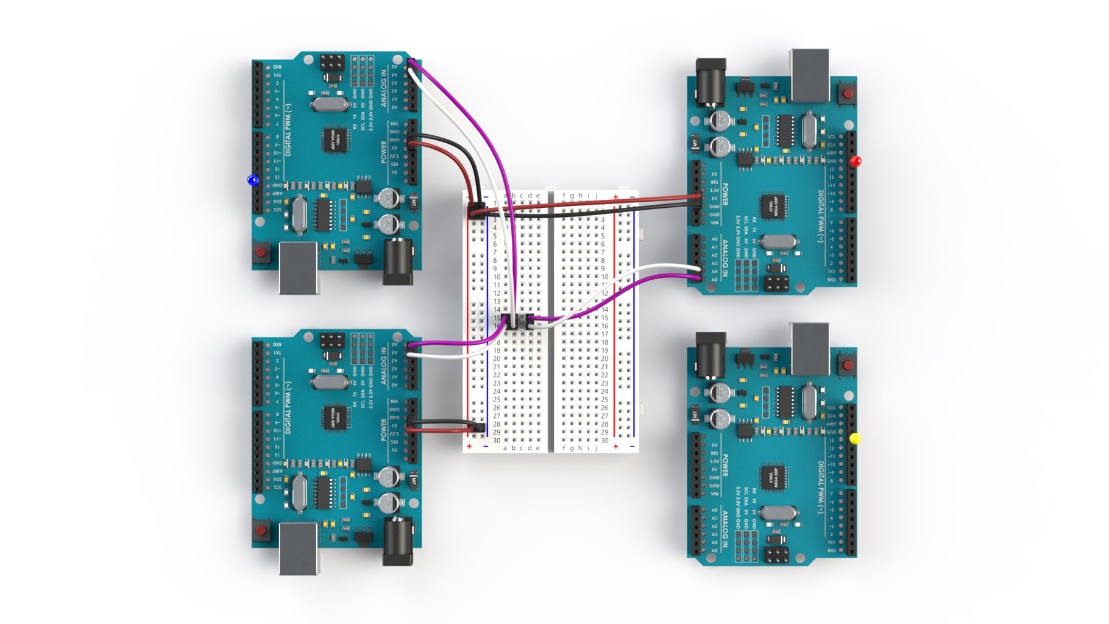
Sharing resources I2C

Tiemon Steeghs

Inhoud

[1. Aanleiding 3](#_Toc136176466)

[2. Probleemstelling 3](#_Toc136176467)

[3. Design 4](#_Toc136176468)

[4. Bronvermelding 5](#_Toc136176469)

# Probleemstelling

Binnen I2C systemen gaat er veel data heen en weer. Maar wat nou als twee masters tegelijkertijd data willen versturen naar een slave? In deze challenge ga ik kijken hoe je ervoor zorgt dat dit geen problemen oplevert, zodat de communicatie vlekkeloos blijft lopen. Een verdere beschrijving van mijn opstelling is in het volgende kopje beschreven.

# Design

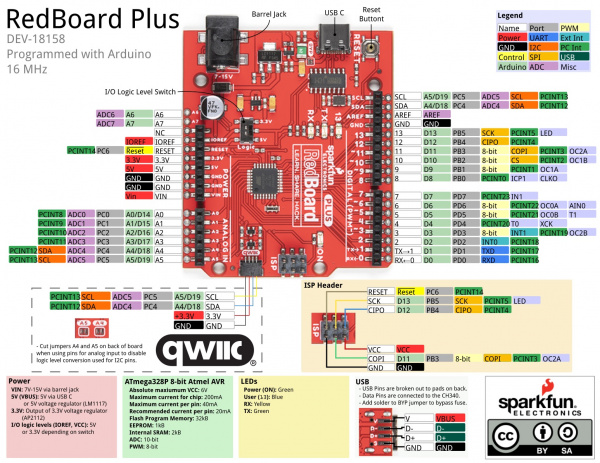
Mijn systeem bestaat uit twee Arduino redboards (masters) en één Qwiic oled display (slave). Mijn systeem gaat als volgt te werk: De twee redboard masters en de Qwiic oled display zitten allemaal aan elkaar aan een I2C bus. Beiden masters kunnen data schrijven naar de display, om zo het gene wat afgebeeld wordt te veranderen. Voordat ze dit doen vragen ze eerst aan de andere master of deze niet al bezig is met het schrijven van data. Op het moment dat de master terugkrijgt van de andere master dat de bus beschikbaar is, zal hij data gaan schrijven. Het is belangrijk om op te merken dat de slave op zichzelf niet geprogrameerd wordt, dus alles wordt geregeld door de twee masters.

De beschreven opstelling is hieronder ook weergegeven. Bij de pijlen zie je dus ook staan wat de masters opvragen en verkrijgen. De data van de beschikbaarheid gaat overigens via de display omdat die tussen de masters in zit, echter doet de display hier niks mee.

Afbeelding met tekst, lijn, schermopname, diagram

Automatisch gegenereerde beschrijving

Om het systeem werkende te krijgen moeten de twee redboards steeds naar de SDA lijn kijken of deze hoog of laag is. Dit moet overheen komen met de bits die ze willen sturen. Als dit niet overeenkomt weten ze dus dan de ander bezig met schrijven. Om de SDA lijn te kunnen lezen moet ik eerst ontdekken of dit wel kan.



# Onderzoek

## Versie 1: Status opvragen

Voor de daadwerkelijke implementatie ben ik door veel verschillende versies gegaan. Mijn eerste oplossing was doormiddel van de masters eerst een bericht naar de ander te laten sturen voor het opvragen van of de bus beschikbaar is. Echter kwam ik erachter dat als ik dit precies tegelijk zou doen, dan zou dit nog steeds zorgen voor problemen.

Afbeelding met tekst, schermopname, Lettertype

Automatisch gegenereerde beschrijving

## Versie 2: Lowcode data lezen

Hierna ben ik gaan kijken naar een lowcode oplossing om de SDA lijn af te lezen. Echter had deze oplossing ook nog een probleem. Deze methode leest alleen data af die naar het adres van de desbetreffende master wordt gestuurd. Dit betekent dus dat als de master data zou gaan sturen naar de slave op exact hetzelfde moment er nog steeds een probleem optreed.

Afbeelding met tekst, schermopname, Lettertype

Automatisch gegenereerde beschrijving

## Versie 3: SDA-lijn direct uitlezen

Tenslotte mijn laatste versie lijkt wel een geldige oplossing te zijn. Door direct de SDA-lijn af te lezen kan ik aan de lijn zien of de andere master bezig. Op het moment dat er geen data wordt geschreven word de lijn als hoog afgelezen. Maar op het moment dat dus de andere master data gaat schrijven naar de slave, kan de master dit waarnemen.



Afbeelding met tekst, schermopname

Automatisch gegenereerde beschrijving

Om nu een mooie POC laat ik de masters een byte aan data sturen en niet een volledige afbeelding omdat dit erg veel bits zijn om te controlleren. Voor deze POC gebruik ik dus de methode hieronder.



Afbeelding met tekst, schermopname, software, Multimediasoftware

Automatisch gegenereerde beschrijvingAls je nu kijkt wat de andere master op de SDA leest kan hij vrij accuraat zien hoe de data wordt gestuurd.

# Bronvermelding

*Qwiic Micro OLED Hookup Guide - SparkFun Learn*. (z.d.). <https://learn.sparkfun.com/tutorials/qwiic-micro-oled-hookup-guide/all>

*ATmega328P datasheet*

<https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf>