Arbeitsjournal: Trink-Reminder

Für Videos, Bilder und Code siehe Ordner „Anhang“

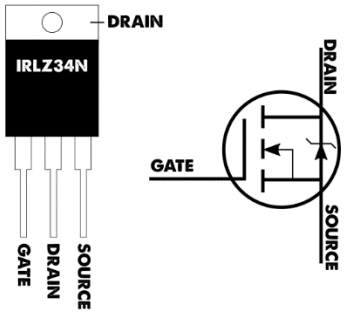
15.04.2021

Informieren über „unbekannte“ Bauteile

Die Funktion der Bauteile war größtenteils klar, da sie bereits in den vorherigen Übungsaufgaben genauer betrachtet wurden. Die einzigen zwei Teile, die ich mir genauer ansehen musste, sind der Transistor, Freilaufdiode und der Spannungsregler.

Transistor

- Ich verwende ein n-Channel MOSFET (nicht p-Channel): IRLZ34N
  - ➔ Datenblatt: https://www.infineon.com/dgdl/irlz34npbf.pdf?fileId=5546d462533600a401535672068 92720
- Wenn an der Base eine Spannung anliegt, kann Strom von Source zu Drain fließen



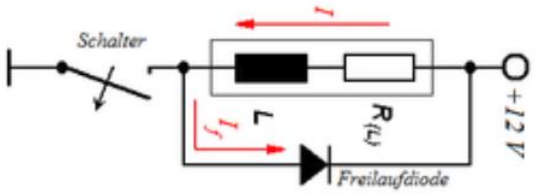
  - ➔ Bild von: https://moderndevice.com/product/irlz34n-mosfets/

Freilaufdiode/ Schutzdiode

- Funktionsweise von Diode schön erklärt:
  - ➔ https://www.youtube.com/watch?v=MSncOmacDJ0 und
  - ➔ https://www.youtube.com/watch?v=tWL4sI4W3_A

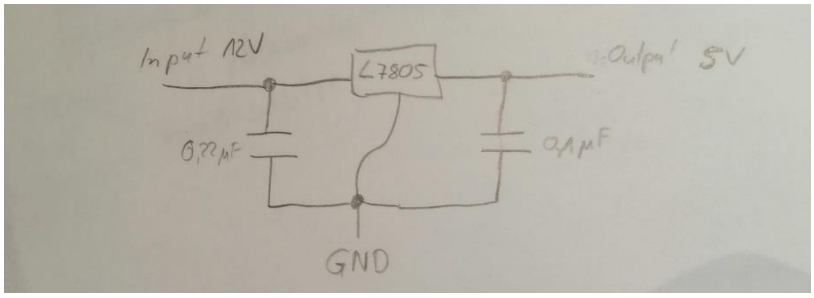
- „Freilaufdioden dienen zum Schutz vor einer Überspannung beim Abschalten einer induktiven Gleichspannungslast (z. B. Elektromotor, Relaisspule, Zugmagnet). Dazu werden Halbleiterdioden (im Schaltbild bezeichnet als *Freilaufdiode*) derart parallel zu induktiven Gleichstromverbrauchern (im Schaltbild: L mit Widerstandsanteil R_L) geschaltet, dass sie von der Speisespannung in Sperrrichtung beansprucht werden.

  Nach dem Abschalten der Speisespannung sorgt die Selbstinduktion der Spule dafür, dass der Strom zunächst in der ursprünglichen Richtung weiter fließt. Ohne Freilaufdiode führt das zu einer Spannungsspitze, die sich zur Betriebsspannung addiert und die Schaltstrecke schädigen oder zerstören kann. Mit einer Freilaufdiode wird die Spannungsspitze jedoch auf die Durchlassspannung der Diode (bei Silizium etwa 0,7 V) begrenzt. Das schützt die elektronischen Bauteile (beispielsweise Halbleiter wie Transistoren), aber auch Schaltkontakte sehr effektiv vor Überspannung (im Bild auf maximal 12,6 V). Der Strom fließt über die Diode und die Energie des Magnetfeldes, die der grün markierten Fläche entspricht, wird größtenteils im ohmschen Widerstand der Spule und zu einem kleinen Teil in der Diode in Wärme umgewandelt.“ – https://de.wikipedia.org/wiki/Schutzdiode

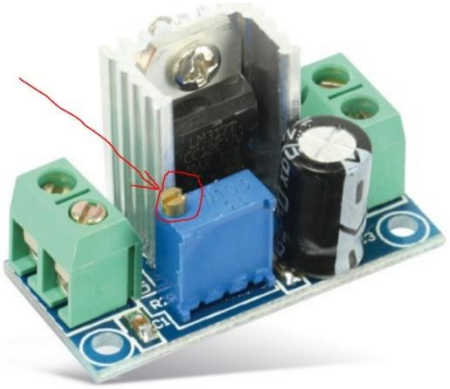


Spannungsregler

- Da ich nur mein Gerät mit nur einer Stromquelle betreiben will, aber sowohl 12V, als auch 5V Spannung für die Bauteile benötige
- Zuerst hatte ich geplant, den Spannungsregler selbst zu implementieren:



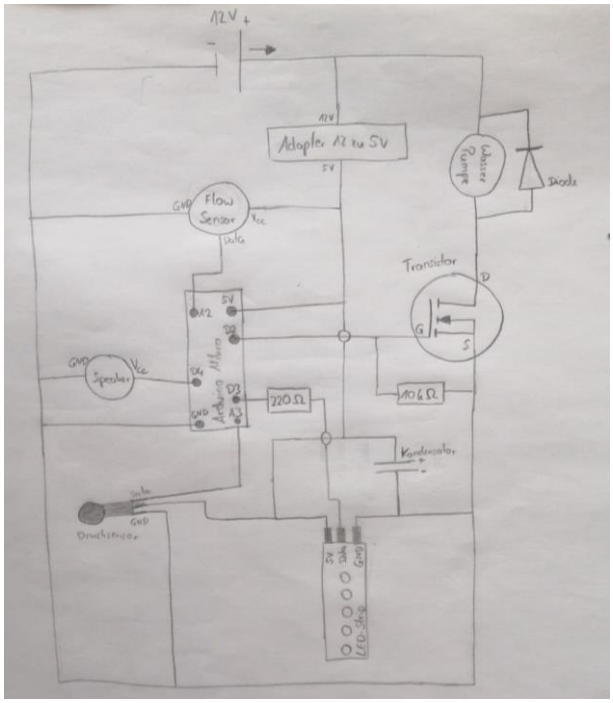
- Andi hat mir dann aber ein fertiges Spannungsreglermodul (das Spannungsregler-Modul DAYPOWER MSDLM317) empfohlen
- Die ausgehende Spannung lässt mithilfe des Potentiometers einstellen:

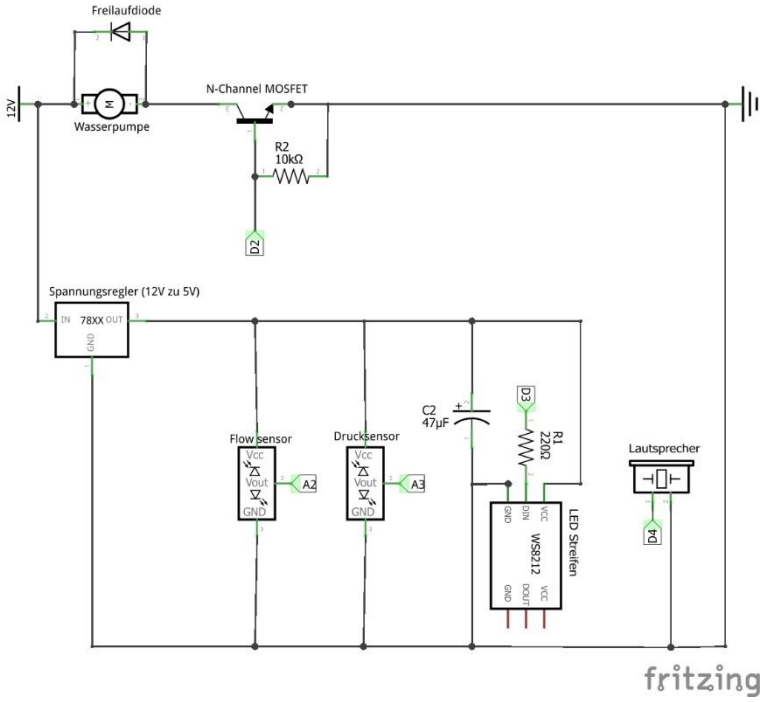


- Nachdem ich das Poti so eingestellt hatte, dass ich die gewünschte Ausgangsspannung (5V) erhalte, habe ich das Ganze mit der Heißklebepistole fixiert

Erstellung des (vorläufigen) Schaltplans

- erst als Skizze auf Papier, dann in sauberer Form mithilfe der Fritzing Software
- Folgende Bauteile sollen im Schaltkreis verbaut werden:
  ➔ Wasserpumpe, Spannungsregler, Flow Sensor, Drucksensor, LED-Streifen (eigentlich zweimal, allerdings im Schaltplan nur einmal eingezeichnet, da Prinzip für den zweiten Strip dasselbe ist) und Lautsprecher





➔ Der Schaltplan hier ist noch nicht final. Da manche Teile erst einige Tage später angekommen sind, mussten nachträglich noch Änderungen am Schaltplan gemacht werden (beispielsweise hatte der Drucksensor nicht wie erwartet zwei separate Ausgänge für GROUND und DATA, sondern nur einen)

17.04.2021

- Nachdem die meisten Bauteile angekommen sind (bis auf die Pumpe), habe ich zunächst einmal den Flow Sensor und den Drucksensor getestet und Notizen zu deren Funktion gemacht.

Flow Sensor

- Informationen zum Bauteil: https://www.robotics.org.za/YF-S401
- Datenblatt und Beispielcode zum Bestimmen der durchgeflossenen Wassermenge: https://github.com/microrobotics/YF-S401/blob/main/YF_S401_datasheet.pdf
- Im Schaltkreisentwurf oben ist der Flow Sensor an einen analogen PIN angeschlossen. Allerdings liefert er keinen analogen Wert, weshalb auch ein digitaler PIN verwendet werden kann. Der Flow Sensor liefert in einer bestimmten Frequenz HIGH/LOW Impulse. **5800 Pulse entsprechen dabei einem Liter (siehe Datenblatt)!**
  - **The range of Model YF-S401-3507 sensor is 300ml to 6000ml per minute.**
  - **At 1000ml we get 5880 pulses**
  - **Thus pulses per second 5880/60 = 98 Hz square wave**
  - **Which has a period of 1/98 = 10.2 millisecond**
  - **Thus for 1ml you calculate 5880/1000 = 5.88/60 = 0.098Hz with a period of 10.2 seconds. In the programming, I will be using the pulsein to measure the time of a pulse.**
- Zunächst hatte ich zum einlesen der Pulse die Methode pulseIn() gefunden, welche die Dauer eines Pulses (in microseconds) bzw. 0 wenn kein Puls vor Timeout erkannt wurde, zurückgibt. Allerdings bin ich dann auf Interrupts (bzw. die attachInterrupt() und detachInterrupt() Methode) gestoßen, welche sich besser für mein Problem eignen und welche auch in dem Code-Beispiel von https://www.robotics.org.za/YF-S401 verwendet werden:
  - ➔ Interrupts: https://www.arduino.cc/reference/en/language/functions/external-interrupts/attachinterrupt/
  - ➔ pulseIn Methode: https://www.arduino.cc/reference/en/language/functions/advanced-io/pulsein/
  - ➔ **Wichtig: Nur die digitalen PINs 0, 1, 2, 3 und 7 sind beim Arduino Micro mit Interrupt verwendbar (Ich verwende daher den digitalen PIN 7 für den Flow Sensor)**
- Letztlich wurde im zur Bestimmung der durchgeflossenen Wassermenge der Code von https://www.robotics.org.za/YF-S401 übernommen.
- Anmerkung: zunächst war ich nicht zufrieden mit der Genauigkeit der Werte, die mir der Sensor liefert. Das lag allerdings nur daran, dass ich noch keine Pumpe hatte, die gleichmäßig viel Wasser durch die Schläuche pumpt. Davor konnte ich den Flow Sensor nur testen, in dem ich Wasser in den Schlauch kippe. Wenn allerdings die Pumpe angeschlossen ist, kann der Sensor ziemlich präzise die Wassermenge bestimmen!

Pressure Sensor

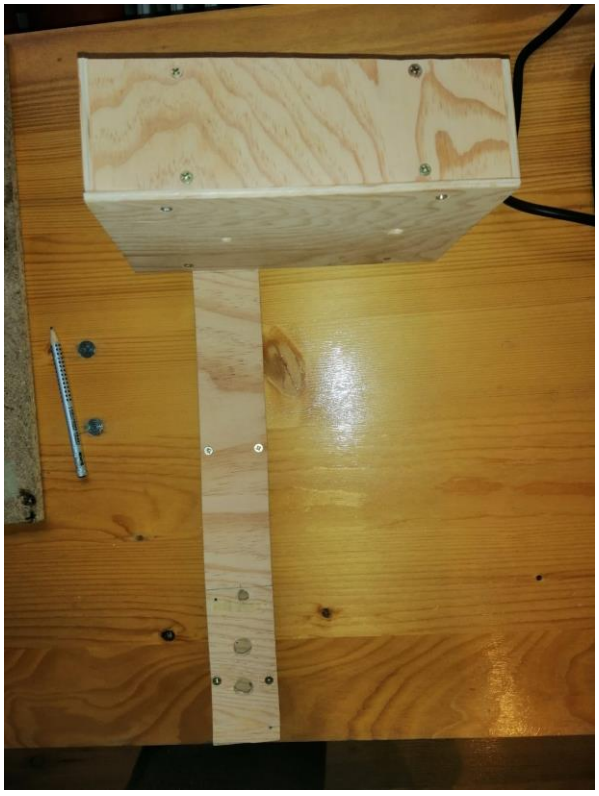
- Zwei (anstatt drei) Anschlüsse: 5V und der andere muss an einen analogen PIN des Arduino (inklusive Pull-down Widerstand)
  - ➔ Skizze des Schaltplans musste dementsprechend angepasst werden
- Je nachdem wie viel Kraft auf den Sensor wirkt, verändert sich sein Widerstand und damit die ausgehende Spannung (welche man dann am analogen Pin messen kann)
- **Wichtig:** Zunächst hatte ich geplant, das Glas einfach auf den Drucksensor zu stellen, um über das Gewicht des Glases den Füllstand (voll, leer, kein Glas) zu ermitteln. Allerdings ist der Sensor zu dünn und registriert gar nichts, wenn man ein Glas drauf stellt. Deshalb muss das Gewicht des Glases also nur auf einen Punkt reduziert werden. Zuerst hatte ich das mithilfe eines Nagels versucht. Der Nagel übt allerdings nur Druck auf einen winzigen Punkt aus, was dazu führt, dass die Werte, die der Sensor liefert, nicht sehr stark variieren (siehe

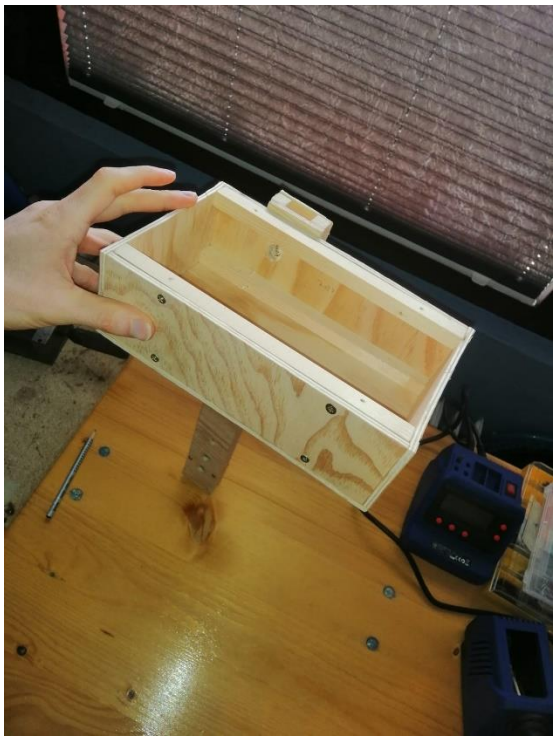
Video „1_Vergleich Drucksensorwerte.mp4“). Die besten Werte liefert der Sensor, wenn mit dem Finger draufgedrückt wird. Aus dem Grund wurde ein Elastikpuffer

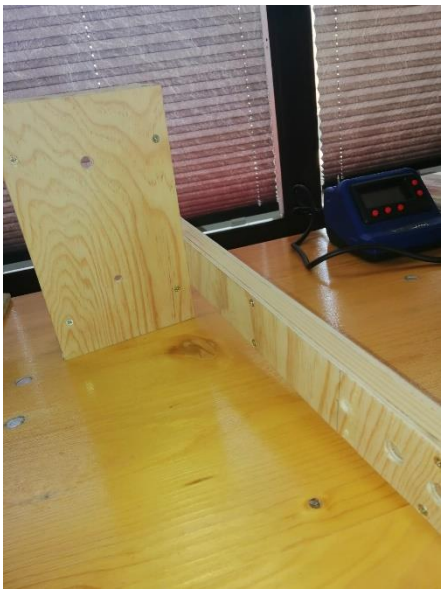
dafür verwendet, welcher einen Finger ganz gut simuliert und somit gute Sensorwerte liefert.
→ Wie die Druckausübung auf einen Punkt umgesetzt wurde, wird bei der Skizze des Prototyps ersichtlich

Bauen des Prototyps: Vorgehen

- Skizzieren auf Papier und Maße bestimmen
- Holz ausmessen
- Alle Teile aussägen (mit Stichsäge) + Kanten abschleifen
- „Unteren“ Kasten zusammenbauen
  - ➔ Für das „Abstell-Loch“ des Glases wurde eine Lochsäge verwendet
  - ➔ Besteht aus zwei Etagen: unterste Etage für die Technik, die andere für den Drucksensor und eine „Wippe“, die den Druck vom Glas auf den Drucksensor überträgt
- Verbindungsstück + „oberen Kasten“ zusammenbauen
  → in dem Kasten sollen die Pumpe und der Flow-Sensor angebracht werden
  - ➔ Das Verbindungsstück ist hohl und die Kabel sollen darin verlegt werden
- Löcher bohren für die Kabel und die Wasserschläuche



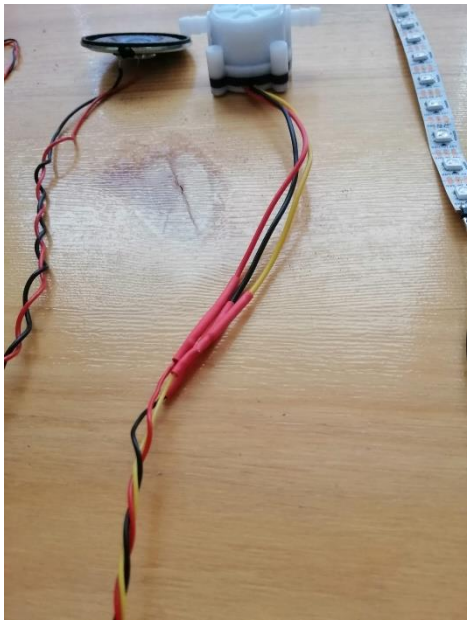


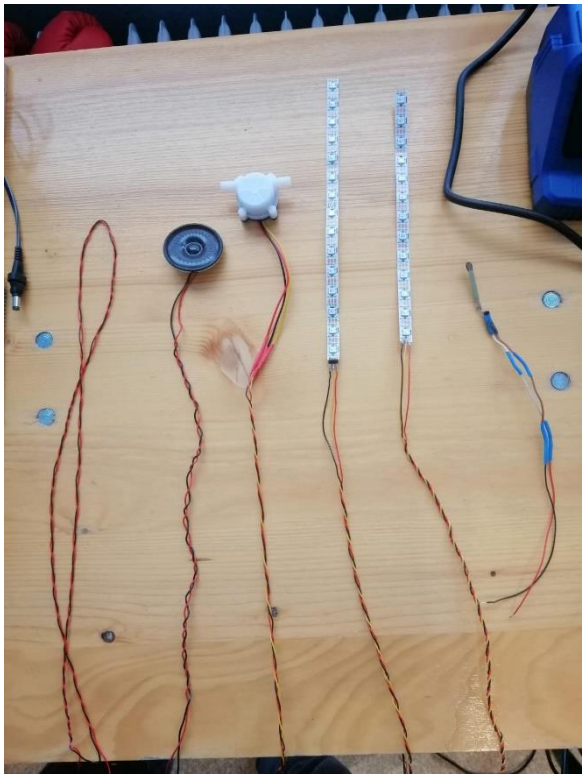


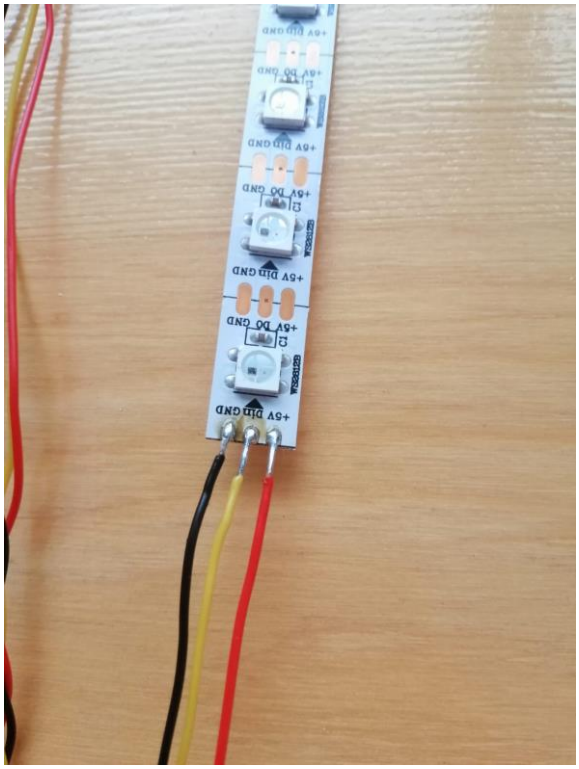
- ➔ Prototyp noch nicht komplett zusammengesetzt, da ja noch die Kabel verlegt werden müssen
- ➔ Allerdings wurden bereits alle Löcher vorgebohrt, dass die Teile dann nur noch zusammengesetzt und festgeschraubt werden
- ➔ Einige Löcher wurden auch vorerst vergessen und mussten dann nach und nach noch gebohrt werden
- ➔ Dauer des Gehäusebaus: ca. 5 Stunden
- ➔ Blechstreifen als Flaschenhalterung vorerst noch weggelassen, da nicht essenziell
- ➔ Siehe Video „2_Prototyp Gehäuse Zwischenresultat.mp4“

18.04.2021

- Litze und Bauteile zusammengelötet
- Offenen Kontaktflächen mit Schrumpfschläuchen abgedeckt







- Probleme am Druckwandler des Prototyps behoben: Die Wippe hat sich nicht frei/ locker bewegt. Es lag daran, dass die Bohrungen für die Schrauben links und rechts etwas zu eng waren. Durch eine kleine Verbreiterung haben die Schrauben jetzt etwas mehr Spielraum und lassen sich einfacher rotieren.

Drucksensor-Werte notiert:

Ich habe mir notiert, wann der Drucksensor welche Werte ausgibt, damit ich dann Schwellenwerte bestimmen kann, ab wann das Programm den Status des Glases als „Glas steht auf der Platte und ist voll“, „Glas steht auf der Platte und ist leer“ und „kein Glas auf der Platte“ ansieht.

Glasstatus	Gemessener Wert
Kein Glas	8-10
Leeres Glas	29-30
Halbvolles Glas (40ml)	32-33
Volles Glas (80ml)	35-36

Anhand der Messungen habe ich dann folgende Schwellwerte festgelegt:

- ➔ Wenn Drucksensor-**Wert <= 15**, dann: **kein Glas**
- ➔ Wenn **Wert > 15 && Wert <=** 31, dann: **Glas leer**
- ➔ Wenn **Wert >** 31, dann: **Glas enthält noch Wasser**

Programmlogik zur Erinnerung ans Trinken geplant:

- Ein Tag hat 24h, davon sind 8h Schlaf → 16h wach
- empfohlen werden 30-40ml Wasser pro kg Körpergewicht am Tag, was bei mir zu ca. 2.5 Litern am Tag führt.
- Der Trink-Reminder soll mich alle 30 Minuten erinnern, die nötige Menge Wasser zu trinken, um auf die 2,5 Liter am Tag zu kommen. D.h.
  - ➔ 16h = 960 min, 2,5 l = 2500 ml
  - ➔ 2500 ml / 960 min = 2,604 ml/min
  - ➔ 2,604 ml/min * 30 min = 78,12 ml ~ 80 ml
- Der Trink-Reminder soll mir also alle 30 Minuten 80 ml in mein Glas füllen und mich dazu bringen, diese zu trinken
- Wird das Glas nicht leergetrunken, wird alle 5 Minuten erneut erinnert, bis es leer ist (ich bezeichne das ab sofort immer als „Snooze-Wecker“ oder „Snooze-Funktion“)
- Das Programm soll sich außerdem merken, wie viel insgesamt schon getrunken wurde. D.h. trinkt man schon voraus, also drei Gläser (mit je 80ml) auf einmal, dann soll auch die nächsten 1,5 Stunden nicht mehr ans Trinken erinnert werden. Und wenn man innerhalb von 1 Stunde (in der man eigentlich 2*80ml trinken sollte), gar nichts getrunken wird, dann muss man auch erstmal zwei Gläser trinken, um das Programm wieder zufrieden zu stellen.
- Wenn die Pumpe aktiv ist, aber über einen gewissen Zeitraum kein Veränderung an der durchgeflossenen Wassermenge zustande kommt, dann ist die Flasche leer
- Wenn der Drucksensor „Glas leer“ misst, dann soll die Pumpe angehen. Sie soll so lange laufen, bis der Flow Sensor 80ml Wasser gemessen hat. Wird während dieses Auffüll-Prozesses das Glas weggenommen (als misst Drucksensor „kein Glas“), dann stoppt die Pumpe sofort. Die bisher eingefüllte Wassermenge soll allerdings gespeichert werden, sodass beim erneuten Abstellen des Glases nicht erneut 80ml aufgefüllt werden, sondern nur noch der Rest, der gefehlt hat.

19.04.2021

- Heute sollte eigentlich die Wasserpumpe geliefert werden, was allerdings spontan von DHL auf den Abgabetag, Mittwoch den 21.04 verschoben wurde. Das heißt ich kann den finalen Schaltkreis und den Wasserpumpzyklus immer noch nicht testen.

Schaltkreis überarbeiten

- Der Schaltkreis wurde basieren auf den gewonnenen Erkenntnissen über die Bauteile überarbeitet und stellt auch die **finale Version** davon dar!
  - ➔ Siehe Anhang „Schaltplan_Fritzing_final.jpg“
- Auch die Steckbrett Skizze wurde dementsprechend überarbeitet
  - ➔ Siehe Anhang „Steckbrett_Fritzing_final.jpg“

Code Implementierung

Für die Code-Implementierung siehe Ordner „Code“ im Anhang

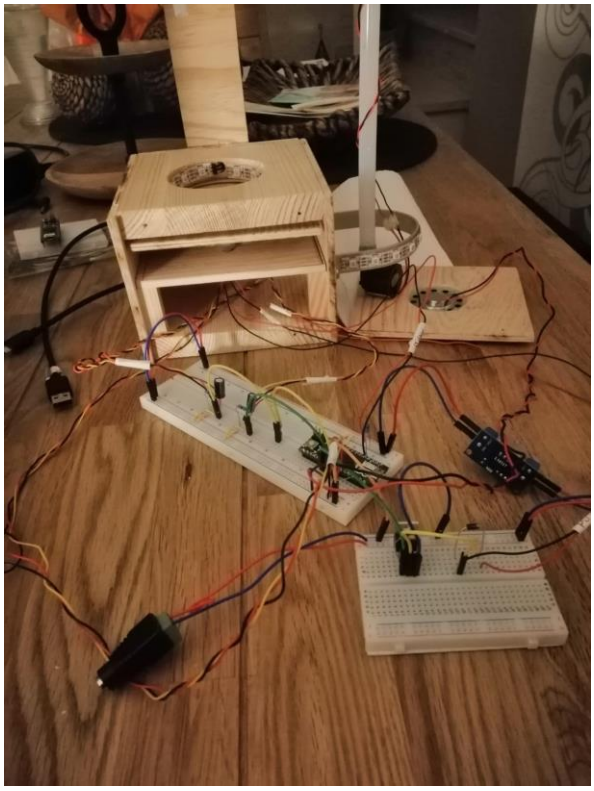
- Implementieren des Haupt- und des Snooze- Weckers:
  - ➔ Der „Hauptwecker“ startet alle 30 Minuten (sofern nicht schon Wasser vorgetrunken wurde). D.h. auch, dass die zu trinkende Wassermenge alle 30 Minuten um 80ml steigt.
  - ➔ Die Erinnerung wird folgendermaßen für den Nutzer deutlich: Ein Piepsignal für 1 Sekunde (Frequenz: 300hz), sowie ein Aufleuchten der LEDs, die sich an der Glas-Abstell-Plattform befinden
  - ➔ Der Snooze Wecker startet nur, wenn die getrunkene Wassermenge geringer ist, als der Soll-Wert. Damit man das wieder aufholt, „nervt“ der Snooze-Wecker alle 5 Minuten.
- Gliedern des Programms mithilfe von Zuständen des Prototyps:
- STANDARD (Glas mit Wasser steht auf der Platte, Glas enthält auch noch Wasser), NO_GLASS (Glas befindet sich nicht auf der Platte), RECHARGING (Gerät pumpt gerade Wasser von der Flasche in das Glas)
  - ➔ Zuerst hatte ich noch einen vierten Zustand NO_BOTTLE, welcher sich dann allerdings als überflüssig erwiesen hat, da das Programm schon mit nur den drei Zuständen umgesetzt werden konnte
- Implementierung der Zustandsübergange (z. B. „wenn Pumpe aktiv und Flow Sensor misst keine Werte mehr, dann soll vom aktuellen Zustand RECHARGING in den Zustand BOTTLE_EMPTY gewechselt werden“)
- Code zum Einlesen der Sensorwerte (Drucksensor und Flow Sensor) und zur Ausgabe über die digitalen PINS (zum Pumpe an- und ausschalten, LED Strips einfärben)
- Testen konnte ich den Code noch nicht, da ich die Schaltung noch nicht auf dem Steckbrett umgesetzt habe und die Pumpe ja auch noch fehlt

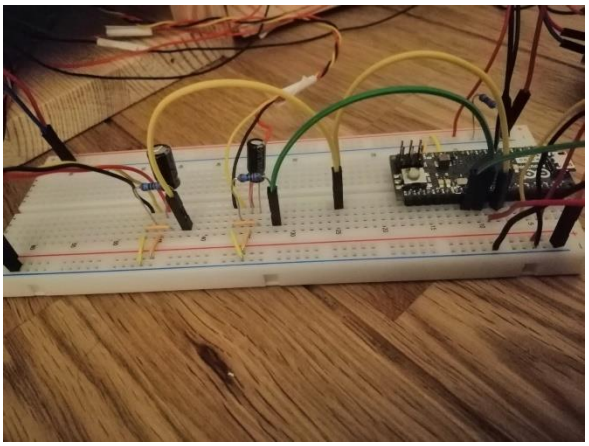
20.04.2021

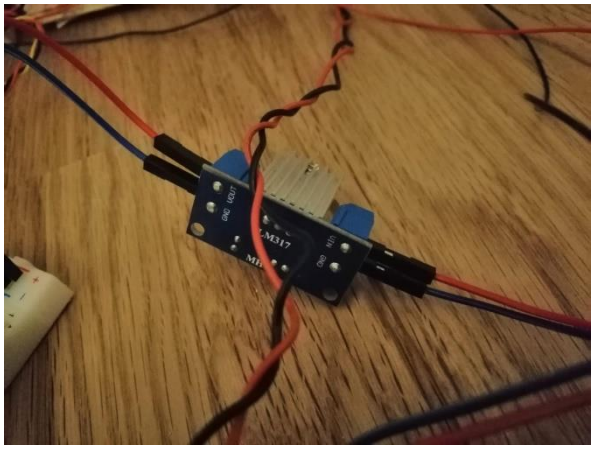
- Anfügen einer Flaschenhalterung an den Prototypen
  - ➔ dazu habe ich einfach ein Stück Bodenleiste, das wir noch daheim rumliegen hatten, gebogen und dann am Prototyp angebracht. An dieser Flaschenhalterung soll dann auch ein LED-Streifen außenrum angebracht werden, der aufleuchtet, wenn die Flasche leer ist. Also musste noch ein Loch gebohrt werden, um die Kabel dafür verlegen zu können.

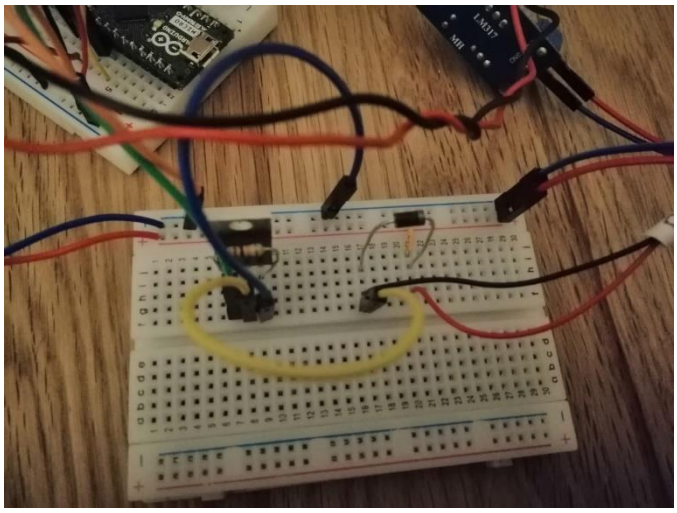
21.04.2021

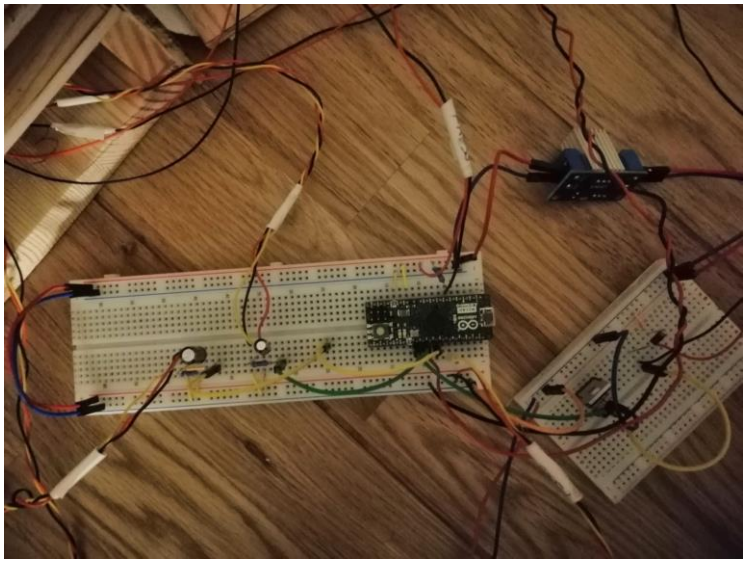
- Pumpe ist angekommen, funktioniert allerdings nicht wie erwartet:
  Die Pumpe kann zwar eine Förderhöhe von bis zu 80 cm erreichen, allerdings arbeitet die Pumpe nicht mit Unterdruck, sondern sie schiebt quasi das Wasser nur weiter. Damit sie also richtig funktioniert, müsste ein geschlossener Wasserkreislauf vorhanden sein. Der eigentliche Plan war es, die Pumpe in der oberen Box des Prototyps zu verbauen, wo sie nicht sichtbar ist. Da das Ansaugen von Wasser allerdings nicht funktioniert, und auch das Zurückschicken und Bestellen einer passenden Pumpe keine Option war (Abgabetermin heute), musste eine andere Lösung gefunden werden. Nachdem ca. 2 Stunden verschiedenste Ansätze getestet und das ganze Haus nach einer nicht benötigten Ersatz-Wasserpumpe durchsucht wurde, lag auf einmal die simpelste aller Lösungen auf der Hand. Die Pumpe kann einfach als Tauchpumpe verwendet werden, indem man sie selbst in die Flasche mit Wasser legt. Das beeinträchtigt zwar deutlich die Ästhetik des Prototyps (da die Kabel für die Pumpe dann nicht versteckt sind, sondern frei herumhängen), sorgt aber immerhin für die gewünschte Funktionalität.
- Bauteile in den Prototyp eingebaut und Kabel verlegt, sodass alle Kabel in der unteren Etage, wo letztlich der Arduino und die Platine platziert werden sollen, herauskommen.
  - ➔ Drucksensor wurde auf der „Zwischenplatte“ mit Klebeband so fixiert, dass der Elastikpuffer genau auf diesen drückt
  - ➔ Wasserschläuche wurden zugeschnitten und an die Pumpe und den Flow Sensor angebracht
  - ➔ Flow Sensor wurde in der oberen Box mit Heißkleber fixiert
  - ➔ LED-Streifen an der Innenseite der Glas-Abstell-Plattform angebracht
  - ➔ LED-Streifen außen an der Flaschenhalterung angebracht
  - ➔ Lautsprecher wurde an der Vorderseite des Prototyps angebracht. Dazu musste erst noch ein passendes Loch gebohrt werden, an dem dann der Lautsprecher mit Heißkleber fixiert werden konnte
  - ➔ Die Kabel von jedem Bauteil wurden mit Tape zusammengebunden, sodass man sich in dem Kabelsalat auskennt
- Schaltung wurde am Steckbrett umgesetzt

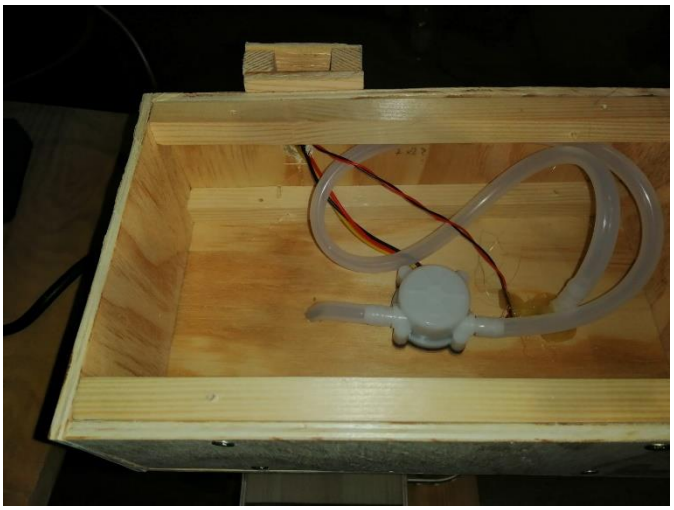


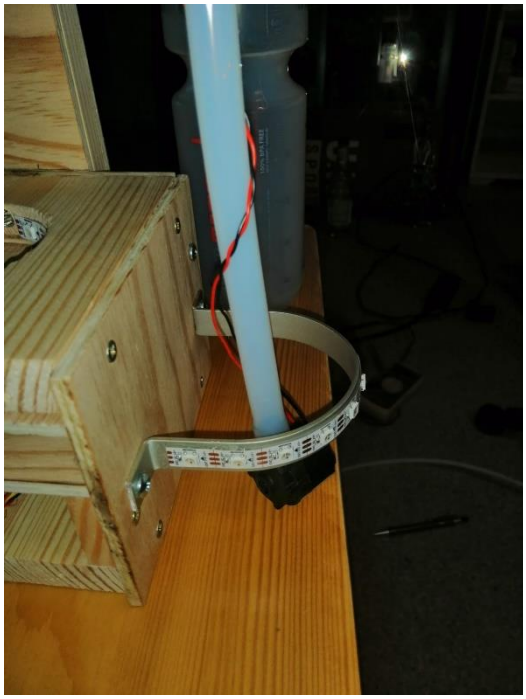


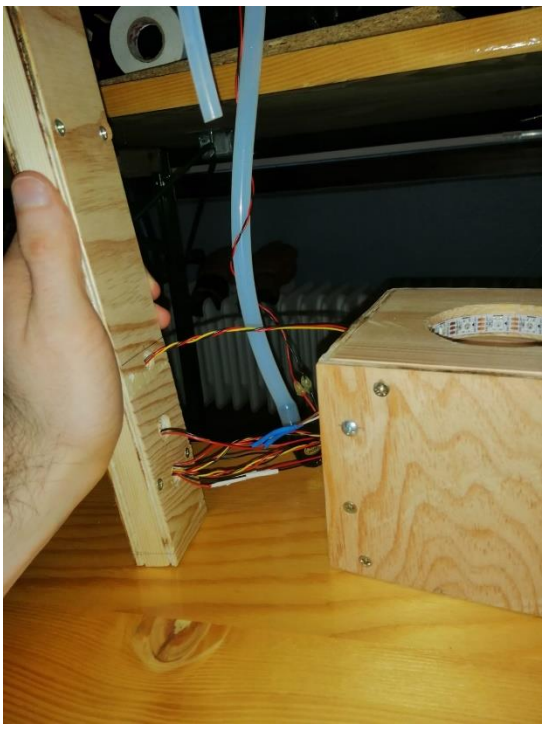

LM317


LM317

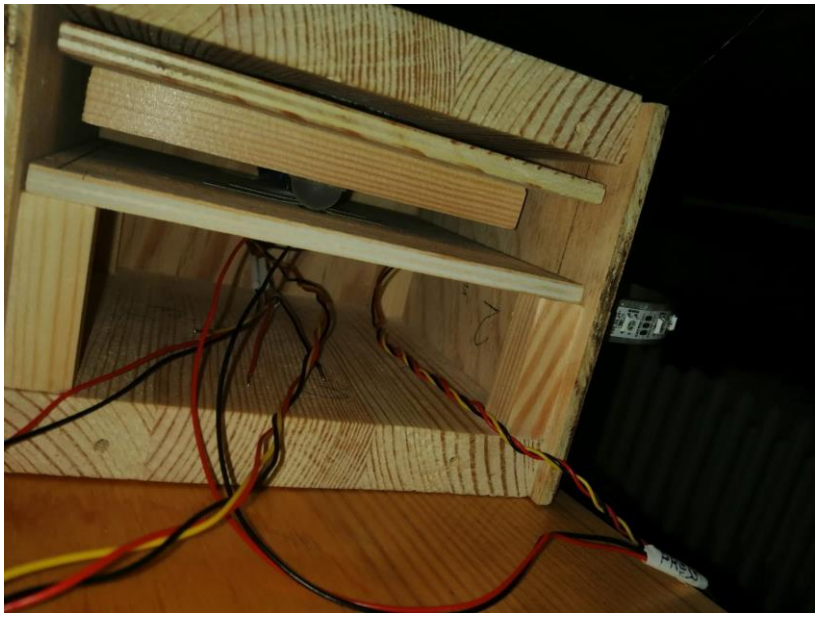









www.memmert.de
memmert



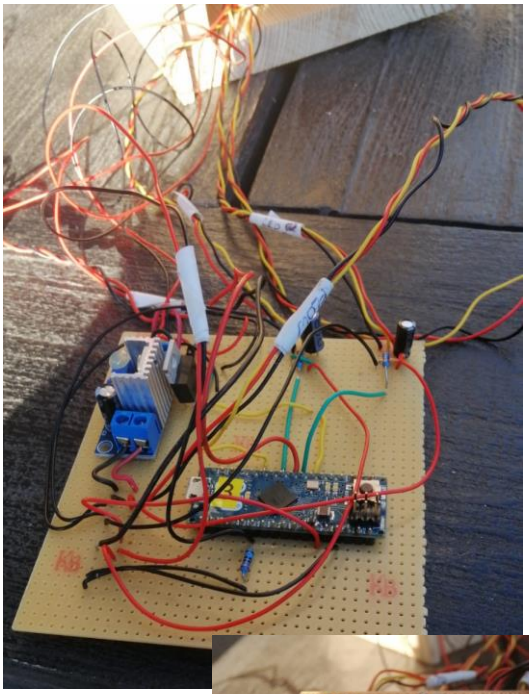
- Testen, ob alle Bauteile richtig angeschlossen sind, und ob die Schaltung an sich funktioniert
  - ➔ Mithilfe eines kleinen Testprogramms, siehe „Schaltungstest.ino“ im Anhang
- Mithilfe des Testprogramms wurden Probleme am Drucksensor festgestellt
  - ➔ Sensor liest immer nur den Wert 0 ein
  - ➔ Schaltung an sich schien korrekt zu sein
  - ➔ Aufschrauben des Prototyps, um zu sehen ob es Probleme an den Kabeln des Sensors gab
  - ➔ Anscheinen habe ich mein Verlegen der Kabel zu stark an denen des Drucksensors gezogen, wodurch ich dessen Beinchen abgerissen habe
- Reparieren des Drucksensors
  - ➔ Freilegen der Leiterbahnen, indem die dünne Plastikschick darüber entfernt wurde
  - ➔ Sensor mithilfe von Kabelschuhen an Litze befestigt
- Erneutes Durchlaufen des Testprogramms
  - ➔ Alles Sensoren und Bauteile haben funktioniert
  - ➔ Siehe Video „3_Schaltung_Testdurchlauf.mp4“

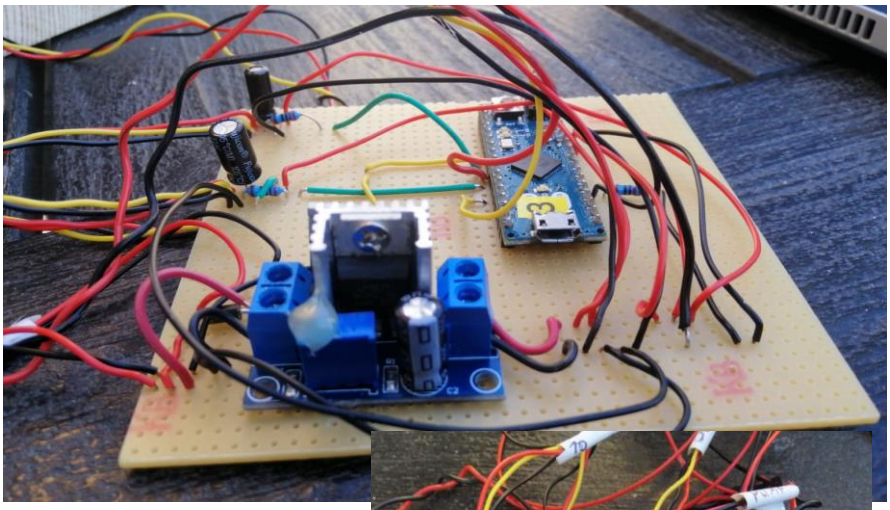
22.04.2021

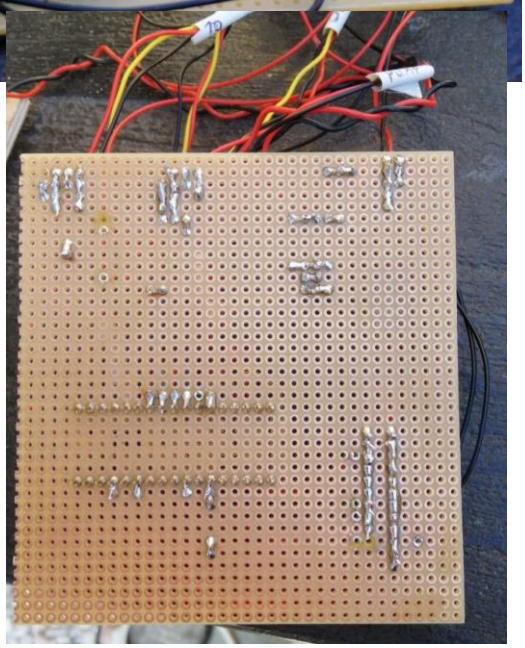
- Code testen:
  Mit dem vorgeschriebenen Code gab es (entgegen meiner Erwartungen) keine größeren Probleme. Das einzige, das nicht gut funktioniert hat, war das Stoppen der Pumpe, wenn der Flow Sensor kein durchfließendes Wasser mehr misst. Dazu sollte in jedem loop-Durchlauf überprüft werden, ob sich der waterFlow verändert hat seit dem letzten Durchlauf. Da allerdings die Loop Funktion mit ca. 117kHz aufgerufen wird, langen die Zeitabstände der Messungen nicht, damit ein Unterschied am WaterFlow gemessen werden könnte.

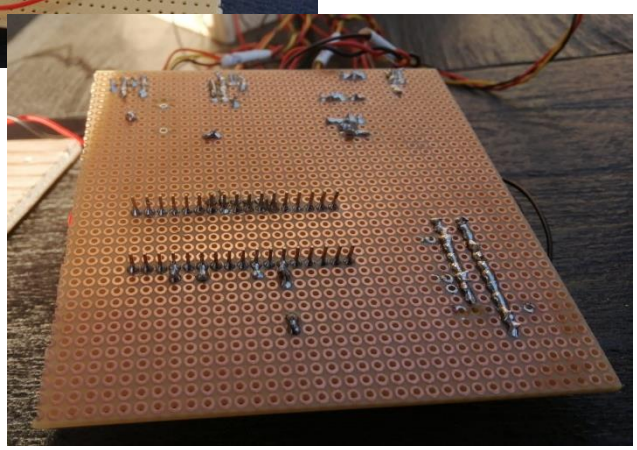
  Das Problem wurde also anders gelöst, indem einfach ein 10-Sekunden Timeout für die Auffülldauer gesetzt wurde. D.h. befindet sich das Programm länger als 10 Sekunden im Zustand RECHARGING, dann gilt die Flasche als leer.

  Ansonsten mussten nur kleinere Bugs, die schnell und einfach gefunden werden konnten (z.B. vergessen, den Waterflow mit 0 zu initialisieren oder wechseln in einen falschen Zustand), gefixt werden.
- Nachdem mit der Schaltung soweit alles funktioniert, habe ich alles auf einer Platine festgelötet
  → Anmerkung: Die PINS vom Arduino wurden extra nicht abgezwickt, sodass man diesen erneut für andere Projekte und auf dem Steckbrett nutzen kann
  → der Spannungsregler hatte keine Pins, weshalb ich diesen einfach mit Heißkleber befestigt habe
  → Wie die Kabel auf der Platine festgelötet werden, hätte vorher besser geplant werden können/ sollen, ich habe mehr oder weniger einfach drauf losgelegt









23.04.2021

Letzte Feinheiten am Prototyp

Bohren eines Lochs für den Anstecker für die Stromversorgung und Festkleben des Steckers mit Heißkleber

Bilder, ein Video („finaler Prototyp.mp4“) mit der Präsentation des finalen Prototyps (inklusive Erklärung der Funktionsweise), sowie der finale Code des Projektes sind im Anhang zu finden