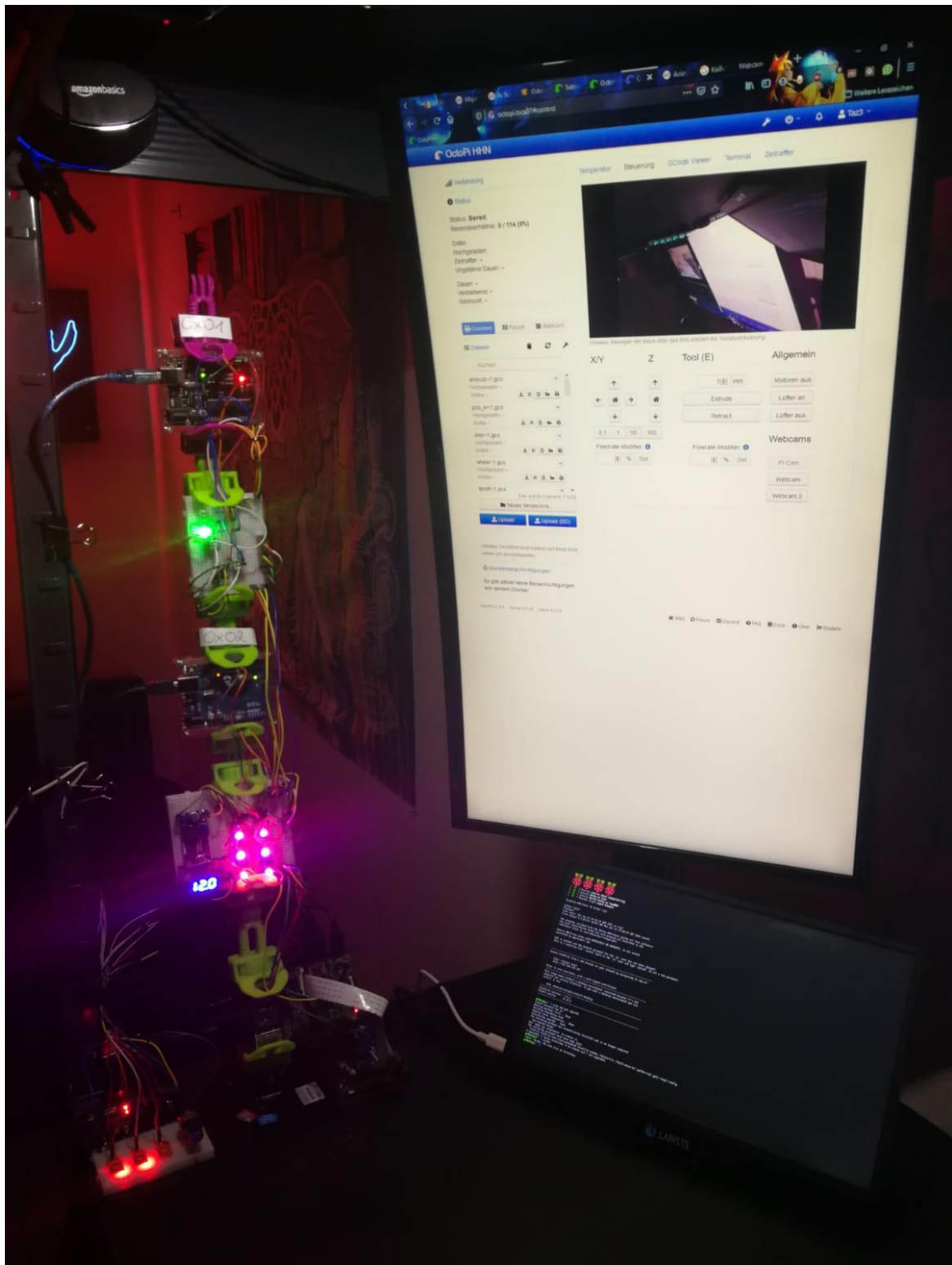


3D Drucker und Servertuning



Embedded Systems WS2020/21

Von:

Marc Nauendorf 200882

INHALTSVERZEICHNIS

EINLEITUNG – DIE VISION.....	2
PROJEKTAUFBAU	3-4
VERWENDETE HARDWARE	5-6
VERWENDETE SOFTWARE	7
PROJEKTVERLAUF (TEST 1)	8
PROJEKTVERLAUF (TEST 2)	9
PROJEKTVERLAUF (TEST 3)	10-13
PROTOTYP 1	14-15
PROBLEM VOM PROTOTYP 1	16
MÖGLICHE „OFFENE AUFGABEN“ + EXPERIMENTELLE AUFGABE	17
DESIGN DER SERVERHÜLLE.....	18-19
DESIGN DES PROTOTYPS 1	20
DRUCKERREPERATUR/ TUNING VOM TAZ 3.....	21-24
NACHWORT.....	24

Einleitung – Die Vision

Das Projekt „3D Drucker und Servertuning“ soll die bisherigen Funktionen des bestehenden Systems ausweiten. Dies beinhaltet das benutzten von min. 2 Druckern und min. 2 Kameras und eine Lichtersteuerung. Zusätzlich sollen einfach zugängliche Schnittstellen geschaffen werden, um das System spielerisch erweitern zu können.

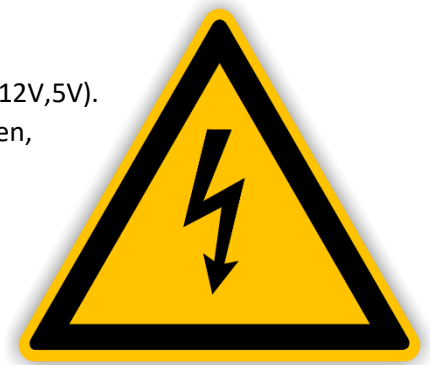
Studenten die SV1 bestanden haben und Tutorials im „Arduino Bereich“ durchgeführt haben, sollen in der Lage sein, Erweiterungen durchzuführen.

Der Mehrwert fokussiert sich primär auf den Studenten, der sich „ausprobieren“ möchte bzw. den 3D Druck kennenlernen möchte. Die effiziente Nutzung des Raspberry Pi spielt eine zentrale, aber untergeordnete Rolle.

Zudem soll das Design der Optik den Studenten neugierig machen, somit muss die „Erweiterung“ nicht nur eingebettet werden, sondern auch mit diesem arbeiten/interagieren können.

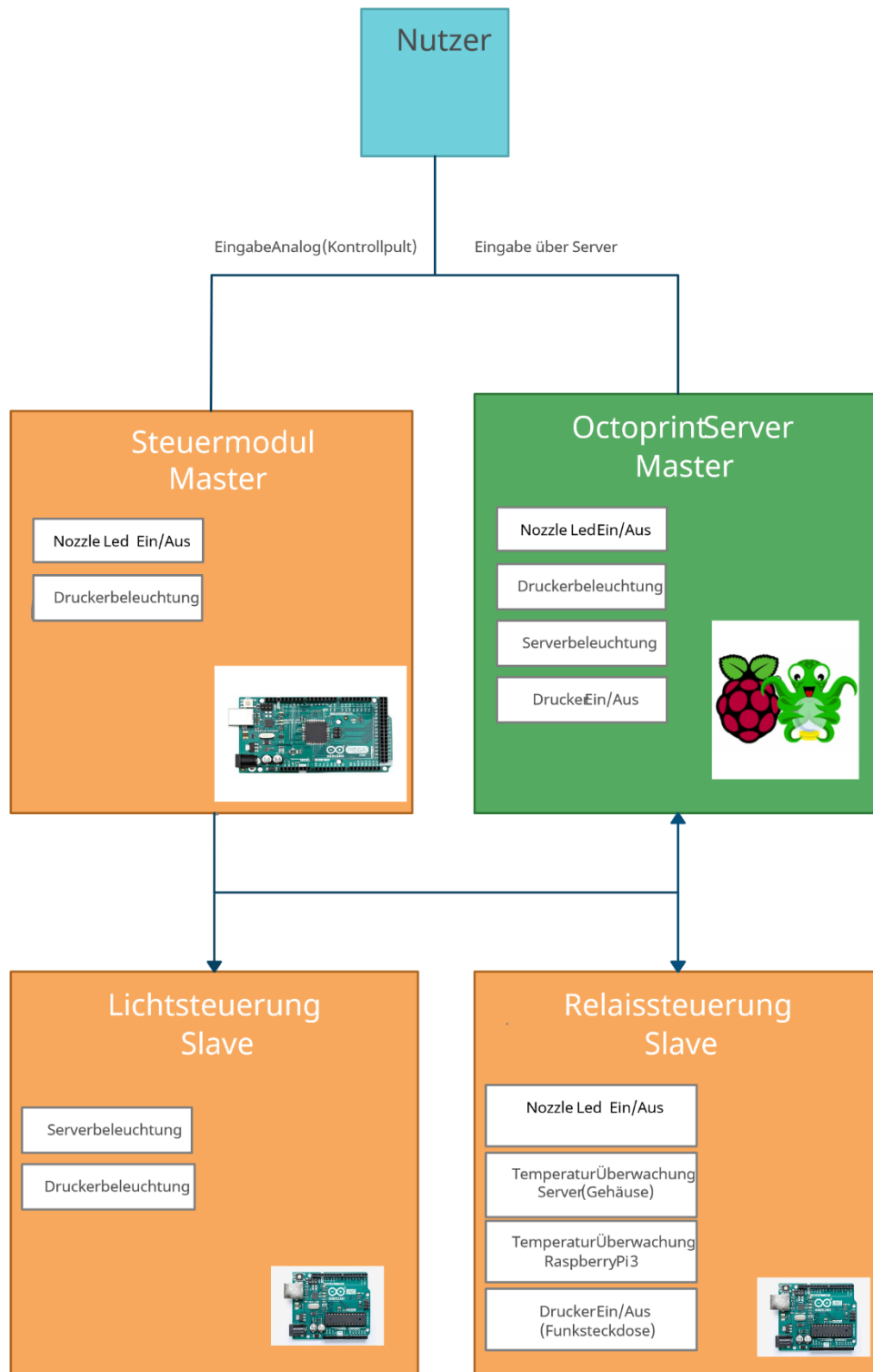
Warnhinweis

In diesem Projekt wird mit 3 verschiedenen Spannungen gearbeitet (24V,12V,5V). Somit muss sorgfältig mit metallischen Gegenständen umgegangen werden, um sich selbst und die Hardware nicht zu beschädigen. Mit offenen Verbindungen also bitte mit Vorsicht interagieren.



Projektaufbau

Der Prototyp setzt sich aus 2 Master und 2 Slaves zusammen. Der Kerngedanke war es die Lichtsteuerung und die Kontrolle einer Funksteckdose nicht auf einem Pi, sondern auf mehreren Arduinos laufen zu lassen. Welche Befehle von dem Octopi Server (Raspberry Pi) und über ein Eingabepult (Schalter und Poti) erhalten sollen. Zudem soll der Octopi dem Nutzer mitteilen können, ob seine gewünschte Eingabe umgesetzt wurde. Diese Form des Feedbacks ist für das Kontrollpult nicht vorgesehen.



Das gezeigte Bild soll eine Übersicht über die Funktionen der „Erweiterung“ bieten, Octopi Internes wird an anderer Stelle erläutert. Die Pfeile stellen das Erhalten von „Daten“ dar. Somit wird ersichtlich das, dass Eingabepult nur zum Senden von Befehlen benutzt wird.

Der Octoprint Server soll stattdessen Senden und Empfangen können. Zum Stand (27.1.2021) ist das Empfangen noch nicht umgesetzt worden.

Bei den „Slaves“ können wir uns die Funktionen anschauen, die sie jeweils bieten. Hier fällt auf das die „Nozzle LED“ nicht mit der Lichtsteuerung gesteuert wird, sondern mit der Relaisteuerung interagiert. Dies ist so da es sich um eine 12V „Hochleistungs Led“ handelt. Diese muss über ein Relais geschaltet werden.

Zudem werden in der Relaissteuerung, 2 verschiedene Temperaturen ausgelesen und verarbeitet. Um das Gehäuse/Bauteile zu schützen schalten diese Temperatursensoren, den benötigten Lüfter ein/aus.

In der Lichtersteuerung sollen nur die Beleuchtungsfarbe gesteuert werden (Stand 27.1.2021). Druckerbeleuchtung und Serverbeleuchtung ist getrennt. Denn der Server soll mit seinem „Farbspiel“ Informationen an den Nutzer übermitteln können. Dies wird im Kapitel „Design“ genauer aufgeführt.

Verwendete Hardware

Es wurde mit 3 verschiedenen Pi's gearbeitet. Für „Performance Tests“ welche sich auf den Octoprint Server beziehen. D.h. Octoprint wurde auf einem Pi 2b, Pi 3b+ und Zero WH getestet und verglichen.



Raspberry Pi 2b

Raspberry Pi 3b+



Raspberry Pi Zero WH

Für die „Auswahl“ an Arduinos gab es keinen Speziellen Grund. Da ich mit dem gearbeitet habe was ich „rumliegen“ hatte.



2x Arduino Uno, diese zwei sind unsere Slave Steuergeräte.



1x Arduino Mega der in unseren Tests das Steuermodul darstellt.



Der Touch Sensor **TTP223**, ist die Eingabemöglichkeit für das Kontrollpult.



Im Testaufbau wurde ein **Boost Buck Converter** verbaut, doch im „fertigen Modell“ werden **Drop down Converter** verbaut. Da in unserem Testumfeld die Basis Spannung 5V ist und im realen Aufbau 24V.

Diese werden mit maximal 3A Output abgegeben. Zudem müssen sie mit einem Kühlkörper und Lüfter ausgestattet werden.



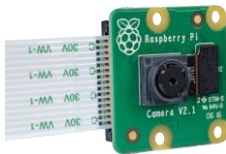
Die ausgehenden Spannungen werden durch **LCD Displays** visualisiert. Im Testaufbau als auch im fertigen Modell.



Die **WS2811 Led-Leiste** wird mit 12V betrieben. Der 3. Pin ist die Datenleitung, doch nur Led „**3er Gruppen**“ können jeweils angesteuert werden.



Das **ISM-Sender-Modul (433 MHz)**, wird zum Steuern einer Funksteckdose verwendet. Diese Schaltet den Drucker ein. Das „Zusatzsystem“ hängt zusätzlich am Netzteil (24V) des Druckers.



Als Kamera dient uns die **Raspi-Cam V2.1**. Mit der 1.3er Version wurden auch Tests durchgeführt, doch die V2.1 wird im fertigen Projekt verbaut.

Sie besitzt eine Auflösung von **8MP** und ist für Videoaufnahmen von 720p mit 60fps ausgelegt.

Widerstände, 2 Pin NTC Sensoren, Led's und ein Potentiometer wurden auch verwendet. Außerdem wird das fertige Projekt mit Kabeln mit einer Isolierung aus Silikon verkabelt. Diese sind flexibler als Kabel mit herkömmlicher Isolierung. Die sollen es ermöglichen Module aus ihrer Halterung zu entfernen ohne das Kabel aus ihren Pins „rutschen“ etc.

Sonstiges Material

Filament Pla (1.75mm): Für das Gehäuse wurde eine Konstruktion 3D gedruckt. + Teile für den Drucker Taz 3.

Software

Um die Arduinos zu programmieren wurde die **Arduino IDE 1.8** verwendet.

Die Benutzten Bibliotheken:

"**FastLED.h**": beinhaltet das Steuern der WS2811 Led's.

<**Wire.h**>: beinhaltet das Verbinden mit der I²C Schnittstelle.

<**RCSwitch.h**>: enthält das Ein/Aus-schalten über den 433Mhz Sender.

Auf den Raspberry's wird folgendes benutzt:

Open Source Software „**Octoprint 0.18.0**“: Dies ist der 3D Druck Server

Plugin **MultiCam**: Zum Benutzen von mehr als einer Kamera.

Plugin **System Command Editor**: Zum Aufrufen von Python Dateien über
Sudo python /home/pi/befehle/„Name der Datei“.py .

Im Python (3.7) Code der „Befehle“:

Import Smbus: Für das verbinden zur I²C Schnittstelle. Zudem muss die I²C Verbindung im raspi-config Menü erlaubt werden.

Zusätzlich:

Software PuTTY: Schafft eine Verbindung vom Windows Rechner zum Pi über SSH.
Diese Verbindung muss auch in der raspi-config freigeschaltet werden.

Software Fusion360: Zum designen der 3D Druck Teile.

Software Simplify 3D: „Slicer“ Software für 3D Druck Teile (erstellt einen G-Code).

Software frizing: Für das Erstellen der Schaltskizzen.

Software balenaEtcher: Notwendig für die Formatierung der „Boot SD-Karte“.

Software Paint: Essenziell für die „Bildbearbeitung“.

Projektverlauf

Im Vorfeld wurde gelernt einen NTC Sensor per Spannungsteiler über einen Arduino auszuwerten. Zudem noch das Ansteuern der WS2811 Led's in verschiedenen Modi.



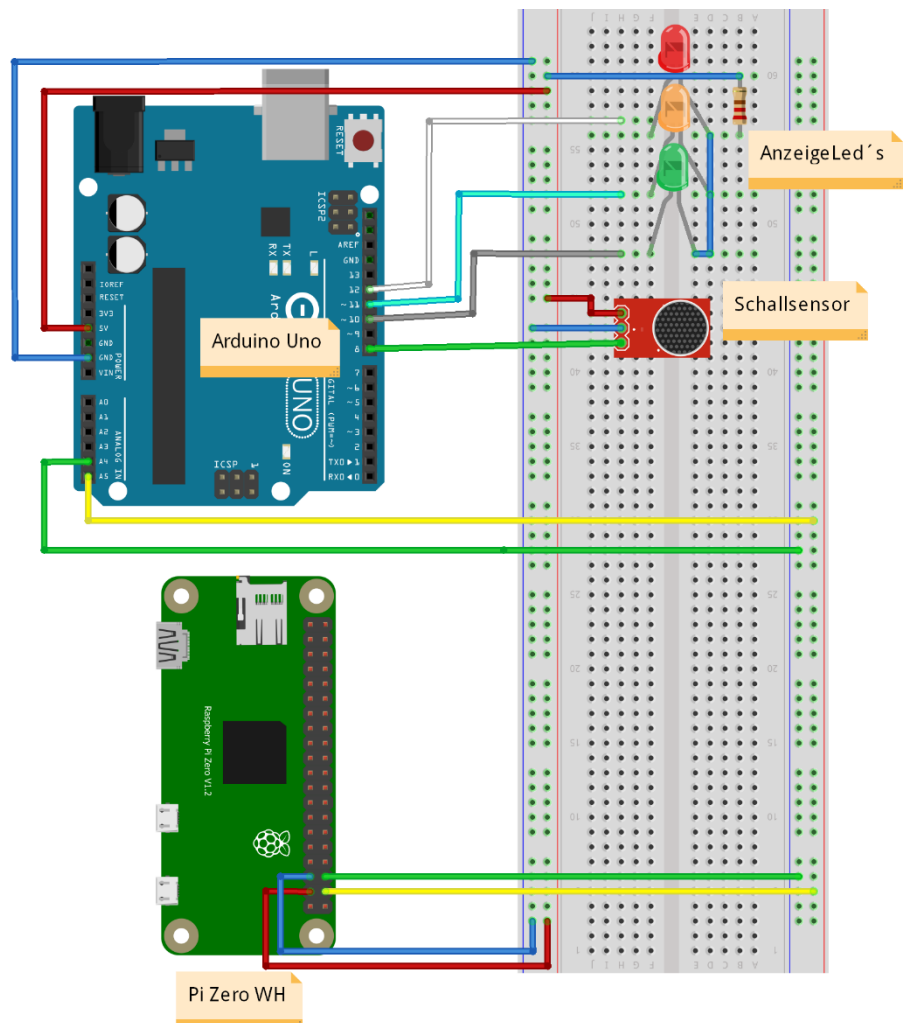
Test 1: I²C Verbindung von einem Arduino zu einem Pi.

Auf dem Pi wurde eine Raspbian Distribution aufgespielt mit Python 3.7. Dieser Pi ist mit dem Arduino über die Pins (SDA, SDA) +Usb verbunden.

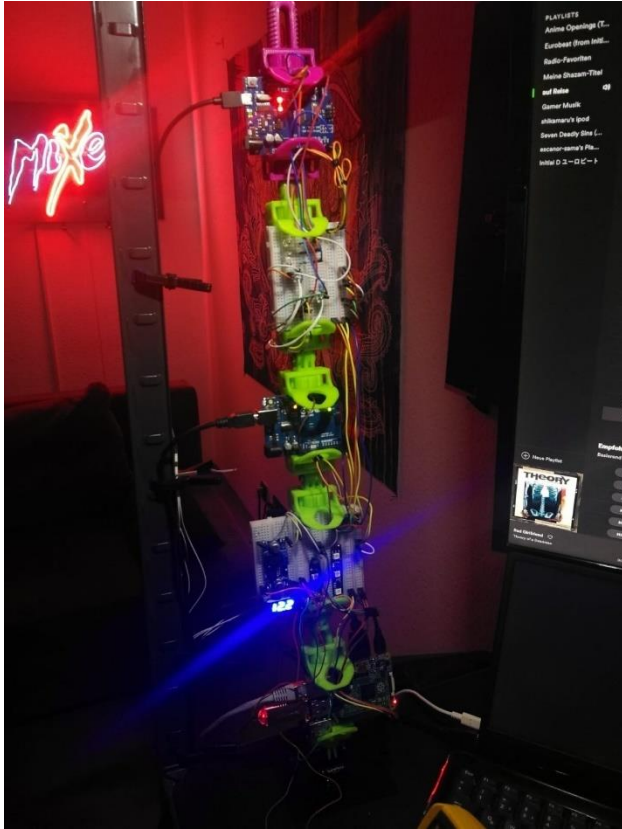
Zudem ist ihre Spannungsversorgung und Masse (-) parallelgeschaltet.

Hier wurde ein Schallsensor vom Arduino ausgewertet und dieses Signal wurde zum Pi geschickt. Dieser schickt beim Eintreffen der Nachricht dem Arduino einen Zahlenwert zurück.

Dieser löst eine Schaltsequenz der Leds aus, welche vom Arduino gesteuert wird.



Dies ist der Aufbau anders dargestellt. Dieses „Experiment“ diente nur als Übung für die I²C Schnittstelle. Zudem wurde versucht die „Serial.prints“ vom Arduino im Pi zu verwenden.



Test 2: Module Konstruieren

Hier haben wir denselben Aufbau wie bei „Test 1“, der Unterschied ist das die geplanten Module den Platz eingenommen haben. Dies beinhaltet einen Arduino Uno der die Lichtversorgung übernimmt und einen welcher die Relais/Hülle steuert.

Eine kleine Besonderheit stellt unser 2. Stromkreis dar. Im Bild zu sehen mit blauem Display. Hier handelt sich um den vorhin erwähnten „Boost Buck Converter“ welcher in unserm Versuchsaufbau die Spannung in einem separaten Rahmen auf 12V erhöht.

So können wir die WS2811 Led's schon jetzt benutzen.

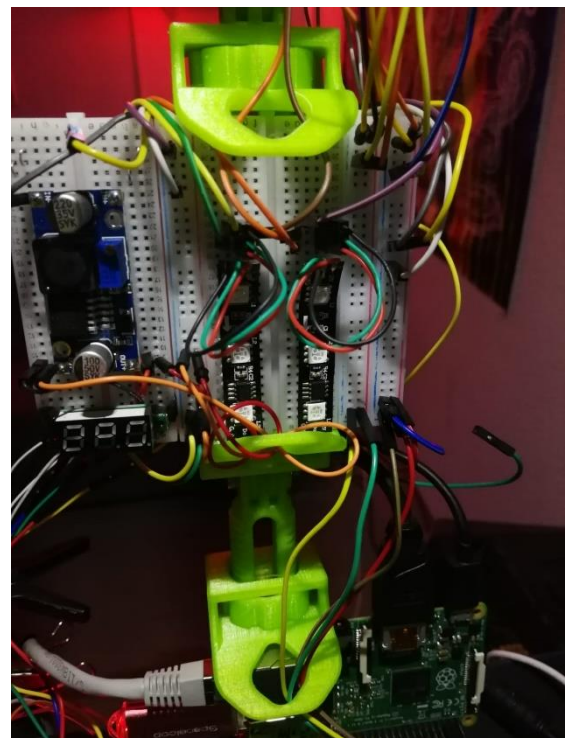
Für den Pi gelten dieselben Voraussetzungen wie im Test 1. Wir führen mit Python ein Script aus, welches einen Zahlenwert an die Adresse „0x01“ oder „0x02“ schickt. Der adressierte Zahlenwert wird vom Jeweiligen Arduino empfangen und verarbeitet.

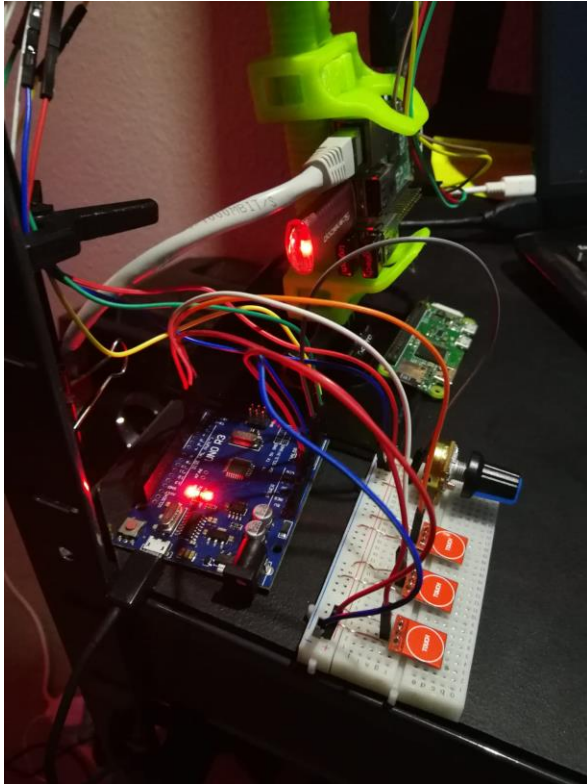
Hier sind die Arduino Sketche angepasst worden. Mit einem „Switch Case“ wird der erhaltende Wert interpretiert und ausgeführt.

Im Falle des Lichtmoduls werden verschiedene Farben ausgeführt.

Eine Funksteckdose und ein Relais sind im Relaismodul ansteuerbar.

Die Schaltskizze hierzu findet man bei „Prototyp 1“.





Test 3: Eingabemöglichkeiten

Der Wunsch nach einer physischen Steuerungseinheit kam nur, um mehrere Farben schalten zu können ohne Octoprint mit Buttons zu „überladen“. Zudem wäre es geschickt die „Nozzleleuchte“ (Led an der Düse) direkt am Drucker, einschalten zu können.

Wenn der TTP223 neben dem Potentiometer ausgelöst wird, wird das Potentiometer ausgelesen und seine 1024 Zustände in 15 unterteilt. Dies machen wir da 15 Farbzustände in der Lichtsteuerung vorhanden sind.

Nach jedem Auslesen wird dieser Wert an die Adresse „0x02“ geschickt. Dieser Prozess stoppt nach dem „Berühren“ des Touch Sensors.

Die zwei anderen Sensoren schalten die Funksteckdose und ein Relais Ein/Aus. Diese Sensoren wurden präpariert damit sie Stellschalter bzw. „Flipflops“ werden. Denn sie senden permanent entweder 1 oder 0. Der „Umbau“ wird um folgendem Absatz erläutert.

Sensor TTP223

Dieser Sensor besitzt 3 Pins (GND, Data, VCC). Diese Bezeichnungen findet ihr auch auf der Rückseite der Platine. Wenn ihr VCC und I/O mit etwas Lötzinn verbindet, wird aus dem Tastschalter ein Stellschalter (Im Bild der schwarze Kasten).




Test 4: Octopi aufsetzen und über I²C kommunizieren lassen

Zuerst wird die Software Octoprint auf eine SD-Karte (Max.32Gb) gebrannt. Danach greifen wir auf die Dateien: „config“, „octopi“ und „octopi-wpa-suppicant“ zu.

In dem File config kommentieren wir diese Zeile ein.



```
# Uncomment some or all of these to enable the optional hardware interfaces
dtparam=i2c_arm=on
#dtparam=i2s=on
#dtparam=spi=on
```



So schalten wir die I²C Schnittstelle ein.

Im Dokument Octopi nehmen wir folgende Änderung vor:

```
"
# Available options are:
# - auto: tries first usb webcam, if that's not available tries raspi cam
# - usb: only tries usb webcam
# - raspi: only tries raspi cam
#
# Defaults to auto
#
camera="raspi"
### Additional options to supply to MJPG Streamer for the USB camera
#
# See https://faq.octoprint.org/mjpg-streamer-config for avail
#
# Defaults to 10fps
#
camera_raspi_options="-fps 10"
```




Wir müssen hier die „raspi“ Cam eintragen. Zusätzlich schalten wir die fps ein, diese kann auch später je nach Version der Kamera angepasst werden. Jetzt erkennt Octoprint die Kamera.

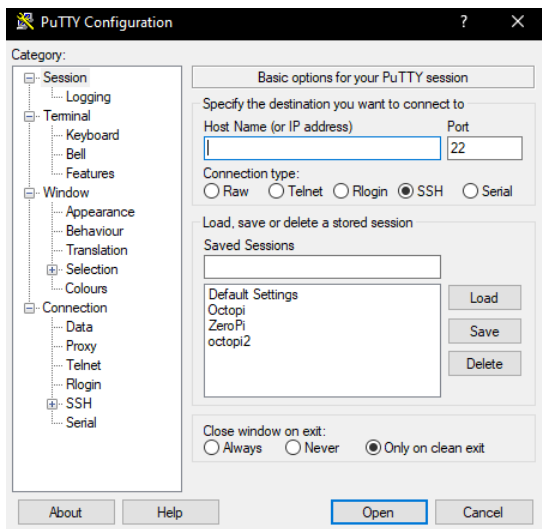
Spezial: Pi Zero

Dieser besitzt keinen verbauten Internetanschluss. Somit müssen wir auf der SD-Karte vom Pi Zero zusätzlich in die Datei „octopi-wpa-suppicant“. Hier lassen sich die Wlan-Einstellungen vornehmen.

Jetzt die SD-Karte in den Pi und Internetanschlüsse falls möglich an das Gerät. Jetzt soll er booten.

```
#
# If you use Textedit to edit this file make sure to use "plain text format"
# and "disable smart quotes" in "Textedit > Preferences", otherwise Textedit
# will use none-compatible characters and your network configuration won't
# work!
## WPA/WPA2 secured
#network={
#  ssid="put SSID here"
#  psk="put password here"
#}
## Open/unsecured
#network={
#  ssid="put SSID here"
#  key_mgmt=NONE
#}
```





Danach verbinden wir uns mit PuTTY mit dem Pi/Pi's.

Die Verbindungs-Ip wird euch nach dem Bootvorgang auf dem Pi auf einem Monitor angezeigt.

Jetzt können wir mit dem Windows Rechner auf die Konsole des Pi's zugreifen.

➔ Wenn keine direkte Adresse vermerkt ist auf der Octoprint laufen soll, sucht sich der PI eine freie im Netzwerk. Dies sorgt dafür, dass wenn ein zweiter Pi mit Octoprint läuft, 2 Server im selben Netzwerk laufen. Klingt logisch, doch der PI Zero sucht sich keine (wenn er als letztes gebootet wird zum ersten Mal). Das umgeht man, wenn man eine Adresse manuell vergibt, doch nett zu wissen ist das schon.

Jetzt schließen wir unsere Vorkonfiguration mit den diesen Befehlen in der Pi Konsole ab:

Sudo apt-get update

Sudo apt-get upgrades

Sudo reboot now

Nach dem Reboot können wir mit der Ip (wird angezeigt) oder der Adresse <http://octopi.local> Octoprint aufrufen (nur der „erste Server“ kann mit <http://octopi.local> aufgerufen werden). Hier kommen jetzt weiter Konfigurationen, die ich nicht weiter erläutere, dort geht es um die Accounterstellung.

Befehle.py

Nun greifen wir mit PuTTY auf die Konsole des Pi's zu.

Der Pi braucht jetzt ein eigenes Verzeichnis, indem wir den Python-Code aus dem 1 Test hinterlegen/einbetten können.

mkdir /home/pi/Befehle	neuen Ordner anlegen (Befehle)
cd /home/pi/Befehle	Wechsel zum Ordner Befehle
touch DruckerLichtBlau.py	leere Datei mit dem Namen „“ anlegen
Sudo nano DruckerLichtBlau.py	Diese Datei öffnen

Dieser Code sendet eine 4 an die Adresse 0x02. D.h. an Beleuchtungsmodul und 4te Farbe.

Dies machen wir mit allen Befehlen, die wir über den Pi ausführen wollen. Das Verzeichnis kann man sich mit dem Befehl „ls“ Anzeigen lassen.

```

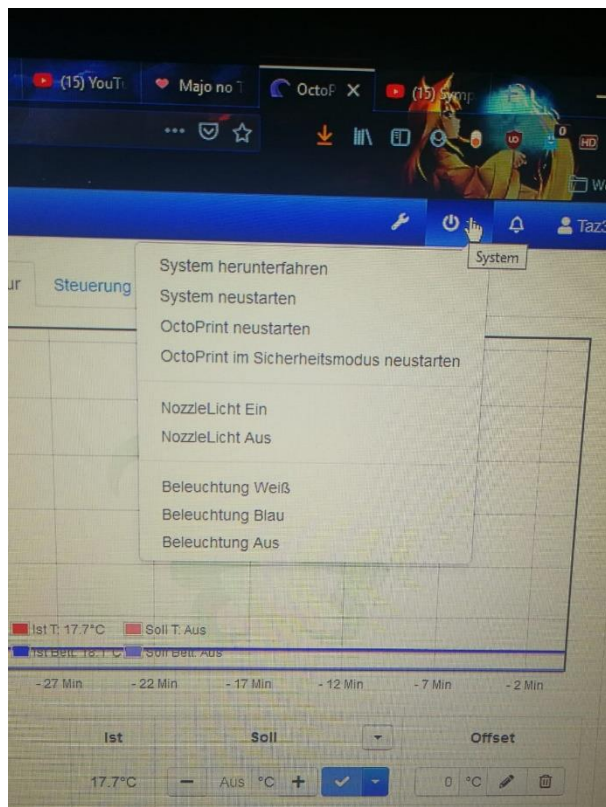
pi@octopi: ~/befehle
GNU nano 3.2 DruckerLichtBlau.py Modified
import smbus
addr = 0x02
bus=smbus.SMBus(1)
bus.write_byte(addr,4)

```

```

pi@octopi:~/befehle $ ls
DruckerAus.py  DruckerLichtAus.py  DruckerLichtPink.py  nozzleLichtAus.py
DruckerEin.py  DruckerLichtBlau.py  DruckerLichtWeiss.py  nozzleLichtEin.py
pi@octopi:~/befehle $

```



System Command Editor

Jetzt können wir diese angelegten Befehle über die Octoprint Maske aufrufen. Indem wir den jeweiligen Button folgendes ausführen lassen:

Sudo python /home/pi/befehle/"Name des Befehls".py

Anmerkung: Im Normalfall hat der Octoprint Account nicht die Berechtigung Sudo Befehle auszuführen. Diese muss man ihm erst über die Maske von Octoprint erteilen.

Das Entfernen von Passwörtern u.ä. über die Pi Konsole hat nicht geholfen.

Jetzt hat dieser Aufbau „Jede“ mögliche Berechtigung, dies muss noch vor dem Fertigstellen, abgeändert werden. Dies diente nur zu Testzwecken.

Es wurde sich auch mit dem Plugin MultiCam und der Möglichkeit einen zweiten Drucker an den Pi anzuschließen befasst. Doch durch mangelnden Erfolg verdient dies nur eine Erwähnung. **Doch 2 Drucker und 2 Kameras zu benutzen steht auf der Aufgabenliste.**

Prototyp 1

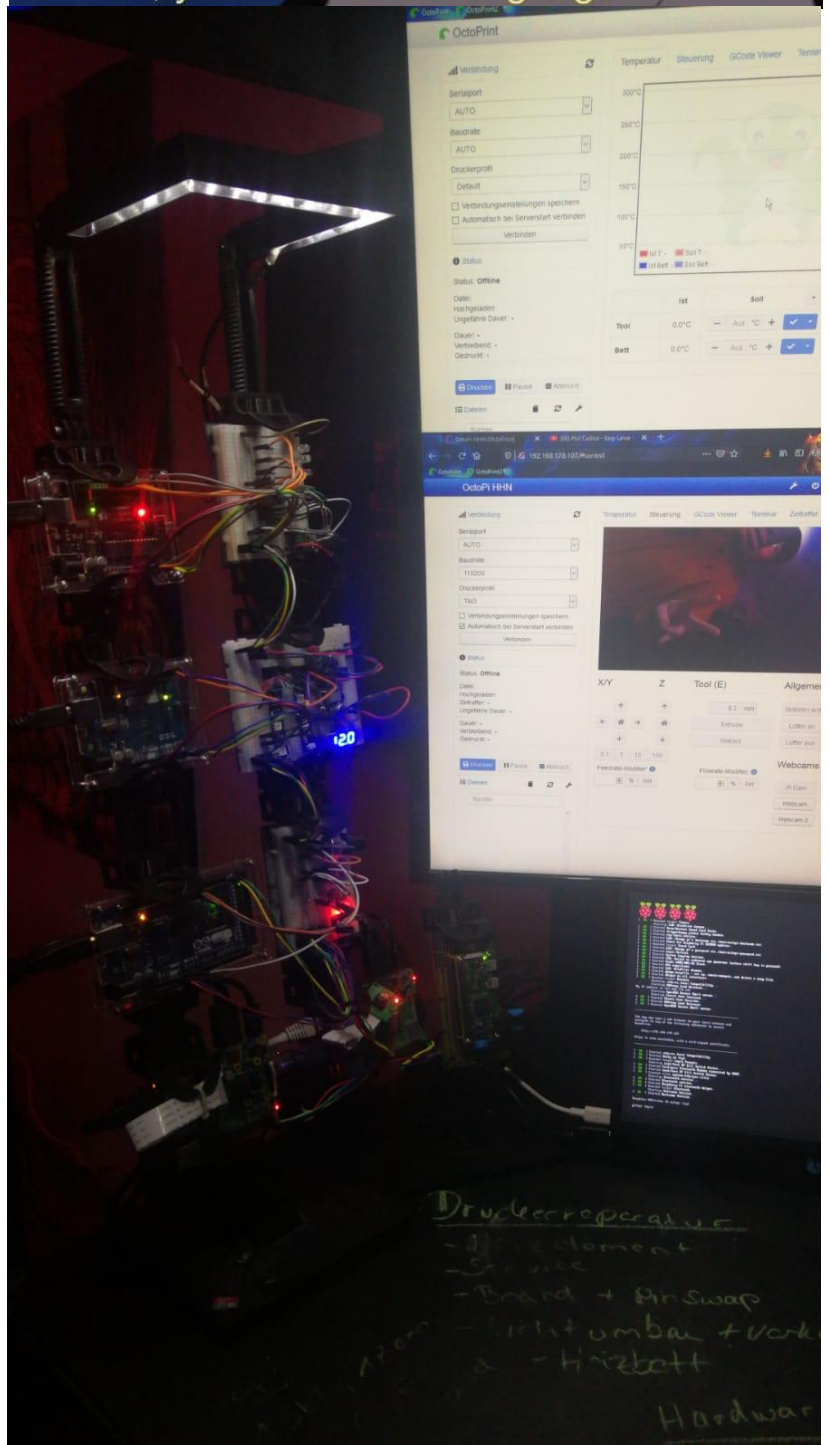


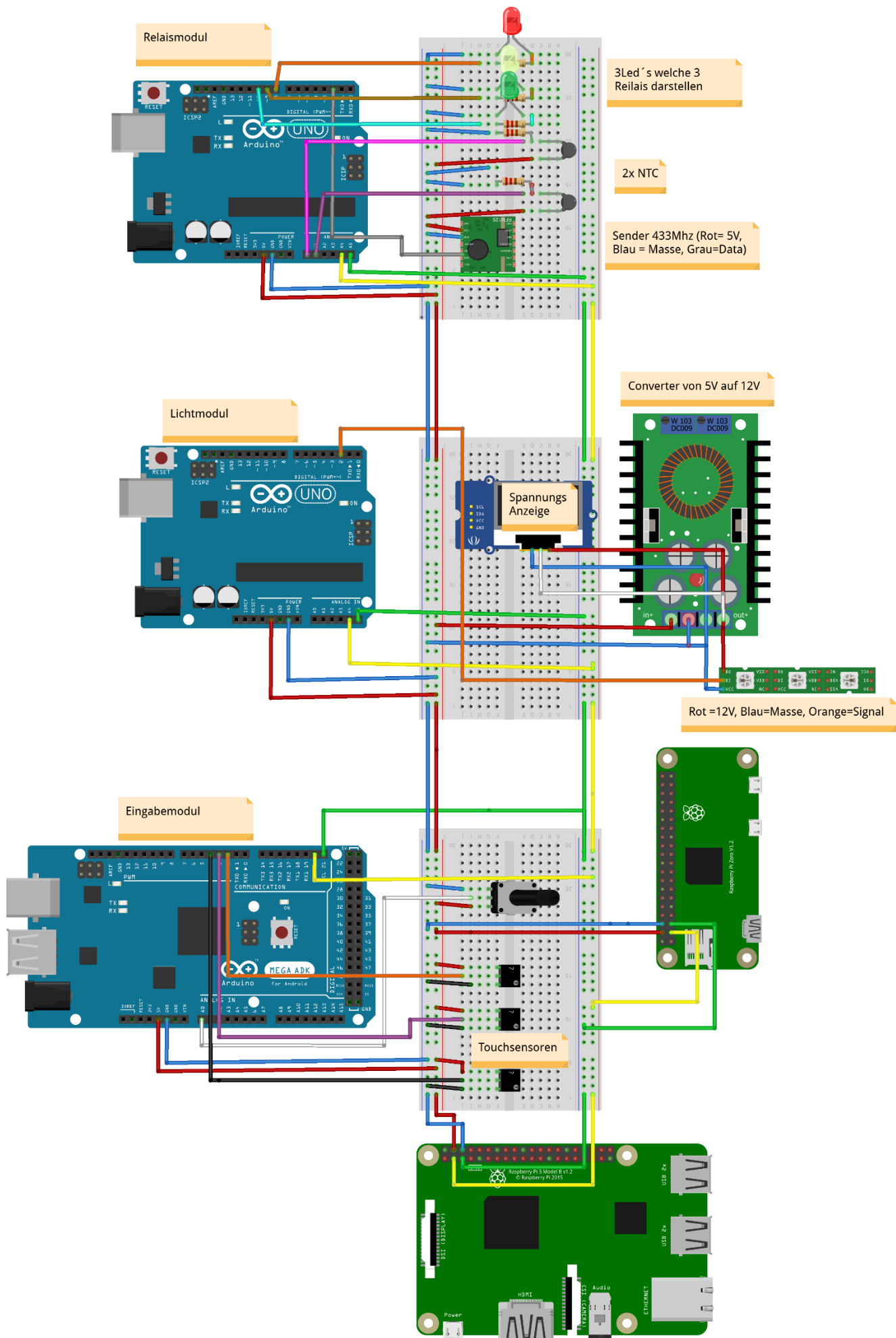
Die Rahmenbedingen zusammengefasst:

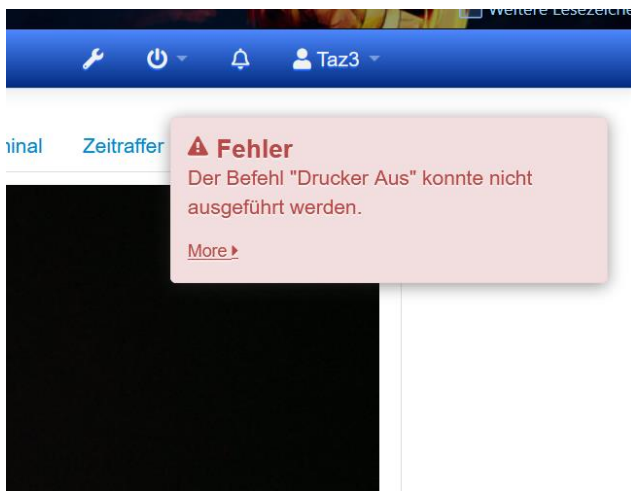
- Octopi aufgesetzt, konfiguriert und geupdatet.
- Kamera (PiCam) konfiguriert
- Schaltung aus Test 2 und Test 3.
- „Befehle“ angelegt und mit der Octoprint Maske verbunden

Der Träger, der diesen Aufbau hält, wurde eigens dafür angefertigt. Die Dateien zum Nachdrucken liegen auf dem Server.

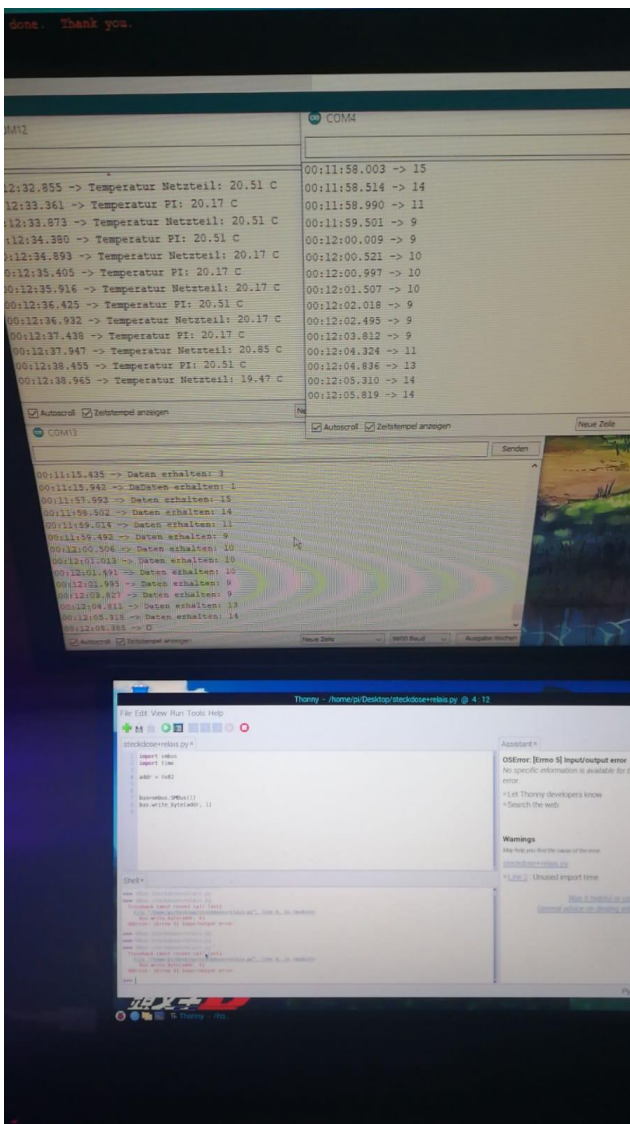
Auf der folgenden Seite ist die Ges. Schaltskizze abgebildet. Diese Stellt nur den Prototypen dar. Der Drucker wird über ein USB-Kabel mit einem Pi verbunden und so gesteuert.







COM Fenster).



Problem vom Prototyp 1

Es gab sehr viele kleine Stolpersteine, die sich aber nach einer ausgedehnten Recherche beheben ließen. Doch ein Fehler schien unlösbar:

Bei diesem Fehler bricht die Kommunikation zusammen.

Dieser macht sich mit der Fehlermeldung (links), Fehlermeldung (Pi Bildschirm ganz unten) bemerkbar. Zusätzlich konnte man sehen das 1 Arduino seinen „Dienst“ eingestellt hat (unterstes

Dieser tritt nur auf bei einer ausgiebigen Benutzung des Kontrollpults auf. Die Zeit bis zum Auftreten sinkt, umso mehr Geräte angeschlossen werden.

Er wird aber nicht ausgelöst, wenn der Aufbau durch den Pi angesteuert wird. Selbst bei 2 Pi's gleichzeitig.

Somit fiel der Fokus der Fehlersuche auf die Potentiometer Schaltung des Steuerpults. Da hier der Fehler nach fast jeder Benutzung auftritt.

Lange Rede kurzer Sinn: Das ist es auch nicht, es konnte zwar durch mehrere „Delay's“ beim auslesen des Potis, die Häufigkeit des Fehlers verringert werden. Doch der Fehler trat dennoch oft auf.

Lösung:

➔ Es wurden **keine Kondensatoren** verbaut. Welche die Spannung im System „abfedern“. Dies erklärt auch die Zunahme des Fehlers durch das Anschließen weiterer Geräte. Dieses Lösungsansatz wurde hier noch nicht durchgeführt, doch der Person, von der dieser Ansatz kommt, kann man vertrauen. Bevor das „fertige Modell“ für die Hochschule bereit ist, muss dieser Ansatz weiterverfolgt werden.

Mögliche „offene Aufgaben“ vom/nach Prototyp 1

- Lösungsansatz (Kondensatoren einbauen) umsetzen.
- Der Pi soll seine Temperatur selbst auslesen. Diese wird momentan durch einen der NTCs gemessen. Da kein 5V Lüfter verfügbar war muss zusätzlich der Lüfter von einem Relais (am Arduino) geschaltet werden (12V Lüfter).
- Plugin MultiCam umsetzen am besten mit 3 Kameras. (evtl. eine Kamera Steuerbar mit Servomotoren).
- Mehrere Drucker mit einem Pi steuern. So könnten wir 3Kameras und 2 Drucker an einen Pi anschließen -> Dies wurde schon häufig umgesetzt (Youtube) und scheint gut zu performen.
- Mehr Einstellmöglichkeiten am Steuerpult implementieren.
- Den Aufbau inkl. Fehlerbehebung verkleinern -> Umsetzung des Projektes erneut mit Microcontrollern wie z.B. ATTiny85, ATMEGA328P. ← persönlicher favo.
- Ein Octoprint „Display“ um das Steuerpult zu ersetzen. Im kommenden Kapitel „Design“ wird man sehen das ein Hüllenaufbau gewählt wurde, der diese Änderung begünstigt.

Experimentelle Aufgabe:

„Ausgedehnte Performancetests“ mit dem Raspberry Pi Zero WH.

Denn es wurden Testdrucke mit den Jeweiligen Pi's durchgeführt. Dafür wurde ein Drucker „Marke Eigenbau“ verwendet (Basis war ein 4Max von Anycubic).

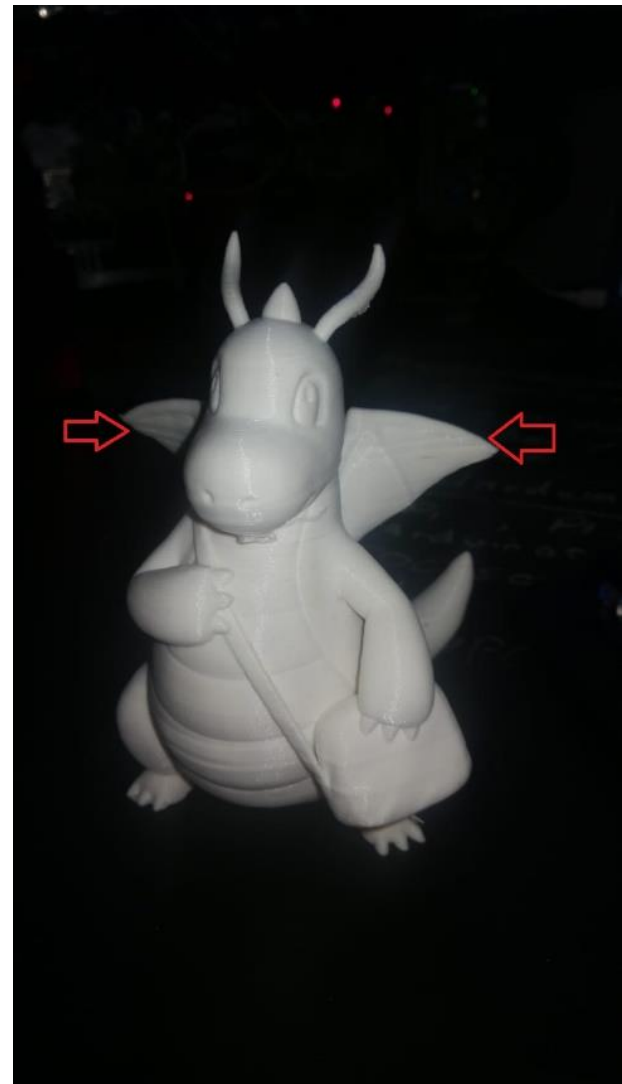
Doch nur der Pi Zero liefert „Druckfehler“, im Bild bei den roten Pfeilen. Diese sind auf eine nicht genaue Ansteuerung der X und Y- Achse zurückzuführen.

- ➔ Hardwarefehler beim verwendeten Drucker sind ausgeschlossen.

Dies tritt aber nicht bei „Kleinen“ Drucken auf, z.B. ein Würfel mit der Kantenlänge von 2cm.

Wenn man dieses Phänomen weiterverfolgt, könnte die 1xDrucker und 1xPiCam Variante mit dem kompakteren Modell „Zero“ umgesetzt werden.

- ➔ Man würde mit der Druckgeschwindigkeit anfangen, beispielsweise „langsam“ drucken und sehen, ob der Fehler damit behoben wäre.

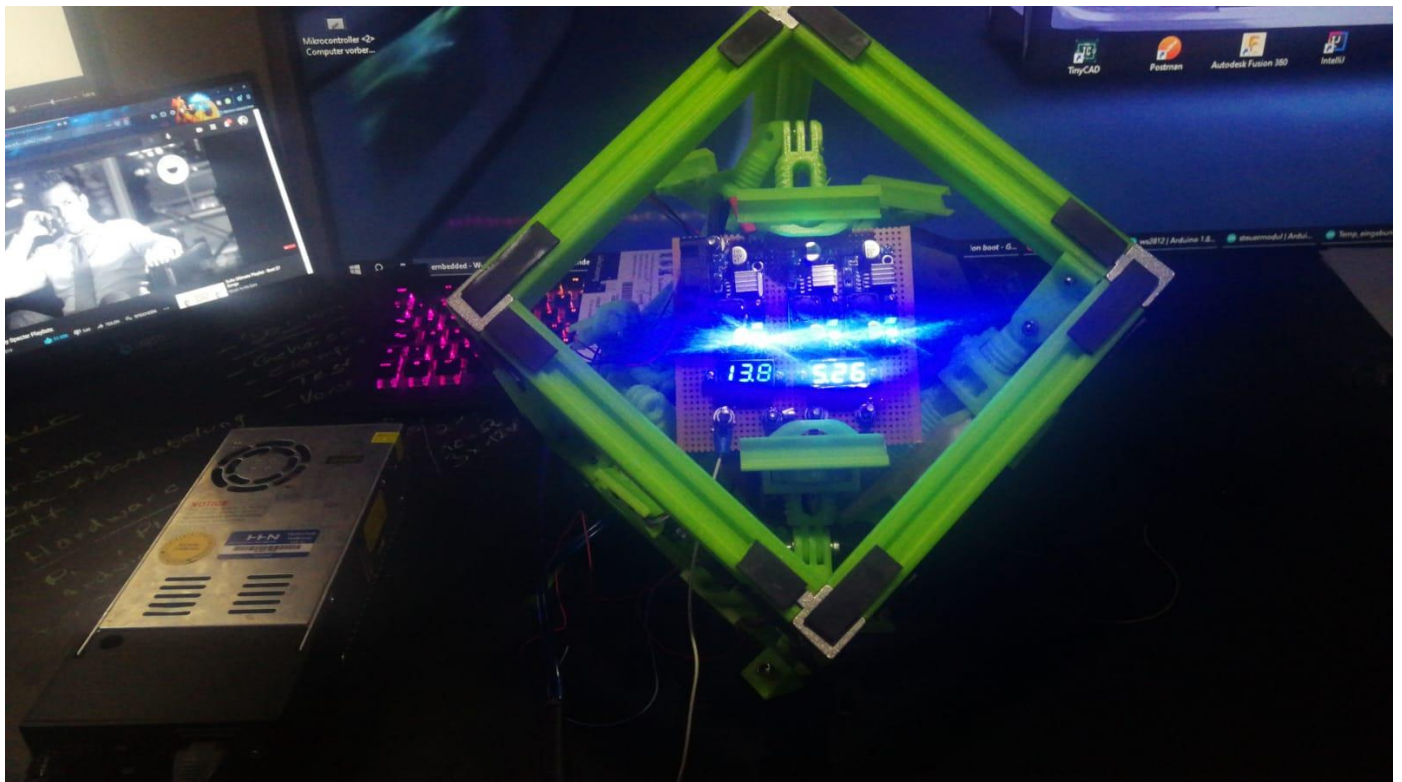
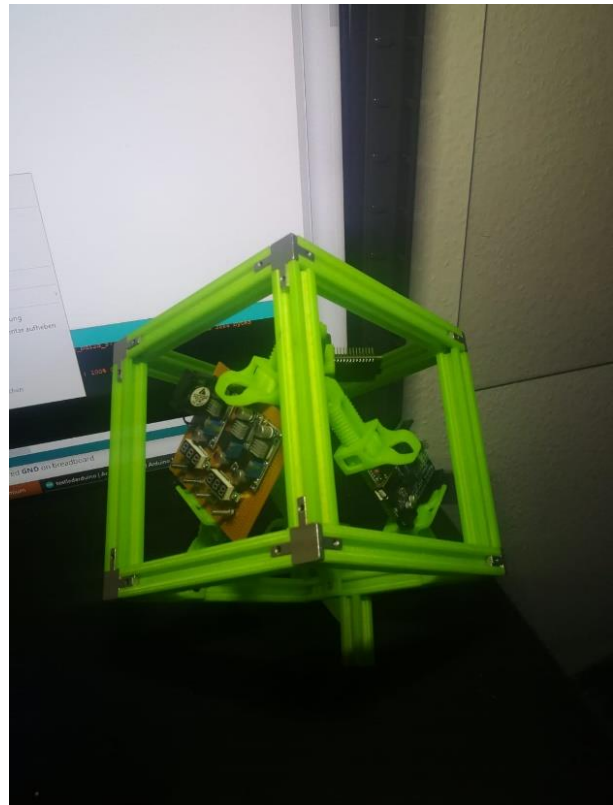


Design der Serverhülle

Es wurde aus 3D gedruckten 2020T Profilen, ein Würfel zusammengesteckt. Jede Seite dieses 20x21cm Würfels (21 wegen den Verbindern entstanden), beherbergt ein Modul unseres Prototypens. Die Füße sind ebenfalls T-Profile.

Die Klammern wurden auf der Seite www.Thingiverse.com herausgesucht.

Auf der Vorderseite wird das Spannungsmodul angebracht. Welches aus 3 „Drop Down Convertern“ besteht. So erhalten wir maximal 3A mit 13.8V und 6A mit 5.26V. Da es sich um eine Parallelschaltung handelt welche nur mit dem Netzgerät (24V) verbunden ist, sparen wir uns an dieser Stelle den Schaltplan dieser Platine.

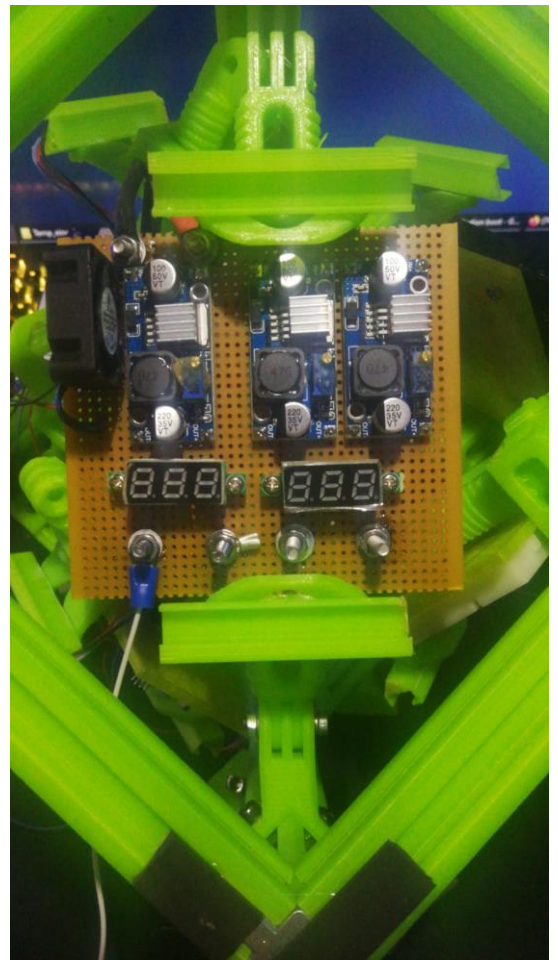


Die „Klemmen“ besitzen zudem eine Halterung für die WS 2811 Led Strips.

Somit kann jede Seite des Würfels unterschiedlich beleuchtet werden. Dies erlaubt uns das Gehäuse gleichzeitig als Statusleuchte des Servers zu benutzen (noch nicht implementiert Stand: 28.1.2021). Oder bei Fehlern das „Modul“, welches einen Fehler meldet, farblich hervor zu heben.

Zudem ist durch diesen Aufbau möglich die I²C Schnittstelle, in ein Modul zu packen und das Anschließen von „neuen Modulen“ angenehmer zu machen. (in der fertigen Version wird dies ein normales Steckbrett sein).

Da wir jetzt Jedes Modul montieren können, brauchen wir noch einen Schutz für unsere Komponenten. Dafür wurden mit Kleber, auf Deckel und T-Profil, Magnete angebracht.



Mit einer „dünnen“ weißen Abdeckung, erhalten wir den Benötigten Schutz + die Led Strips scheinen durch die Oberfläche.

Plexiglas wäre ansehnlicher gewesen, doch da sich der „Lasercutter Marke Eigenbau“ erst im Prototypstadium befindet, wurde mit dem herkömmlichen 3D Druck Verfahren gearbeitet.

Die Abdeckungen wurden selbst erstellt. Somit wird die Original Fusion 360 Datei verfügbar sein.

Somit kann mit „Leichtigkeit“ die Hülle so bearbeitet werden das ein „Octopi Display“, dem Aufbau hinzugefügt werden kann o.ä.

Dieser Aufbau wird im laufe der „Semesterferien“ noch verkabelt.

Design des Prototypens 1

Die Klammern, welche verwendet wurden, werden seit Test 1 genutzt. So kam es zur Idee das man Modul und Steckplatine auf eine Ebene holen sollte. Dafür habe ich ein T-Stück erstellt, welches die „bekannte“ GoPro-Aufnahme besitzt. Somit bleibt es trotz Spezialfall, universal.

Zudem wurde eine Bodenplatte erstellt welche:

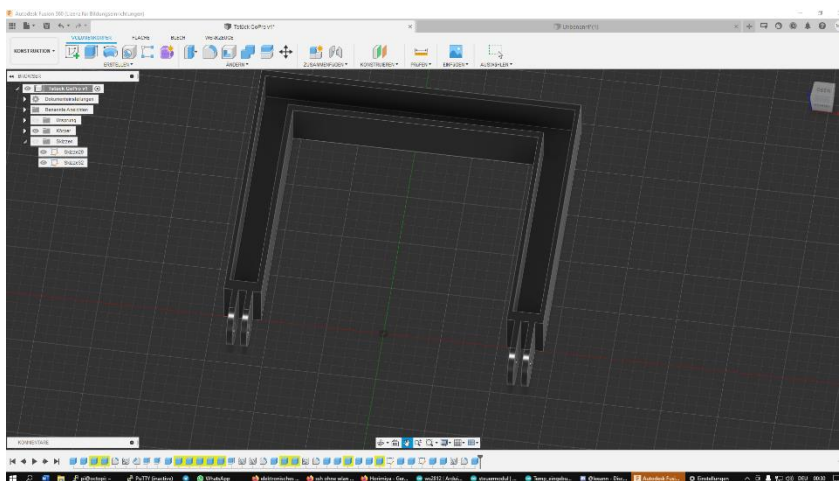
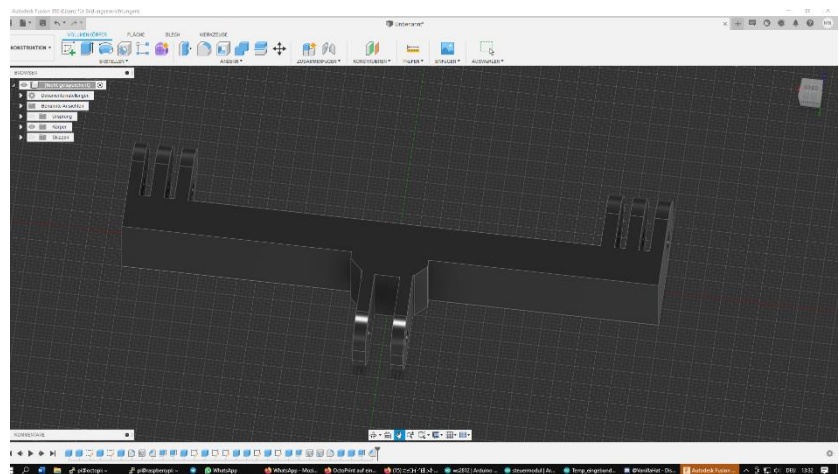
1. Eine Aufnahme für die GoPro-Steckverbindungen besitzt.
2. In der man eine Metallplatte (ca. 1Kg) einkleben kann.

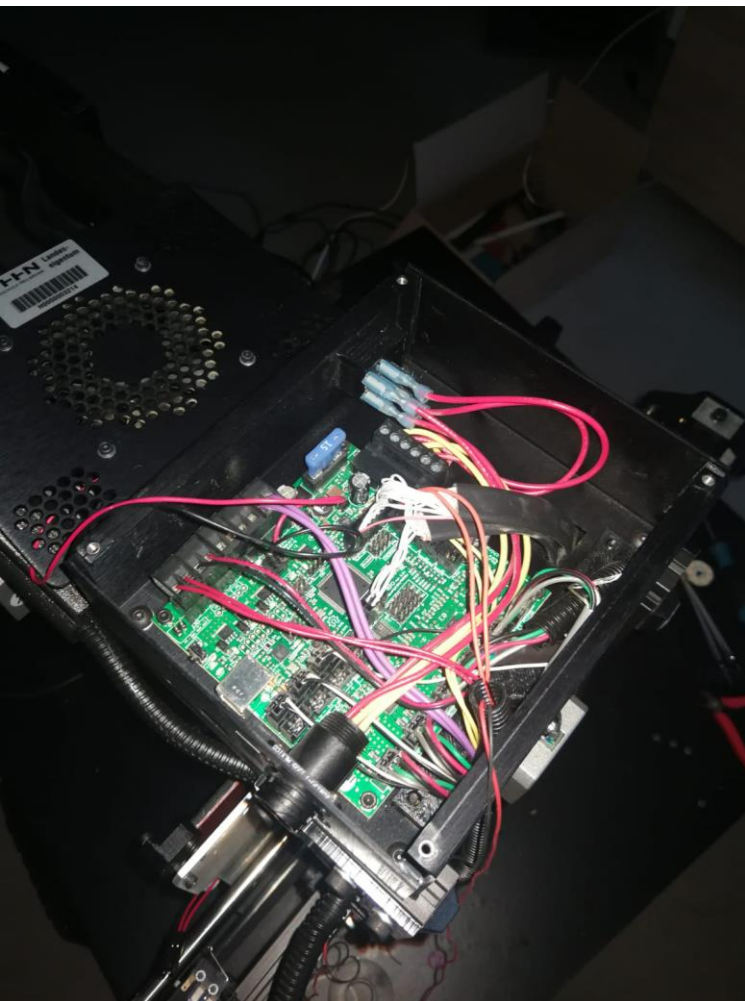
Dies funktionierte auf Anhieb, doch beide Seiten waren unabhängig von der anderen Seite ausrichtbar. Da dies störte, wurde eine Verbindung erstellt, in der man zusätzlich eine Led-Leiste für die Beleuchtung montieren kann.

Somit ist der Aufbau nicht nur „stabiler“ geworden, sondern wir erhalten einen Tragegriff. Somit können wir den Prototyp 1 ohne Rücksicht von A nach B bewegen.

Dieser Aufbau ist so universal gestaltet das jedes mögliche „Go-Pro Stecksystem“ verwendet werden kann. Dies ist sicherlich hilfreich bei zukünftigen Prototypen.

Diese Dateien sind alle auf dem Server als Original Datei und. STL (3D Druck Datei) auffindbar. Somit können Studenten unabhängig vom 3D Drucker Projekt ihre Prototypen nach diesem Schema in Zukunft aufbauen.





Druckerreparatur/ Tuning vom Taz3

Auf die Reparatur des Druckes wird nur sporadisch eingegangen da die Forum von Embedded Systems die „Essenz“ schon vorhanden ist:

In „Diskussionen Rund um Embedded Systems & Games Interfaces“.

Thema: **Beschaffungen**

Die Fehlersuche/Inbetriebnahme zusammengefasst:

- Überprüfung des Netzteils
(Potentialunterschied unter Last)
- Sicherungen haben auf dem Board gefehlt.
Somit neue Sicherungen nacheinander eingebaut,
parallel dazu immer Potenzialmessungen.
➔ Funktion = i.O.
- Prüfen der Verbindung über USB
➔ Funktion= i.O.
- Probedrucken per SD-Karte
➔ Proportionsfehler= N.I.O.+Düse heizt schlecht

Deshalb wurde eine „neue Marlin“ aufgespielt. Es wurde die „Standard Marlin“ vom Taz 3 Drucker aufgespielt, eine Marlin ist die Software des Druckers. Sie enthält Einstellungen jeglicher Art von „Board Configuration“ bis „Stepper Treiber“ Einstellungen. Dies ist auch der Grund, warum das Druckerdisplay 3 Extruder anzeigt. Da alles „Unwichtige“ noch im „default“ Modus ist.

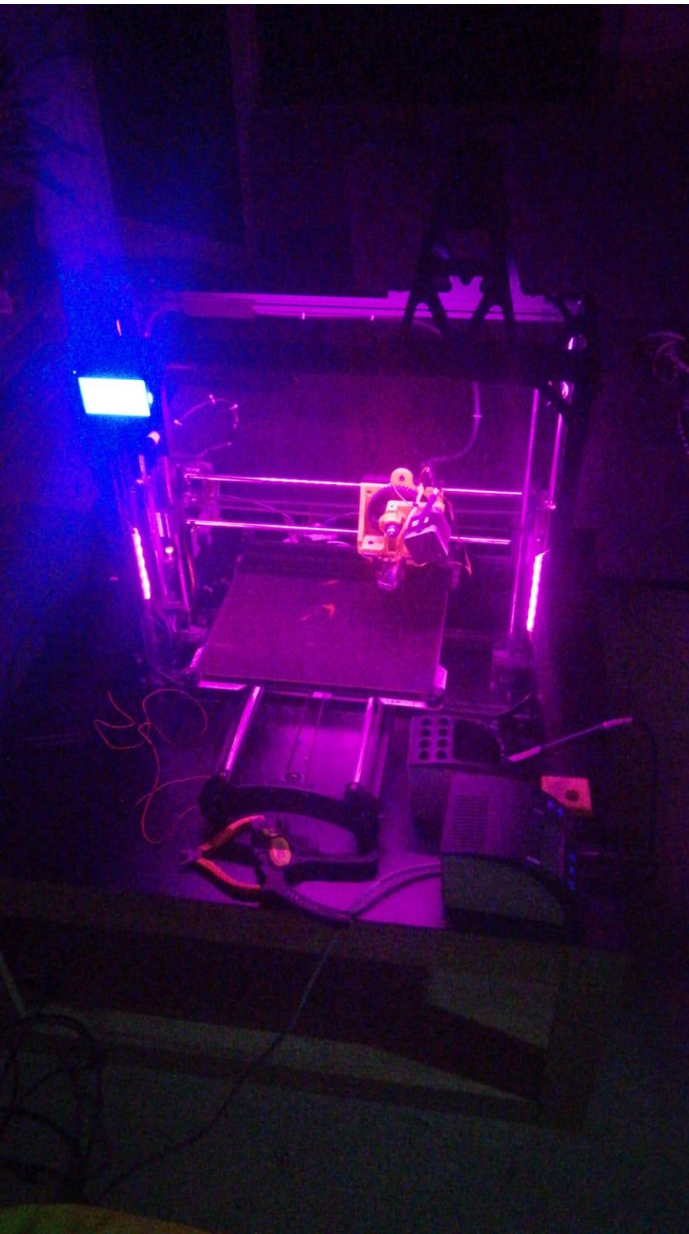
- Probedrucken per SD-Karte Nr3
➔ Düse heizt gar nicht mehr

Jetzt wurden alle Düsen welche kompatibel mit dem Taz 3 Montiert (4Stück) geprüft. Alle bis auf eine gingen nicht.

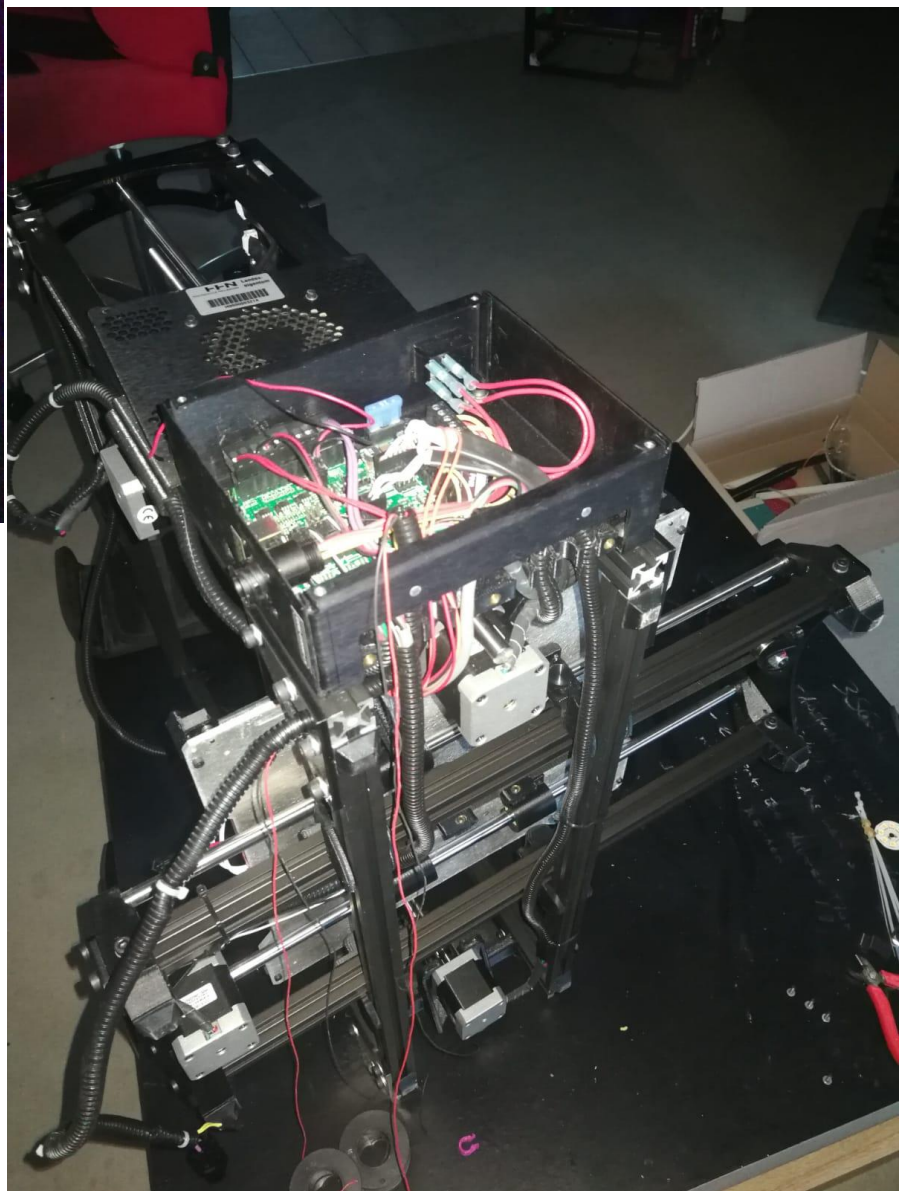
- Widerstandsmessung an allen nicht funktionierenden Heizelementen
➔ Werte = O.L

Einige Zeit später gab es einen Masseschluss im Stromkreis des Heizbettes. Dazu kann man sich ein Video im besagten Forum ansehen.





Der Taz 3 wurde mit WS2811 Led Strips ausgestattet und besitzt jetzt eine Düsenleuchte. Diese ist eine 12V Hochleistungs Led.



Es wurden neue Kabel durch den Elektronik Kasten/Kabelkanäle gezogen. Somit fügt sich die zusätzliche Elektronik Nahtlos ein.



Nacharbeiten

Höchst wahrscheinlich durch den Transport zurück zur HHN, wurde die Aufnahme der Z-Achse beschädigt. Dieses muss noch erneuert werden (muss gedruckt werden).

Nachwort

Im Laufe der nächsten Woche (Stand:29.1.2021) wird ein Vortrag zum Prototypen im Forum zu finden sein.

Das Projekt hat viel Spaß, Verzweiflung und schlaflose Nächte bereitet. Doch dieses Projekt wird weitergeführt, da dies ein „Produkt“ ist, welches ich selbst verwenden will. Zudem soll dies die Basis meiner „Smarthome Zentrale“ werden.

Im Rahmen der Veranstaltung: **AI-VPI4.1 Org.-, Arbeits-, Personalpsychologie (173151)** habe ich auch Usability Tests zur Software „Cura“ durchgeführt. Diese Ausarbeitung und ihre Ergebnisse stelle ich euch zusätzlich auf dem Server bereit.

- ➔ Dies könnte interessant sein für diejenigen welche einen „Guide“ über das 3D Drucken schreiben wollen. Oder Interesse daran haben mal ein Plugin für eine Open Source Software zu schreiben.

Ps. Die Quellen sind in einem Separaten Dokument in diesem Ordner, danke fürs Lesen.