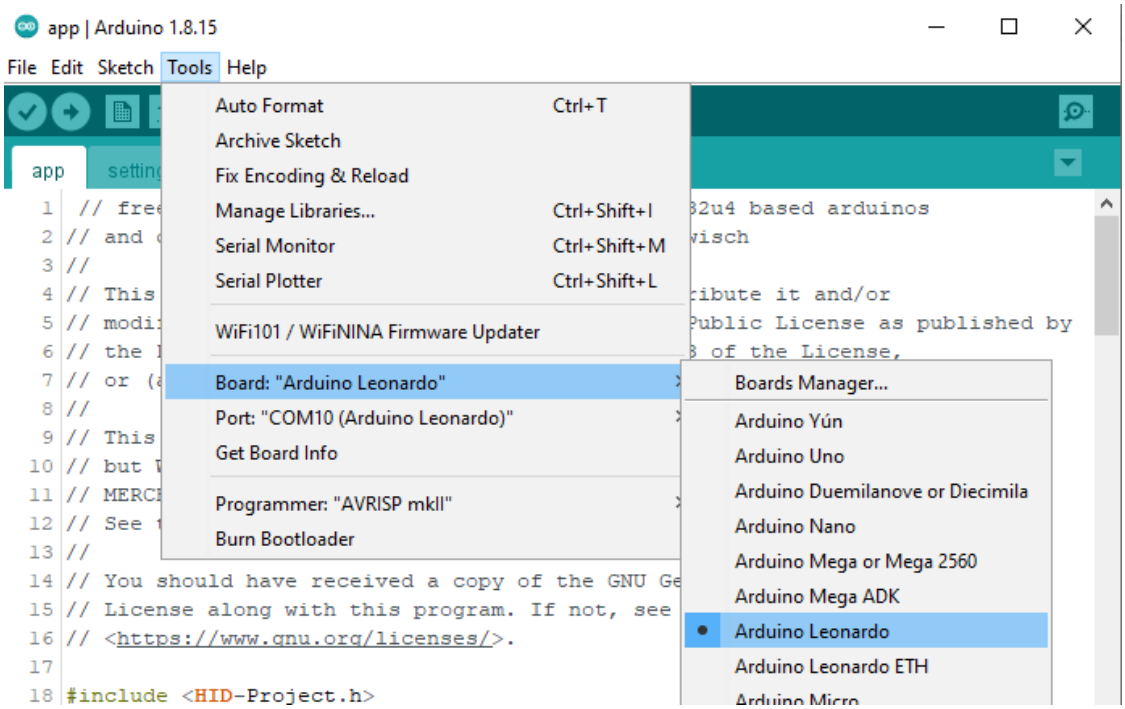

Figure 28: Arduino IDE Board settings

Next you need to connect your FreeDeck board to your PC and it should be selectable in “Port: “ on the dropdown menu of “Tools” . It should appear as “COMxx (Arduino Leonardo)”. When this is done you need to compile the code.

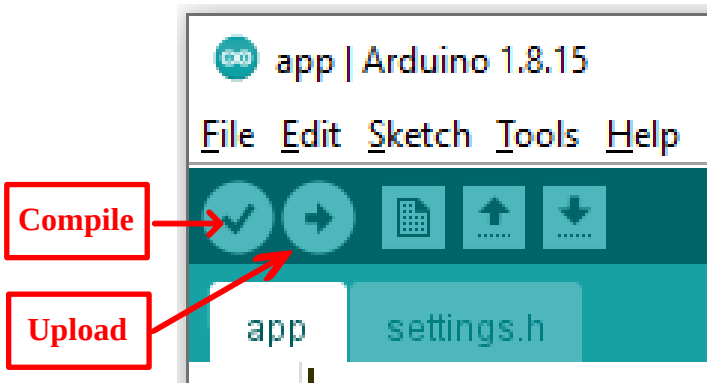

Figure 29: Arduino IDE compile and upload buttons

Just press on the compile button and wait. When something like *"Global variables use 1226 bytes (47%) of dynamic memory, leaving 1334 bytes for local variables. Maximum is 2560 bytes."* is the last thing shown in the console of the IDE, the code compiled successfully. If an error appears, check if the versions of your downloaded libraries are right.

Now you are ready to flash the code to your Freedeck PCB. Just press the "Upload" button and wait for the flashing process to end. When there is no Error message in the console of the IDE, your upload was successful and your screens should light up in a flickering white. This white flickering is due to the missing config. This will be fixed in the next step of this guide.

3.2.1 FreeDeck Example Config

For this step you need the "config.bin" from the folder "/FreeDeck_example_config" on the GitHub. There are two ways of getting your config to the SD-card. You can either put the config file directly on the SD-card with a SD card or send it via serial with the Freedeck configurator.

3.2.1.1 Via SD-Card Reader

This is the safest and easiest way to do it. Just make sure your SD-card is formatted as FAT32 and MBR. When this is done, put the "config.bin" on there without any folders. Don't change the name of the config. Now insert the SD-card into your FreeDeck and do a quick power-cycle.

3.2.1.2 Via FreeDeck Configurator

For this method you need to unplug your Freedeck first and then open the configurator with a chrome-based browser.

https://fdconfig.freeyourstream.com/

Press "Load Config" on the upper left corner and upload the "config.bin" to the configurator. The configurator should now show 3 prepared pages with buttons. Now plug your Freedeck back into the PC and open "Settings" on the Freedeck-Configurator. Go to the tab "Serial(Beta)" and press the connect button. The LED on your Freedeck indicates the serial connection and should now be turned on. Then close the settings menu in the configurator and click "Upload Config(Serial)". You now have to wait until the shown percentage reaches 100%. If the percentage shows weird numbers way above 100%, restart the upload.

3.2.1.3 Testing the Config

The Freedeck should now show the numbers from 1 to 8. A press of the Button should write the number. You can now test that every button and screen is working correctly. When you turn the rotary encoder, you should be able to switch through the pages of the config. The buttons should now change appearance and function. By pressing the rotary encoder, the screens should show the first page with the numbers again.

3.2.2 Code Adjustments

In this chapter I will explain which values you have to change in “settings.h” to fix possible misbehaviour of the FreeDeck. After changing the values in settings.h you need to reflash your FreeDeck.

3.2.2.1 Rotary Encoder

When the rotary encoder skips pages or needs more several clicks for a page change, change the value “EncStepPerClick”.

```
3 // Software stepps the encoder does per click:
4 #define EncStepPerClick 2
5 //change if to 1,2,4,6 or 8 if the Rotary Encoder skipps pages
```

Figure 30: Rotary Encoder settings

3.2.2.2 Boot Delay

If your FreeDeck has start-up issues or has problems to load the config, you can set a boot-delay. You can set the “BOOT_DELAY” to 1500 or higher.

```
41 // the delay to wait for everything to "boot"
42 // increase to 1500-1800 or higher if some displays dont
43 // startup right away
44 #define BOOT_DELAY 0
45 #define CONFIG_NAME "config.bin"
46 #define TEMP_FILE "config.bin.tmp"
47 #define MAX_CACHE 32
```

Figure 31: Boot Delay settings

3.2.2.3 Precharge Period

When you have issues with coil whine, you can adjust the value of “PRE_CHARGE_PERIOD”. Just test different values to and choose which is the best for your screens. The needed value is in HEX and has to be between 0x11 and 0xff. This value can also help with start-up issues of bad screens.

```
49 // Change this value from 0x11 up to 0xff to reduce coil whine. different
50 // from display to display
51 #define PRE_CHARGE_PERIOD 0xc0
```

Figure 32: Precharge Period settings

3.2.2.4 Minimum Brightness

Some cheap screens cant handle the lowest minimum brightness. If you got problems with your screen, try uncommenting a higher value for “MINIMUM_BRIGHTNESS”.

```
// Minimum Brightness value for displays. If your displays image quality gets
// worse at lower brighness choose a bigger value here
#define MINIMUM_BRIGHTNESS 0x00
// #define MINIMUM_BRIGHTNESS 0x00 //almost dark, good displays only
// #define MINIMUM_BRIGHTNESS 0x30 //brightest for cheap displays
```

Figure 33: Minimum Brightness settings

3.2.2.5 Refresh Frequency

Is your Screen flickering, try a lower number for “REFRESH_FREQUENCY”. The value needs to be written in HEX with a maximum of 0xff. I recommend trying values that are a multiple of decimal 64. (0x40, 0x80, 0xc0)

```
// if your screen is flickering, choose a lower number. the worse the screen,
// the lower the number.
#define REFRESH_FREQUENCY 0x80
// #define REFRESH_FREQUENCY 0xf1
// #define REFRESH_FREQUENCY 0xc1
// #define REFRESH_FREQUENCY 0x80
```

Figure 34: Refresh Frequency settings

4 Creating your own FreeDeck Config

In this chapter I will not show you all the details of the configurator, because there is already a detailed guide in the documentation of the FreeDeck. You can find it on following link:

https://github.com/FreeYourStream/freedeck-configurator

I will show you how to set up the settings for the FreeDeck-deej-combo config and to use the serial configurator functions. You need to open the FreeDeck configurator on following link with a chrome-based browser (like Google Chrome or Edge).

https://fdconfig.freeyourstream.com/

4.1 Set Basic Settings

To get the basic settings for this build you can either load the example config and used it as a basis for your config or you can follow the next steps to set the right settings. The example config can be found on the GitHub.

First you need to click on the settings button on the upper right corner.

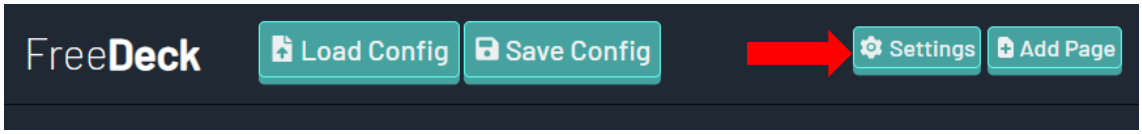

Figure 35: FreeDeck configurator settings Buttons

A new “General settings” window should open. You need to go to the tab “Device” and Set the width and height like shown in Figure 36.

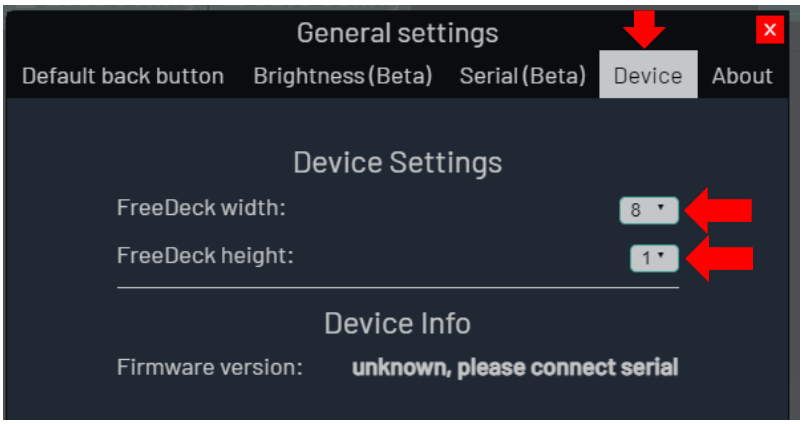

Figure 36: FreeDeck configurator settings window

When the settings are set, you can close the window and click on “Add Page”. A new empty page should appear like shown in Figure 37.

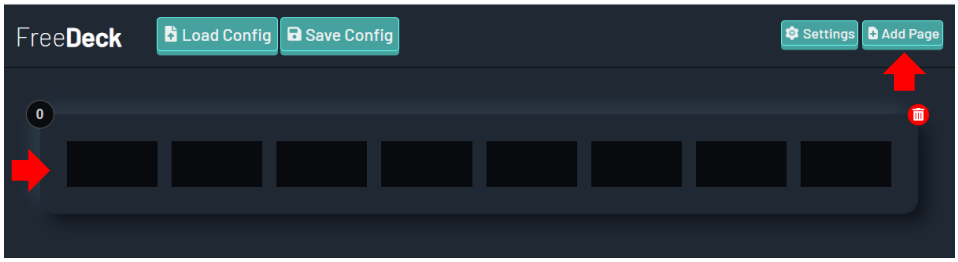

Figure 37: FreeDeck configurator new page

To edit the appearance and functions of the buttons you need to simply click on them. A more detailed guide on configuring the buttons is in the FreeDeck documentation.

4.2 Serial Functions of the Configurator

Like in chapter 3.2.1 already explained, you can transfer the config by downloading it and putting it directly on the SD-card or transfer it by using the serial functions of the configurator. In this chapter I will show you how to download and upload the config to your FreeDeck via Serial.

Attention: If you have already set up your faders and the "deej.exe" is running, you need to close it with the task-manager! Otherwise, the serial connection of the "deej.exe" will disturb the transfer of the config!

First you need to connect your FreeDeck to your PC. Then go to the FreeDeck configurator. On the first setup of the serial connection you need to open the settings and go to the "Serial(Beta)" tab. You need to click on the "Connect"-button like shown in Figure 38.

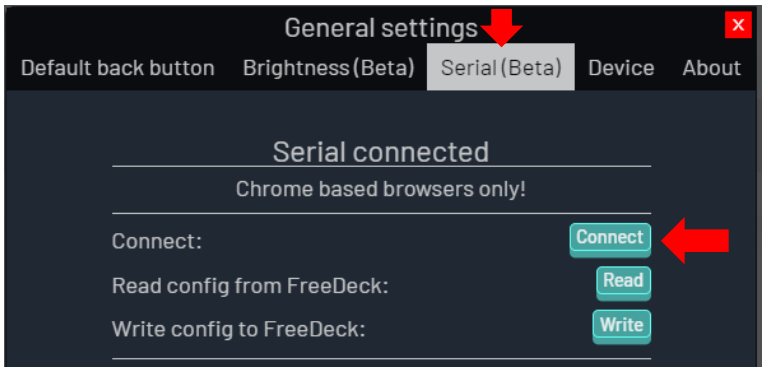

Figure 38: FreeDeck configurator serial settings

After that your "Load Config" and "Save Config" buttons should now have changed.

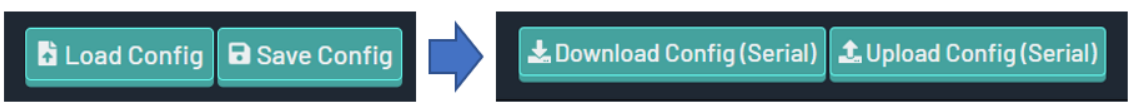

Figure 39: FreeDeck configurator changed buttons

Connecting through the settings should only be necessary on the first setup. If you want to load the existing config from your FreeDeck, just simply click on “Download Config(Serial)”. To upload the config click on “Upload Config(Serial)”. If the percentage shows weird numbers way above 100%, restart the download/upload.

While using the serial functions of the configurator the status-LED of the FreeDeck, should light up. This LED indicates, that the serial send function of the deej-faders is disabled. To enable them, after closing the FreeDeck configurator, just click on the Rotary Encoder Button and the LED should turn off.

5 Set up deej Faders

To set up the deej faders you need the “deej” folder from the GitHub. I recommend moving the deej folder into your programs folder.

5.1 Set up the deej Config

Inside the deej folder is the “config.yaml”. you can open it with any programming IDE you like. The windows text-editor works fine too. The first thing you need to adjust is the “com_port:”, like shown in Figure 40.

```
# settings for connecting to the arduino board
com_port: COM10
baud_rate: 9600
```

Figure 40: deej config com_port

Here you need to enter the assigned COM-port of the Arduino. There are two ways of finding out which COM-port you need to enter. You can go to your “Windows Device Manager” and look for “Ports (COM & LPT)”. The Arduino should appear as “Arduino Leonardo(COMxx)”. Just enter the shown COM-number to your config.

You can also open your Arduino IDE and look at Tools → Port → COMxx (Arduino Leonardo). Just enter the shown COM-number to your config.

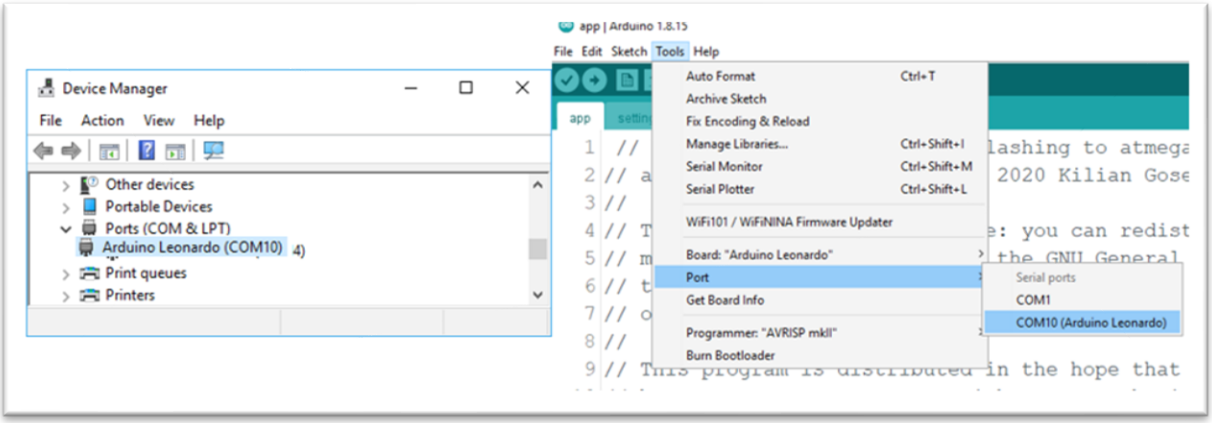

Figure 41: finding out COM-port of Arduino

Now you can start assigning the faders to programs/functions. This process is process is pretty self-explanatory. You just need to write the name of the programs .exe or the name of the function behind the number of the Fader you want. The only thing you need to look out for is the syntax, if you want to assign multiple programs with one fader. I recommend just copy and pasting the existing syntax and change the names of the programs.

```
slider_mapping:
  0: master
  1:
    - firefox.exe
    - chrome.exe
    - spotify.exe
  2:  deej.unmapped
  3:
    - discord.exe
    - ts3client_win64.exe
```

Figure 42: Example syntax for the fader mapping

On the top of the config file, you can find basic functions you can assign instead of programs. By entering “master” you can control the master volume. The function “mic” lets you control the output level of your in windows assigned standard microphone. The function “deej.unmapped” is pretty handy if you want to change the volume of games. It controls every program, that is not mapped on this config, so you don’t need to assign every game you play onto this list. I recommend playing around with these functions to find your perfect settings.

5.2 Starting deej

Attention: To start deej you need to close the FreeDeck configurator and the Arduino IDE. Otherwise, it will spit out a warning that the COM-port is busy. Make sure the Status LED on the FreeDeck is off. If it’s on, press the button of the rotary encoder.

Just start the deej.exe in the deej folder and you should be able to control your sounds. To check if everything is working correctly, open the “Windows Volume Mixer” and move the faders.

5.3 Putting deej on Windows Autostart

If you don’t want to start deej by hand, every time you want to use it, I recommend adding it to windows autostart. To do that, you simply create an shortcut of “deej.exe” by doing a right click on it. Then you just need to move this shortcut to the windows autostart folder. For that I created a shortcut to the standard location of this folder. Just click on “link_Windows_Autostart_folder” in the deej folder and put in the deej shortcut.

If this method didn’t work, just search for ”adding programs to windows autostart” on Youtube.

6 3D-Printing the Case

In this chapter I will give some advises for 3d printing the case for the FreeDeck-deej-combo. You will find all STL files for printing in the GitHub in the “Housing_STL_data” folder. To print this case a 3D-printer with a minimum build volume of 200x200x100mm is required.

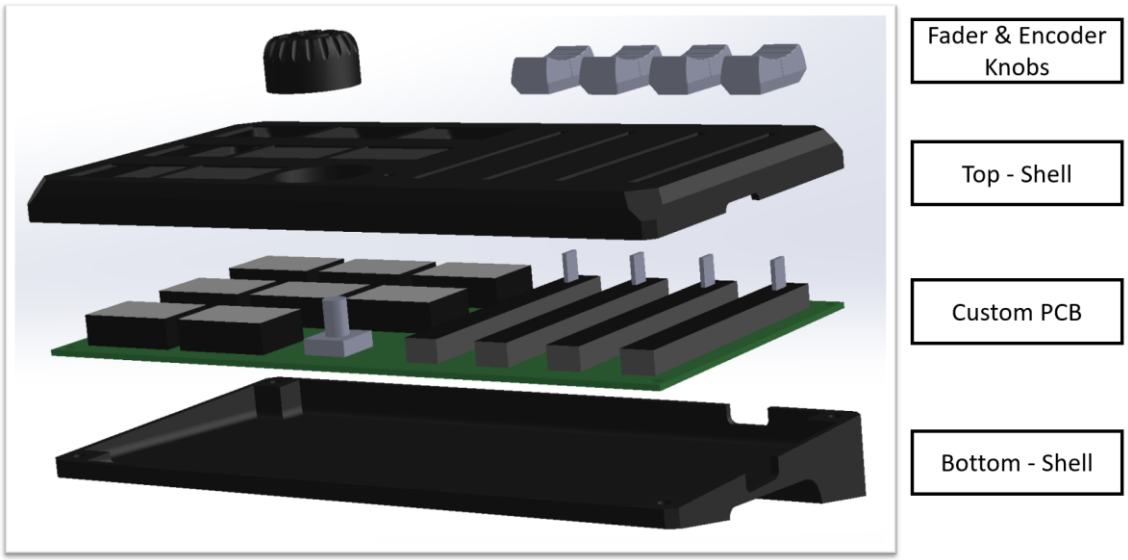

Figure 43: Exploded view of the Case

6.1 Top Shell

I recommend printing the top shell of the case face down, as shown in Figure 44. This way no supports are needed. If you got a textured PEI sheet as build plate, I highly recommend using it. The textured finish of the top surface will make the device look more refined.

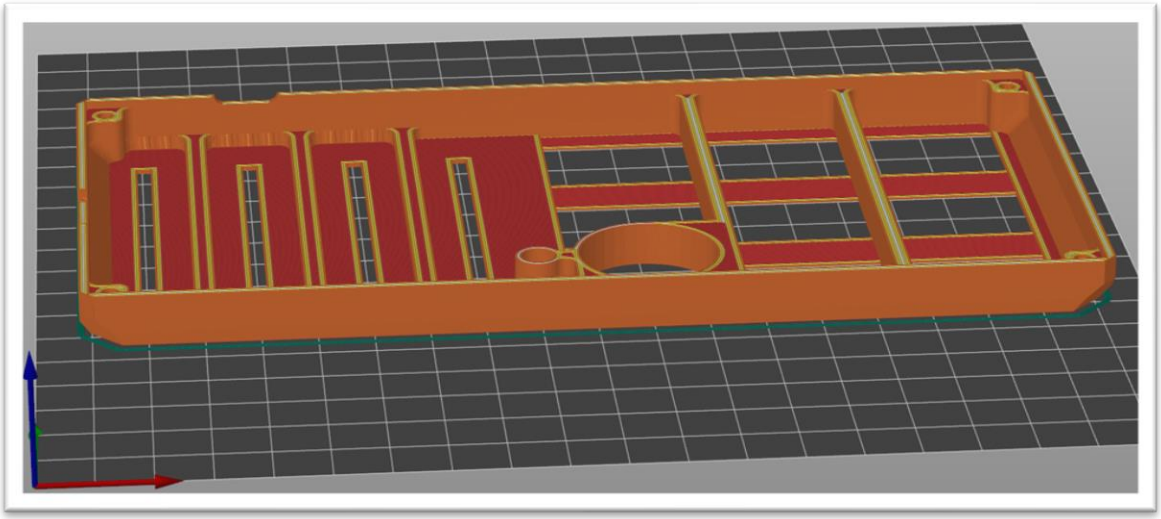

Figure 44: Case top shell in Prusa-Slicer

6.2 Bottom Shell

There are two ways of printing the bottom shell. The first way is printing it lying flat on the buildplate. This way the result will look better because the layer orientation matches the top shell. The downside are the Supports which are not so easy to remove. I highly recommend using Prusa-Slicer for this, because the supports can be ripped out with some pliers in one piece. I highly recommend a support spacing of 0.2mm or higher! The inner surface will look stringy because of the high spacing, but you will never see it again when assembled.

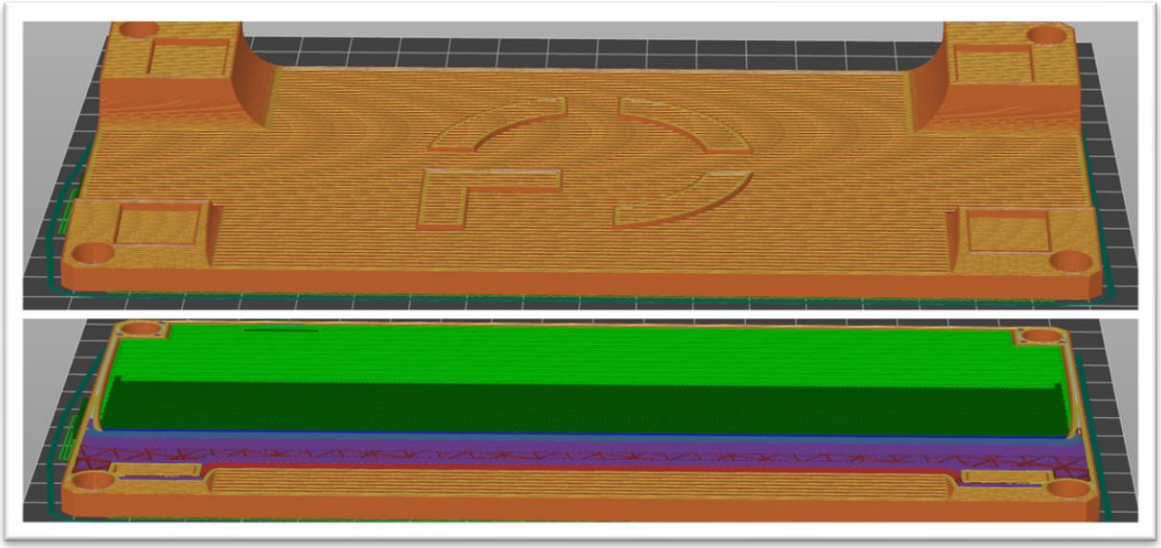

Figure 45: Case bottom shell in Prusa-Slicer lying

Another possible way is printing it standing, as shown in Figure 46. For this method I highly recommend using Cura, because of the tree support function. This is the easiest way of printing it, because the supports will come off nicely.

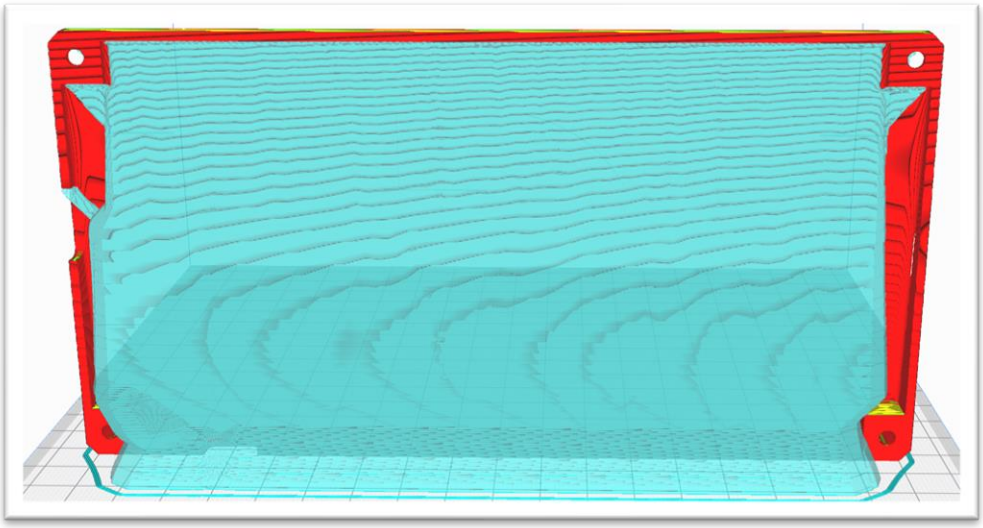

Figure 46: Case Bottom shell in Cura standing with tree supports

6.3 Rotary Encoder Knob

To print the knobs for the rotary encoder I recommend using Cura and printing it face up, as shown in Figure 47. This way the top surface will look better. To enhance the surface quality further you can activate the ironing function. The tree supports are the best choice on this part too, because they are easy to remove in the narrow space. I highly recommend using the “change filament at layer hight” function to print the FreeDeck logo in a different colour.

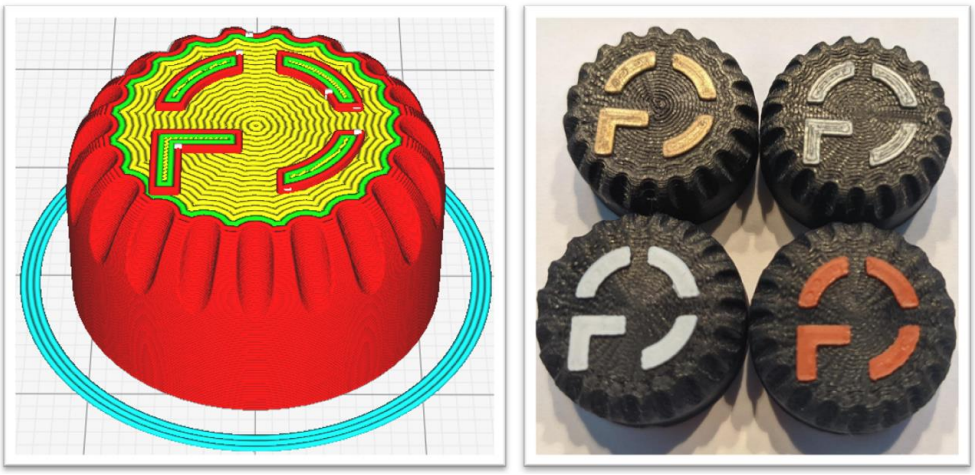

Figure 47: Rotary encoder knob Cura & printed

6.4 Fader Knobs

In the GitHub you will find three STL-files for the Fader knob. Each Version has different width for the fader shaft. I highly recommend printing one of each version to evaluate which one works best with the tolerances of your printer. You don’t want to break your board, because you used too much force pushing the knobs on. Likewise, you don’t want the knobs to wiggle around while moving them.

I recommend printing them in the same colour as the logo on the rotary encoder knob.

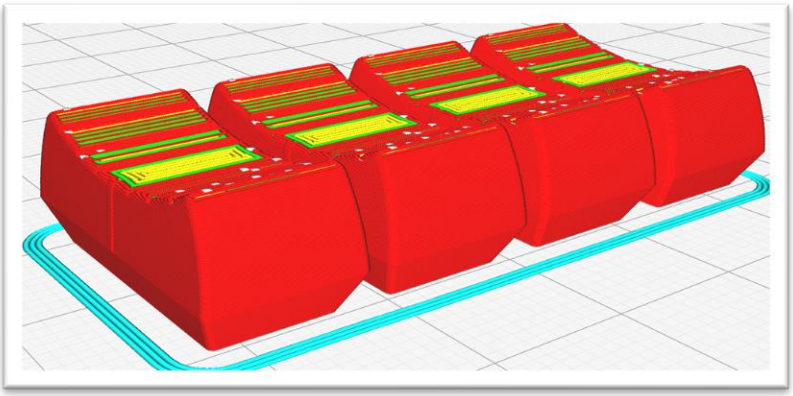

Figure 48: Fader knobs in Cura