Abschlussbericht VDE

Bau einer Seifenblasenmaschine im Rahmen des NWT Unterrichts in Klasse 10

Kepler Gymnasium Ulm

Fachlehrer: Kerstin Mechelke und Anita Lamprecht

Schulhof Kepler Gymnasium Ulm

Inhalt

Projektbeschreibung

Das Kepler Gymnasium bietet unter anderen Profilen auch das Profilfach NWT vierstündig von Klasse 8 bis 10 an. Ab Schuljahr 25/26 wird voraussichtlich auch für die Oberstufe das Basisfach NWT angeboten.

In Klasse 10 wird das Projekt „Bau einer Seifenblasenmaschine“ realisiert.

Zuerst lernen die Schülerinnen und Schüler die Grundlagen der Arduino Programmierung. Die Schülerinnen und Schüler arbeiten in 2-3er Teams nach dem Prinzip des agilen Projektmanagements (Scrum). Für den Bau der Maschine verwenden wir einen Arduino Uno. Die Schülerinnen und Schüler dürfen nun 2 aus 3 Elektromotoren aussuchen:

- Servomotor
- Schrittmotor
- Propellermodul.

Optional können die Schülerinnen und Schüler noch einen Taster zum An- und Ausschalten der Maschine, ein Potentiometer zur Regulation der Motorgeschwindigkeit, sowie ein LCD Display einbauen.

Der Kreativität sind aber keine Grenzen gesetzt. Es können auch noch andere Komponenten wie zum Beispiel ein Infrarotmodul mit Fernbedienung verbaut werden. Oder die Maschine kann über eine Lightshow oder über Sound verfügen.

Das Grundgerüst der Maschine besteht aus Legosteinen. Für die ausgewählten Motoren werden mit einer CAD Software[1] legokompatible Halterungen konstruiert und mit einem 3D Drucker ausgedruckt.

Die Seifenlösung optimieren die Schülerinnen und Schüler selbst unter Berücksichtigung der chemischen Grundlagen (Oberflächenspannung herabsetzen, polare Zutat für die Erhöhung der zwischenmolekularen Kräfte, Modellvorstellung auf Teilchenebene).

Im Unterrichtsverlauf werden auch die chemischen Grundlagen für die Herstellung von PLA für den 3D Drucker besprochen (Polykondensation) sowie umwelttechnische Probleme in Hinblick auf Kunststoffe und die Rolle der abbaubare Biokunststoffe.

Für den Unterricht ist es wichtig, dass wir mit externen Kooperationspartnern zusammenarbeiten:

1) Zusammenarbeit mit einem mittelständischen Unternehmen (Besuch des Betriebs – Vortrag *„Vergleich Ausbildung zum Fachinformatiker, BA Studium Informatik, Informatikstudium an der Universität“* – Programmierpatenschaften)
2) Zusammenarbeit mit einem DAX Unternehmen (Vortrag eines Scrum Masters zum Thema Biotech und Agilem Projektmanagement)

Der Zeitumfang beträgt ein Schulhalbjahr.

Alle Bauteile (außer Legobausteine) können auf https://funduinoshop.com/ bestellt werden. Hier findet man zu jedem Bauteil auch eine Anleitung, wie das Bauteil an den Arduino angeschlossen werden kann, und Beispielcodes.

[1] https://www.tinkercad.com/

Benötigte Materialien

	Link	Kosten
Legobausteine		
3d Drucker	https://www.conrad.de/de/p/flashforge-adventurer-5m-3d-drucker-beheizbares-druckbett-inkl-filament-flexibles-metallbett-3203033.html	350€
Arduinos (mit Kabel zum PC, optional mit Gehäuse)	https://funduinoshop.com/elektronische-module/sonstige/mikrocontroller/arduino-uno-rev3-das-original	20€
Steckbretter	https://funduinoshop.com/bauelemente/steckbretter-und-platinen/steckbretter/miniatur-breadboard-mit-170-steckplaetzen-div.-farben	1€
Kabel m/m	https://funduinoshop.com/bauelemente/kabelsysteme/jumper-kabel/40-stueck-breadboardkabel-maennlich/maennlich-20cm	2€
Kabel m/w	https://funduinoshop.com/bauelemente/kabelsysteme/jumper-kabel/40-stueck-breadboardkabel-maennlich/weiblich-20cm	2€
Schrittmotor	https://funduinoshop.com/elektronische-module/elektromotoren-und-zubehoer/schrittmotoren/schrittmotor-28byj-48-mit-uln2003-treiberplatine	3€
Propeller	https://funduinoshop.com/elektronische-module/sonstige/luefter/l9110-breakoutboard-mit-motor-und-propeller	3€
Servomotor	https://funduinoshop.com/elektronische-module/elektromotoren-und-zubehoer/servomotoren/funduino-sg90-servomotor-vgl.-towerpro	2,60€
Display	https://funduinoshop.com/elektronische-module/displays/lcd/20x04-2004-i2c-lcd-modul-hintergrundbeleuchtung-blau	7€
Taster	https://funduinoshop.com/bauelemente/taster-und-schalter/taster/taster-klein-mit-rundem-knopf	0,30€
IR Fernbedienung	https://funduinoshop.com/elektronische-module/wireless-iot/infrarot-ir/infrarotfernbedienungs-set	2€

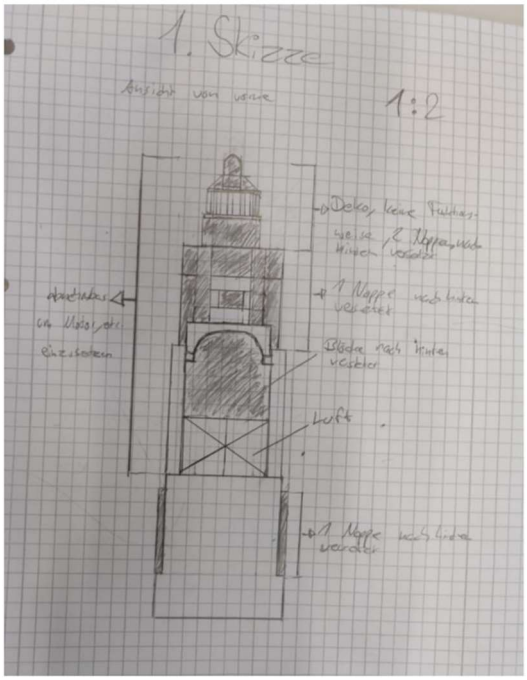
Mögliche auftretende Probleme

1) Abstand Propeller – Dispenser muss optimiert werden
2) Der Behälter muss passend, nicht zu groß, nicht zu klein und dicht sein
3) Wenn alle Bauteile an einen Arduino angeschlossen werden, kann man leicht den Überblick verlieren: zu viele Kabel, Fehler im Code. Eine Lösung hierfür wird in Unterrichtsbeispiel 3 gezeigt. Hier wird pro Bauteil ein Arduino verwendet.

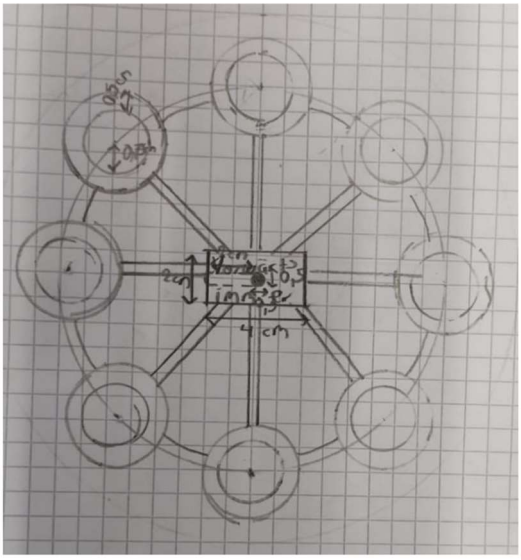
Unterrichtsbeispiele

Beispiel 1 „Ein Arduino, alle Bauteile“

Die folgenden Bilder stammen aus einer Unterrichtsdokumentation und zeigen den Weg von der Handskizze zur fertigen Maschine.



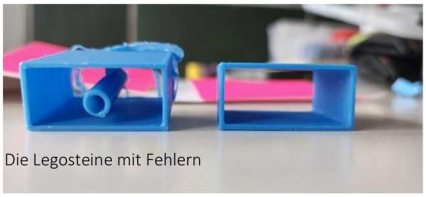
Der Legoturm



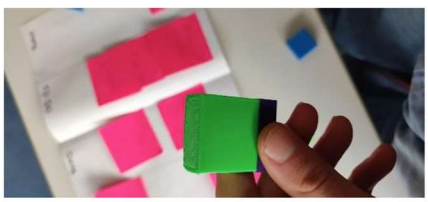
Der Dispenser (Maßstab: 2:1)

Zitat aus der Schülerdokumentation:

„Bis zum 23.11. hatten wir leider kein scharfes Foto des Scrumboards. Davor machten wir uns mit den Motoren und anderen Arduino-Komponenten bekannt und optimierten die Seifenblasenlösung. Ende September schafften wir es, die Lösung „perfekt“ zu machen. Danach lernten wir die Kompetenzen für Tinkercad und das Arduino Programm. Wir versuchten von Stunde zu Stunde unseren eigenen Legostein zu entwickeln. Dabei hatten wir mehrere Probleme: entweder die Wände waren zu dick oder zu dünn, oder ein Teil des Steins wurde nicht gedruckt.“



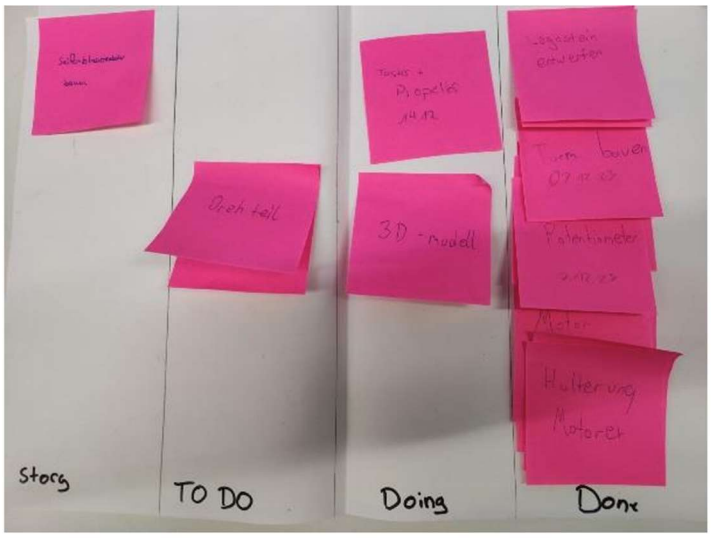
Die Legosteine mit Fehlern



Der funktionierende Legostein

Seifenblase aus eigener Lösung

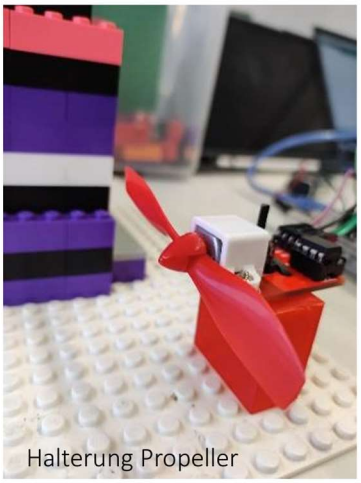
Das Scrumboard dient der Projektorganisation.



Scrumboard des 23.01.2024

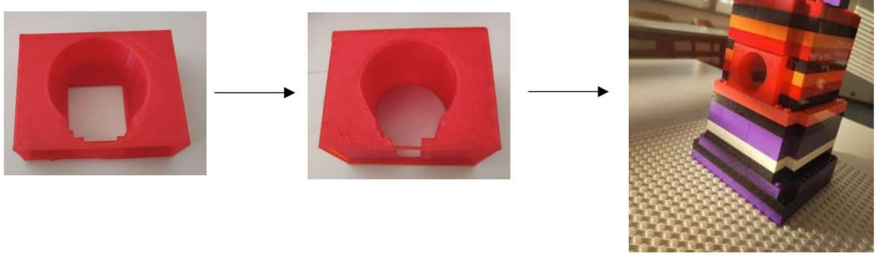
Bauteile werden mit Tinkercad gezeichnet und gedruckt.

- Legokompatible Halterung für den Propeller



Halterung Propeller

- Legokompatible Halterung für den Schrittmotor



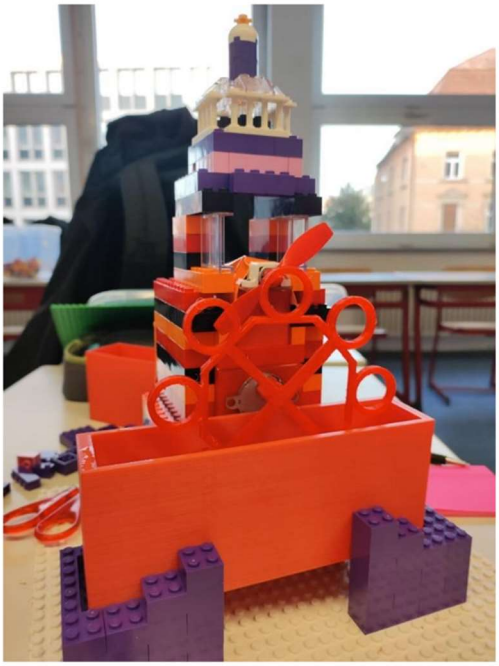
Halterung mit Fehler | passende Halterung | Eingebaute Halterung

Zitat aus Schülerdokumentation:

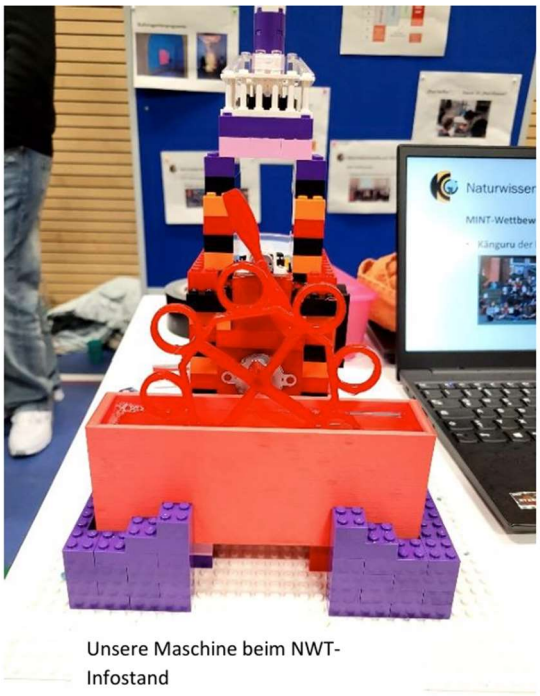
„Es trat leider ein Fehler mit einem Kabel auf, also entschieden wir uns, gleich auch das Kabelmanagement zu machen. D.h. wir ordneten die unterschiedlichen Aufgaben der Anschlüsse unterschiedlichen Farben zu (siehe Skizze Code und Schaltplan). Schülername erstellte dafür eine Skizze und ich sortierte die Kabel.“

Bild vom Kabel sortieren

Fertige Maschine

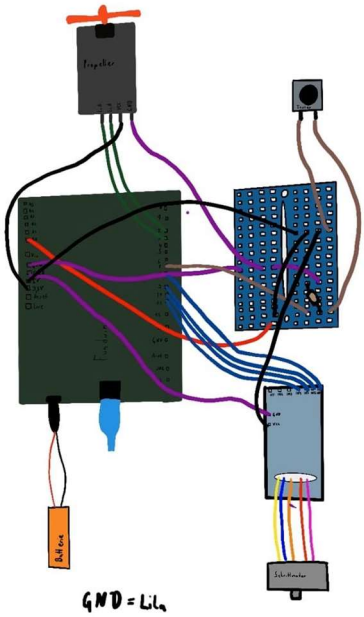


Erstmals zusammengebaute Maschine



Unsere Maschine beim NWT-Infostand

Schaltbild der Schüler

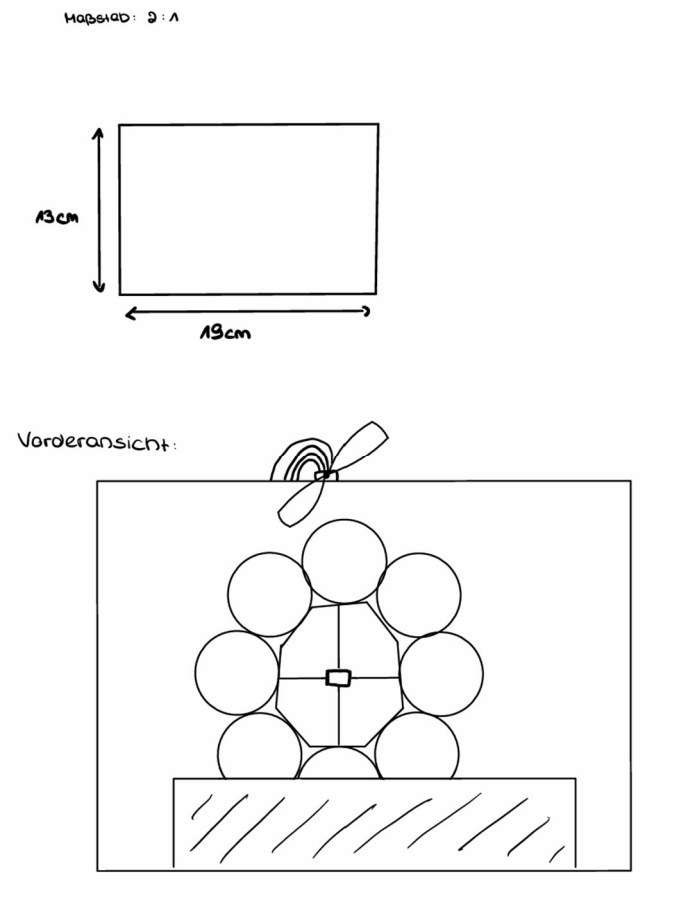


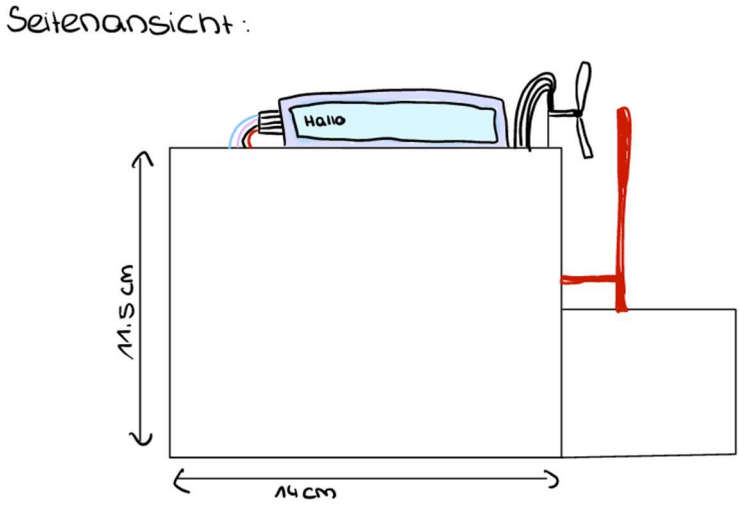
Zugehöriger Code der Schüler

```
#include <Stepper.h>
int P1=2;//Propelleranschluss an Pin 2
int P2=3;//Propelleranschluss an Pin §

int taster=7;// Taster an Pin 7
int tasterstatus=0;// Tasterstatus ist 0

int stepsPerRevolution = 2048; //Schritte pro Umdrehung
Stepper myStepper(stepsPerRevolution, 8, 9, 10, 11); //Schrittmotor an
Pins 8,9,10,11
int potPin = A0; //Pin, an dem der Potentiometer angeschlossen ist
double _speed;//Variable mit Dezimalstellen

void setup(){
  Serial.begin(9600);// Serieller Monitor wird gestartet


  pinMode(P1,OUTPUT);//Propeller als OUTPUT definiert
  pinMode(P2,OUTPUT);//Propeller als OUTPUT definiert

 // pinMode(taster, INPUT);


}

void loop(){
  int potValue = analogRead(potPin); //Lese den Potentiometer
   Serial.println(potValue);//potValue auf seriellen Monitor ablesen
   //Stepper Motor(potValue,8,9,10,11);

     _speed =potValue/200+1;//chnet die Geschwindigkeit für den
     Schrittmotor

    Serial.println(_speed);//speed des Schrittmotors auf seriellem
    Monitor ablesbar
    myStepper.setSpeed(_speed); //Setze die Geschwindigkeit des
    Schrittmotors
    myStepper.step(1);//Schrittmotor dreht


  tasterstatus=digitalRead(taster);//der Taster wird ausgelesen
  if(tasterstatus == HIGH)//Wenn der Taster gedrückt ist
  {
    analogWrite(P1,200);//Drehe den Propeller
    analogWrite(P2,0);
  }
  else //sonst…
  {
    analogWrite(P1,0);// Drehe den Propeller nicht
    analogWrite(P2,0);
 }


}
```

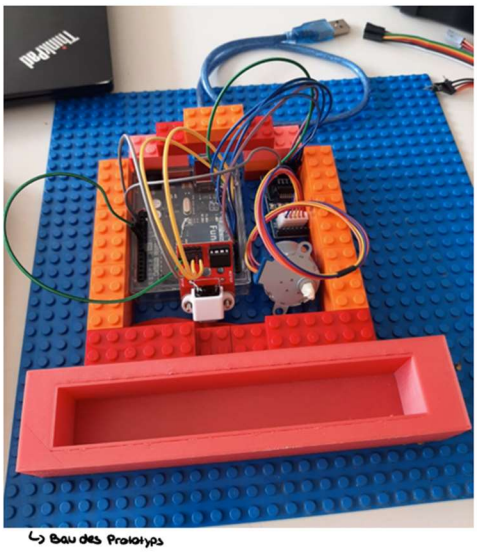
Beispiel 2 „Ein Arduino, alle Bauteile“

Handskizzen

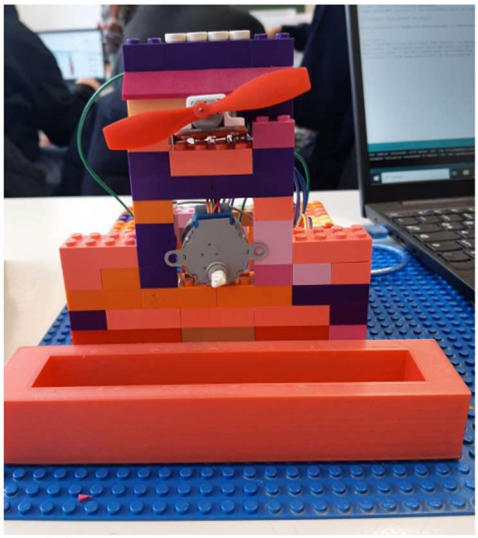




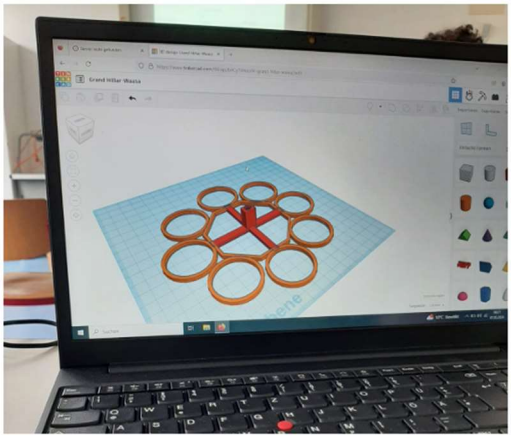
Fotos der Zwischenschritte



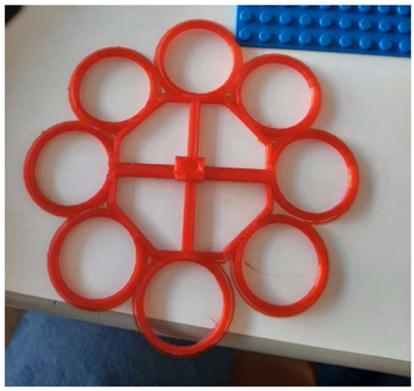
↳ Bau des Prototyps

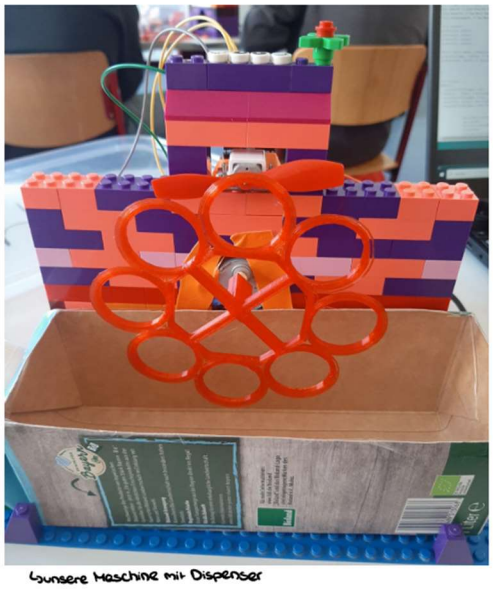


↳ erste Versuche das Lagergerüst aufzubauen

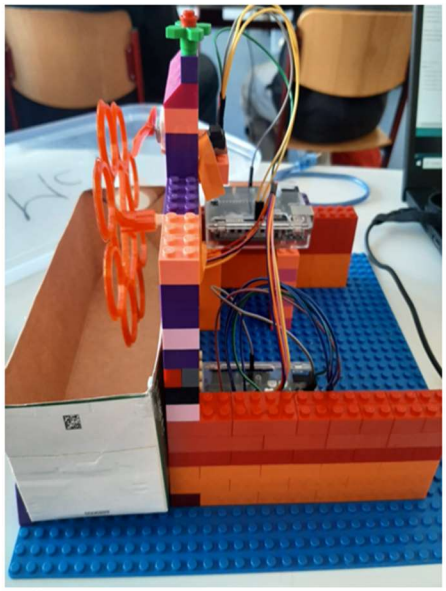


↳ erstellen unseres Dispensers mit „Tinker.cad"

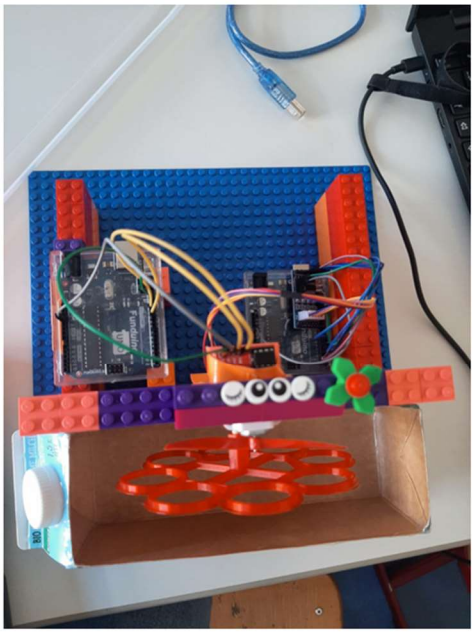


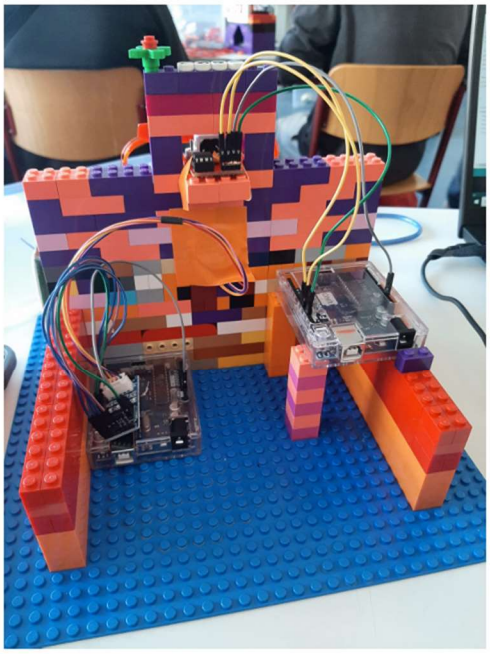


↳unsere Maschine mit Dispenser

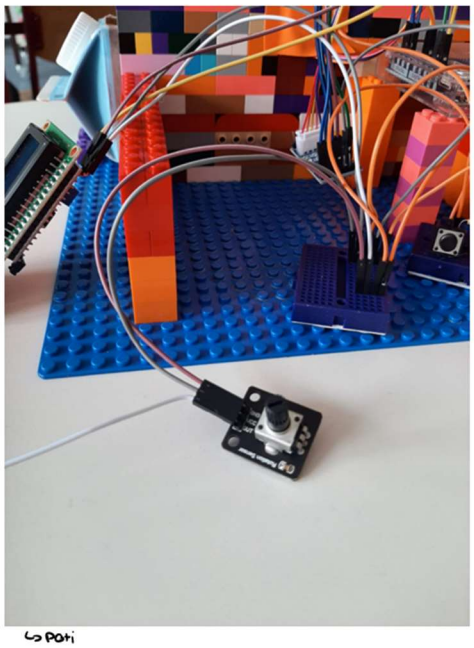


↳ Seitenansicht

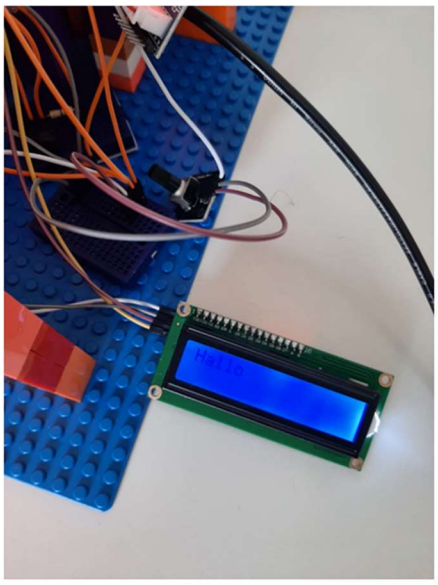




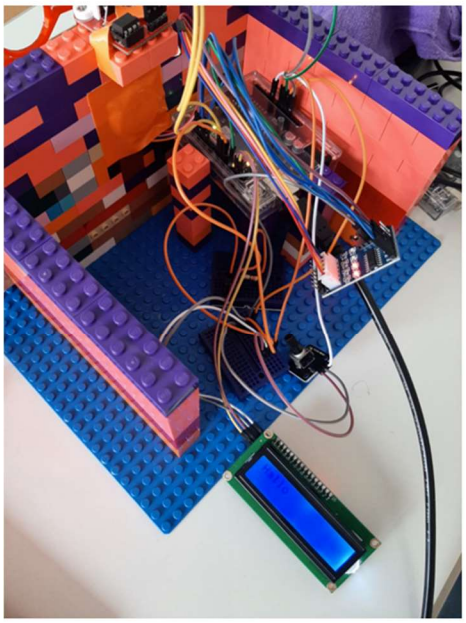
↳andere Ansichten



↳ Poti

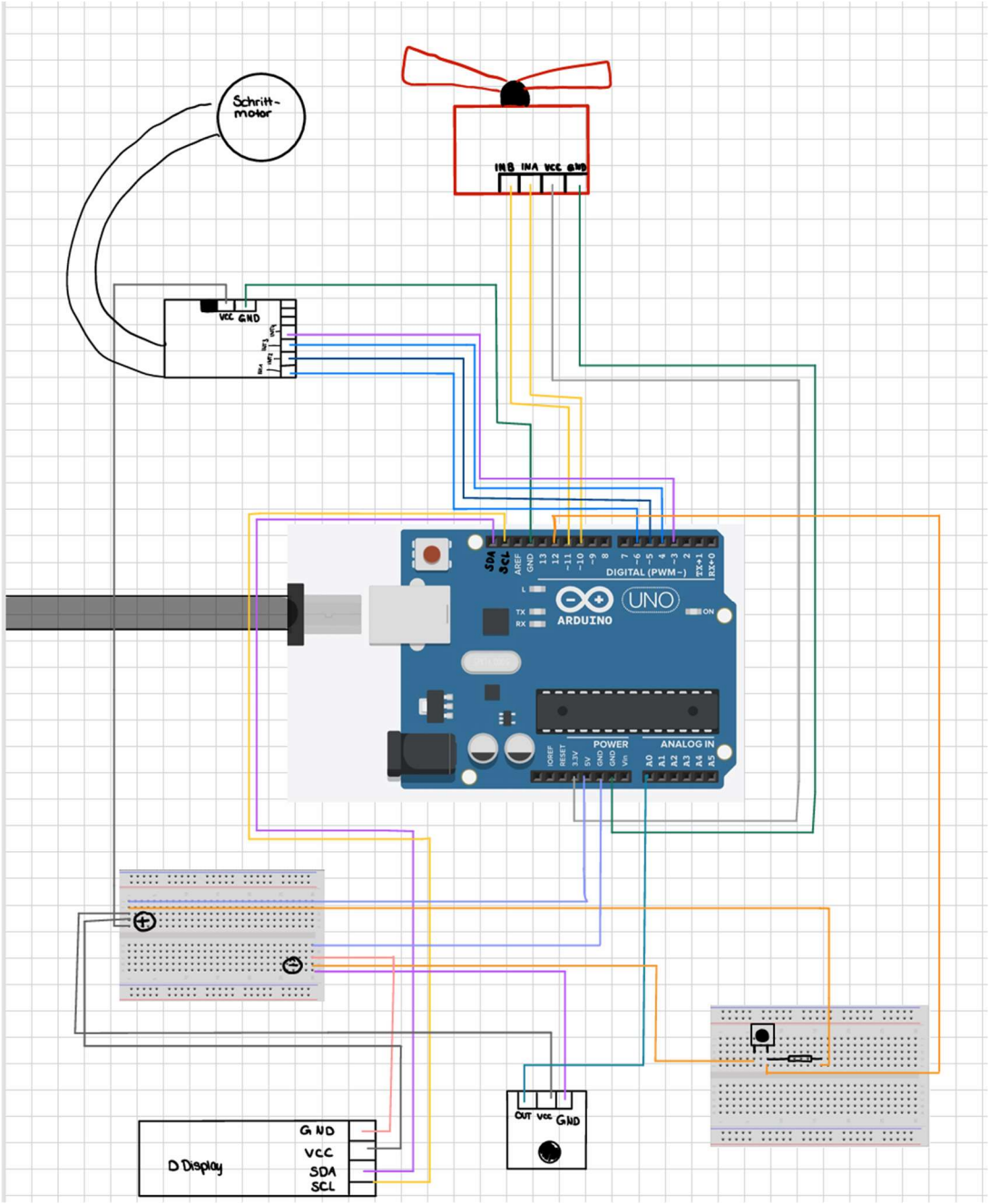


↳ unser Taster zeigt „Hallo" an



↳ Seifenblase

Schaltbild:


Schritt-
motor
IN8 IN4 VCC GND
VCC GND
ARDUINO
UNO
DIGITAL (PWM~)
POWER
ANALOG IN
OUT VCC GND
D Display
GND
VCC
SDA
SCL

Zugehöriger Code der Schüler

```
#include <Stepper.h> // Hinzufügen der Programmbibliothek.

int SPU = 2048; // Schritte pro Umdrehung.

Stepper Motor(SPU, 3,5,4,6); // Der Schrittmotor erhält die Bezeichnung "Motor" und es wird angegeben an welchen Pins der Motor angeschlossen ist.


#include <Wire.h> // Wire Bibliothek einbinden

#include <LiquidCrystal_I2C.h> //LCD-Bibliothek laden.

LiquidCrystal_I2C lcd(0x27, 16, 2); //Hier wird festgelegt um was für einen Display es sich handelt. In diesem Fall eines mit 16 Zeichen in 2 Zeilen und der HEX-Adresse 0x27. Für ein vierzeiliges I2C-LCD verwendet man den Code "LiquidCrystal_I2C lcd(0x27, 20, 4)"


int MotorA=10; //Pin INA am digitalen PWM Pin D10

int MotorB=11; //Pin INB am digitalen PWM Pin D11

int t;// Variable für Taster

int eingang= A0; //Das Wort „eingang“ steht jetzt für den Wert „A0“ (Bezeichnung vom Analogport 0). Hier wird das Signal des Potentiometers angeschlossen.

int sensorwert = 0; //Variable für den Sensorwert mit 0 als Startwert

int Geschwindigkeit = 0; //Variable für die Geschwindigkeit mit 0 als Startwert


void setup(){ //Hier beginnt das Setup.


Motor.setSpeed(5); // Angabe der Geschwindigkeit in Umdrehungen pro Minute.

pinMode(MotorA,OUTPUT); // Pin 10 (Motor-A) als Ausgang definieren

pinMode(MotorB,OUTPUT); // Pin 11 (Motor-B) als Ausgang definieren


pinMode(12,INPUT_PULLUP);//Pin 12 wird in den Modus INPUT_PULLUP gesetzt


Serial.begin(9600);// serieller Monitor wird angeschaltet


//lcd.begin(5, 1); //Im Setup wird angegeben, wie viele Zeichen und Zeilen verwendet werden. Hier: 5 Zeichen in 1 Zeile.

lcd.init(); //Im Setup wird der LCD gestartet

lcd.backlight(); //Hintergrundbeleuchtung einschalten (lcd.noBacklight(); schaltet die Beleuchtung aus).

}
```

```
void loop() {
//Taster
t=digitalRead(12);//momentaner Zustand des Tasters wird abgefragt
Serial.println(t);//...und angezeigt


if(digitalRead(12)==0){


//Poti
sensorwert =analogRead(eingang); //Die Spannung am Drehregler wird auslesen und als Zahl zwischen 0 und 1023 unter der Variable „sensorwert“ gespeichert.
Geschwindigkeit= map(sensorwert, 0, 1023, 0, 255); //Umwandeln des Sensorwertes mit Hilfe des "map" Befehls. Der Befehl wandelt den Sensorwert im Bereich 0-1023 um in einen Zahlenwert zwischen 0 und 255. Dadurch kann der Wert "Geschwindigkeit" direkt zur Ansteuerung im Befehl "analogWrite" verwendet werden.


Motor.step(1); // Der Motor macht 2048 Schritte, das entspricht einer Umdrehung.
analogWrite(MotorA, 0); // Zahl zwischen 0 und 255. Je höher die Differenz zwischen Motor-A und Motor-B ist, desto schneller dreht der Motor.
analogWrite(MotorB,Geschwindigkeit); // Zahl zwischen 0 und 255. Je höher die Differenz zwischen Motor-A und Motor-B ist, desto schneller dreht der Motor.
lcd.setCursor(0, 0); //Startposition der Darstellung auf dem LCD festlegen. lcd.Cursor(0,0) bedeutet: Erstes Zeichen in der ersten Zeile.


lcd.print("Hallo"); //Dort soll der Text "Hallo" erscheinen. Der Befehl lcd.setCursor ist dem Mikrocontrollerboard durch das Aufrufen der Bibliothek bekannt.
}
else{
 analogWrite(MotorA, 0);
 analogWrite(MotorB, 0);
}


}
```

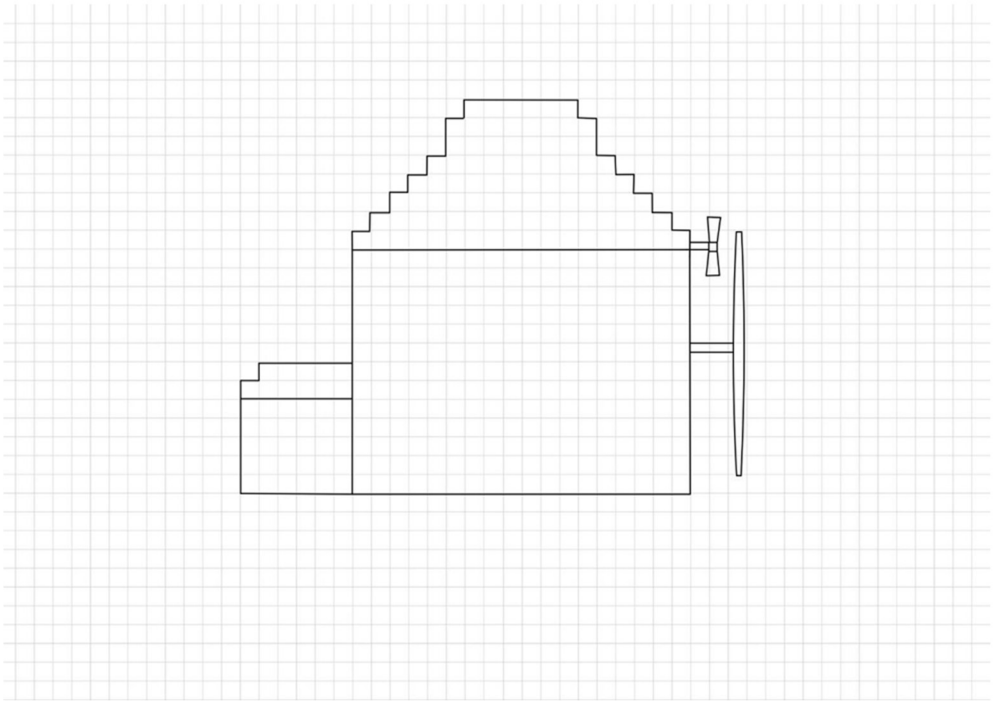
Zitat aus der Schülerdokumentation:

„Während diesem Projekt wurden wir vor viele Herausforderungen gestellt. Wir (Schülername und ich) hatten vor dem Projekt im ersten Halbjahr („Bau eines Fahrroboters“) noch keinerlei Erfahrungen mit dem Ardino Programm oder „Tinker.cad“ und mussten uns so alles neu erarbeiten. Im zweiten Halbjahr konnten wir unsere gewonnenen Erfahrungen im neuen Projekt direkt wieder anwenden und noch erweitern, was nicht immer einfach war, da immer neue Herausforderungen entstanden. Alles in allem bewältigten wir jedoch alle Probleme als Team oder durch das Fragen nach Hilfe. Nebenher führten wir regelmäßig das Scrumboard und den Arbeitsplan und dokumentieren alles mit Videos und Bildern. Durch dieses Projekt habe ich vieles darüber gelernt, wie wichtig es ist alles gut zu dokumentieren und organisiert vorzugehen.

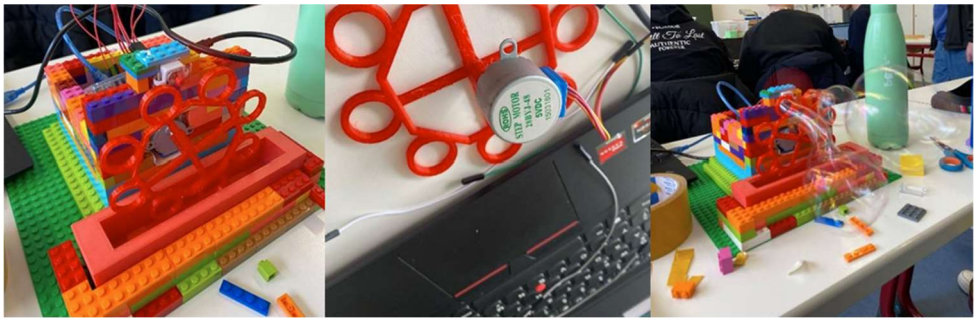
Eine gute Arbeitsteilung ist von großer Wichtigkeit und sollte nie unterschätzt werden, wenn man zu einem guten Endergebnis kommen will. Bei auftretenden Problemen sollte man nicht direkt verzweifeln sondern zunächst sachlich vorgehen und versuchen auf eine Lösung zu kommen. Wenn das nicht gelingt kann man andere um Hilfe bitten doch sollte man immer darum bemüht bleiben den Lösungsweg auch nachvollziehen zu können, um die Fehler nicht immer wieder zu wiederholen. Ich werde dieses Projekt als sehr nervenaufreibend, aber auch lehrreich in Erinnerung behalten, da ich zwar oft nicht direkt wusste, wie ich vorgehen musste, doch mit gutem Informieren und Strukturiertem Vorgehen immer eine Lösung finden konnte.“

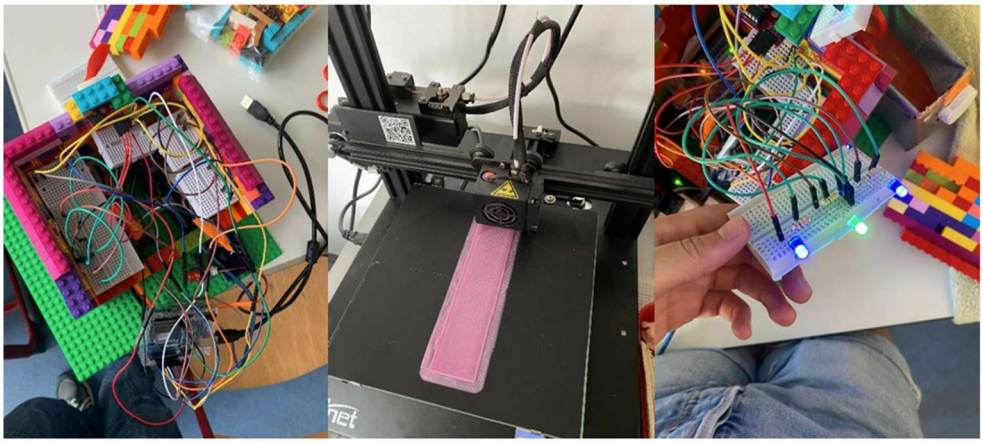
Beispiel 3 „Ein Arduino pro Bauteil – aber mit Lightshow“

Handskizze



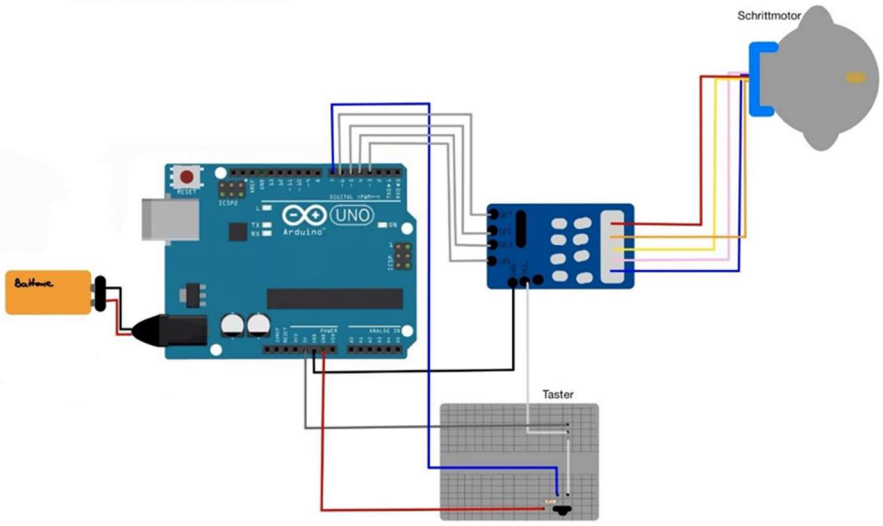
Bilder zum Entstehungsprozess



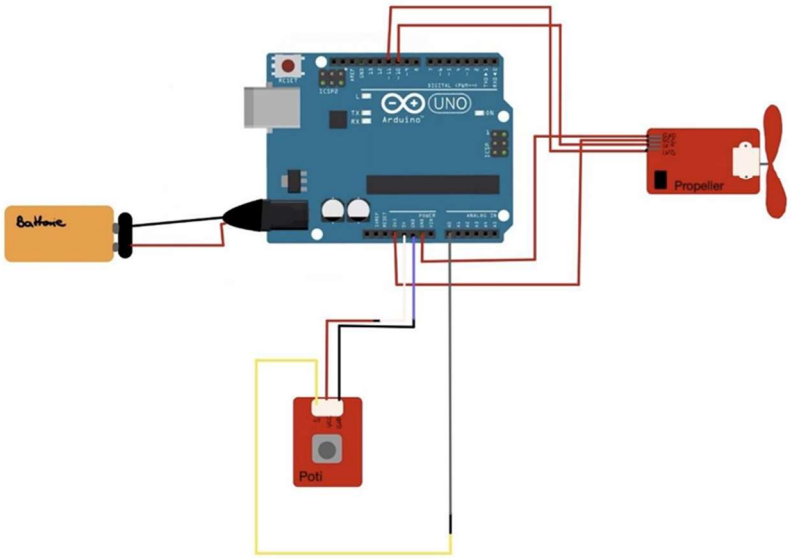


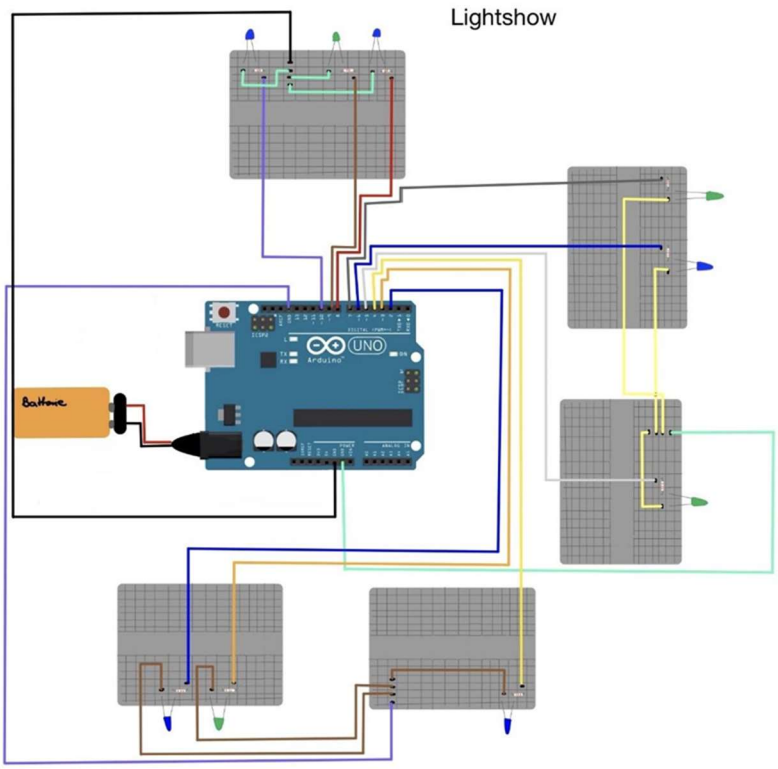
Schaltbild

Schrittmotor und Taster



Propeller und Poti





Zugehöriger Schülercode

Schrittmotor :[2]

```
// Schrittmotor
#include <Stepper.h>
int SPU = 2048;
Stepper Motor(SPU, 3,5,4,6);
int Taster=7; // Taster an Pin7
int Tasterstatus=0; // Variable für den Status des Tasters


void setup()
```

[1] Nr.13 Servo ansteuern - Funduino - Kits und Anleitungen für Arduino (ich habe nur den Arduino herausgenommen)

[2] Schrittmotor - Funduino - Kits und Anleitungen für Arduino (Beispielsketch 2)

```
{
Motor.setSpeed(4);
pinMode(Taster, INPUT);
}

void loop()
{

//Taster

Tasterstatus=digitalRead(Taster); //Zunächst wird der Taster abgefragt.

while(Tasterstatus == LOW) //Solange der Wert des Tasters LOW, also gedrückt ist,...

 { //...springt der Sketch in diese Klammer und verbleibt hier.

//Schrittmotor
Motor.step(341); //Solange der Taster gedrückt ist, dreht der Motor immer 1/6 Drehung stoppt, dreht 1/6
stoppt usw.
Tasterstatus=digitalRead(Taster); //Innerhalb der Schleife muss der Taster immer wieder abgefragt
werden, damit eine Änderung des Status vom Mikrocontroller erkannt und verarbeitet werden kann. Nur
dadurch kann die Schleife beim loslassen des Tasters wieder verlassen werden.
 }
 }
```

Propeller:[3]

```
int MotorA=10; //Pin INA am digitalen PWM Pin D10
int MotorB=11; //Pin INB am digitalen PWM Pin D11
int eingang= A0; //Das Wort „eingang“ steht jetzt für den Wert „A0“ (Bezeichnung vom Analogport 0).
Hier wird das Signal des Potentiometers angeschlossen.
int sensorwert = 0; //Variable für den Sensorwert mit 0 als Startwert
int Geschwindigkeit = 0; //Variable für die Geschwindigkeit mit 0 als Startwert
```

[3] DC-Motor mit Propeller ansteuern - Funduino - Kits und Anleitungen für Arduino (Beispielsketch 2)


```
void setup()
{
pinMode(MotorA,OUTPUT); // Pin 10 (Motor-A) als Ausgang definieren
pinMode(MotorB,OUTPUT); // Pin 11 (Motor-B) als Ausgang definieren
}

void loop()
{
sensorwert =analogRead(eingang); //Die Spannung am Drehregler wird auslesen und als Zahl zwischen 0
und 1023 unter der Variable „sensorwert“ gespeichert.
Geschwindigkeit= map(sensorwert, 0, 1023, 0, 255); //Umwandeln des Sensorwertes mit Hilfe des "map"
Befehls. Der Befehl wandelt den Sensorwert im Bereich 0-1023 um in einen Zahlenwert zwischen 0 und
255. Dadurch kann der Wert "Geschwindigkeit" direkt zur Ansteuerung im Befehl "analogWrite"
verwendet werden.
```

```
analogWrite(MotorA, Geschwindigkeit); // Der Wert bleibt bei 0. Die Veränderung des Wertes "MotorB"
führt in diesem Sketch zur Veränderung der Motorgeschwindigkeit.
analogWrite(MotorB, 0); // Durch die Variable "Geschwindigkeit" erfolgt je nach vorherig erfasstem
Sensorwert des Potentiometers eine Veränderung der Motorgeschwindigkeit.
}
```

Lightshow:

```
void setup() {
  pinMode(2,OUTPUT); // Pin 2 als Ausgang definieren
  pinMode(3,OUTPUT); // Pin 3 als Ausgang definieren
  pinMode(4,OUTPUT); // Pin 4 als Ausgang definieren
  pinMode(5,OUTPUT); // Pin 5 als Ausgang definieren
  pinMode(6,OUTPUT); // Pin 6 als Ausgang definieren
  pinMode(7,OUTPUT); // Pin 7 als Ausgang definieren
  pinMode(8,OUTPUT); // Pin 8 als Ausgang definieren
  pinMode(9,OUTPUT); // Pin 9 als Ausgang definieren
  pinMode(10,OUTPUT);
}

void loop() {
```

```
 digitalWrite(2,HIGH); // Pin 2 wird HIGH geschaltet
 delay (100); // Dauer: 100 Millisekunden
 digitalWrite(3,HIGH); // Pin 3 wird HIGH geschaltet
 delay (100) // Dauer: 100 Millisekunden
 ;digitalWrite(4,HIGH); // Pin 4 wird HIGH geschaltet
 delay (100); // Dauer: 100 Millisekunden
 digitalWrite(5,HIGH); // Pin 5 wird HIGH geschaltet
 delay (100); // Dauer: 100 Millisekunden
 digitalWrite(6,HIGH); // Pin 6 wird HIGH geschaltet
 delay (100); // Dauer: 100 Millisekunden
 digitalWrite(7,HIGH); // Pin 7 wird HIGH geschaltet
 delay (100); // Dauer: 100 Millisekunden
 digitalWrite(8,HIGH); // Pin 8 wird HIGH geschaltet
 delay (100); // Dauer: 100 Millisekunden
 digitalWrite(9,HIGH); // Pin 9 wird HIGH geschaltet
 delay (100); // Dauer: 100 Millisekunden
 digitalWrite(10,HIGH); // Pin 10 wird HIGH geschaltet
 delay (300); // Dauer: 300 Millisekunden

 digitalWrite(10,LOW); // Pin 10 wird LOW geschaltet
 delay (100); // Dauer: 100 Millisekunden
 digitalWrite(9,LOW); // Pin 9 wird LOW geschaltet
 delay (100); // Dauer: 100 Millisekunden
 digitalWrite(8,LOW); // Pin 8 wird LOW geschaltet
 delay (100); // Dauer:  100 Millisekunden
 digitalWrite(7,LOW); // Pin 7 wird LOW geschaltet
 delay (100); // Dauer: 100 Millisekunden
 digitalWrite(6,LOW); // Pin 6 wird LOW geschaltet
 delay (100); // Dauer: 100 Millisekunden
 digitalWrite(5,LOW); // Pin 5 wird LOW geschaltet
 delay (100); // Dauer: 100 Millisekunden
 digitalWrite(4,LOW); // Pin 4 wird LOW geschaltet
 delay (100); // Dauer: 100 Millisekunden
 digitalWrite(3,LOW); // Pin 3 wird LOW geschaltet
 delay (100); // Dauer: 100 Millisekunden

digitalWrite(2,LOW); // Pin 2 wird LOW geschaltet
delay (100); // Dauer: 100 Millisekunden



}
```

Fertige Maschine:

Zitat aus Schülerdokumentation:

„Was hätte besser sein können? Wir hatten größtenteils keine Probleme. Wenn Probleme aufgetreten sind, waren es meistens eher kleinere Dinge, die schnell wieder zu beheben waren. Wir mussten unseren Propeller mehrfach austauschen, da dieser sich zu langsam gedreht hat und dadurch keine Seifenblasen entstanden sind. Unsere LEDs sind mehrmals von der Wand an der sie angebracht waren abgefallen und wir mussten sie neu anbringen. Das war ein bisschen nervig, da wir dafür unser gesamtes Dach auseinander bauen mussten. Unter anderem wollten wir noch unsere Wanne für die Seifenblasenflüssigkeit mit dem 3D-Drucker drucken, das hat aber leider nicht funktioniert. Deswegen haben wir einfach eine aus einer Milchtüte gebastelt. Und unser Kabel Management war nicht optimal, da wir sehr viele Kabel haben die kreuz und quer verteilt sind. Eventuell hätten wir uns dafür eine bessere Lösung überlegen können

Was war gut? Wir haben während des Projekts sehr viel Neues über Technik gelernt. Da unsere Vorkenntnisse gering waren, war es eine gute Gelegenheit um aus unserer Komfortzone zu kommen und neue Dinge zu lernen. Außerdem finde ich es schön, dass wir die Möglichkeit hatten praktische Erfahrungen zu sammeln. Dies ist eine gute Abwechslung zu dem sonst eher theoretischen Unterricht. Ich hatte sehr viel Spaß bei der Realisierung des Projekts, da das Arbeitsklima innerhalb unserer Gruppe sehr angenehm und entspannt war. Wir haben uns gut miteinander abgesprochen und zusammen harmoniert. Wir hatten durch unser gutes Zeitmanagement nie Stress und ein angenehmes Arbeitsklima. Insgesamt hat generell alles sehr gut funktioniert, wir hatten wenig Probleme und sehr viel Spaß bei diesem Projekt.“

Codebeispiele[2]

Potentiometer und Propeller

```
int MotorA=10; //Pin INA am digitalen PWM Pin D10

int MotorB=11; //Pin INB am digitalen PWM Pin D11

int eingang= A0; //Das Wort „eingang“ steht jetzt für den Wert „A0“ (Bezeichnung vom Analogport 0). Hier wird das Signal des Potentiometers angeschlossen.

int sensorwert = 0; //Variable für den Sensorwert mit 0 als Startwert

int Geschwindigkeit = 0; //Variable für die Geschwindigkeit mit 0 als Startwert


void setup()

{

pinMode(MotorA,OUTPUT); // Pin 10 (Motor-A) als Ausgang definieren

pinMode(MotorB,OUTPUT); // Pin 11 (Motor-B) als Ausgang definieren

}


void loop()

{

sensorwert =analogRead(eingang); //Die Spannung am Drehregler wird auslesen und als Zahl zwischen 0 und 1023 unter der Variable „sensorwert“ gespeichert.

Geschwindigkeit= map(sensorwert, 0, 1023, 0, 255); //Umwandeln des Sensorwertes mit Hilfe des "map" Befehls. Der Befehl wandelt den Sensorwert im Bereich 0-1023 um in einen Zahlenwert zwischen 0 und 255. Dadurch kann der Wert "Geschwindigkeit" direkt zur Ansteuerung im Befehl "analogWrite" verwendet werden.


analogWrite(MotorA, 0); // Der Wert bleibt bei 0. Die Veränderung des Wertes "MotorB" führt in diesem Sketch zur Veränderung der Motorgeschwindigkeit.
```

[2] Als Grundlage dienen die Codebeispiele unter https://funduinoshop.com/

```
analogWrite(MotorB, Geschwindigkeit); // Durch die Variable "Geschwindigkeit" erfolgt je nach vorherig erfasstem Sensorwert des Potentiometers eine Veränderung der Motorgeschwindigkeit.

}
```

Potentiometer und Servomotor

```
// eingebaute Bibliothek einbinden

# include <Servo.h>

// Bezeichnung des Motors

Servo Motor;

// speichert den analogen Wert des Drehpotentiometers

int ReglerWert;

// Position des Motors

int Position;


void setup()

{

 // Motor an Pin 9 angeschlossen (attach)

 Motor.attach(9);

}

void loop()

{

 int ReglerWert = analogRead(A0);

 /*

 map -> Umwandlung des gelesenen Wertes

 von 0 bis 1023 (analoger Sensorwert)

 auf 0 bis 180 (Drehung des Motors)

 */

 Position = map(ReglerWert, 0, 1023, 0, 180);

 // Motor zur Position bewegen

  Motor.write(Position);}
```

Propeller, Potentiometer, Schrittmotor, Taster, Display

```
//Propeller und Poti

int MotorA=10; //Pin INA am digitalen PWM Pin D10

int MotorB=11; //Pin INB am digitalen PWM Pin D11

int eingang= A0; //Das Wort „eingang“ steht jetzt für den Wert „A0“ (Bezeichnung vom Analogport 0). Hier wird das Signal des Potentiometers angeschlossen.

int sensorwert = 0; //Variable für den Sensorwert mit 0 als Startwert

int Geschwindigkeit = 0; //Variable für die Geschwindigkeit mit 0 als Startwert



// Schrittmotor

#include <Stepper.h>

int SPU = 2048;

Stepper Motor(SPU, 3,5,4,6);

int Taster=7; // Taster an Pin7

int Tasterstatus=0; // Variable für den Status des Tasters



// Display

#include <Wire.h> // Wire Bibliothek einbinden

#include <LiquidCrystal_I2C.h> // Vorher hinzugefügte LiquidCrystal_I2C Bibliothek einbinden

LiquidCrystal_I2C lcd(0x27, 16, 2); //Hier wird festgelegt um was für einen Display es sich handelt. In diesem Fall eines mit 16 Zeichen in 2 Zeilen und der HEX-Adresse 0x27. Für ein vierzeiliges I2C-LCD verwendet man den Code "LiquidCrystal_I2C lcd(0x27, 20, 4)"



void setup()

{

// Propeller
```

```
pinMode(MotorA,OUTPUT); // Pin 10 (Motor-A) als Ausgang definieren

pinMode(MotorB,OUTPUT); // Pin 11 (Motor-B) als Ausgang definieren

//Schrittmotor

Motor.setSpeed(4);

pinMode(Taster, INPUT);

//Display

lcd.init(); //Im Setup wird der LCD gestartet

lcd.backlight(); //Hintergrundbeleuchtung einschalten (lcd.noBacklight(); schaltet die Beleuchtung
aus).

}


void loop()

{


//Taster


Tasterstatus=digitalRead(Taster); //Zunächst wird der Taster abgefragt.


while(Tasterstatus == HIGH) //Solange der Wert des Tasters HIGH, also gedrückt ist,...


 { //...springt der Sketch in diese Klammer und verbleibt hier.


//Poti + Propeller

sensorwert =analogRead(eingang);

//Die Spannung am Drehregler wird auslesen

//und als Zahl zwischen 0 und 1023 unter der Variable „sensorwert“ gespeichert.
```

Geschwindigkeit= map(sensorwert, 0, 1023, 0, 255);

//Umwandeln des Sensorwertes mit Hilfe des "map" Befehls.

//Der Befehl wandelt den Sensorwert im Bereich 0-1023 um in einen Zahlenwert zwischen 0 und 255. Dadurch kann der Wert "Geschwindigkeit" direkt zur Ansteuerung im Befehl "analogWrite" verwendet werden.

analogWrite(MotorA, 0); // Der Wert bleibt bei 0. Die Veränderung des Wertes "MotorB" führt in diesem Sketch zur Veränderung der Motorgeschwindigkeit.

analogWrite(MotorB, Geschwindigkeit); // Durch die Variable "Geschwindigkeit" erfolgt je nach agt werden, damit eine Änderung des Status vom Mikrocontroller erkannt und verarbeitet werden kann. Nur dadurch kann die Schleife beim loslassen des Tasters wieder verlassen werden.

//Schrittmotor

Motor.step(341); //Solange der Taster gedrückt ist, dreht der Motor immer 1/6 Drehung stoppt, dreht 1/6 stoppt usw.

delay(1000);

Tasterstatus=digitalRead(Taster); //Innerhalb der Schleife muss der Taster immer wieder abgefragt werden, damit eine Änderung des Status vom Mikrocontroller erkannt und verarbeitet werden kann. Nur dadurch kann die Schleife beim loslassen des Tasters wieder verlassen werden.

//Display

lcd.setCursor(0, 0);//Hier wird die Position des ersten Zeichens festgelegt. In diesem Fall bedeutet (0,0) das erste Zeichen in der ersten Zeile.

lcd.print("Viel Spaß");

lcd.setCursor(0, 1);// In diesem Fall bedeutet (0,1) das erste Zeichen in der zweiten Zeile.

lcd.print("mit den Seifenblasen");

}

delay(1);//Wenn der Taster nicht mehr gedrückt ist, springt der Sketch in den regulären Loop zurück und führt dort weitere Befehle aus.

//Der Schrittmotor stoppt}