

Automatische Somfy rolluiken

Somfy zonnewering is al in de markt sinds 1969 op de markt, vroeger als handmatige zonnewering en later als elektrische zonnewering. Helaas is deze zonnewering alleen te bedienen met een afstandsbediening. Wat erg onhandig kan zijn omdat je deze dan bij de hand moet hebben en kwijt kan raken. Daarom bespreken wij in dit artikel de mogelijkheid om doormiddel van een gateway die aangestuurd kan worden met behulp van een app genaamd Home assistent

Veel Nederlandse huizen zijn gebouwd op een zogenaamd open raam huisstijl, dit brengt een huis veel licht en is het huis dus minder geld kwijt aan verlichting. Ook het is gezonder voor je om natuurlijk licht te hebben. Het grote nadeel van deze huizen is alleen dat als je in de zomer aan het werk moet op een PC dan reflecteren deze schermen vaak met zoveel zonlicht. Superhandig om een zonnewering te laten installeren dus. Alleen deze soort zonnewering vereist enige aandacht. Zo moet de gebruiker nadenken over het dicht doen na gebruik om te voorkomen dat het luik de volgende dag of nacht kapot gaat doordat het hard gaat waaieren of regenen. Dit kan voorkomen worden door middel van automatische besturing. Daarom is met behulp van Douwe Schotanus en Marco Winkelman is in samenwerking met Smartomation Engineers gekeken naar het automatiseren en via mobiel besturen van de Somfy rolluiken.

Wat er momenteel bestaat

Voor het automatiseren zijn al veel bekende mogelijkheden te vinden, zoals bijvoorbeeld de Bosch smart home. Alleen veel mensen willen geen nieuwe rolluiken kopen alleen maar omdat ze dan automatisch zijn. Verder zijn de overige mogelijkheden erg prijzig voor een vrij simpele feature. Somfy heeft zelf geen mogelijk om de zonnewering te automatiseren. Er zijn natuurlijk al wel apparaten gemaakt die het automatiseren van bepaalde apparaten een stuk makkelijker maken door bijvoorbeeld de benodigde hardware al in een soort gateway te stoppen maar ook deze producten zijn hopeloos duur voor wat het kan.



Gebruikte hardware

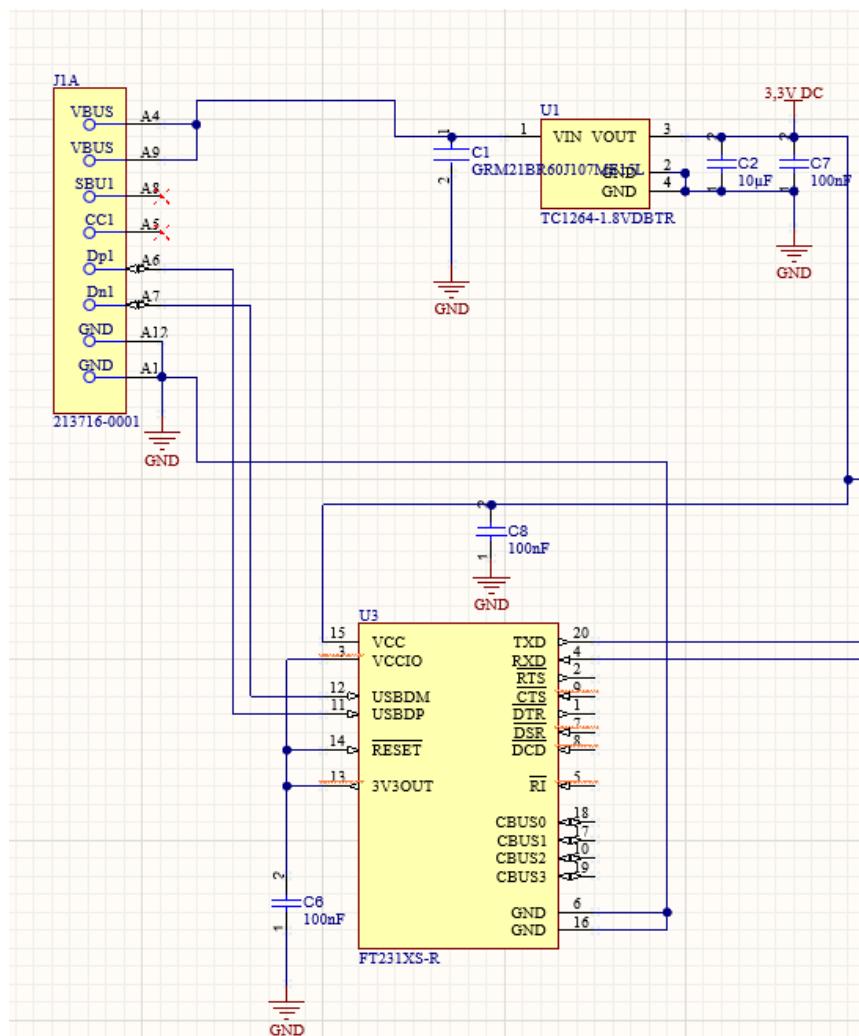
In de eerste fase van het project moest de hardware worden uitgekozen. Hiermee begonnen we bij de kern van de gateway, de microcontroller. In deze microcontroller zochten we naar een controller die gebruik kan maken van Wi-Fi, die seriële data van een usb-c poort kan ontvangen en lezen en met een groot genoeg geheugen zodat de software erop kon. Aan de hand van deze eisen kwam de esp-32-wroom-32 eruit. Daarna werd nagedacht over hoe deze Esp gevoed ging worden.



Figuur 1, de gekozen Esp-32-wroom-32 van ESPRESSIF



Tijdens het ontwerpen kwamen we op 2 soorten van het product uit, 1 versie waarbij de gehele gateway op batterijen werkt en een ander waarbij het doormiddel van een usb-c kabel werd gevoed. Na verder onderzoek viel door de hoeveelheid vermogen die gebruikt werd de eerste optie af. Dus wordt de gateway gevoed met een usb-c input. Deze usb-c geeft een standaard output van 5V, maar voor het voeden van de esp is 3,3v vereist. Daarvoor gebruiken we een power converter zoals te zien in **Figuur 2**.

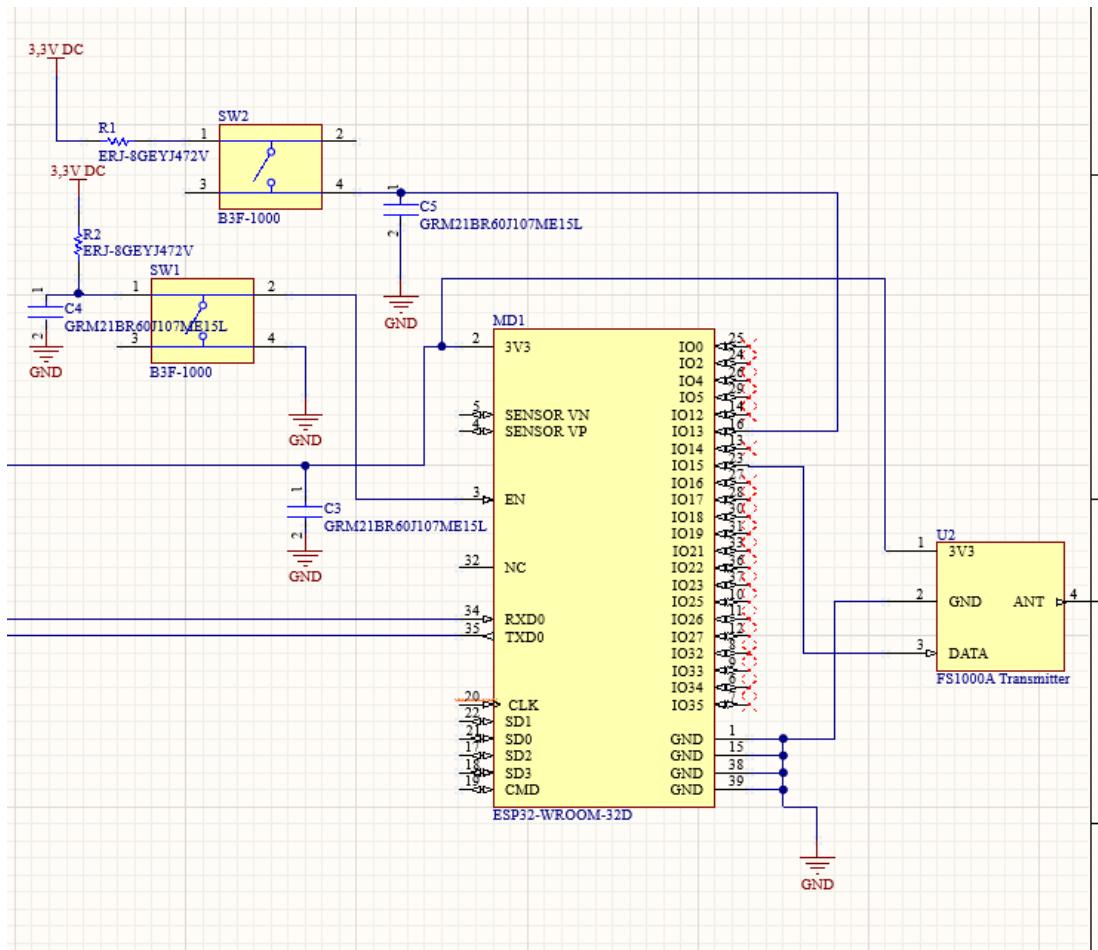


Figuur 2, de power supply en data converter van de gateway



Omdat de gebruikte Esp niet direct de informatie van de usb-c kan gebruiken om de rolluiken aan te sturen moet deze informatie eerst worden omgezet naar seriële informatie. Dit wordt gedaan door de onderste chip te zien in **Figuur 2**. Ook zijn er om storing tegen te gaan en een zo zuiver mogelijke spanning naar de chip en microcontroller te krijgen verschillende soorten condensatoren geplaatst. Om te kunnen communiceren met de rolluiken van Somfy is een 433.42 MHz transmitter nodig. Omdat deze waarde zo specifiek is werkt niet elke transmitter die dicht bij de 433 MHz zit. Omdat dit het geval is hebben we gekozen voor de FS1000A met daarop een aparte 433.42MHz kristal geplaatst zoals te zien in **Figuur 3**.

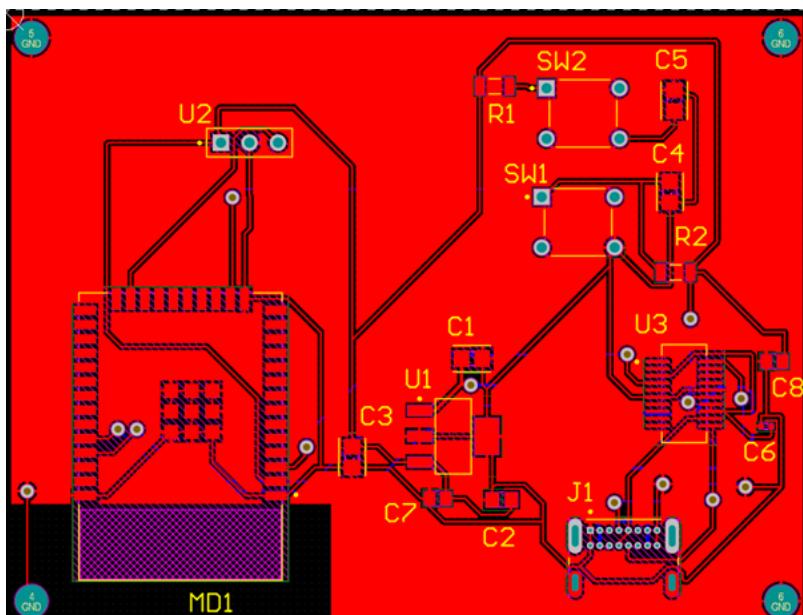
Omdat de microcontroller ook nog geprogrammeerd moet worden, en dit alleen mogelijk is door de ENABLE van de microcontroller laag te maken hebben we nog een knop geplaatst. Ook is er een 2^e knop geplaatst zodat de kast een hard reset heeft die de rolluiken van de gateway kan wissen. Deze knoppen zijn beide op een verschillende manier aangesloten omdat de ENABLE van de Esp altijd hoog moet zijn zoals ook te zien in **Figuur 3**.



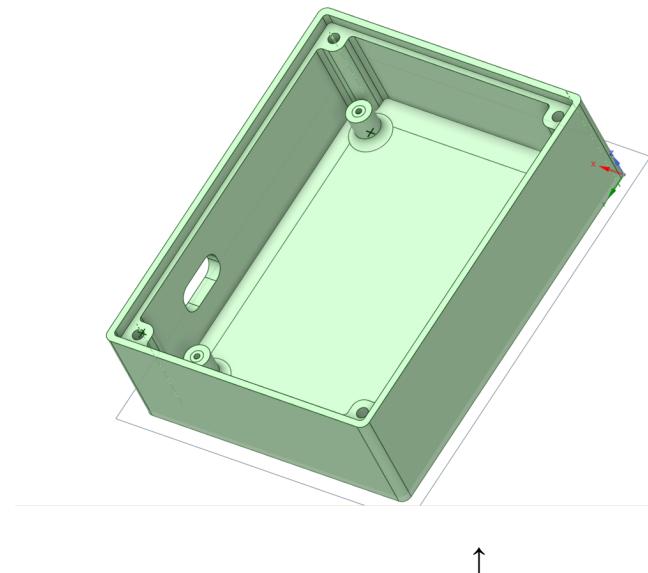
*Figuur 3,
schakeling voor
data verwerken
en versturen.*

De PCB

Nadat alle hardware voor de gateway uitgezocht was moest een PCB ontworpen worden. Bij de PCB moeten we ook nadenken over het monteren in de behuizing, ook moesten we oppassen dat de antenne van de microcontroller niet verstoord werd door de ground laag van de PCB, dit probleem hebben we opgelost door de ‘polygon pour’ onder en rond de antenne weg te halen zoals te zien in **Figuur 4**. De monteermethode die ons het best leek was door 4 gaten in de pcb te maken om deze later in de behuizing te schroeven. De PCB is ontworpen in Altium Designer omdat ik daar al de meeste ervaring in had en het een professioneel eindproduct oplevert.



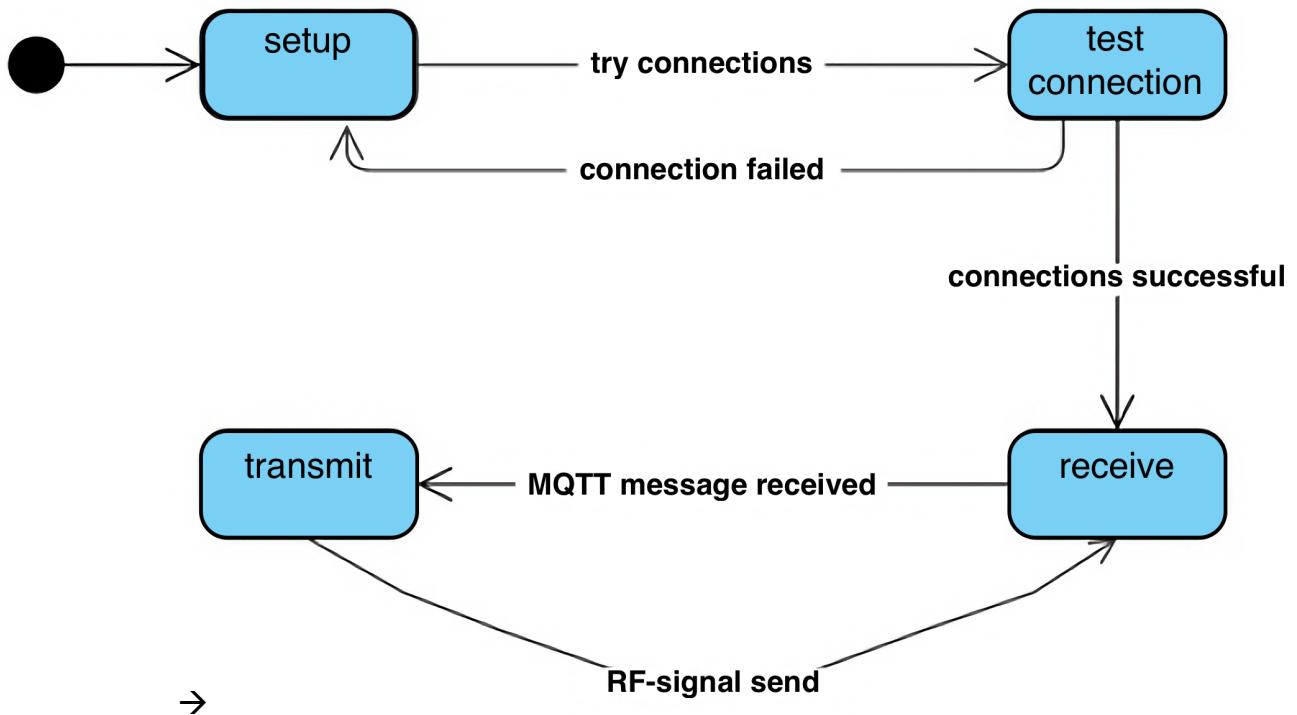
Figuur 4, de PCB ontworpen in Altium Designer.



Figuur 5, het Design van de behuizing in Designspark Mechanical.

De behuizing

Vervolgens hebben we nog de behuizing, deze heb ik ontworpen in Designspark Mechanical zoals te zien in **Figuur 5**. Ik wou graag dat de bovenkant van de behuizing doorzichtig was dus is de bovenkant van het 3D ontwerp open. Ook wou ik graag dat de pcb vast in de behuizing zit zodat de ingang voor usb-c in de behuizing gelijk zit met de usb-c poort van de PCB. Ik heb ervoor gekozen om aparte grotere gaten te gebruiken voor de plexiglas plaat die erop komt, zo kan je met een schroevendraaier nog goed de plaat er een aantal keer afhalen zonder dat het schroefgat te groot wordt. Verder zijn alle randen een beetje platter gemaakt zodat de hoeken niet scherp aanvoelen, ook zorgt dit voor iets meer stevigheid als hij valt omdat de printen 90 graden hoeken moeilijker kan printen.



Figuur 6, State
Diagram van
volledige code

De software

Om alles ook daadwerkelijk met elkaar te verbinden moet er software op de ESP32 komen. Deze is geschreven met C++ in Arduino IDE. Dit de meest bekende compiler onder hobbyisten en heeft veel online support. Voordat we code op de esp32 kunnen zetten, moeten we in de voorkeuren van Arduino IDE de Esp32 url toevoegen in de Bijkomende Boorden Beheer URL. Hierna kunnen we in de borden beheer de esp32 opzoeken en downloaden. Nu kunnen we software op de Esp32 zetten.

State Diagram

De state diagram in **Figuur 6** laat zien hoe de software in grote lijnen werkt. Als eerst probeert de gateway verbindingen te maken met het wifinetwerk en Home Assistant (setup). Hierna test de gateway of deze verbindingen zijn geslaagd. Dit proces wordt herhaald totdat allebei de connecties voltooid zijn. Nu kan de gateway berichten ontvangen van Home Assistant. Zodra een bericht is ontvangen wordt het bijbehorende signaal naar het rolluik gezonden en kan de gateway weer berichten ontvangen.

```

//          Alle define's
#define EMITTER_GPIO 14
#define REMOTE1 0x5184c8
#define REMOTE2 0x25b5d5
#define REMOTE3 0xc6c78f
#define REMOTE4 0x59714b
#define ESP32

//          Alle int's
int RS = 0; //identificatie van de 4 remotes en de state

//          SETUP
EEPROMRollingCodeStorage rollingCodeStorage1(0);
EEPROMRollingCodeStorage rollingCodeStorage2(2);
EEPROMRollingCodeStorage rollingCodeStorage3(4);
EEPROMRollingCodeStorage rollingCodeStorage4(6);
SomfyRemote somfyRemote1(EMITTER_GPIO, REMOTE1, &rollingCodeStorage1);
SomfyRemote somfyRemote2(EMITTER_GPIO, REMOTE2, &rollingCodeStorage2);
SomfyRemote somfyRemote3(EMITTER_GPIO, REMOTE3, &rollingCodeStorage3);
SomfyRemote somfyRemote4(EMITTER_GPIO, REMOTE4, &rollingCodeStorage4);

```

↑

Figuur 7, Code fragment voor de Somfy bediening

```

//          SETUP
EspMQTTClient client(
    "WifiSSID",           // Wifi naam
    "WifiPassword",       // Wifi wachtwoord
    "192.168.1.100",     // MQTT Broker server ip
    "MQTTUsername",       // De MQTT gebruikersnaam
    "MQTTPassword",       // De MQTT wachtwoord
    "TestClient",         // Client naam dat jou toestel een unieke identificatie geeft
    1883                 // De MQTT poort, default naar 1883
);

```

↑

Figuur 8, Setup code voor de MQTT op de gateway

Gateway naar Somfy

Voor de connectie met de Somfy rolluiken gebruiken we de Somfy_Remote_Lib library. Het emuleert de Somfy afstandsbediening zodat de gateway acties naar de rolluiken kan versturen. Als eerst geven we in de define ieder afstandsbediening een eigen adres, in **Figuur 7**, en geven we aan welke Esp we gebruiken (32). Daarna staat er in de setup waar de rollingcode moet worden opgeslagen per afstandsbediening. Nu kan met de code “somfyRemote1.sendCommand();” verschillende acties worden verstuurd.

Home Assistant naar gateway

Om verbinding te maken met Home Assistant gebruiken we de netwerkprotocol MQTT. Om dit op de Home Assistant te gebruiken moet de addon Mosquitto broker toegevoegd worden en gebruiken we de EspMQTTClient library voor de Esp32. De Home Assistant is de publisher en Esp de subscriber. Om de connectie velliger te maken moeten we in Home Assistant een nieuwe gebruiker aanmaken. De naam en wachtwoord van de gebruiker worden de MQTT-gebruikersnaam en wachtwoord. Deze gegevens, de MQTT IP-adres (de ip van je Raspberry pi), wifi naam en wachtwoord vullen we in de setup van de EspMQTTClient (**Figuur 8**).

Hoe werkt MQTT?

- Lichtgewicht
- Betrouwbaar communicatie
- Lage overhead
- Werkt met beperkte bandbreedte

MQTT staat voor “Message Queuing Telemetry Transport” en is een netwerkprotocol voor berichten. Er zijn twee rollen, publishers en subscribers. De publishers kun je zien als de pers die nieuwsberichten (payload) sturen met specifieke onderwerpen (topics). Subscribers zijn de lezers die zich kunnen abonneren (subscribe) op deze specifieke onderwerpen en vervolgens de nieuwsberichten kunnen ontvangen.

Samenvoegen

Nu we weten hoe alle connecties werken, kunnen we kijken hoe dit allemaal wordt samengevoegd.

Stap 1 - Verbindingsopbouw:

De gateway zal eerst verbinden met de wifi daarna de MQTT.

Stap 2 - Abonneren en Payload Analyse:

Nadat dit is gelukt abonneert hij op de topic somfy/state en kan de gateway payloads detecteren.

Stap 3 - Eerste Filtering:

Als een payload gedetecteerd wordt zal deze als eerste gefilterd worden op Somfy afstandsbediening nummer door naar het eerste karakter van de payload te kijken (zie register).

Stap 4 - Tweede Filtering:

Na de eerste filtering wordt naar de tweede karakter van de payload gekeken om te gefilterd op het specifieke commando voor het rolluik (zie **Figuur 9**).

Stap 5 - Wijziging van Remote State (RS):

Aan de hand van de filtering wordt de int RS, ofwel de remote state, aangepast naar bijvoorbeeld 12. Hierbij staat het eerste cijfer voor het rolluiknummer en het tweede cijfer voor de uitgevoerde actie. En wordt de void rolluikZending uitgevoerd.

Stap 6 - Zending:

In void rolluikZending gebruiken we een switch case om de juiste commando's naar de rolluiken te zenden aan de hand van de int RS. Dit is te zien in **Figuur 10**

```
// This function is called once everything is connected (Wifi and MQTT)
void onConnectionEstablished()
{
    // Subscribe to "somfy/state" and display received message to Serial
    client.subscribe("somfy/state", [](const String & payload) {
        Serial.println(payload);

        // Checkt de eerste karakter
        if (payload[0] == '1') { // filtert signaal voor remote 1
            Serial.println("geslaagd");
            if (payload[1] == 'u'){ //bericht is up
                RS = 11;
                rolluikZending();
            }
            if (payload[1] == 'd'){ //bericht is down
                RS = 12;
                rolluikZending();
            }
            if (payload[1] == 'm'){ //bericht is my
                RS = 13;
                rolluikZending();
            }
            if (payload[1] == 'p'){ //bericht is prog
                RS = 14;
                rolluikZending();
            }
        }

        if (payload[0] == '2') { // filtert signaal voor remote 2
            Serial.println("geslaagd");
            if (payload[1] == 'u'){ //bericht is up
                RS = 21;
                rolluikZending();
            }
            if (payload[1] == 'd'){ //bericht is down
                RS = 22;
                rolluikZending();
            }
        }
    });
}
```

Figuur 9, Code payload filtering

```
// hier worden de berichten voor de gevraagde actie gefilterd en verzonden
void rolluikZending(void) {
    switch(RS) {
        //remote 1
        case 11: // command 1
            somfyRemote1.sendCommand(Command::Up, 1);
            delay(100);
            break;

        case 12: // command 2
            somfyRemote1.sendCommand(Command::Down, 1);
            delay(100);
            break;

        case 13: // command 3
            somfyRemote1.sendCommand(Command::My, 1);
            delay(100);
            break;

        case 14: // command 4
            somfyRemote1.sendCommand(Command::Prog, 1);
            delay(100);
            break;

        //remote 2
        case 21:
            somfyRemote2.sendCommand(Command::Up, 1);
            delay(100);
            break;
    }
}
```

Figuur 10, Code rolluikZending

Register:

Payload

1ste karakter geeft aan welke rolluik bestuurt moet worden

- 1 = rolluik 1
- 2 = rolluik 2
- 3 = rolluik 3
- 4 = rolluik 4

2e karakter is de actie

- U = omhoog
- D = omlaag
- M = my knop
- P = de prog knop

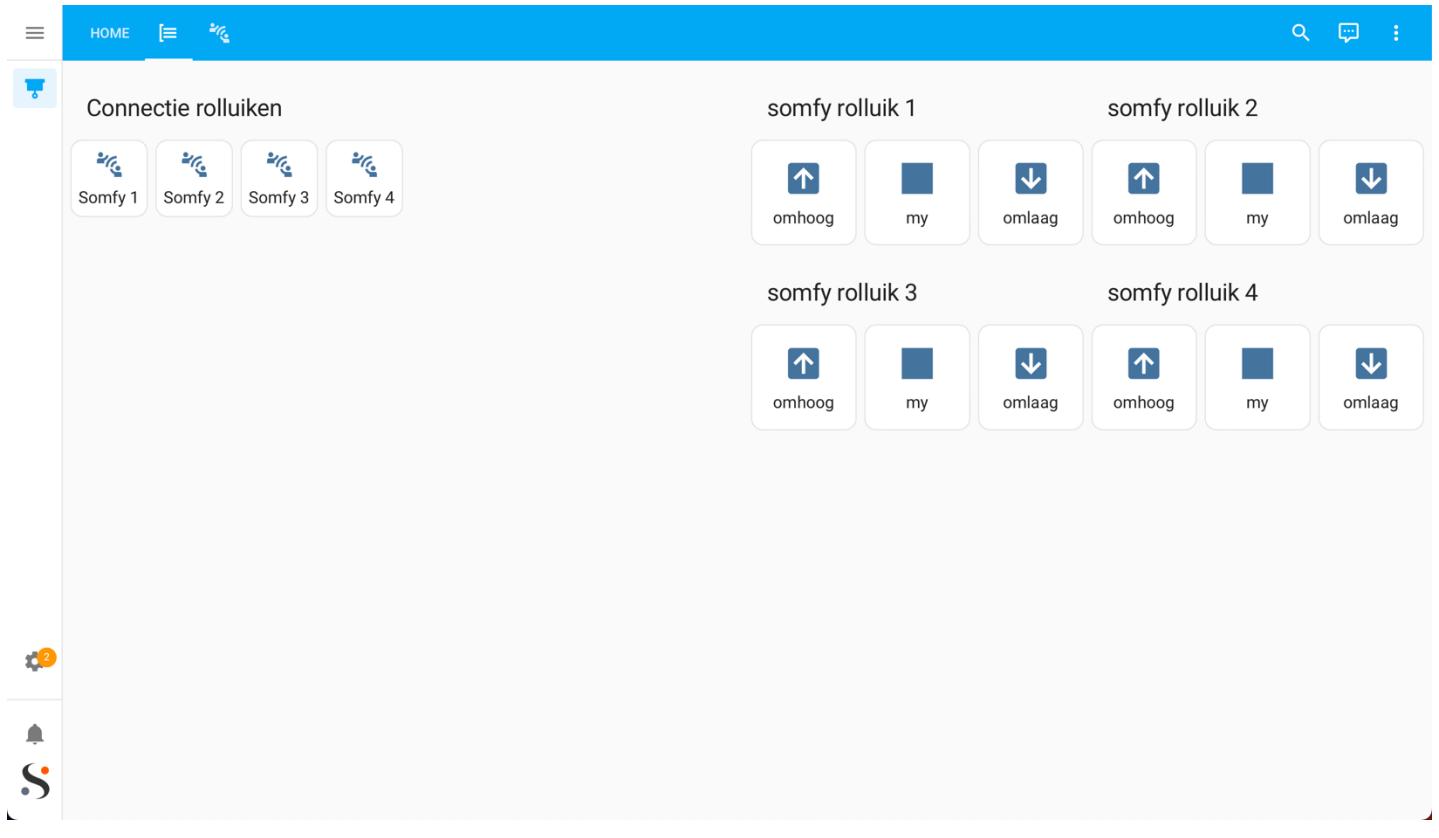
Int RS

1ste cijfer geeft aan naar welke rolluik verzonden moet worden

- 1 = rolluik 1
- 2 = rolluik 2
- 3 = rolluik 3
- 4 = rolluik 4

2e cijfer geeft aan welk actie het rolluik moet uitvoeren

- 1 = omhoog
- 2 = omlaag
- 3 = my knop
- 4 = de prog knop



Figuur 11, Interface in Home Assistant

Home Assistant interface

De Home Assistant Interface, zie **Figuur 11**, is regelmatig geüpdateet om deze gebruiksvriendelijker te maken. Dit is bereikt door verschillende mensen de interface te laten testen en te observeren hoe ze ermee omgingen. Vervolgens vroegen we om feedback en op basis daarvan, samen met de observaties, werd de interface voortdurend bijgewerkt totdat iedereen deze als gebruiksvriendelijk ervoer.

De code voor de interface is te vinden in de QR-code onderaan het artikel.

Testen software

Er is een software testrapport gemaakt dat een overzicht geeft van de testen die zijn uitgevoerd op de software van het project, gebaseerd op de vooraf vastgestelde eisen door het ingenieursbureau in overleg met de stakeholders:

- Communicatie met Home Assistant ✓
- Communicatie met rolluik ✓
- Veilig communicatie ✓
- Kan meerdere rolluiken aansturen ✓
- De rolluiken kunnen geautomatiseerd worden ✓
- Juiste filtering ✓
- Maximale reactietijd van vijf seconde ✓
- Simpele interface ✓

Zoals te zien is hierboven zijn alle eisen getest en met succes doorlopen, wat bevestigt dat het project voldoet aan de vastgestelde eisen.

Het eindproduct

Nadat alle code en hardware bij elkaar was gevoegd en de PCB in de behuizing zat met daar bovenop de plexiglas laag was het product klaar voor gebruik zoals te zien in **figuur 12**. Tijdens het ontwerpen van dit product hebben we er rekening mee gehouden dat mensen zelf ons product aan wouden passen en thuis wilden bouwen. Daarom hebben wij ons product en de code zo ontworpen dat het makkelijk is om zelf de code aan te passen en op de esp te zetten. Ook staan alle codes en designs van hardware en behuizing op onze site.



Zelf bouwen

Het leukste aan een project als dit is dat je het heel makkelijk thuis na kan maken. Ook is het een superhandig middel om in je eigen huis te gebruiken. Op de QR-code aan de onderkant van het blad vind je alle informatie die wij hebben gebruikt tijdens dit project. Als je zelf thuis een mooie toevoeging hebt gemaakt voor deze gateway wees dan vooral niet bang om je resultaten te mailen naar smartomationEngineers@outlook.com

Figuur 12, het eindresultaat van het product.



QR-code,

Mis je informatie? Dan is het misschien wel te vinden op onze website op de QR-code. Zo niet, kun je ons altijd bereiken via ons mailadres:

smartomationEngineers@outlook.com