




PROJEKTAUFTRAG FAHRSTUHL



Joël Bandle, Jan Ludwig, Thomas Stern
NOSER YOUNG

Inhaltsverzeichnis

| | |
|---------------------------|---|
| Einleitung..... | 2 |
| Vorbereitung..... | 2 |
| Projektplanung | 3 |
| Drawio | 3 |
| Hardware Komponenten..... | 4 |
| Lichtsensord..... | 4 |
| Servo Motor | 5 |
| Verbindungen | 6 |
| MQTT..... | 6 |
| Node RED | 6 |
| Implementierung | 7 |
| Fazit | 8 |
| Quellen..... | 9 |
| Testing | 7 |

Einleitung

Für den zweiten Teil des ÜKs müssen wir in einer Gruppe ein Projekt basierend auf einer Vorgabe erstellen. Unsere Gruppe besteht aus Jan Ludwig, Thomas Stern und Joël Bandle. Unsere zugeteilte Aufgabe ist es, ein Programm für den ESP32 zu entwickeln, das mit Sensoren verbunden ist. Diese Sensoren überprüfen, ob sich der Lift im 6. Stock befindet. Falls nicht, soll das Programm den Lift rufen. Zusätzlich müssen wir eine Dokumentation erstellen, die unseren gesamten Ablauf beschreibt. Das Projekt bietet eine gute Gelegenheit unsere Zusammenarbeit zu verbessern, um das ganze zu realisieren.

Vorbereitung

Um überhaupt loslegen zu können, mussten wir das Ganze erst einmal vorbereiten und überlegen, wie wir es umsetzen wollen. Dafür sind wir zuerst zum Lift gegangen, wo schliesslich unser Endprodukt sein wird, und haben uns überlegt, wie wir die Umsetzung gestalten können. Es war von Anfang an klar, dass wir den Lift nur durch das Drücken des Knopfes rufen können. Hierfür ist der Servomotor perfekt geeignet, da er wie zwei Bretter hat, die rotieren. Mit den richtigen Befehlen können wir ihn so einstellen, dass er sich nur so weit dreht, bis er den Knopf drückt.

Wie wir jedoch erkennen können, ob sich der Lift im 6. Stock befindet, war etwas schwieriger. Wir entschieden uns, dafür einen Lichtsensor zu verwenden, da die Taste oben immer aufleuchtet, wenn der Lift oben ist, und wieder erlischt, wenn er nach unten fährt. Das passt gut zu unserer Situation, da der Sensor uns zuverlässig anzeigen kann, ob die Taste leuchtet oder nicht.



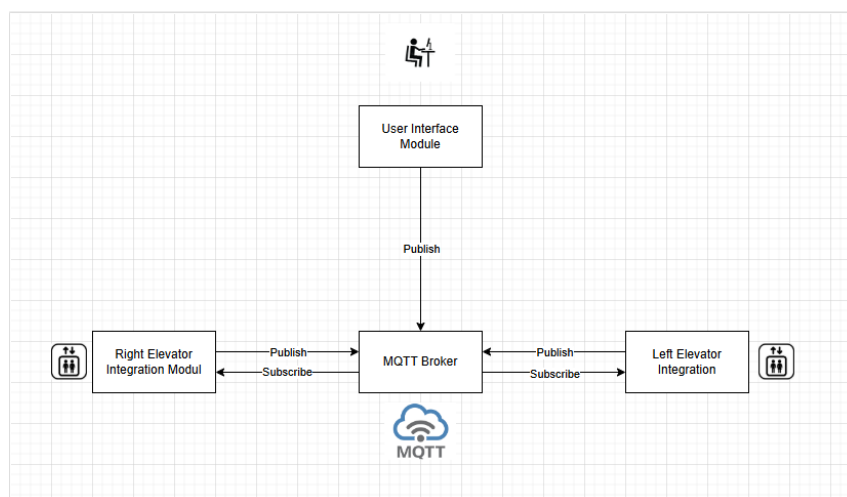
Nachdem wir alles geplant hatten, wie wir vorgehen wollen, teilten wir die Aufgaben gerecht auf. Thomas und Jan übernahmen die Programmierung der Sensoren, und Joël begann bereits mit der Dokumentation.

Projektplanung

| Schritt | Beschreibung | Wer? | Dauer (Tage) |
|---------------------|--|----------------------|---------------|
| Planung | Anforderungen analysieren, Konzept erstellen | Alle | 1 Tag |
| Hardware einrichten | Sensoren anschliessen, Verbindungen testen | Joel, Jan und Thomas | 2 Tage |
| Software entwickeln | Sensoren programmieren, MQTT und Node-RED einrichten | Thomas und Jan | 2 Tage |
| Dokumentation | Ablauf dokumentieren, Screenshots hinzufügen | Joël | Zwischendurch |
| Testing | Gesamtsystem testen und Fehler beheben | Alle | 1 Tag |

Drawio

Bevor wir aber mit dem Programmieren anfangen machten wir auf Drawio ein Flussdiagramm wie das ganze dann noch mit MQTT funktionieren wird.



Das Diagramm zeigt ein System zur Steuerung von den zwei Aufzügen mithilfe des MQTT-Protokolls. Das User Interface Module sendet Benutzeranfragen an den MQTT-Broker, der diese an die jeweiligen Elevator Integration Modules weiterleitet. Die Aufzugsmodule führen die Befehle aus und senden Statusmeldungen zurück, wodurch eine klare Kommunikation zwischen den Aufzügen entsteht.

Hardware Komponenten

Nachdem wir mit Drawio unsere Verbindungen geplant hatten, begannen wir mit der Programmierung der einzelnen Sensoren. Wir machten dazu zuerst mal eine Liste was wir alles brauchen werden und für welchen Zweck wir diesen benötigen.

| Komponenten | Verwendungszweck |
|-----------------|--|
| ESP 32 | Steuerung der Sensoren |
| Lichtsensord | Erkennt ob der Lift im 6 Stock ist |
| Servo Motor | Drücken des Lift Knopfes, um den Lift zu rufen |
| Breadboard | Verbindung der einzelnen Komponenten |
| Jumper Kabel | Verbindungen zwischen Komponenten |
| Stromversorgung | Power Bank |

Zunächst programmierten wir die Sensoren einzeln, um zu prüfen, wie sie funktionieren. Als erstes machten wir uns an den Lichtsensor zu schaffen.

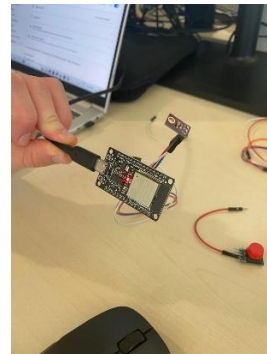
Lichtsensord

Der Lichtsensor war einfach zu verbinden nämlich GND in GND, VCC in 3V3 und OUT in den jeweiligen beschriebenen Pin im Script. Am Anfang nahmen wir das Beispiel das wir am Anfang des Üks bekommen haben um den Lichtsensor mal auszuprobieren.

```

1  #define LIGHT_PIN 13
2
3  void setup() {
4      Serial.begin(115200);
5      pinMode(LIGHT_PIN, INPUT);
6  }
7
8  void loop() {
9      float reading = analogRead(LIGHT_PIN);
10     Serial.println(reading);
11     delay(1000);
12 }

```



Dies funktionierte einwandfrei wie man es im Bild erkennen kann und später können wir das noch genauer beschreiben, wie zum Beispiel es soll den Servo Motor laufen lassen, wenn der Wert zwischen diesen zwei Zahlen liegt.

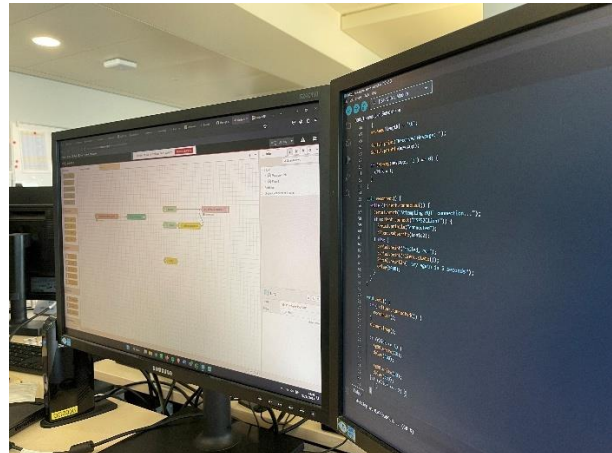
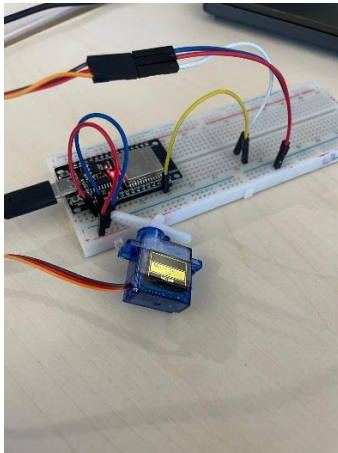
```

12:06:11.528 -> 453
12:06:11.528 -> Light intensity sent: 453
12:06:12.543 -> 368
12:06:12.543 -> Light intensity sent: 368
12:06:13.553 -> 371
12:06:13.553 -> Light intensity sent: 371
12:06:14.550 -> 371
12:06:14.550 -> Light intensity sent: 371
12:06:15.566 -> 373
12:06:15.566 -> Light intensity sent: 373
12:06:16.522 -> 599
12:06:16.565 -> Light intensity sent: 599
12:06:17.524 -> 509
12:06:17.562 -> Light intensity sent: 509
12:06:18.527 -> 266
12:06:18.561 -> Light intensity sent: 266
12:06:19.573 -> 362
12:06:19.573 -> Light intensity sent: 362

```

Servo Motor

Der Servo-Motor hat, wie der Lichtsensor, nur drei Pins: GND für die Masse, VCC an 5V, da er mehr Strom benötigt als der Lichtsensor und den PWM-Pin zur Steuerung der Bewegung. Wie beim Lichtsensor haben wir zuerst, bevor wir weitere Schritte unternahmen, ein Testprogramm von einer Website (https://wiki.hshl.de/wiki/index.php/Servomotor_SG90) kopiert, um zu prüfen, ob alles wie geplant funktioniert.



Das funktionierte ebenfalls einwandfrei: Die zwei "Bretter" drehten sich 360 Grad um den Mittelpunkt. Damit konnten wir mit den Verbindungen zu Node-RED und MQTT beginnen.

Verbindungen

MQTT

Node RED

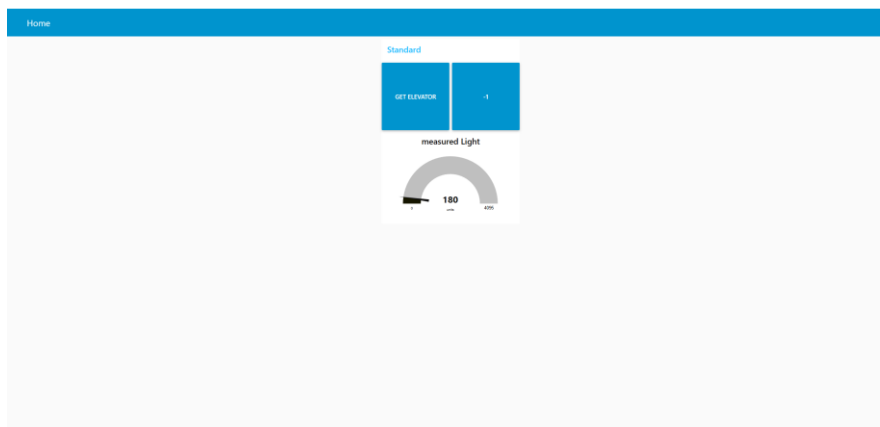
Zuerst haben wir den Node-RED-Flow erstellt, um die Daten des Lichtsensors anzuzeigen. Danach haben wir die Steuerlogik hinzugefügt, um die Bewegungen des Servomotors zu kontrollieren. Der Flow für den **linken Knopf** prüfte, ob der Aufzug nach oben gerufen werden sollte. Der **rechte Knopf** war für die Bewegung nach unten zuständig. Beide Flows nutzen **Switch-Nodes**, um zu prüfen, ob der Aufzug bereits im Zielstockwerk ist.

Ein Problem war, dass der Servomotor nicht immer die Knöpfe zuverlässig drücken konnte. Wir haben das durch Kalibrierung der Servowinkel und Verbesserungen in Node-RED gelöst. Außerdem haben wir sichergestellt, dass die MQTT-Nachrichten stabil zwischen Node-RED und dem ESP32 liefen. Am Ende funktionierte alles wie geplant, und wir konnten den Aufzug erfolgreich steuern.

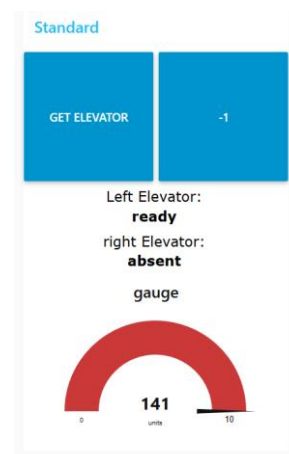
Implementierung

Zur Implementierung haben wir die Komponenten zunächst einzeln programmiert und erst am Schluss zusammengefügt. So konnten wir im Falle eines Fehlers direkt erkennen, welcher Sensor betroffen war, ohne lange suchen zu müssen. Anschliessend haben wir das Ganze mit dem Node-RED-Server verbunden, um die Werte des Lichtsensors anhand einer Grafik anzuzeigen.

Anfangs haben wir zwei ESP32 verwendet. Diese wurden über einen MQTT-Server verbunden, damit beide Geräte auf Node-RED angezeigt werden konnten, aber dennoch separat blieben.



Das haben wir danach umgeändert, da wir das Konstrukt für beide Lifte machen mussten. Deshalb haben wir auch zwei Grafiken für jeden einzelnen Sensor erstellt, um zu sehen, ob sie funktionieren. Ausserdem haben wir zwei Knöpfe für den Servomotor erstellt auf Node RED: einen für den oberen Knopf und einen für den -1-Knopf.



Nachdem alles fertig war, gingen wir zum Lift, um jeden einzelnen Sensor mit der Stromzufuhr einer Powerbank zu testen. Da wir alles mit Node-RED verbunden hatten, konnten wir die Werte direkt sehen, um eine übersichtliche Struktur zu schaffen. Anschliessend haben wir einige Tests durchgeführt, um sicherzustellen, dass alles funktioniert, und erst dann haben wir alles zusammengefügt.

Unser Endprodukt bestand darin, dass auf der rechten Seite des Lifts nur ein Lichtsensor befestigt war, der überprüfte, ob der Lift dort angekommen ist, und auf der linken Seite war ebenfalls ein Lichtsensor, zusätzlich aber auch der Servomotor, der die Knöpfe bediente. Das Ganze wurde mit zwei Powerbanks versorgt. Auf Node RED wird die Helligkeit angezeigt und man kann dann auch von dort mit den zwei Knöpfen den Lift holen. Auch wird angezeigt ob der Lift schon oben ist oder nicht.



Fazit

Im Rahmen des Projekts haben wir erfolgreich ein System erstellt, das einen Lift mithilfe eines ESP32, eines Lichtsensors und eines Servomotors steuert.

Der Lichtsensor erkennt, ob sich der Lift im 6. Stock befindet. Wenn der Lichtsensor erkennt, dass der Lift nicht im sechsten Stock ist, kann man einen Knopf in RedNode drücken und der Servomotor drückt den Liftknopf, um den Lift zu rufen.

Die Implementierung erfolgte in mehreren Schritten, beginnend mit der Programmierung der einzelnen Sensoren, gefolgt von der Integration in Node-RED und MQTT.

Trotz einiger Herausforderungen, wie z.B. der optimalen Platzierung des Lichtsensors und der genauen Kalibrierung des Servomotors, konnten wir das Projekt erfolgreich abschließen. Wir alle konnten viel lernen und die Teamarbeit lief sehr gut.

Durch die Verwendung von Drawio konnten wir den Datenfluss visualisieren und die Kommunikation zwischen den Komponenten optimieren.

Insgesamt war das Projekt ein Erfolg und hat uns wertvolle Einblicke in die Entwicklung von IoT-Systemen gegeben.

Quellen

https://wiki.hshl.de/wiki/index.php/Servomotor_SG90

Testing

| Nr. | | | | | | Test | Eingabe | Erwartet | Ergebnis | Status | Tester |
|-----|--|--------------------------|--|---|-----------------|--------|---------|----------|----------|--------|--------|
| 1 | Taschenlampe auf Lichtsensor strahlen | Licht | Einen hohen Helligkeitswert | 0 | Bestanden | Thomas | | | | | |
| 2 | Lichtsensor abdecken | Dunkelheit | Einen niedrigen Helligkeitswert | 200 | Bestanden | Jan | | | | | |
| 3 | Knopf in Node-RED drücken | Knopfdruck | Servo bewegt sich und drückt den Knopf | Fast gedrückt / Aber nicht fest genug | Nicht Bestanden | Alle | | | | | |
| 4 | Lift im 6. Stock simulieren (mit Licht) | Lift im 6. Stock (Licht) | Servo bewegt sich nicht | Hat sich nicht bewegt | Bestanden | Alle | | | | | |
| 5 | Lift NICHT im 6. Stock simulieren (dunkel) | Lift nicht im 6. Stock | Servo bewegt sich und drückt den Knopf | Hat knopf gedrückt | Bestanden | Alle | | | | | |
| 6 | MQTT-Nachricht senden | NodeRed Knopf drücken | Servo bewegt sich und drückt den Knopf | Hat den Command am Servo gegeben und Servo hat den Knopf gedrückt | Bestanden | Alle | | | | | |