Projektdokumentation Projekt E21-H1

Mark Mallon, Alexander Riemer ,Sascha Link

Inhalt

1.Projektidee
• Erweiterung des T1 Bausatzes

2.Planung
• Materialliste

3.Durchführung
•Sensorik
• Programmierung
• Fritzing
• Additive Fertigung
- Ultimaker Cura
- - Creality CR-10S Pro

4.Fertigstellung
- • Montage
- • Funktionsnachweis

1.Projektidee

• Erweiterung des T1 Bus um eine Automatische
Fahrlichtschaltung sowie, einer Alarmanlage

Materialplanung

Benötigt wurde folgendes Material :

• Mikrocontroller (ESP32)
• Breadboard
• Jumper Kabel
• Rote, Weisse und Gelbe Led‘s mit passendem Vorwiderständen
• Infrarotsensor (HC-SR501)
• Sensorhalterung (Erstellung mit 3D-Druck)
• Buzzer
• Micro-USB Kabel zur Spannungsversorgung und Programmierung des Mikrocontrollers

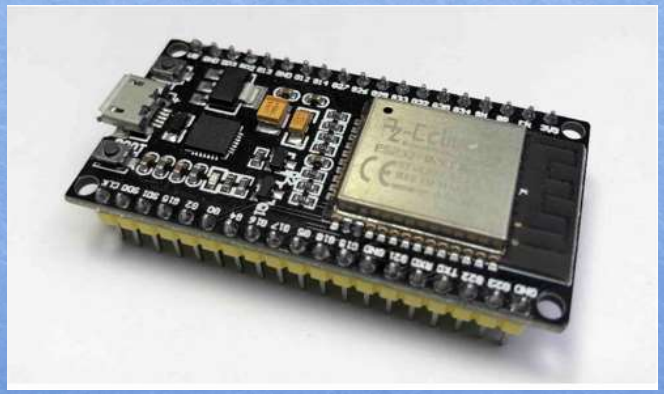
3.Aktiver Buzzer

- Den aktiven Buzzer muss man nur die entsprechende Spannung (3,3 - 5,0 Volt) zuweisen, um über den Oszillator des Buzzers in ein Tonsignal zu erhalten.
- Bei einem passiven Buzzer muss der ESP 32 die Funktion des Oszillators übernehmen, und über die Funktion tone () programmiert werden.
- Dies ist der Grund, warum wir uns für den aktiven Buzzer entschieden haben.

3.ESP 32 Mikrocontroller

Beschreibung:

Der ESP32-WROOM-32 dient zur Steuerung und Vernetzung von verschiedenen Sensoren oder antriebstechnischen Baueinheiten.



Kenndaten:
- Betriebsspannung: 3,3 V oder 5,0 V
- Betriebstemperaturbereich: –40 °C bis 125 °C
- RAM (Arbeitsspeicher): 512 Kilobyte
- CPU (Prozessor): 2 Kerne mit einer Taktfrequenz von 240 MHz
- WLAN-Modul
- Bluetooth-Modul

ESP32-WROOM-32 ist ein leistungsstarker Wi-Fi+BT+BLE-MCU-Modul Mikrocontroller, der eine Vielzahl von Anwendungen, die von Low-Power-Sensornetzwerken bis hin zu den anspruchsvollsten Aufgaben wie Sprachcodierung standhält.

Der ESP 32 kann mit Arduino IDE Programmiert werden, und ist für Anwendungen auf dem Breadboard geeignet.

Mit seinen 34 universelle Ein- und Ausgänge (GPIOs => General Purpose Input / Out- put) sind eine Vielzahl von verschiedenen Anwendungen gleichzeitig durch die Programmierung als analoge oder digitale Ein- und Ausgänge verwendbar.

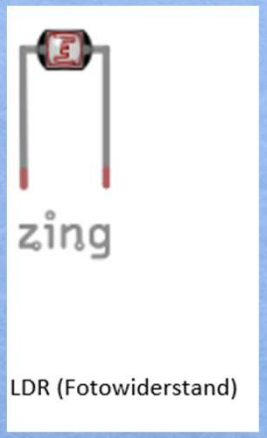
3.Infrarotsensor HC-SR501:

Dieser Infrarotsensor gibt bei Erkennung einer Bewegung auf seinem Datenpin eine Spannung von 3,3 V aus. Dabei sendet der Sensor Wellen an seine Umgebung aus, welche an Hindernissen reflektiert werden und wieder zurück zum Sensor gelangen. Die ankommenden Wellen wertet der Sensor anschließend aus. Die Auswertung der zurückgekommenen Wellen ergeben verschiedene Ergebnisse, wenn sich ein Hindernis bewegt. Dadurch kann der Sensor eine Bewegung feststellen.

Bei dem Sensor sind zwei Potentiometer integriert, mit denen die Signallänge und die Sensibilität eingestellt werden kann. Das linke Potentiometer ist für die Einstellung der Signallänge und das rechte für die Sensibilität zuständig. Bei beiden steigt der Wert bei Drehung im Uhrzeigersinn.

3.Lichtsensor Planänderung

Lichtsensor 393



LDR (Fotowiderstand)

Ursprünglich sollte der **Lichsensor** in unserer Schaltung, die Nachtfahrlichtschaltung ermöglichen.

Dank dem Tip von Frau Dr. Oerke, mussten wir den **Lichtsensor** nicht bestellen, da der **LDR** die selben Eigenschaften besitzt die des Lichtsensors.

Der Vorteil des **Lichsensors** wäre, dass die Empfindlichkeit mittels eins Potentiometers eingestellt werden könnte.

Je kleiner der Widerstand desto größer der Strom, den man mit hilfe eines 2 Widerstands dann mit dem ESP32 analog auslesen kann.

3.LDR Funktionsweise

Ein LDR ist ein Halbleiter, dessen Widerstandswert lichtabhängig ist.

Ein LDR besteht aus zwei Kupferkämmen, die auf einer isolierten Unterlage (weiß) aufgebracht sind. Dazwischen liegt das Halbleitermaterial in Form eines gewundenen Bandes (rot).

Fällt das Licht (Photonen) auf das lichtempfindliche Halbleitermaterial, dann werden die Elektronen aus ihren Kristallen herausgelöst (Paarbildung). Der LDR wird leitfähiger, das heißt, sein Widerstandswert wird kleiner. Je weniger der Widerstand desto größer der Strom, den wir dann mit dem ESP32 analog auslesen können.

3.Arduino Programmierung

Festlegung der verschiedenen Pins Input/Output

```
void setup()
{
  pinMode(22, INPUT);          // Eingang Schalter
  pinMode(13, OUTPUT);        // Blinker LED  Ausgang
  pinMode(14, OUTPUT);        // Blinker LED  Ausgang
  pinMode(27, OUTPUT);         // Blinker LED  Ausgang
  pinMode(26, OUTPUT);         // Blinker LED  Ausgang
  pinMode(17, OUTPUT);         // Tagfahrlicht LED
  pinMode(25, OUTPUT);         // Tagfahrlicht LED
  pinMode(33, OUTPUT);         // Tagfahrlicht LED
  pinMode(32, OUTPUT);         // Tagfahrlicht LED
  pinMode(2, OUTPUT);    // alarmBuzzer ist ein Ausgang
  pinMode(15, INPUT);// bewegungssensor ist ein Eingang
  Serial.begin(9600);
}
```

Definition der Pins und Variablen

```
int buttonZaehler = 0;
int letzterStatus = 0;
int Schalterzustand = 0;
int licht;
int alarmLED = 13;                  // Blinker LED
int alarmLED2 = 14;                 // Blinker LED
int alarmLED3 = 27;                 // Blinker LED
int alarmLED4 = 26;                 // Blinker LED
int alarmBuzzer = 2;               // Pin des Lautsprechers
int bewegungssensor = 15;          //Pin des Infrarotsensors
int bewegungsStatus = 0;           /* In der Variable wird gespeichert, ob eine
                                      Bewegung erkannt wird oder nicht*/
```

3.Arduino Programmierung

Funktion Schalter im Loop

```
void loop() {
  Schalterzustand = digitalRead(22);
  if (Schalterzustand != letzterStatus)
  {
    if (Schalterzustand == 1)
    {
      buttonZaehler++;
    }
  }
  if (buttonZaehler % 2 == 0)
  {

    buttonZaehler = 0;
  }

  letzterStatus = Schalterzustand;
  Serial.println(buttonZaehler);
```

Zustand des Schalters wird ausgelesen

Schalterzustand ungleich Letzter Status (!=)

Wird der Schalter betätigt erhöht sich der Buttonzähler um 1 (++)

Buttonzähler wird durch 2 geteilt. Wird der wert 0 erreicht, setzt sich der Buttonzähler zurück auf den wert 0 (ansonsten würde er hochzählen!)

Der letzte Status wird auf das gesetzt was gerade mit den Schalter passiert

Ausgabe serieller monitor

3.Arduino Programmierung

Lichtsteuerung /
Abfrage Alarm

```
licht = analogRead(4);
Serial.println(licht);
if (licht < 3500) {
  digitalWrite(17, HIGH);
  digitalWrite(25, HIGH);
  digitalWrite(33, HIGH);
  digitalWrite(32, HIGH);
}
else {
  digitalWrite(17, LOW);
  digitalWrite(25, LOW);
  digitalWrite(33, LOW);
  digitalWrite(32, LOW);
}

if (buttonZaehler == 1) {

  Alarm(); // Funktionsaufruf
}
```

Funktion ,,licht,, wird analog gelesen über pin4

Ausgabe serieller monitor

Ist der wert kleiner 3500 so werden die Leds Aktiviert! wird der wert größer, bleiben sie aus (es werden je nach Lichtverhältnis Elektronen freigesetzt die dann als wert ausgegeben werden)

Wird der Schalter gedrückt und erhält den wert 1 so wird die Funktion ,,Alarm,, durchlaufen andernfalls wird diese übergangen und der loop beginnt von vorn

Arduino Programmierung

Funktion Alarm

```
void Alarm ()
{

  bewegungsStatus = digitalRead(bewegungssensor);
  if (bewegungsStatus == HIGH)
    // Wenn eine Bewegung erkannt wird, wird der Block ausgeführt
  {
    Serial.println("Alarm");
    for (int i = 0; i < 5 ; i++)
      // Die Schleife wird 5 mal ausgeführt
    {
      digitalWrite(alarmLED, HIGH);        // Blinker LED an
      digitalWrite(alarmLED2, HIGH);       // Blinker LED an
      digitalWrite(alarmLED3, HIGH);       // Blinker LED an
      digitalWrite(alarmLED4, HIGH);       // Blinker LED an
      digitalWrite(alarmBuzzer, HIGH);     // Alarmton an
      delay(500);                          // Eine halbe Sekunde ist Alarm an
      digitalWrite (alarmLED, LOW);        // Blinker LED aus
      digitalWrite (alarmLED2, LOW);       // Blinker LED aus
      digitalWrite (alarmLED3, LOW);       // Blinker LED aus
      digitalWrite (alarmLED4, LOW);       // Blinker LED aus
      digitalWrite (alarmBuzzer, LOW);     // Blinker LED aus
      delay(500);                          // für eine halbe Sekunde ist Alarm aus
    }
  }
}
```

Bewegungsstatus wird ausgelesen

Bei erkannter Bewegung wird eine schleife von Blinkenden Leds in Begleitung von einem Alarm Buzzer durchlaufen

Der serielle monitor schreibt ,,alarm,,

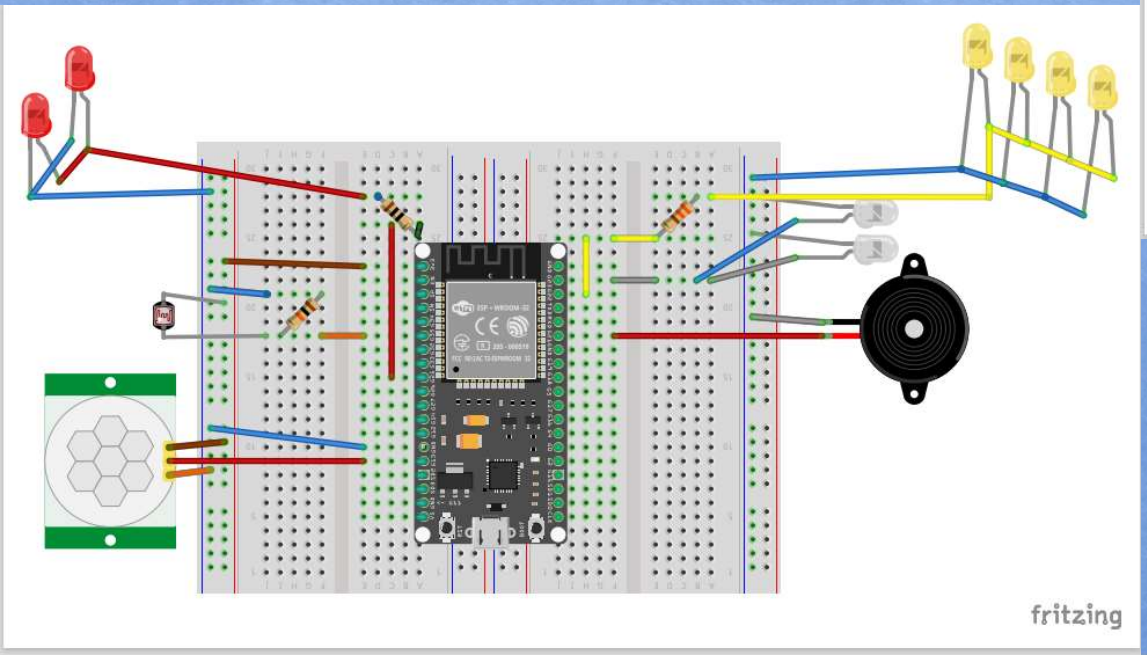
Probleme bei der Programmierung

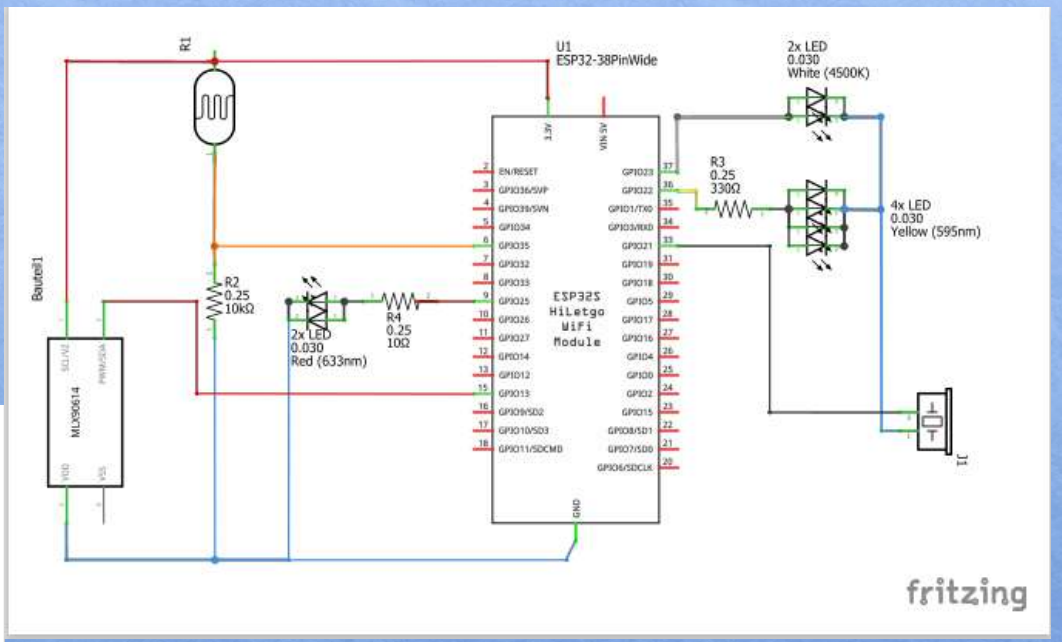
```
void loop() {
  Schalterzustand = digitalRead(22);
  if (Schalterzustand != letzterStatus)
  {
    if (Schalterzustand == 1)
    {
      buttonZaehler++;
    }
  }
  if (buttonZaehler % 2 == 0)
  {

    buttonZaehler = 0;
  }
```

Die Bedeutung verschiedenster Funktion waren fremd.
Dies sorgte für Probleme um gewisse Funktion vernünftig umzusetzen.
Researche und rumprobieren waren unter anderem die Lösung. Tipps und Tricks von erfahren Programmierern waren auch ausschlaggebend !

- == (Vergleicht die Variable auf der linken Seite mit dem Wert oder der Variablen auf der rechten Seite des Operators)
- ++ (Erhöht den Wert einer Variablen um 1.)
- != (Vergleicht die linke Variable mit dem Wert oder der Variablen rechts vom Operator. Gibt true zurück, wenn die beiden Operanden nicht gleich sind)
- % (Gibt die Möglichkeit einen wert von Variable zu teilen)

3.Fritzing Schaltung





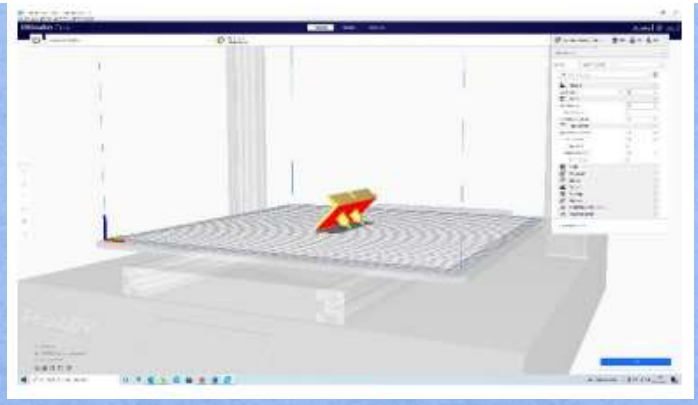
3. Durchführung – Additive Fertigung

Das T1 Bus Modell soll mittels additiver Fertigung eine Halterung für den Bewegungssensor im Heck bekommen.
Dafür wurde eine bestehende 3D Konstruktionsdatei als STL. Format von der Seite xploredna.net verwendet.

Die sogenannte STL-Datei kann nicht direkt an den 3D-Drucker geschickt werden. Dazu benötigt man ein Programm, den sogenannten Slicer, das die STL-Datei in einzelne Schichten zerlegt und einen sogenannten GCode erzeugt. Der G-Code ist primär eine Auflistung von Anweisungen an den Druckkopf, wie er sich geometrisch in X-, Y- und Z-Achse bewegen soll.

Bei der Verwendung der bereits bestehenden STL. Datei, ist nach dem ersten Druck aufgefallen, dass der Abstand der Zapfen für die Befestigung des Bewegungssensors zu klein war. Da der PLA Kunststoff allerdings über schlechte mechanische Eigenschaften verfügt, sind bei dem Halter bei leichter Druck Ausübung die Haltestifte für den Sensor abgebrochen. Abhilfe konnte geschaffen werden, in dem wir die Z-Achse in Ultimaker Cura um 2mm vergrößert wurde.

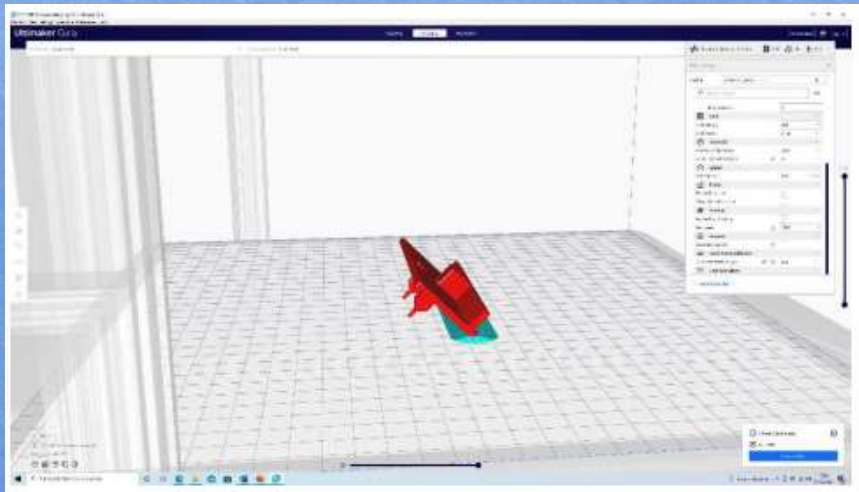
3. Durchführung



Zum Slicen haben wir die Open-Source-Software Cura verwendet. Mit dieser Speziellen Software konnten wir die vorhandene STL.Datei öffnen und bearbeiten.

Sie fungiert als Vermittler zwischen dem 3D-Modell und dem 3D-Drucker Man hat z.B. die Möglichkeit zu skalieren, Proportionen, Fülldichte Druckqualität, verwendendes Material, Druckgeschwindigkeit ,Temperatur sowie die Stützkonstruktionen zu ändern.

Die Option für verschiedene Ansichten und frei wählbare Perspektiven sind auch Möglich. Während der Präparation des Objektes kann man die LayerAnsicht der Füllung des Objektes sich anzeigen lassen. Je nach gewählten Parametern gibt Cura auch eine zeitliche Einschätzung wie lange der 3D-Druck ungefähr brauchen wird.



3. Durchführung - Material „ PLA „

Vorteile:
- o Geringe Drucktemperatur
- o Geringes schrumpf und Verzugsverhalten
- o Thermoplastisch leicht zu verarbeiten o Hohe mechanische Festigkeit und Steifigkeit
- o Umweltfreundlich

Nachteile:
- o sehr spröde
- o kann ab 60 °C schmelzen
- o schlechte mechanische Eigenschaften

Druckparameter Filament PLA
für Creality CR-10S PRO

Einfülldichte	20%
Drucktemperatur	65C
Druckplatten Temperatur	200C
Druckgeschwindigkeit	100%
Lüftergeschwindigkeit	100%

Probleme und Lösungen

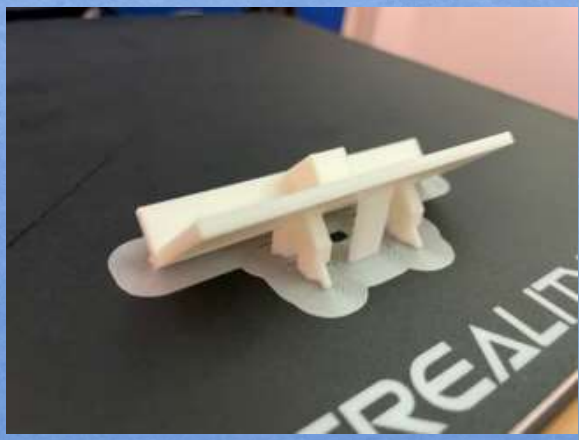
Probleme während des 3D Druck	Ursache	Lösung
Brim bzw. Druck löst sich beim Drucken	Abstand der Drückerdüse zu groß bzw. verschmutzte Druckplatte	AUX leveling durchführen Reinigung der Druckplatte
Bruch des Filaments	zu neige gehendes Filament auf der Rolle	
Die erste Schicht wird nicht druckt und die zweite nur teilweise	Abstand zwischen Nozzle und Druckbett vergrößern	AUX leveling durchführen

3. Durchführung - Druckhaftungstyp „Brim“

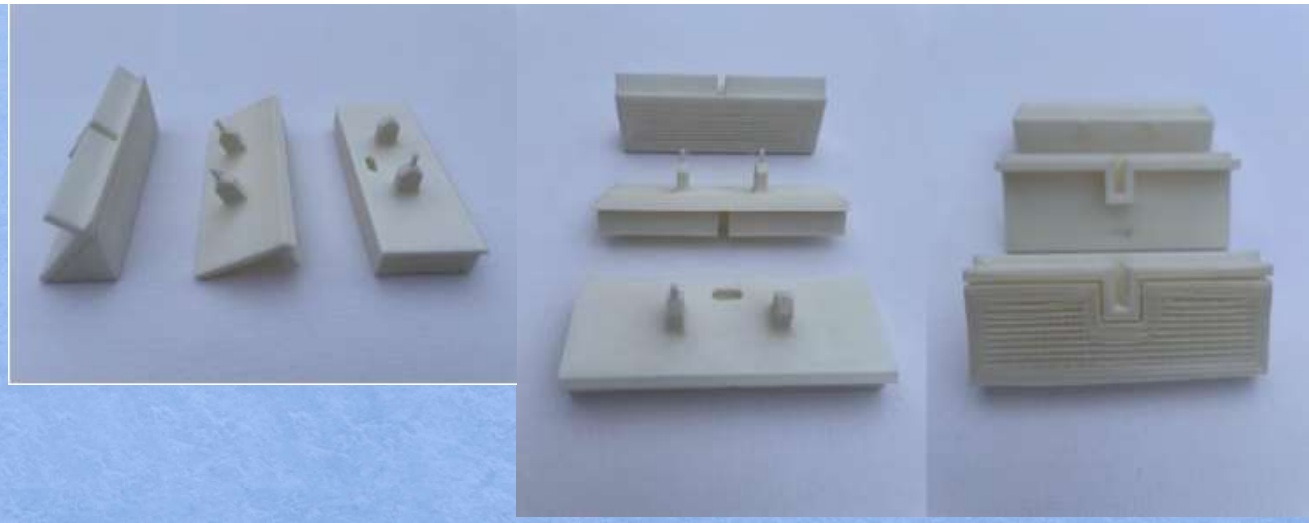
Da der Brim nur aus einer Schicht besteht, ist er sehr sparsam im Filament Verbrauch und lässt sich relativ leicht von der Druckplatte entfernen.

Die Hauptaufgaben des Brim beim 3D-Druck sind folgende:

- o Haftung des Druckgegenstandes auf der Druckplatte zu optimieren.
- o Filament Fluss anregen, bevor der eigentliche Druck beginnt.
- o Hochbiegen von Kanten und Ecken abwenden.
- o Druckfehler frühzeitig erkennen (Nivellierung des Druckbettes, Tropfen der Düse, etc.)



3. Durchführung – Sensorhalter



Erster Druck mit noch vorhandener Stützstruktur

Fertiger Druck mit angepassten Stift Abstand +2mm

4. Fertigstellung - Montage

Einbau der Komponenten in das T1 Modell Da das Gelieferte T1 Modell nur über 2 vorgesehenen Aussparungen für jeweils vorn und hinten verfügte, wurden für 4 weitere gelbe Led's neue Bohrungen im Durchmesser von 4,9mm gesetzt und anschließend verklebt.

Der LDR wurde in der Fahrzeugfront verbaut, hierbei mussten 2 kleine Löcher in
Die Front des Autos gebohrt werden.
Die Halterung für den Bewegungssensor im Heckbereich wurde mittel Ultimaker Cura in der Z-Achse um 2mm verbreitert, so das die Zapfen des Halters in die vorgesehenen Öffnungen des Bewegungssensors passten.

4. Fertigstellung – Funktionsnachweis

Alarm

Lichtsteuerung

Quellenangabe:

-Irre Clips - Von wegen „Schmusetiger“: Es gibt auch böse Katzen | krone.at

-xplore-dna.net > Die DNA des digitalen Lernens!

-Software-| Arduino

1. Videos+Projektstatusberichte
H1. Alexander Riemer, Mark Mallon, Sascha Link
2.» Dateien
3.» Gruppen
4.» FST21
5.» Schüler- Sammelunterlagen
6.» Projektarbeit Modul 2 (T1)
7.» H1. Alexander Riemer, Mark Mallon, Sascha Link