VRIJE
UNIVERSITEIT
BRUSSEL

ONTWERP EN BOUW VAN EEN ZONNECEL TESTOPSTELLING

Tim Spit
Niels Veldekens

Academiejaar 2016 - 2017

Promotor: Prof. Dr. Ir. S. Bram
Ingenieurswetenschappen - Industriële Wetenschappen (IWT)

Inhoudsopgave

Dankwoord

Zonder de hulp en het geduld van de professoren van de VUB zouden we nooit tot dit eindresultaat gekomen zijn. Op de eerste plaats willen we de heer Svend Bram bedanken voor de vlotte begeleiding van dit project, zijn bereidvaardigheid om buiten de contacturen af te spreken en zijn getoonde interesse tijdens de werkzaamheden.
Voor de technische ondersteuning konden we steeds op de heer Lieven Standaert rekenen.
Wij danken de heer Yannick Verbeelen voor zijn rol door op het juiste moment een nieuw licht te laten schijnen op het plottings-probleem waarmee we geconfronteerd werden. Zijn inbreng was nodig om tot de gewenste meetresultaten te komen.
Tenslotte danken we Eva Spit voor de montage van het begeleidingsfilmpje bij deze bachelorproef.

Inleiding

In het kader van de bachelorproef hebben we de opdracht gekregen om een ‘solartracker’ te bouwen en het maken van een testopstelling voor zonnecellen. Zoals de naam het verraadt, dient een ‘solartracker’ de zon of een artificiële lichtbron te traceren. Daarna moet het toestel ervoor zorgen dat de zonnecellen, gemonteerd op de solartracker, loodrecht staan ten opzichte van het inkomende licht. Dit moet automatisch en handmatig kunnen gebeuren. Het doel van de testopstelling is om de reeds bestaande bevindingen op het vlak van de studies over zonnecellen op een didactische manier te kunnen visualiseren. Daarom willen wij met onze bachelorproef de reeds bestaande kennis uit zijn theoretische context halen en in de praktijk brengen, tastbaar in het klaslokaal. Dit doen we door de gegevens die we verkrijgen uit onze zonnecellen op verschillende grafieken te plotten.
Het idee van een verstelbaar zonnepaneel is niet nieuw. Er werd hieromtrent al veel onderzoek naar gedaan en daarom konden we verschillende opties vinden op het internet voor bijvoorbeeld het design, de manier van tracken en coderen. In volgende paragrafen zullen we de keuzes die we hebben gemaakt bij het bouwen van de solartracker en testopstelling verantwoorden. Hierbij sommen we de verschillende voor- en nadelen op die we hebben afgewogen. Ook vermelden we wat we beter of anders had gekund doen en de problemen die we tegenkwamen.

Deel I
Handleiding

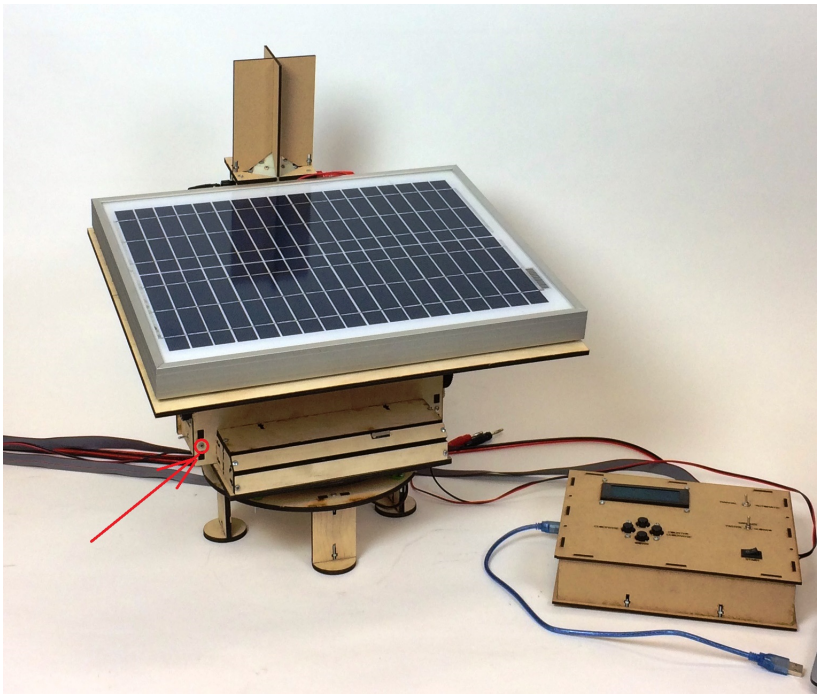
Bij de ontwikkeling van de solartracker werd de gebruiksvriendelijkheid van het toestel en de software als een prioriteit beschouwd. Ondanks het intuïtieve gebruik van de solartracker is een handleiding noodzakelijk.

1 Auto-Rotating solartracker

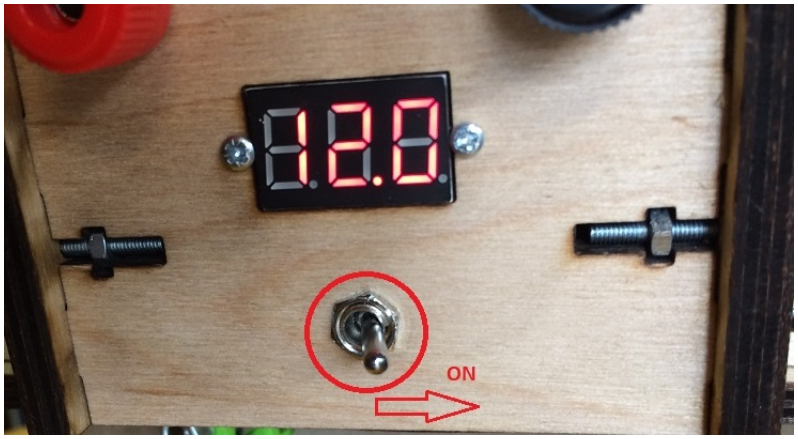
1.1 Solartracker starten

1. Eerst en vooral moeten de motoren van de solartracker van batterij-voeding worden voorzien. Dit gebeurt door middel van de schakelaar die zich aan de zijkant van de tracker bevindt. Door de schakelaar naar de **rechtse positie** te verplaatsen, zal de tracker starten.

   Als de tracker gebruikt wordt, zal een display oplichten. Het getal op deze display is een maat voor de spanning over de klemmen van de batterij.

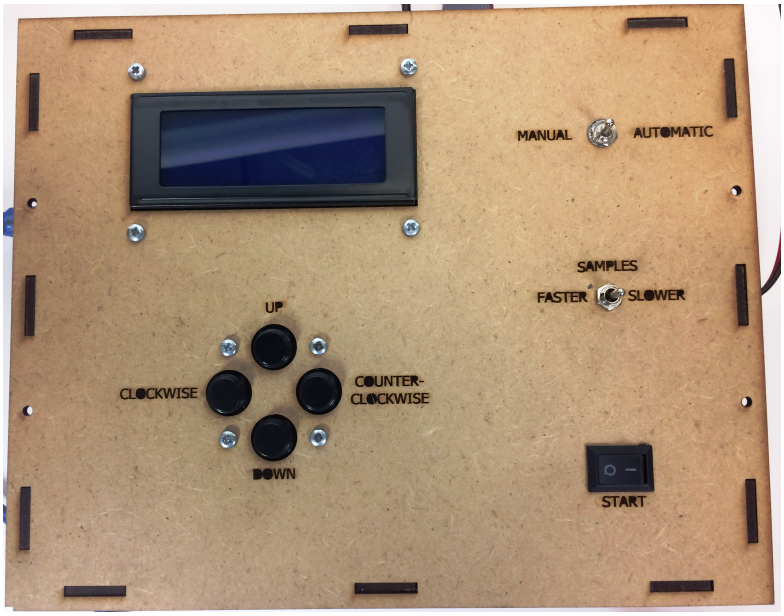


Figuur 1: Aan/uit schakelaar



Figuur 2: Aan/uit schakelaar

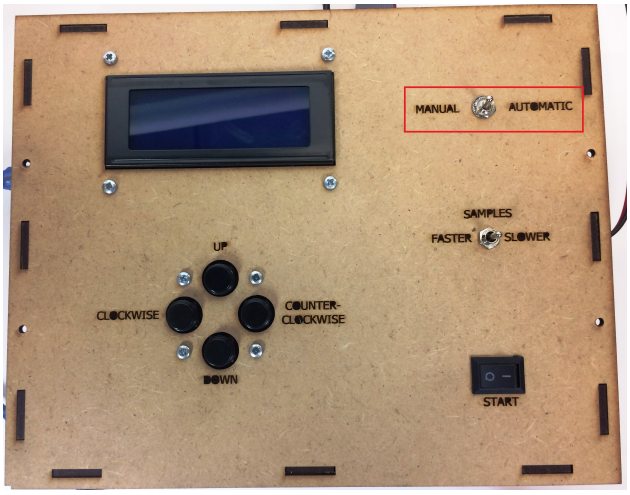
2. Eens de solartracker stroom krijgt, zal hij beginnen met ‘homen’. Het paneel zal blijven bewegen tot de limitswitch wordt aangeklikt. Dan zal het paneel zichzelf weer volledig horizontaal zetten. De rest van de bediening van de solatracker gebeurt met een bedieningspaneel (*ook ‘controller’ genoemd in deze handleiding*).



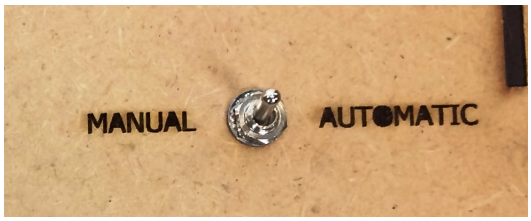
Figuur 3: Controller

1.2 Bediening van de solartracker

3. Men moet nu kiezen of de solartracker manueel of automatisch bestuurd zal worden via de schakelaar (manual / automatic).

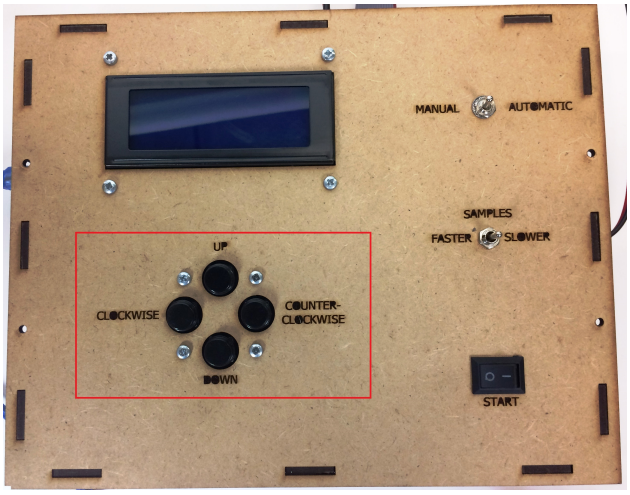


Figuur 4: Manual/Automatic

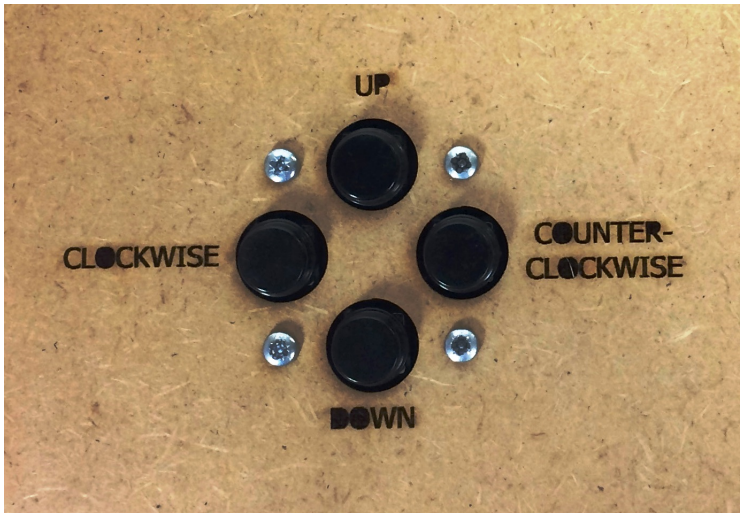


Figuur 5: Manual/Automatic

4. Als er voor manueel wordt gekozen, zullen de vier knoppen worden gebruikt. Voor de up en down knoppen moet er vanuit het standpunt van de piramide worden gekeken.



Figuur 6: Knoppen



Figuur 7: Knoppen

5. Als er voor automatisch wordt gekozen, zal de tracker zelf beginnen te bewegen naar de zon of naar het meeste licht toe. Men moet voor automatisch niets meer doen.

2 Maximum Power Point Tracking

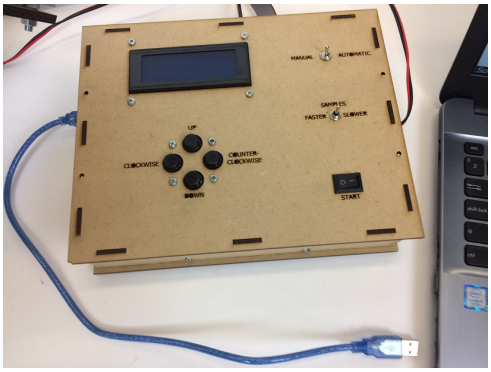
Met behulp van een Arduino en de nodige sensoren wordt de stroom en spanning, opgewekt door de zonnecellen, geregistreerd. Deze meetresultaten worden verwerkt en visueel gemaakt met het softwarepakket 'KST2'[1] [1].

2.1 Controller en belasting aansluiten

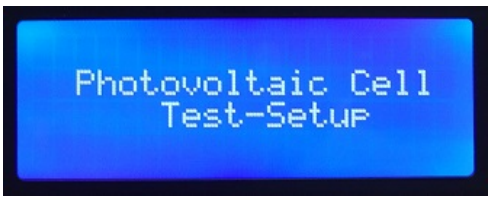
6. Verbind het bedieningspaneel met de computer via de USB-kabel. Een LCD-scherm licht op en de boodschap 'Photovoltaic Cell Test-Setup' verschijnt.

7. Vergewis je ervan dat de belasting (*eventueel een rij van kleine gloeilampjes*) op de controller aangesloten is via de banaanstekkers.

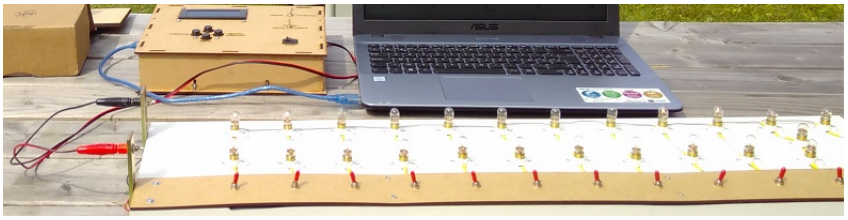
[1] KST2 is open-source software die toelaat gegevens in 'real-time' te visualiseren. Het programma is gebruiksvriendelijk en beschikt over uitgebreide toepassingen waardoor het uitermate geschikt is om onze meetgegevens te visualiseren. Het programma leest de gegevens uit een *.txt-bestand. Meer info: `https://kst-plot.kde.org`



Figuur 8: Controller aansluiten



Figuur 9: Startscherm



Figuur 10: Belasting aansluiten

2.2 Meting starten

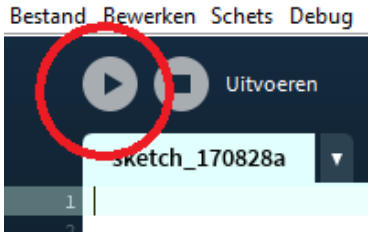
8. Open het bestand 'SolarTracker.pde'. Dit kan op verschillende manieren:
   - Een snelkoppeling naar het bestand bevindt zich op het bureaublad.
   - Het originele bestand bevindt zich in de map: `C:\Users\PV-cell\Documents\MPPT_on_a_auto-rotating_solartracker\SolarTracker`
   - Open het programma 'Processing'[2] en laad het bestand van hieruit.

Figuur 11: Processing

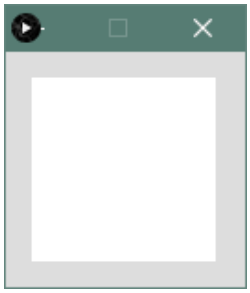
9. Klik op je scherm in de linker-bovenhoek op de 'Start-knop' om het programma te runnen. Er zal centraal op het computerscherm een klein grijs venstertje verschijnen. Dit betekent dat het Processing actief is en de meetgegevens in een *.txt-bestand worden geschreven.

[2]Processing wordt gebruikt om de meetgegevens die seriëel uit Arduino worden gelezen op te slaan in een *.txt-bestand.

Het bestand dat wordt gecreëerd krijgt automatisch als bestandsnaam **de dag en tijdstip** waarop het wordt aangemaakt. Op die manier is het onmogelijk dat er per vergissing een eerder gemaakt bestand wordt overschreven.



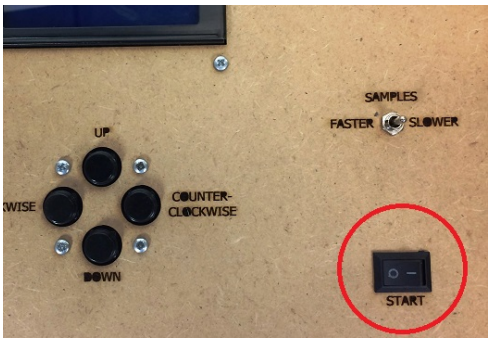
Figuur 12: Run het programma



Figuur 13: Processing actief

10. Op het LCD-scherm van het bedieningspaneel staat ondertussen de instructie om de meting te beginnen. Zet de schakelaar voor automatische of manuele bediening in de gewenste stand. Druk daarna de 'Start-knop' op de controller in, om de meting te starten.

Figuur 14: Instructies controller



Figuur 15: Start controller

Wanneer de startknop is ingedrukt, verschijnt de boodschap dat de meting gestart is. Wacht dan even op het eerste meetresultaat.

Figuur 16: Wachten op eerste meting

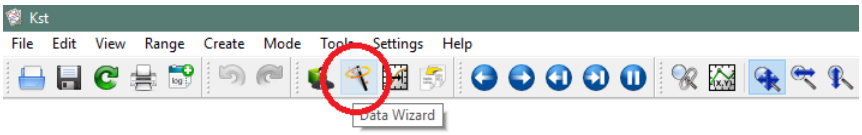
Figuur 17: Meetresultaat

2.3 Meetgegevens visualiseren

11. Open het programma 'KST2'. De snelkoppeling bevindt zich op het bureaublad.

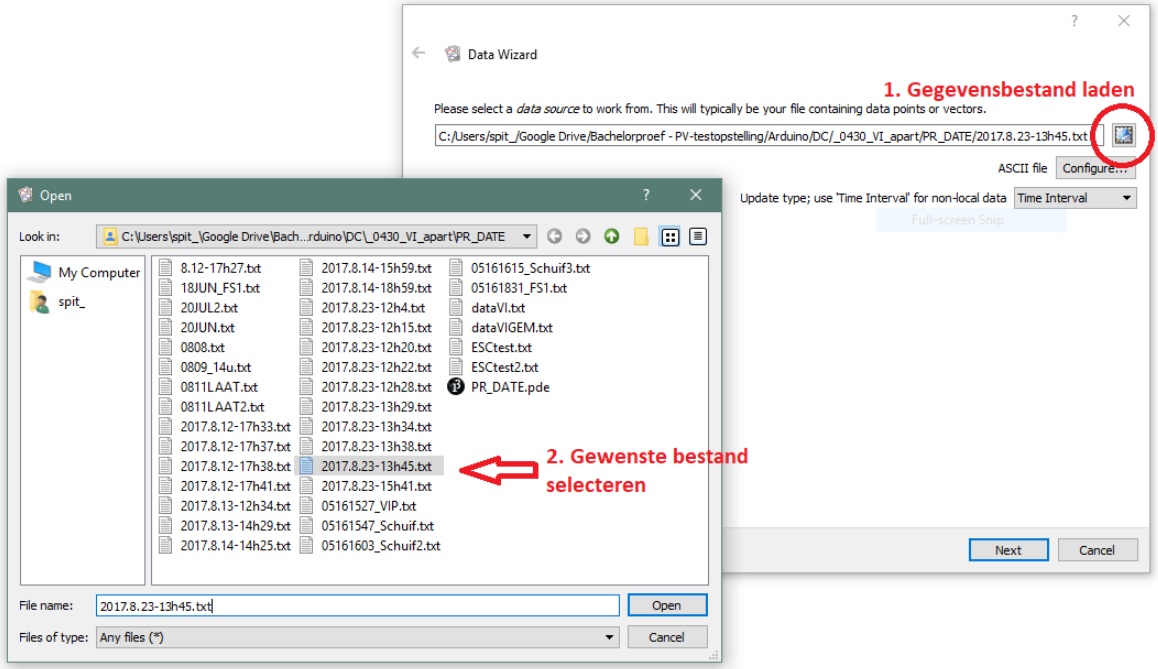
Figuur 18: Plotting software

12. Laad de meetgegevens via de ‘Data Wizard’. Klik hiervoor in het menu op het icoontje met de toverstaf.



Figuur 19: Data Wizard

13. In het venster dat nu verschijnt, klik je op het kleine vierkante icoontje rechts van het bestandspad (*langwerpige rechthoek*). Selecteer het gewenste bestand. Dit bevindt zich in dezelfde map als het eerder gebruikte Processing-programma `C:\Users\PV-cell\Documents\MPPT_on_a_auto-rotating_solartracker\SolarTracker` en heeft als bestandsnaam de dag en tijdstip waarop het werd aangemaakt.

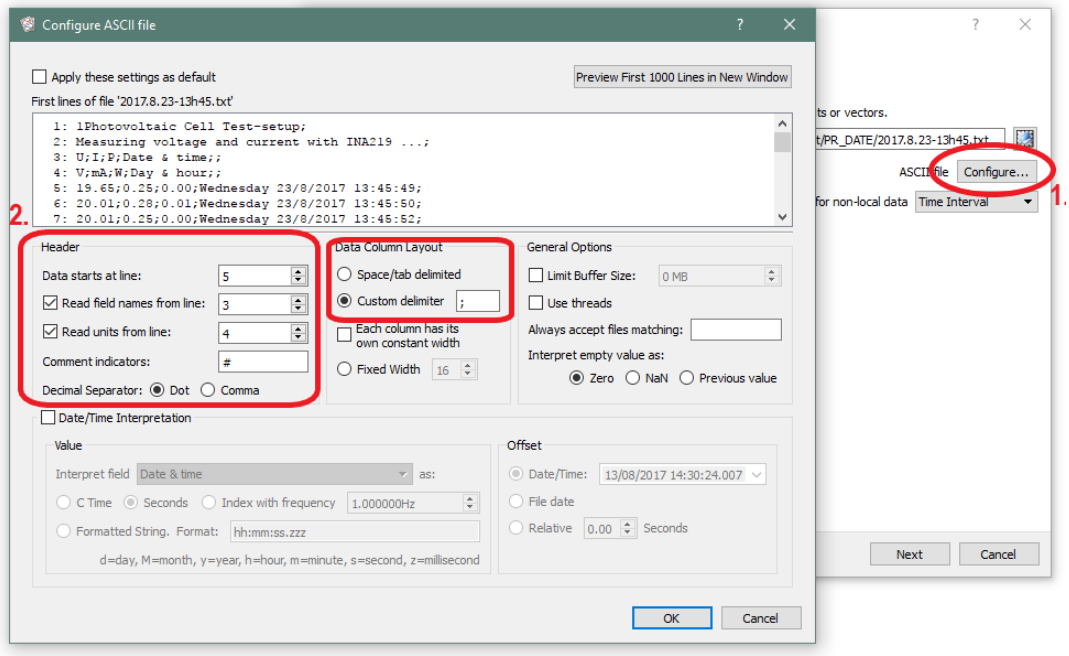


Figuur 20: Gegevensbestand kiezen

14. Klik op ‘Configure...’ om te controleren of KST de meetgegevens correct zal inlezen. In de witte kader worden de eerste regels van het ingeladen *.txt-bestand getoond. Aan de hand hiervan kan je onder andere het volgende nagaan:

    - 1^e regel van de data
    - Regelnummer die de grootheden bevat
    - Regelnummer die de eenheden bevat
    - Scheidingsteken voor decimale getallen (*punt of komma*)
    - Scheidingsteken tussen de verschillende data

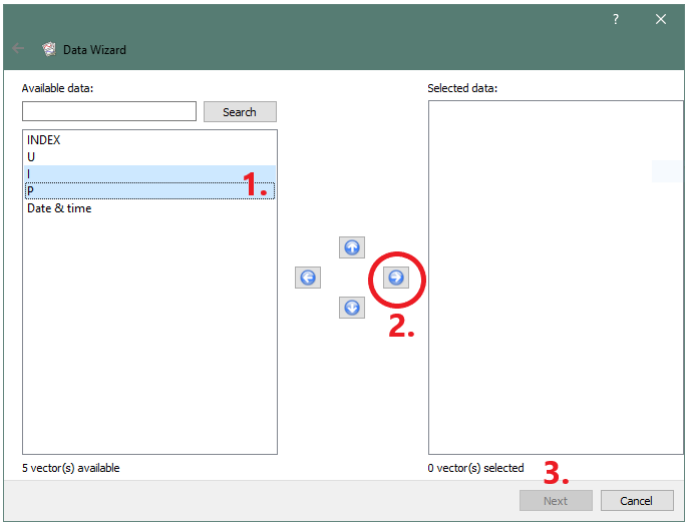
    Controleer goed of alle parameters juist zijn ingesteld. Klik dan op ‘OK’.



Figuur 21: Configure...

15. In het volgende scherm kies je de grootheden die je wilt plotten. Om de $I(U)$- en de $P(U)$-grafiek te plotten:

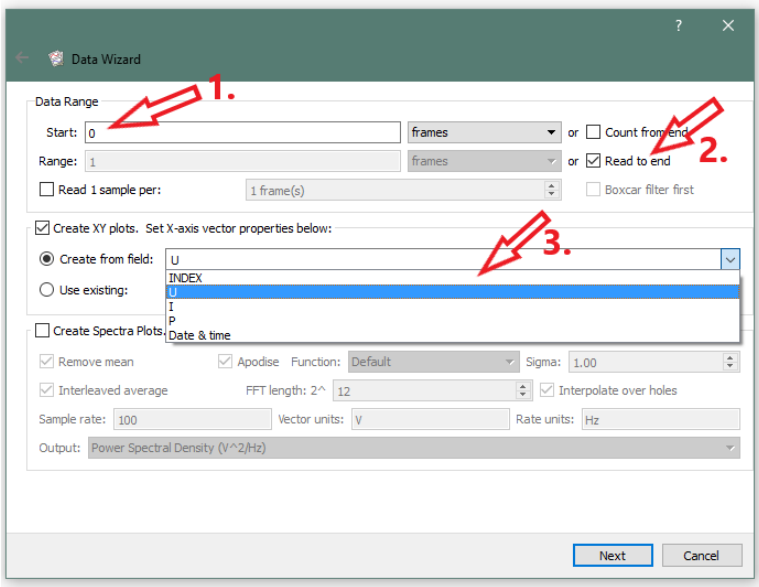
    - selecteer je I en P
    - klik op de pijl naar rechts (*De gekozen grootheden staan nu in het rechtse veld*)
    - ‘Next’



Figuur 22: Grootheden kiezen

16. Vervolgens ga je volgende parameters na:

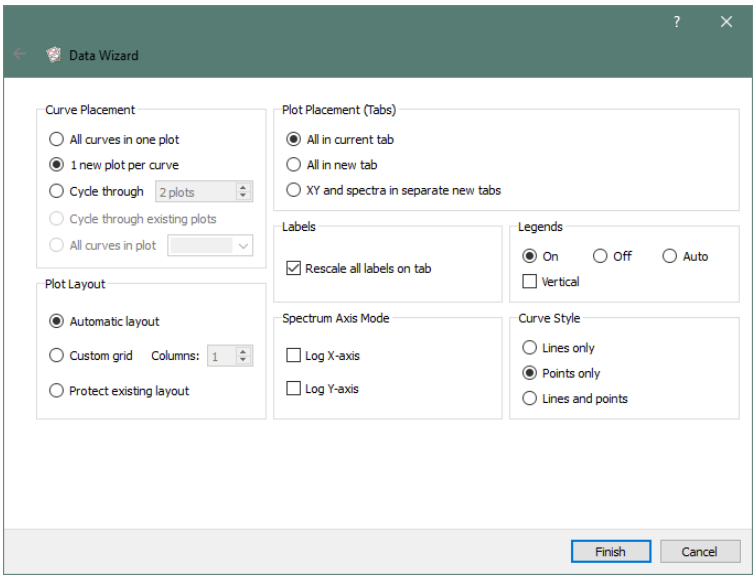
    - Start = 0
    - 'Read to end' is aangevinkt
    - 'Create XY plots. Set x-axis vector properties below' is aangevinkt
    - Selecteer 'Create from field:' en kies uit het dropdownmenu voor U.
    - 'Next'



Figuur 23: Gegevens inlezen

17. Tenslotte kan je in het laatste venster de lay-out van de grafieken nog aanpassen. Mogelijkheden zijn:

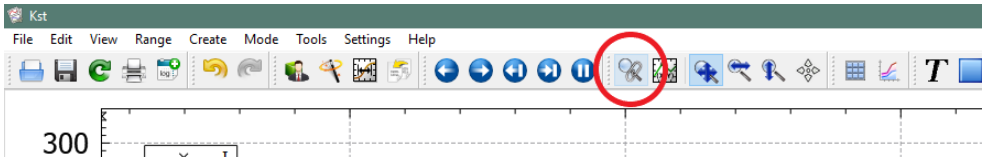
    - beide grafieken in één en hetzelfde assenstelsel of voor elke plot een apart assenstelsel
    - legende bij elke plot
    - meetresultaten als PUNTEN of als VERBINDINGSLIJNEN. (*Kies voor **punten***)
    - 'Finish'



Figuur 24: Lay-out

De meetresultaten worden nu geplot in de assenstelsels. De ijk van de assen past zich automatisch aan de ingelezen data. Het is echter mogelijk de ijk van de assen **manueel** aan te passen. Ga daarvoor als volgt te werk:

18. Klik in het menu op het icoontje met het vergrootglas en paperclip. Hierdoor worden beide grafieken aan elkaar gekoppeld. Als je de ijk van de ene grafiek aanpast, zal dit ook gebeuren met de tweede grafiek. Indien de ijk van de beide grafieken moet verschillen, klik je opnieuw op dit icoontje.
    Om te weten of de grafieken gekoppeld zijn of niet, wordt er rechtsboven elke grafiek een bolletje ingekleurd. Een donker bolletje betekent 'gekoppeld'.

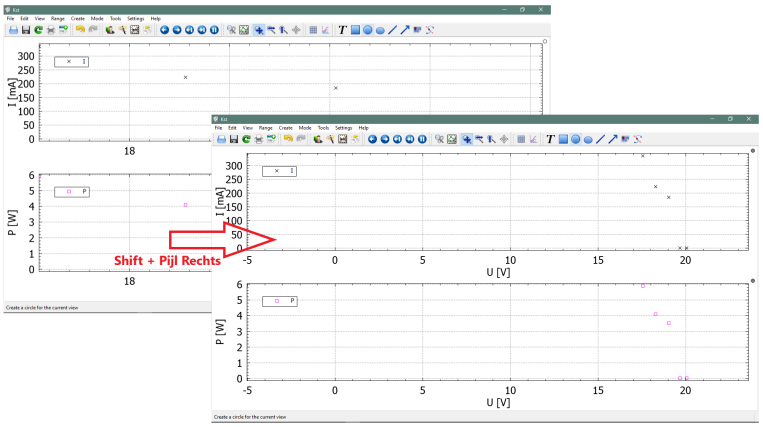


Figuur 25: Ijk aanpassen - grafieken koppelen

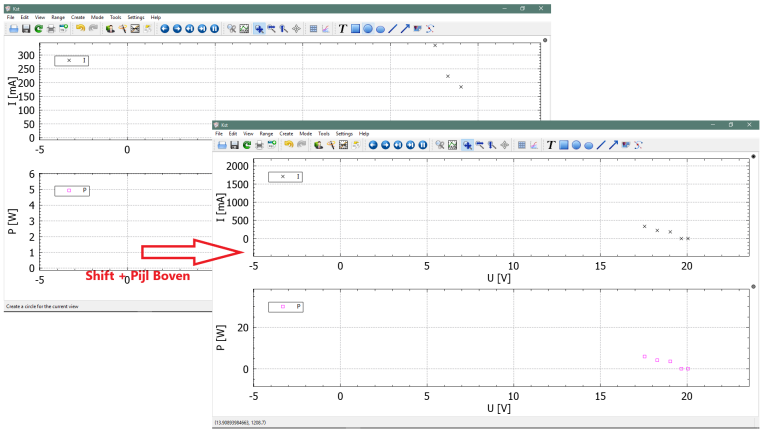
19. Gebruik volgende toetsenbordcombinaties om in- en uit te zoomen:

    - SHIFT + PIJL RECHTS: X-as uitzoomen
    - SHIFT + PIJL LINKS: X-as inzoomen
    - SHIFT + PIJL BOVEN: Y-as uitzoomen
    - SHIFT + PIJL ONDER: Y-as inzoomen

    Met behulp van de pijltjestoetsen kan je in alle richtingen navigeren in het assenstelsel.



Figuur 26: Uitzoomen en verplaatsen X-as



Figuur 27: Uitzoomen en verplaatsen Y-as

2.4 Bediening

20. Om de snelheid van de metingen aan te passen, kan je op het bedieningspaneel de knop ‘Samplespeed’ bedienen. De tijdsintervallen waarin er kan worden gesampled zijn: 0.5 s; 1 s; 2 s; 3 s; 5 s; 10 s; 30 s; 1 min; 2 min of 5 min.

    Op het LCD-scherm zal de gekozen snelheid verschijnen.

Figuur 28: Bediening

Figuur 29: Gekozen samplespeed

21. Om op zoek te gaan naar het Maximum Power Point van het zonnepaneel, moet de belasting op het paneel aangepast worden. Dit kan door een serie van lampjes aan te sluiten op de controller, maar eender welke andere regelbare belasting voldoet. Belangrijk hierbij is dat het gedissipeerd vermogen van de belasting moet overeenkomen met het vermogen opgewekt door de zonnecellen. Hou hierbij rekening bij de keuze van de belasting.

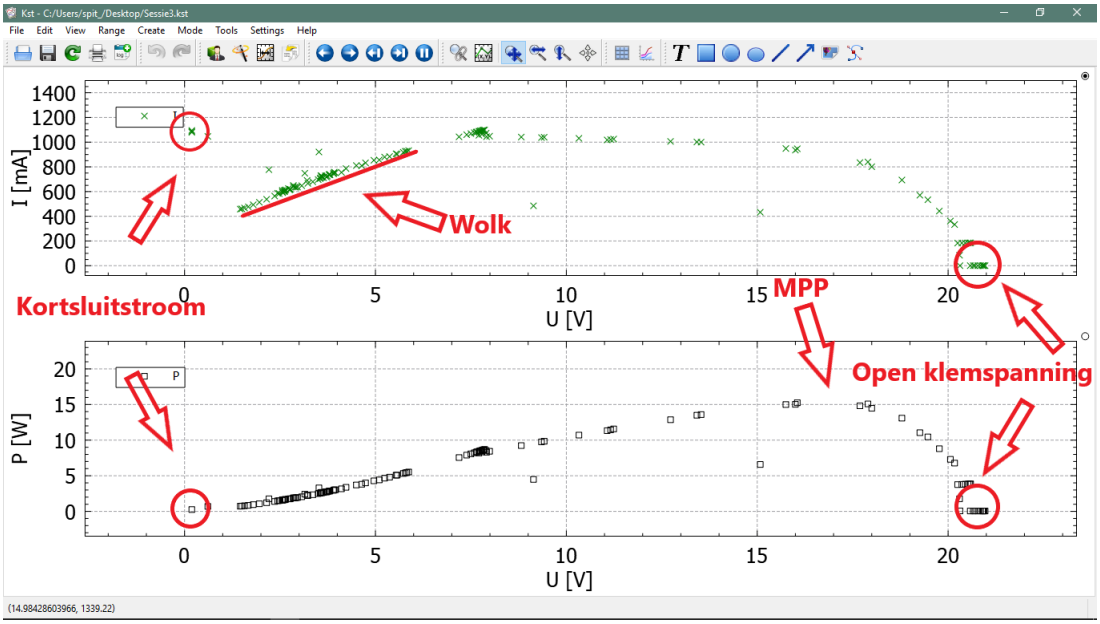
    Wanneer de belasting op het paneel **toeneemt**, vermindert de spanning over de klemmen van het paneel maar zal de stroom toenemen, tot de maximale waarde die het paneel kan opwekken, bereikt wordt. Met andere woorden, de meetpunten op de $I(U)$- en $P(U)$-grafiek verschuiven naar links.

    Door de banaanstekkers **LOS** te koppelen van de belasting, creëer je *open-klemspanning*.

Punten uiterst rechts-onder in de $I(U)$-grafiek worden geplot. Hierbij is de STROOM gelijk aan NUL.

Wanneer je banaanstekkers **RECHTSTREEKS met ELKAAR** verbindt, zonder belasting, stroomt de *kortsluitstroom*. De SPANNING is gelijk aan nul en punten links-boven in de $I(U)$-grafiek verschijnen.

Merk op dat in de $P(U)$-grafiek het vermogen in beide gevallen NUL is.



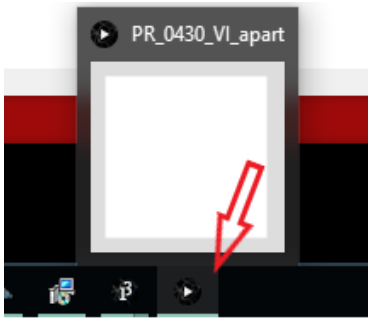
Figuur 30: MPPT

Opm.: In bovenstaande grafiek is het 'niet-rechtlijnig' verloop van de $I(U)$-grafiek het gevolg van bewolking. Bij heldere hemel kent de grafiek in de buurt van de kortsluitstroom een horizontaal verloop.

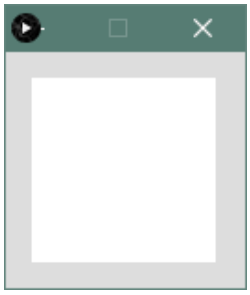
3 Einde van de metingen

22. Het *.txt-bestand dat gegenereerd wordt, moet correct afgesloten worden. Hiervoor klik je in de taakbalk op het actieve Processing-venster, te herkennen aan de pijl die naar rechts wijst. Het kleine venster komt op de voorgrond. Druk dan op de **esc-toets**. Het nieuwe *.txt-bestand wordt correct afgesloten en Processing kan nu worden verlaten.

23. Wanneer alle metingen achter de rug zijn, kunnen zowel de schakelaars op het bedieningspaneel als op het toestel zelf uitgeschakeld worden. Op de tracker is de middelste positie van de schakelaar de OFF-stand. Als via het bedieningspaneel de metingen gestopt zijn, kan de USB-kabel los gekoppeld worden van de computer.



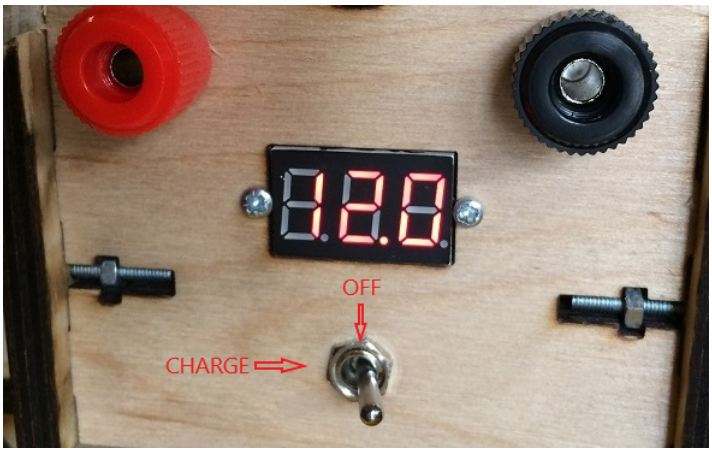
Figuur 31: Actieve Processing-venster



Figuur 32: Druk op de esc-toets

4 Batterij van de tracker opladen

24. Een auto-batterij is volledig opgeladen als je 12.6 V meet over de klemmen. [2] Als de spanning echter te laag wordt, onder 12 V, zal deze onvoldoende vermogen leveren aan de motoren van de tracker. Je kan een spanning van 12.4 V beschouwen als de grens om te overwegen de batterij opnieuw op te laden. Als de spanning nog verder afneemt, heeft dit nefaste gevolgen voor de levensduur van de batterij. Om schade te vermijden, zorgt een interne schakeling er voor dat bij een klemspanning onder de 11.4 V de batterij ontlast wordt en wordt het circuit automatisch onderbroken. De batterij loskoppelen om deze op te laden is niet nodig. Je kan eenvoudigweg via banaanstekkers de batterij aansluiten op een gepaste voedingsbron via de zijkant van de tracker. Je kan de batterij pas opladen als de 'Aan-uitschakelaar' in de uiterst linkse positie staat. Let bij het opladen dat de laad-spanning tussen de 13.5 V en 14.5 V bedraagt. [3]



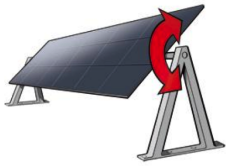
Figuur 33: Charge - OFF - ON

Deel II
Technische achtergrond

5 Algemeen

Over de manier waarop we de solar tracker bewegen, hebben we niet lang moeten nadenken. We hadden slechts twee keuzes, namelijk een solar tracker met één as of met twee assen. Bij een 1-assig systeem zal het zonnepaneel de zon van oost naar west kunnen volgen. Wij kiezen voor onze bachelorproef echter voor een rotatiesysteem over twee assen om op die manier de meest nauwkeurige oriëntatie ten opzichte van de zon te bekomen. Met één as kan de solartracker ook niet binnen gebruikt worden omdat de hoek tussen de tracker en de lamp natuurlijk niet altijd dezelfde is in verschillende lokalen.

Figuur 34: 1 as (a)



Figuur 35: 1 as (b)

Met een 2-assig systeem kan een zonnepaneel naar alle kanten, in de bovenste helft van een bol, gericht worden. Binnen de 2-assige systemen hebben we nog verschillende opties, zoals een slew drive, een lineaire actuator, motor op assen zonder overbrenging en motor met overbrenging. We hebben geopteerd voor een grote overbrenging (zie 6.6). Hierdoor zal het paneel niet zo snel kunnen bewegen. De tragere snelheid is gunstig omdat men het paneel dan nauwkeuriger kan bewegen.

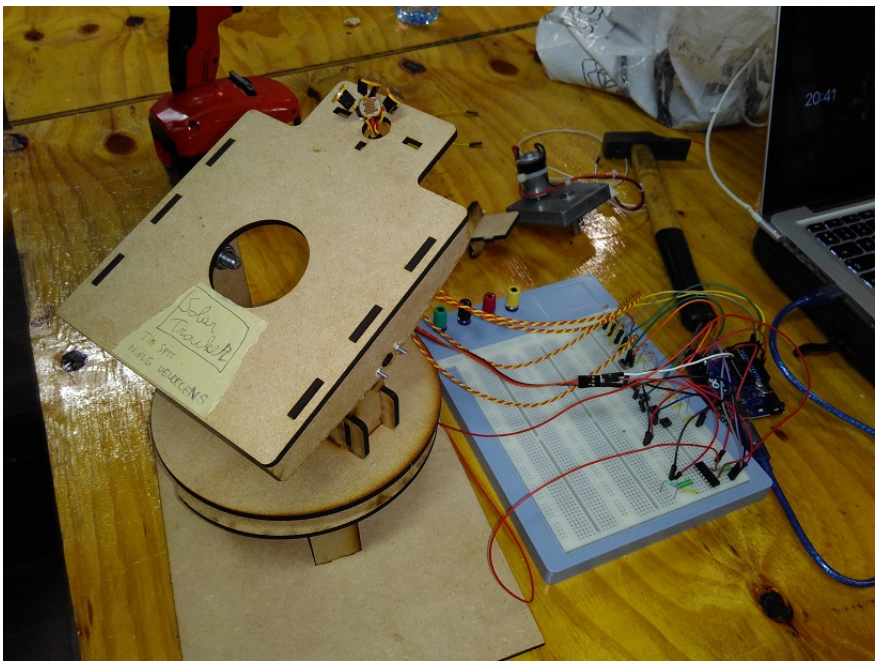
Figuur 36: 2 assen

Het nadeel bij dit concept is dat, als de verticale as meerdere keren ronddraait, de kabels die verbonden worden met de motor, zonnecellen, controler en de arduino in de knoop zullen geraken en zichzelf uiteindelijk zullen kapot trekken. De oplossing die we hiervoor hebben

gevonden is: ten eerste de motor zo te plaatsen dat deze meedraait met de zonnecellen en bijgevolg de kabels ook, ten tweede de elektronica en batterij ook op de solartracker plaatsen. Natuurlijk zijn er nog altijd kabels zoals die voor de knoppen, batterij, ... die niet gemakkelijk weggewerkt kunnen worden. Later bespreken we mogelijke oplossingen.

5.1 Prototype

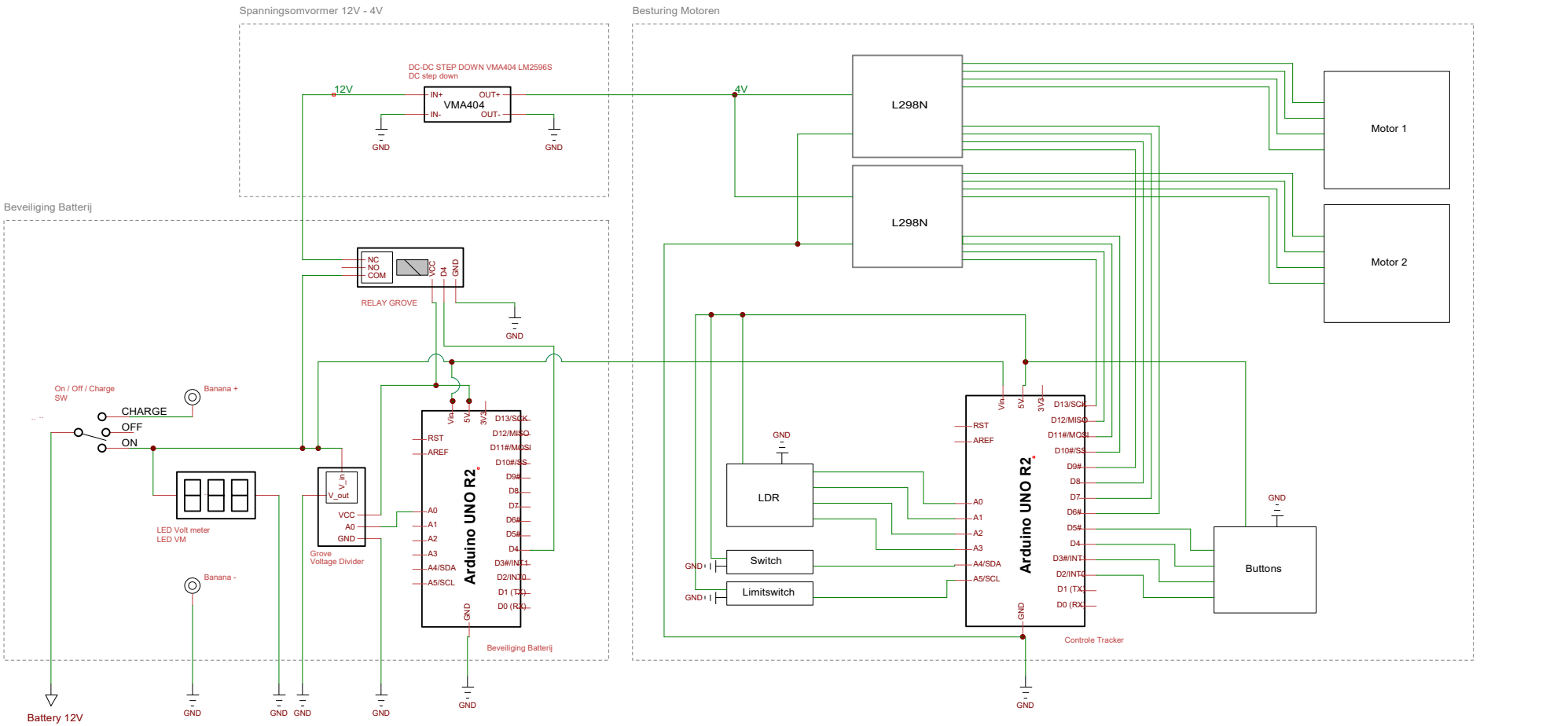
Struikelblokken zijn onoverkomelijk tijdens het uitbouwen van een project. Daarom hebben we besloten om een prototype te bouwen. Het prototype is een verkleinde versie van onze uiteindelijke versie van de solar tracker. Het prototype heeft ons geholpen om de 'beginnersfouten' uit ons design te halen en oplossingen te zoeken. Het eerste deel van het onderzoek heeft veel tijd in beslag genomen, maar ook onze eindversie. Alhoewel de problemen uit het prototype definitief verwijderd zijn, zijn er bij de tweede versie andere problemen naar boven gekomen. Dit zorgde voor veel vertraging in het schema. Beter zou zijn geweest om eerst over een degelijk en robuust design na te denken en mogelijke problemen of moeilijkheden voorspellen in plaats van meteen iets te bouwen.



Figuur 37: Eerste prototype

6 Auto-Rotating solartracker

6.1 Draadschema



Title Solar Tracker		
Author Niels Veldenkens; Tim Spit		
File ctrische schema's\SCHEMA_TRACKER_Niels.dsn		Document
Revision 1.0	Date	Sheets 1 of 1

6.2 Keuze batterij

Verschillende elektronische onderdelen moeten in de tracker van voeding worden voorzien: Arduino, motoren, drivers voor motoren. Omdat het mogelijk moet zijn de solartracker op een locatie te plaatsen waar hij gedurende uren de zon kan volgen, zochten we een mobiele oplossing. De oplossing voor een batterij ligt dus voor de hand. Hierbij moet echter rekening gehouden worden dat deze stroombron het lang genoeg kan uithouden. De stroom die de motor trekt, bedraagt volgens onze metingen gemiddeld 1.6 A. Dit is ook de stroom die nodig is om de steppermotoren in een vaste positie te laten staan. Laat ons als voorbeeld nemen dat we de zon 6 uur lang willen volgen. Dit betekent dat onze stroombron minstens 1600 mA$\cdot 6$ h $= 9600$ mAh moet kunnen leveren.
De steppermotoren kunnen draaien onder een breed spectrum van voltages. (zie later 6.6) De keuze van de batterij hangt daarom ook af van de Arduino. De V_{in} van de Arduino moet minsten 6 V bedragen. Uit ervaring bleek dat de spanning over de klemmen van de bron afnam wanneer de motoren het zwaarst belast werden. Het probleem met een 6 V batterij was dat deze verminderde spanning onvoldoende werd als voeding om de Arduino probleemloos te laten functioneren. De keuze voor een 12 V-batterij ligt dan voor de hand.
Vermits de stroomvoorziening van deze batterij ook toereikend voor het project moet zijn, leidt dit tot de keuze van een auto-batterij.

6.3 Beveiliging van de auto-batterij

Om te vermijden dat door het langdurige gebruik van de auto-batterij, het voltage onder 11.4 V zakt, [2][3] wordt de spanning over de klemmen constant gemeten door een spanningsdeler. Deze is gekoppeld aan een Arduino-micro. Wanneer de spanningsdeler een spanning onder 11.4 V registreert, zal een relais aangestuurd worden die de keten onmiddellijk onderbreekt. De solartracker zal niet meer autonoom of via de manuele bediening kunnen bewegen. Op het kleine 3-digit scherm zal echter de spanning over de klemmen nog steeds af te lezen zijn. Dat het circuit onderbroken is, kan dus niet worden afgeleid uit het oplichten van de rode cijfers, wel door de waarde die ze aangeven.
De Arduino-code is geschreven zodat de gemeten spanning het resultaat is van het gemiddelde van 1000 metingen. Het berekenen van het gemiddelde vermijdt dat de Arduino onmiddellijk zou reageren op toevallige uitschieters. Wanneer de schakeling onderbroken is, zal deze enkel automatisch opnieuw ingeschakeld worden als de spanning over de klemmen meer dan 12.6 V bedraagt. Een groot interval tussen beide spanningen (11.4 V en 12.6 V) was ook nodig om te vermijden dat de schakeling rond een grenswaarde aan-uit knipperde.
Pseudo-code:

Code Arduino 6.1

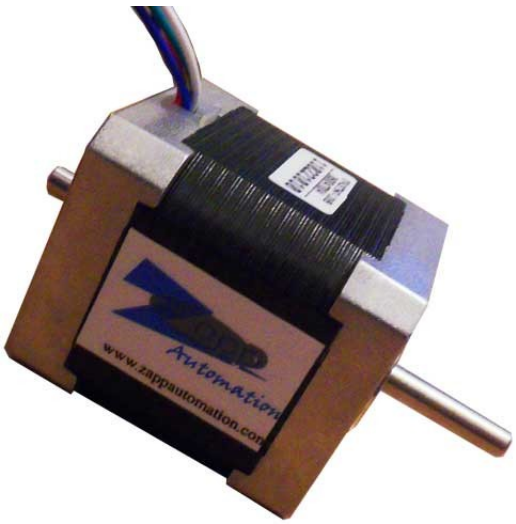
```
for(int i=0;i<1000;i++)
    {
```

```
        sum=sensorValue+sum;
        sensorValue=analogRead(A1);
    }
sum = sum/1000;

if (Vbat < 11.40) { //Relais in NC-toestand
     digitalWrite(pinRelais, HIGH); // High-signaal
onderbreekt de arduino.
    }
else if (Vbat > 12.60) {
    digitalWrite(pinRelais, LOW);
    }
```

6.4 Motoren

Bij het prototype hebben we eerst servo's gebruikt voor de beweging. Al snel wisten we dat servo's niet sterk genoeg waren om een zwaarder gewicht en een groter moment te dragen. Daarom werden deze vervangen door dc-motoren. Bij het testen van de solartracker met zonnecellen merkten we op dat het paneel telkens omviel door het gewicht van de zonnecellen waardoor het paneel telkens op en neer wipte in plaats van altijd op één positie te blijven. Bijgevolg hebben we uiteindelijk gekozen om steppermotoren te gebruiken. Deze type motoren kunnen in holding torque staan. Als het paneel goed gericht staat, zal het ook zo blijven staan, ongeacht het gewicht dat erop wordt uitgeoefend.

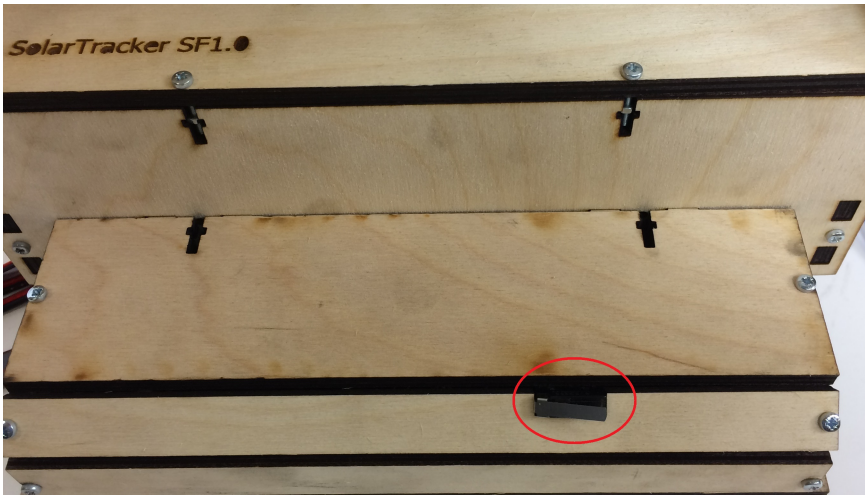


Figuur 38: Stepper motor

Een ander voordeel dat de stepper heeft ten opzichte van de dc-motoren is dat men de positie

van een stepper kan controleren en bijhouden door het aantal stappen dat de motor maakt op te tellen. Homen van het paneel is nu noodzakelijk, omdat we de beginhoek niet kennen. Bij het opstarten van de solartracker zal er dus automatisch gehomed worden met behulp van een limitswitch. Wanneer de limitswitch wordt aangeklikt, zal het programma de steps van de motor bijhouden. De solartracker werd zo geprogrammeerd dat het zichzelf niet kan kapot draaien door een te laagstaande lichtbron/zon of door een slechte besturing. Daarna zal de tracker zich automatisch parallel zetten met de grond.

Figuur 39: Limit switch

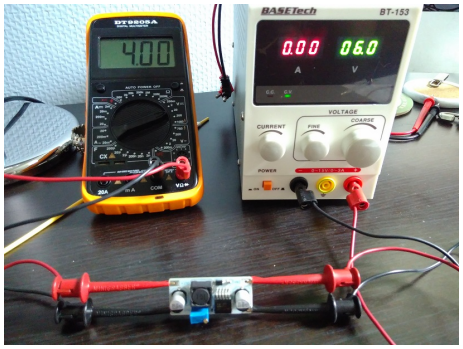


Figuur 40: Limit switch

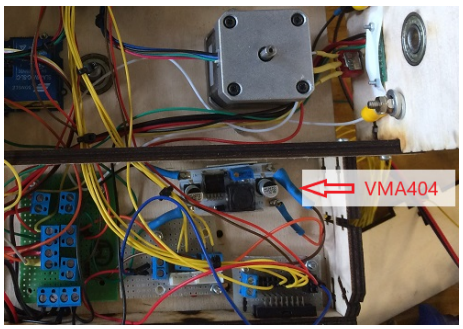
6.5 Voeding motoren

Bij het in gebruik nemen van de solartracker werden we een ratelend geluid gewaar. Naar de precieze oorzaak hiervan, is geen grondig onderzoek verricht. Een oplossing was echter om de motoren niet rechtstreeks te voeden via 12 V. Bij lagere spanningen werden we niet geconfronteerd met dit probleem. Om de batterijspanning te verlagen werd een DC-DC StepDown omvormer tussen de batterij en de motoren geplaatst. Op die manier wordt een constante

spanning van 4 V gegarandeerd. [3] Er werd gekozen voor de *DC-DC adjustable voltage STEP DOWN module LM2596S - Velleman VMA404*. [4]



Figuur 41: Omvormen van 6V naar 4V

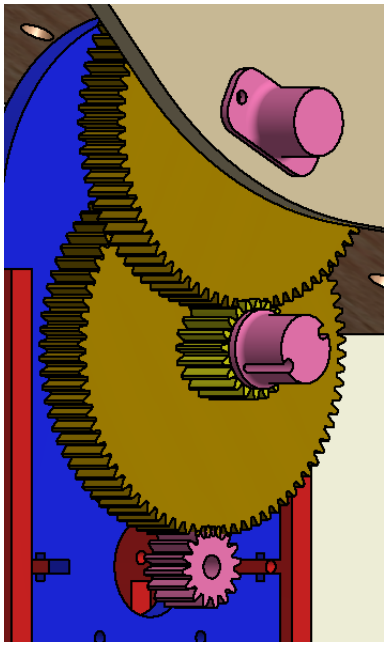


Figuur 42: Module gemonteerd

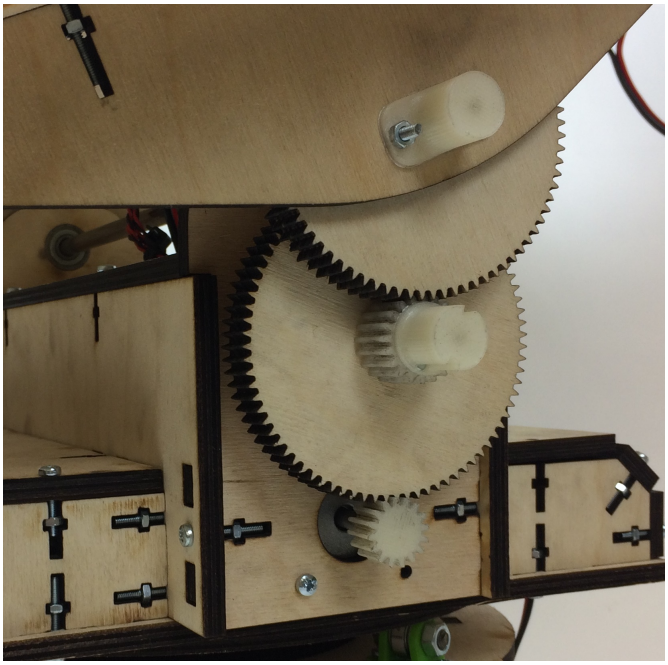
6.6 Tandwielen

De stepper motoren die gebruikt worden zijn nema 17 SY42STH47-1206B High Torque Hybrid Stepper Motors. Deze hebben een holding torque van 3,17 kg.cm wat overeenkomt met 0,311 N.m. Het paneel waarop de zonnecellen komen heeft een afmeting van 35 cm en het zonnepaneel weegt 2,5 kg. Een goede overbrenging is zeker nodig. We hebben gekozen voor een tandwielaandrijving. Tandwielen zijn compact, hebben hoge vermogens en zijn gemakkelijk te maken. Na enkele berekeningen weten we dat een overbrenging van 1 op 20 moet volstaan. Omdat er weinig plaats is voor grote tandwielen, maken we gebruik van een ”two stage gear reduction”.

[3] 4 V was de spanning die de motoren vlot liet draaien, met de afwezigheid van het ratelende geluid van de tandwielen.

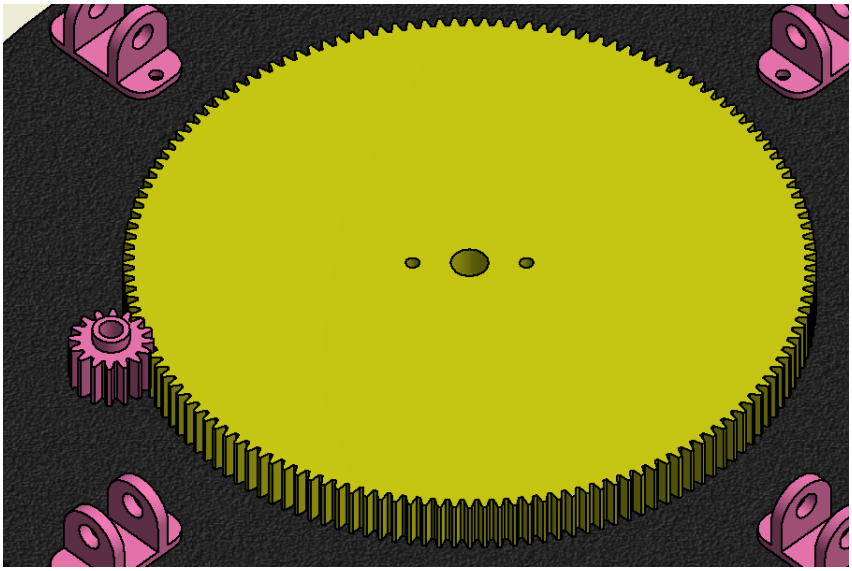


Figuur 43: Two stage gear reduction



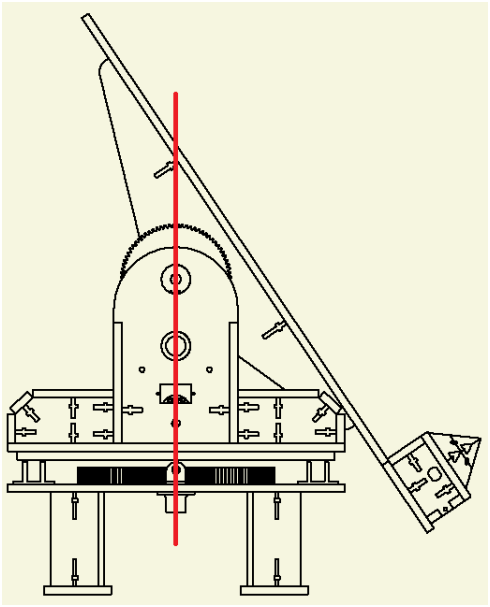
Figuur 44: Tandwielen

Voor de beweging van de schijf is er niet zo'n grote overbrenging vereist, de kracht die nodig is, is enkel de wrijvingskracht van de tracker op de lagers.



Figuur 45: One stage gear reduction

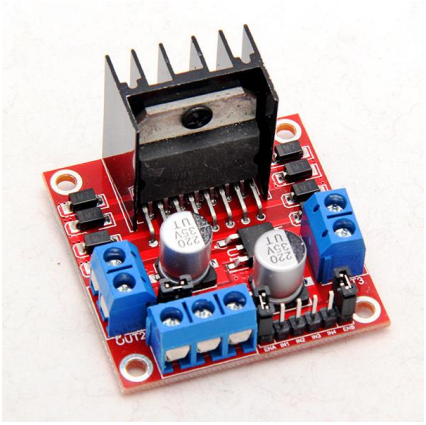
Het design is echter niet optimaal: het paneel waarop de zonnecellen liggen, is hoger dan de as die het paneel doet draaien. Als het paneel onder een hoek komt te staan, zien we duidelijk dat de motor bijna heel het gewicht van de zonnecellen moet opheffen om terug te draaien. Een beter idee is om het paneel op hetzelfde vlak van de as te leggen waardoor een deel van het gewicht, aan de ene kant van de as, wordt gecompenseerd door het gewicht aan de andere kant van de as.



Figuur 46: Ongelijke massaverdeling

Op de tekening ziet men duidelijk dat de motor heel het gewicht van het paneel dat rechts van de rode lijn staat moet opheffen.

Om de motoren in beide richtingen te kunnen laten draaien worden stepper drivers gebruikt. We gebruiken de drivers L298N. Ze werken bij een spanning van 4V tot 46V en kunnen 4 Ampère aan. Dit is perfect voor de solartracker.



Figuur 47: L298N

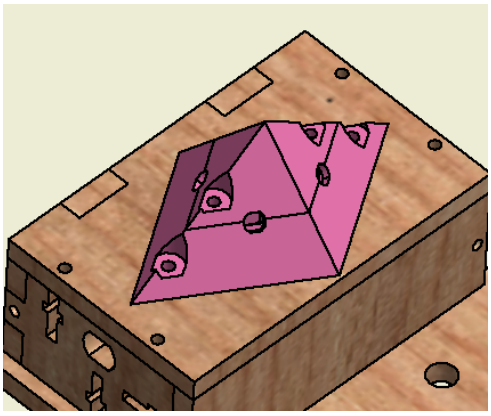
6.7 Detectie lichtbron

Eén van de uitdagingen in dit project is dat het zonnepaneel zich kan richten naar de zon. Opnieuw waren hier veel verschillende opties om dit te doen, maar de beste optie leek ons om te werken met LDR's of Light Depending Resistors.

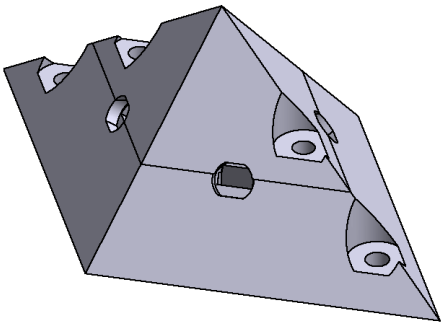
Figuur 48: LDR

Een LDR zal een bepaalde hoeveelheid stroom doorlaten afhankelijk van de hoeveelheid licht dat er op schijnt. We dachten dus om vier LDR's zo op het paneel met de zonnecellen te plaatsen dat ze elk een andere richting uit kijken. Omdat de zon of het licht uit een bepaalde richting komt, zal bijgevolg een of twee LDR's meer licht krijgen dan de andere. Dit zal zo blijven

totdat het paneel loodrecht staat op de stralen van het inkomend licht. We zullen de motoren dus doen draaien totdat alle LDR's hetzelfde resultaat doorgeven. We weten welke LDR het meeste licht krijgt en bijgevolg ook welke kant de motoren moeten draaien. De beweging van het zonnepaneel gebeurt dus gericht.
De moeilijkheid was om de LDR's allemaal in eenzelfde hoek te plaatsen. Daarom gebruiken we een 3D geprinte piramide waarin de LDR's passen en deze piramide wordt bovenop het paneel bevestigd.

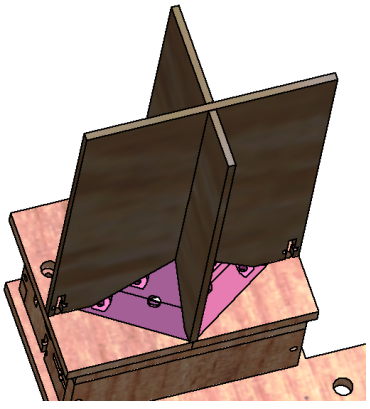


Figuur 49: Piramide

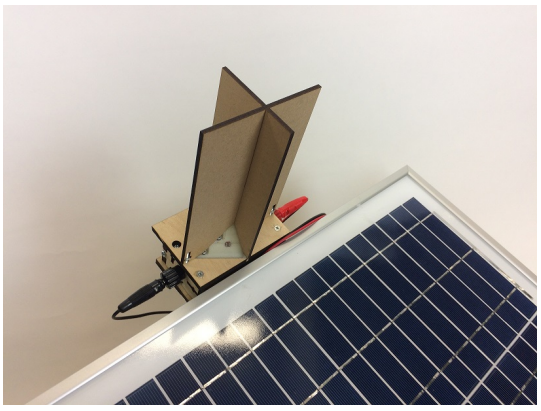


Figuur 50: Piramide

Een nadeel van LDR's is dat deze geen hoge nauwkeurigheidsgraad hebben. Hierdoor kan het gebeuren dat de solartracker niet 100% loodrecht staat op de lichtinval, maar deze fout heeft geen grote invloed om de meting van de MPP. Wij besloten om vier LDR's te gebruiken voor een grotere nauwkeurigheid, drie zou ook gekund hebben. Deze piramide is handig en werkt goed als er veel licht of een felle zon is. Tijdens het testen bij veel bewolking merkten we echter dat de piramide niet goed genoeg was. Daarom hebben we vinnen in de vorm van een kruis boven de piramide gemonteerd die voor meer schaduw zorgt als de tracker niet loodrecht staat op de zon.



Figuur 51: Kruis

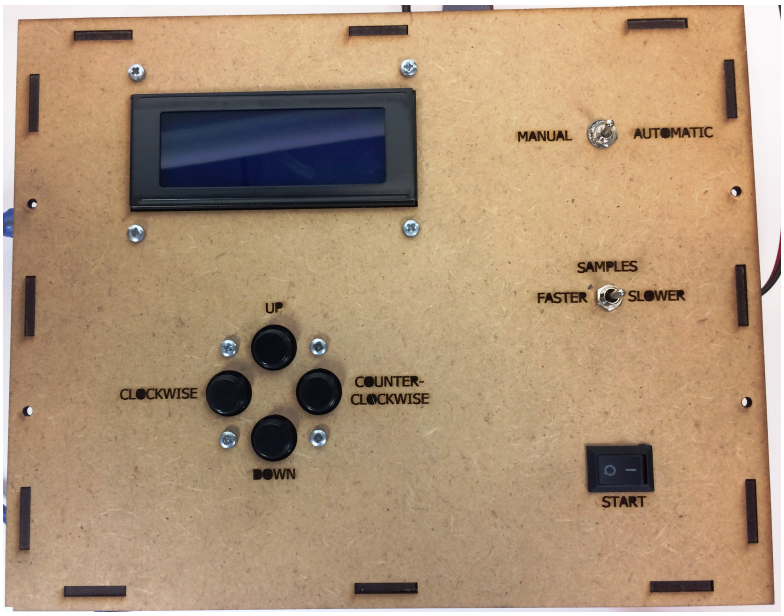


Figuur 52: Kruis

Een andere, en meer industriële methode om een zonnepaneel te richten naar de zon, is door de astronomische data van de zon te kennen en deze data te implementeren in een algoritme. Op die manier zal het zonnepaneel automatisch, zonder gebruik van sensoren of andere vorm van tracker, zich juist positioneren naar de zon. We vinden dit een minder interessante mogelijkheid omdat het zonnepaneel enkel te gebruiken valt op een welbepaalde breedtegraad op aarde. Door een solartracker echter, kan deze technologie op eender welke plaats op aarde worden toegepast. Zo zou de solartracker ook niet binnen kunnen worden gebruikt.

6.8 Manuele bediening

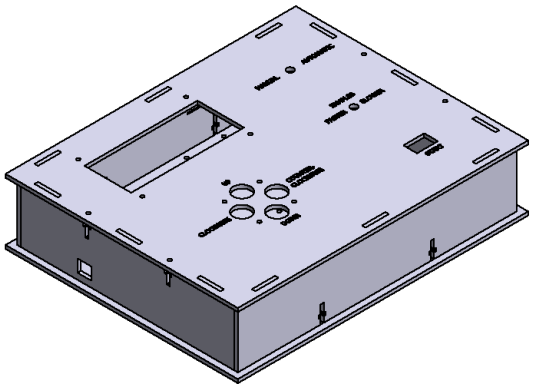
De solar tracker moet ook manueel kunnen bestuurd worden. Een switch zal bepalen of de solar tracker manueel of automatisch bestuurd wordt. De automatische kant werd hierboven al uitgelegd. De manuele besturing gebeurt door push buttons. Per motor gebruiken we twee knoppen, elk om een verschillende kant op te gaan. We ontwierpen dan ook een controller. De controller is via een flatcable verbonden met de solar tracker. Hierop staat een schakelaar om het plotten te starten en één om de snelheid van de samples te bepalen. Een LCD-scherm toont de informatie over het plotten en de zonnepanelen.



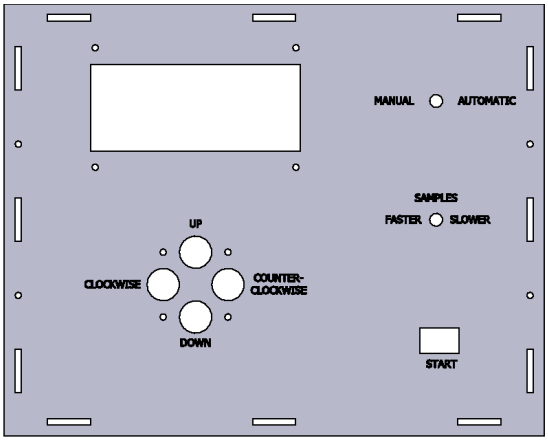
Figuur 53: Controller

6.9 Arduino-code

Nu volgt een bondige uitleg over de code die we gebruiken om de solartracker te doen bewegen. We maken gebruik van Arduino, omdat dit een taal is waar we al ervaring mee hebben.



Figuur 54: Controller



Figuur 55: Controller

Het programma bestaat voornamelijk uit twee grote delen nl. het automatische gedeelte en het manuele gedeelte. In de manuele loop worden de buttons en de plaats van het paneel gecontroleerd om te kunnen bewegen. In de automatische loop is het iets ingewikkelder. Er wordt een onderscheid gemaakt tussen een positieve hoek van het paneel en een negatieve hoek. Afhankelijk van de positie van de zon en van de hoek die het paneel maakt, zal de richting die de motoren moeten draaien verschillen.
Om de gebrekkige nauwkeurigheid van de LDR's op te vangen, delen we het meetresultaat hiervan door 10 en ronden we het af.

$$LB = round(sensorLB/10);$$

$$RB = round(sensorRB/10);$$

$$LO = round(sensorLO/10);$$

$$RO = round(sensorRO/10);$$

Opdat de motoren zouden weten in welke richting ze moeten draaien, nemen we het gemiddelde van de twee linkse, rechtse, onderste en bovenste LDR's. De beweging gebeurt dus aan de hand van de gemiddelden en niet de rechtstreekse waarde van 1 LDR, zodoende de nauwkeurigheid te verhogen.

$$LinksAVG = (LB + LO)/2;$$

$$RechtsAVG = (RB + RO)/2;$$

$$OnderAVG = (RO + LO)/2;$$

$$BovenAVG = (RB + LB)/2;$$

Helemaal in het begin van de loop zal het homen gebeuren. De stepper die het paneel bestuurt, zal blijven draaien tot het de limitswitch aanklikt. Hierna zal die blijven bewegen tot zijn horizontale positie. Dit komt overeen met 624 stappen van de motor:

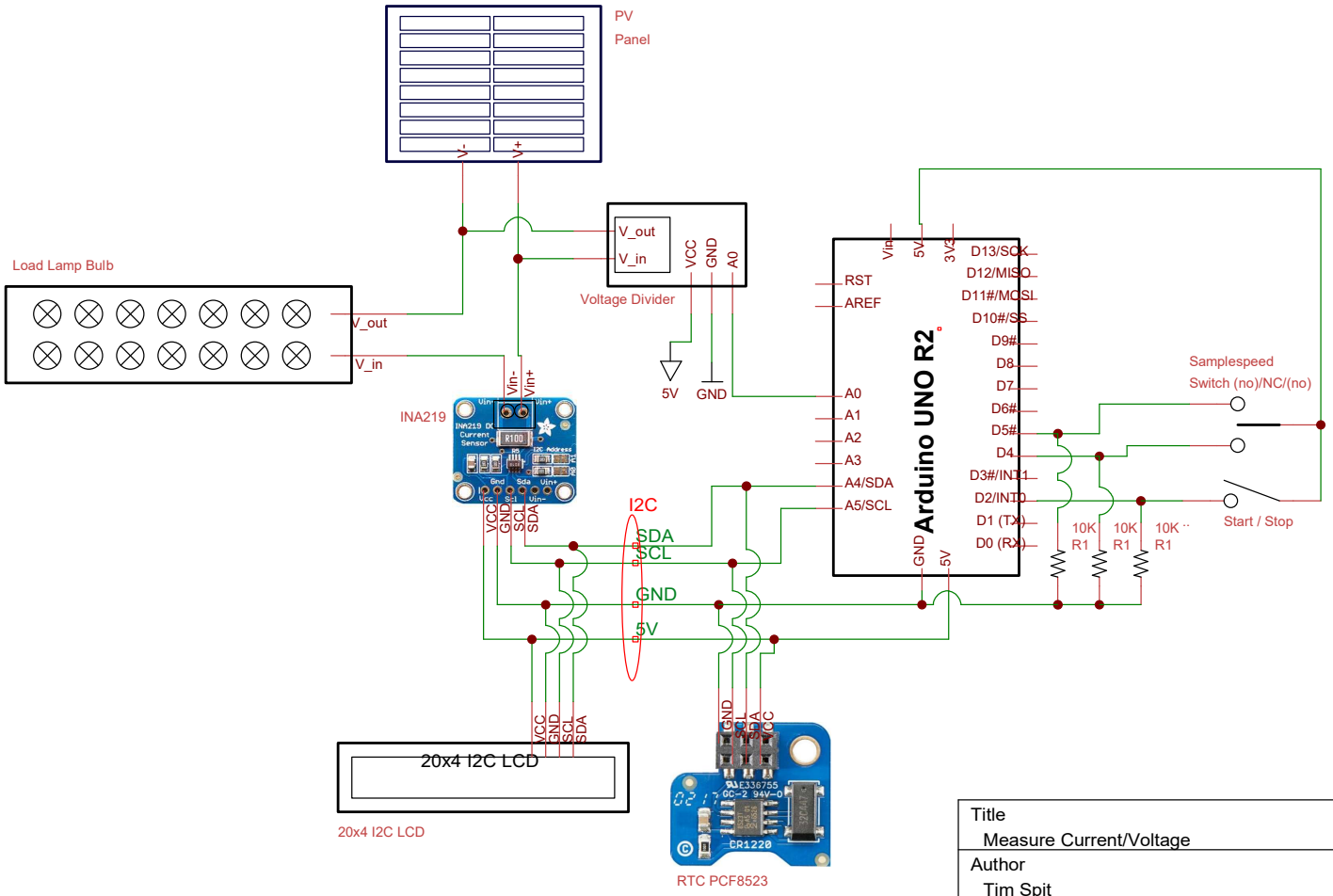
Code Arduino 6.2

```
while (x == 0){
    myStepper2.step(1);
    if (klikker == HIGH){
        x = 1;
        stepCount2 = 0;
    }
}
while (x == 1){
    myStepper2.step(-1);
    Serial.print("steps2:");
    Serial.println(stepCount2);
    stepCount2++;
    if ((622 <= stepCount2)&&(stepCount2 <= 625)){
        x = 2;
        Serial.print("steps2:");
        Serial.println(stepCount2);
    }
}
```

7 Maximum Power Point Tracking

De meting en verwerking van de stroom en spanning gebeurt volledig binnenin het bedieningspaneel.

7.1 Draadschema



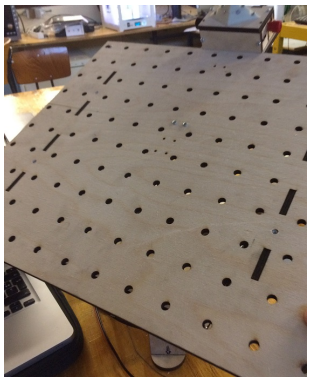
Title Measure Current/Voltage		
Author Tim Spit		
File ıg\Elektrische schema's\SCHEMA_PLOTTING2		Document
Revision 1.0	Date	Sheets 1 of 1

7.2 Zonnecellen - zonnepaneel

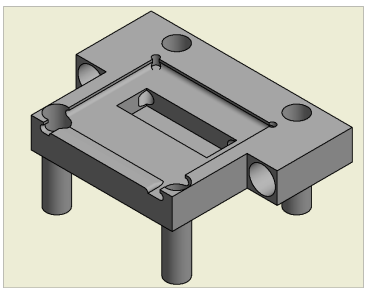
Bij de aanvang van de bachelorproef, werd de keuze gemaakt om met kleine, afzonderlijke zonnecellen te werken. Deze zijn klein (45 mm x 60 mm), zijn licht maar wekken slechts beperkte stroom en spanning op (1 V / 200 mA). Onze keuze is gevallen op een reeks van 10 kleine en 1 grotere (125 mm x 65 mm) monokristallijne Photo-Voltaïsche zonnecelen. Ten opzichte van de polykristallijne cellen, met zichtbare kristalstructuur, hebben de monokristallijne cellen een iets hoger rendement.

Figuur 56: monokristallijne PV-cellen

Het voordeel van verschillende kleine cellen, is dat je als gebruiker vrij kan experimenteren met de hoeveelheid cellen en of je ze onderling in serie of parallel schakelt om zo de impact op de productie van stroom en spanning waar te nemen.
Om de gebruiker deze vrijheid te gunnen, moet het gebruik van de cellen zo flexibel mogelijk zijn. Daarom worden voor de zonnecellen 'celhouders' ontworpen. De zonnecellen worden in deze celhouders bevestigd. Deze celhouders kunnen gemakkelijk op de bevestigingsplaat van de solartracker worden gemonteerd via de serie van gaatjes. De pootjes van de celhouders passen hier perfect in. Na de metingen, kunnen de cellen er weer makkelijk afgehaald worden. Alle cellen kunnen afzonderlijk aangesloten worden op de meetinstallatie, met behulp van banaanstekkers. Hierdoor is het tevens mogelijk de cellen in verschillende configuraties te verbinden.

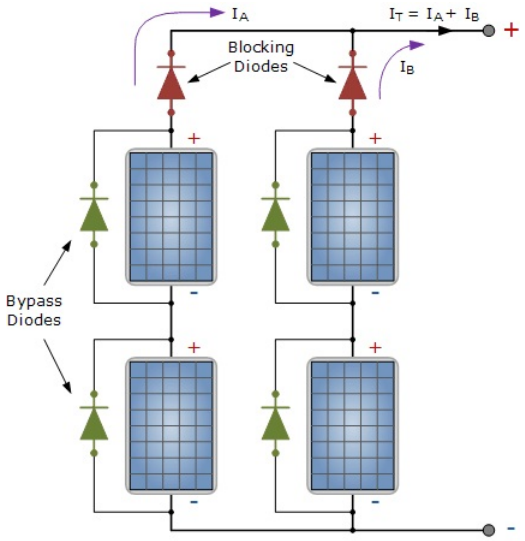


Figuur 57: Bevestigingsplaat



Figuur 58: Celhouder

Een belangrijk aspect dat bij het maken van onze eigen zonnecellen is de aanwezigheid van diodes. We moeten hierbij het onderscheid maken tussen enerzijds 'bypass-diodes' en anderzijds de 'blocking-diodes'. Wanneer cellen in serie worden geplaatst, gebeurt het dat er schaduw valt over 1 of meerdere cellen. Deze cellen zouden geforceerd worden om een stroom te leveren groter dan de kortsluitstroom. Dit kan enkel wanneer deze cellen opereren in een gebied van negatieve spanning. Om te vermijden dat deze cellen een negatieve invloed hebben op het opgewekte vermogen, zijn bypass-diodes nodig. Blocking-diodes bewijzen hun dienst wanneer een zonnepaneel een batterij oplaadt. Indien de spanning van de batterij hoger is dan de opgewekte spanning van het paneel, zou het paneel als verbruiker optreden en zou de batterij ontladen. Om dit te vermijden zijn blocking-diodes nodig. [5] [6]



Figuur 59: Diodes bij zonnecellen

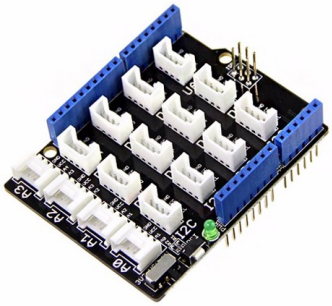
Jammer genoeg moesten we vaststellen dat de metingen met deze kleine zonnecellen een onvoldoende nauwkeurig resultaat opleverden. Dit is het gevolg van de lage opgewekte stroom en spanning en de verliezen die optreden bij het vervoer van de stroom. Daarom zijn we overgeschakeld op het gebruik van een klein 20 W zonnepaneel.
Specificaties:

Categorie	Polykristallijn zonnepaneel
Vermogen	20 Wp
Nominale spanning	12 V
Nullastspanning	22 V
Nominale stroom	1.18 A
Kortsluitstroom	1.32 A
Breedte	386 mm
Hoogte	40 mm
Lengte	461 mm
Gewicht	2.1 kg

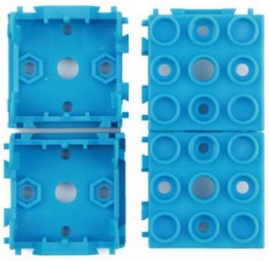
Figuur 60: 20 W zonnepaneel

7.3 Arduino met Grove Base Shield

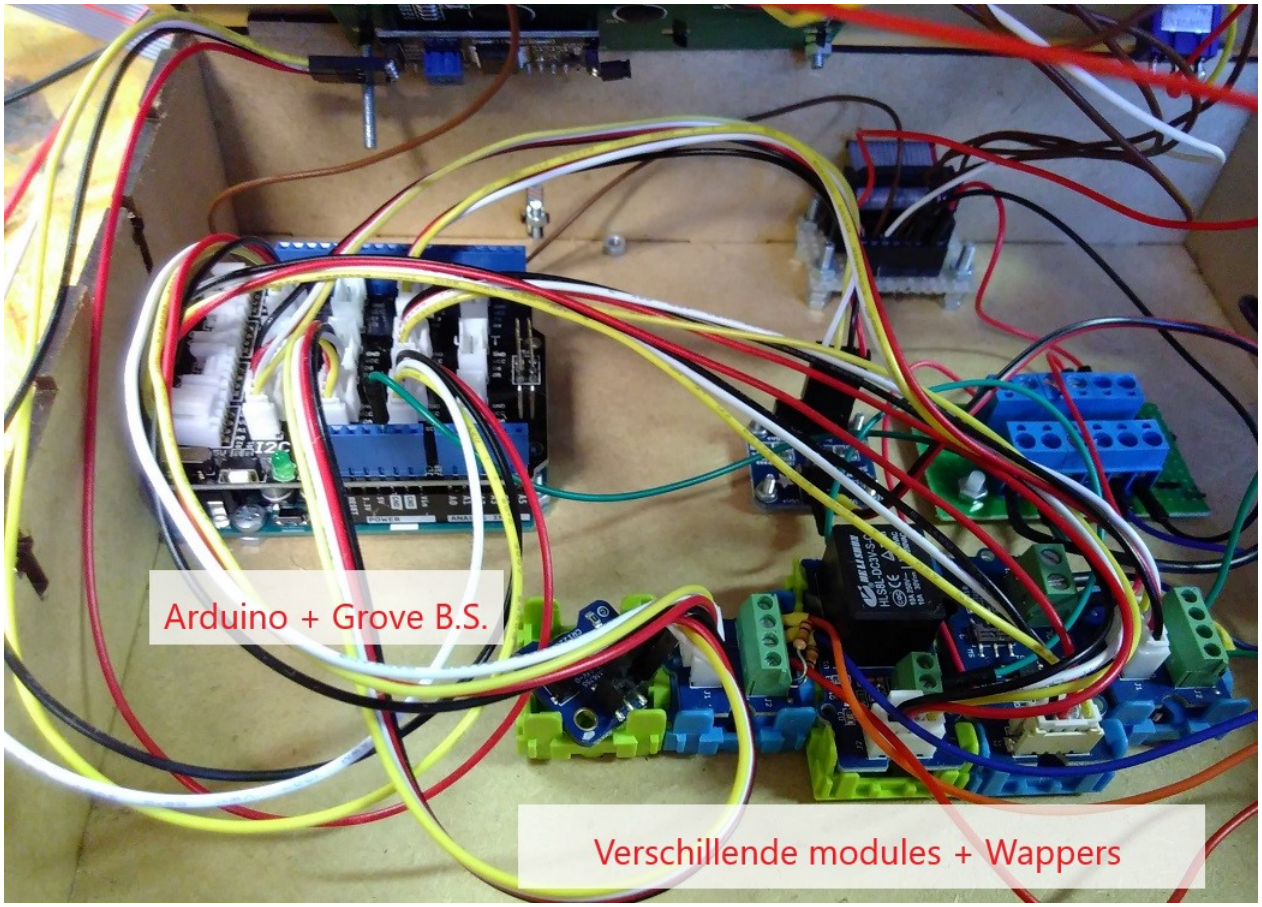
Om de Arduino op een duurzame doch flexibele manier te verbinden met de stroom-sensor, spanningsdeler, relais en de nodige schakelaars, wordt er gebruik gemaakt van een Grove Base Shield. Dit 'schild' wordt bovenop de Arduino geschoven. Verschillende modules zijn via aangepaste kabels stevig verbonden met het schild waardoor het uitgesloten is dat kabels door schokken of stoten loskomen. De verschillende modules kunnen op hun beurt via op maat gemaakte 'wrappers' met bouten en moeren vast gemonteerd worden in het bedieningspaneel. Deze methode is als het ware een alternatief voor het maken van PCB's. Vooral de aanwezigheid van 4 toegangen tot I2C-bus, is een meerwaarde van dit schild.



Figuur 61: Grove Base Shield V2 (Seeed)



Figuur 62: Module Wrapper



Figuur 63: Grove Base Shield en wrappers gemonteerd in het bedieningspaneel

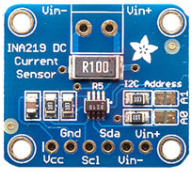
7.4 Meting stroom & spanning met INA219

Een cruciaal component in de meetinstallatie is de stroom-sensor. Er werd gekozen voor een INA219 DC current-sensor (Adafruit). [7] De sensor van het type 'Current-Shunt Monitor'. Current shunts 'berekenen' de stroom door de spanningsval over een weerstand die in het pad van de stroom is geplaatst. Via de I^2C-bus kan de sensor communiceren met Arduino en wordt

afzonderlijk gevoed met 3.3 V of 5 V. Er wordt gelezen/geschreven aan een frequentie van 100 kHz. De INA219 meet niet enkel de gelijkstroom maar ook de spanning over de aansluitpoorten. Met beide grootheden beschikken we over alle informatie om met vermogen van het zonnepaneel te berekenen.

$$P = I \cdot U$$

De uit te lezen stroom/spanning moet worden aangesloten op de V_{in+} en V_{in-} poort. Deze analoge ingangen zijn verbonden met een shunt-weerstand. Let op: er kan maximaal 26 V en 3 A worden uitgelezen. [8]



Figuur 64: INA219 DC current sensor

Om toevallige pieken in hoge of lage waarden te filteren, wordt eerst het gemiddelde berekend van **50 metingen**, alvorens deze waarde te plotten. In de Arduino-code werd dit geprogrammeerd met behulp van de `millis()`-library. Werken met een for-loop bleek geen optie omdat het mogelijk moet zijn sommige samples pas na 5 minuten te berekenen. Met een for-loop zou dit betekenen dat de controller gedurende 5 minuten niet bediend kan worden. Door middel van `millis()` doet dit probleem zich niet voor. Een nieuwe meting van de INA219 wordt op een volgende plaats in een array geschreven als de tijd (in milliseconden) tussen 2 metingen een bepaalde waarde overschrijdt. *bvb. Je wenst een samplespeed van 2 seconden. Dan moet er om de 40 milliseconden een nieuwe meting gedaan worden.* Het is echter onmogelijk om de tijd tussen 2 metingen exact vast te leggen. Daarom zal bij het gebruik van de Solartracker opvallen dat de intervallen niet helemaal overeen komen met de waarde op het LCD-display.

Pseudo-code:

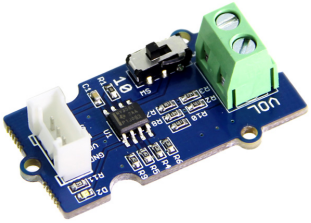
Code Arduino 7.1

```
if ((millis() - CurrentMillis >= SSpeed) ) {
    float ArrayU[Sampels];
    float ArrayI[Sampels];
    float current_mA = 0;
    current_mA = ina219.getCurrent_mA();
    ArrayI[i] = current_mA;
    i = i + 1;
    CurrentMillis = millis();
```

```
}
```

7.5 Beveiliging INA219 tegen overspanning

Omdat de currentsensor maximaal 26V aankan, moeten we een schakeling voorzien die het circuit onderbreekt indien deze waarde wordt overschreden. Het principe dat hierbij gehanteerd werd, komt volledig overeen met 6.3 met als verschil dat in dit geval de spanning onderbroken wordt als deze te **hoog** wordt, i.p.v. te laag.[4] Bijkomstig hierbij is dat op het LCD-display de boodschap zal verschijnen dat de spanning te hoog is en het veiliger is de meting tijdelijk te stoppen.



Figuur 65: Grove Voltage Divider

Figuur 66: Grove Relay-module

7.6 Belasting

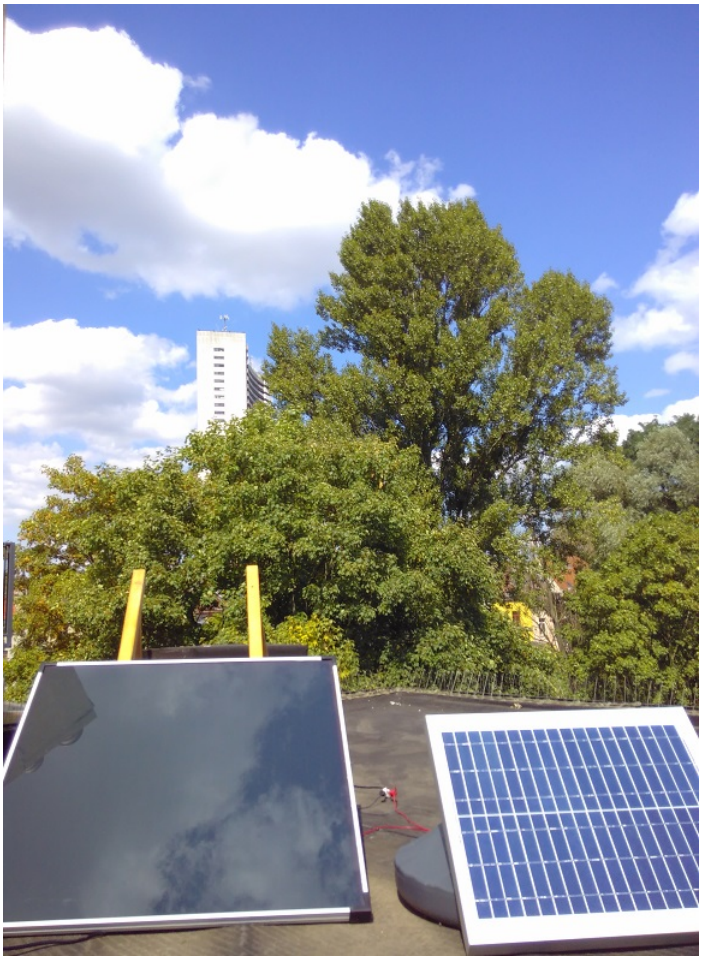
Er zal pas stroom doorheen de schakeling vloeien, wanneer verbruikers aangesloten zijn. In het geval van deze solartracker zijn deze verbruikers een reeks van 2 kleine lampjes (elk 14 V 200 mA) parallel met elkaar geschakeld. Concreet betekent dit dat over elke rij van 2 lampjes maximum 28 V spanning kan staan. Dit is meer dan de nullastspanning van het zonnepaneel, dus aanvaardbaar. De stroom die vloeit doorheen een koppel lampjes bedraagt ongeveer 100 mA. Elk koppel van lampjes is voorzien van een aan-uitschakelaar. Je kan de stroom doen toenemen door meer lampjes te laten branden. Merk op dat de lichtintensiteit vermindert naar gelang de hoeveelheid aangesloten lampjes. De invloed van het aantal aangesloten verbruikers op de stroom, spanning en vermogen wordt in real-time geplot in $I(U)$-en $P(U)$-grafiek op het computerscherm met behulp van het programma 'KST'. De laatste 2 rijen verbruikers bestaan uit 3 lampjes van 6V / 100 mA. Reden hiervoor is dat er de stroom en spanning iets fijner kan worden afgestemd. Dit kan handig zijn om het MPP zo goed mogelijk te benaderen. De spanning en stroom door de lampjes kan worden afgelezen op het LCD-display van de controller.

[4] De spanning over de polen van het zonnepaneel wordt gemeten met een spanningsdeler. Via een relais zal het circuit onderbroken worden wanneer spanning te hoog wordt (vanaf 25.5 V). De relais zal pas sluiten als de spanning minder dan 24 V bedraagt. Er moet namelijk een soort 'dode zone' zijn om een constant aan-uit-geschakel te vermijden rond de grenswaarde van 25.5V.

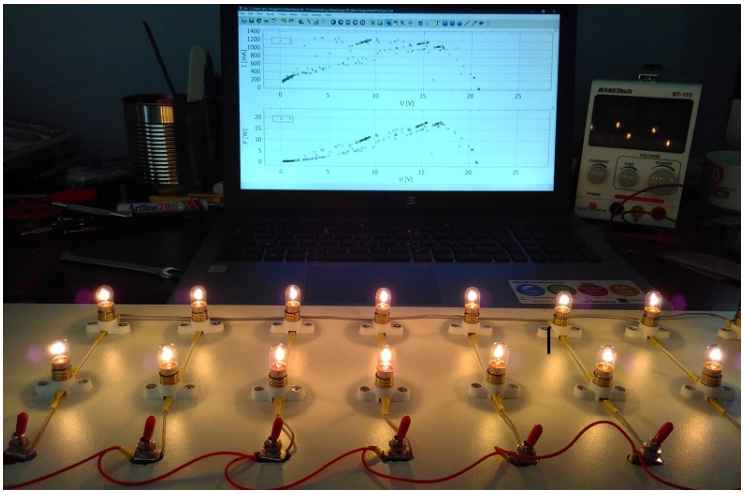
Figuur 67: Belasting tijdens meting

7.7 Enkele resultaten

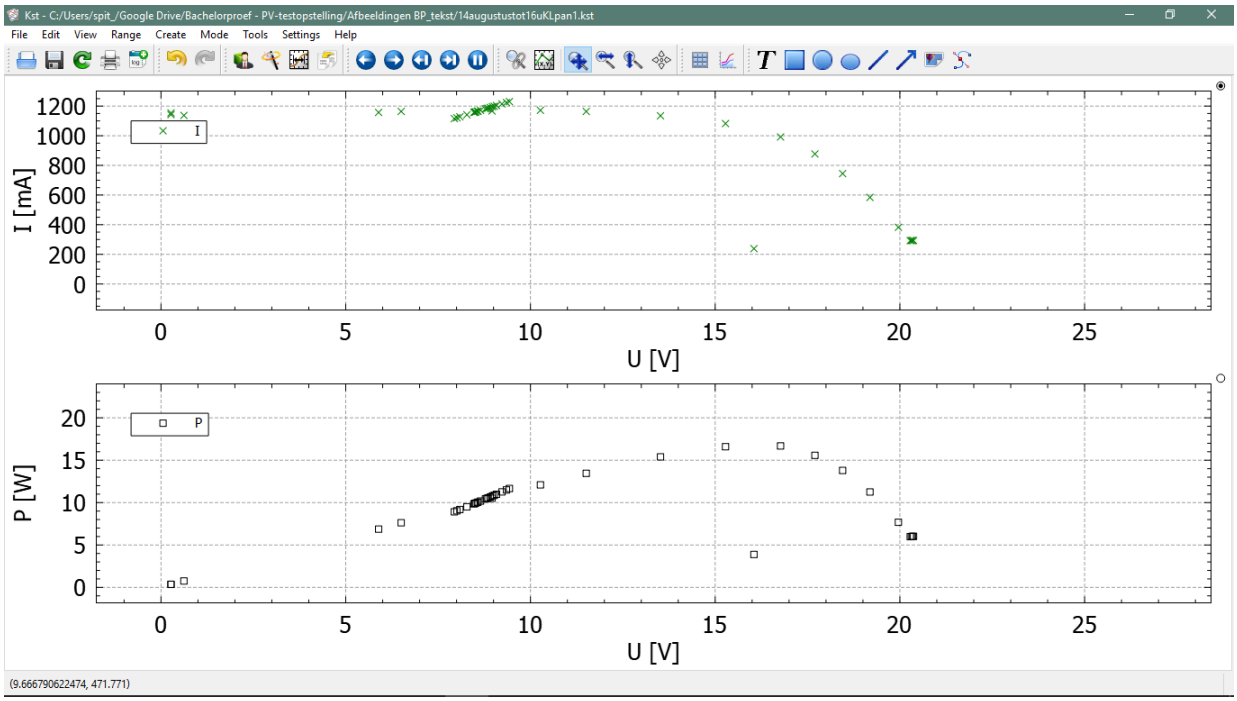
Op 14 augustus 2017 werden enkele tests uitgevoerd om te controleren of de schakeling met de lampjes degelijke resultaten opleverden. Hieronder enkele foto's en schermafdrukken van de metingen. Het viel op dat wolken onmiddellijk een grote impact hebben op de opgewekte spanning/stroom. Wanneer de $I(U)$ grafiek van rechts naar links gelezen wordt, 'zakt' de grafiek als het ware naar de oorsprong. Dit illustreert duidelijk dat het zonnepaneel slechts optimaal werkt bij heldere hemel.



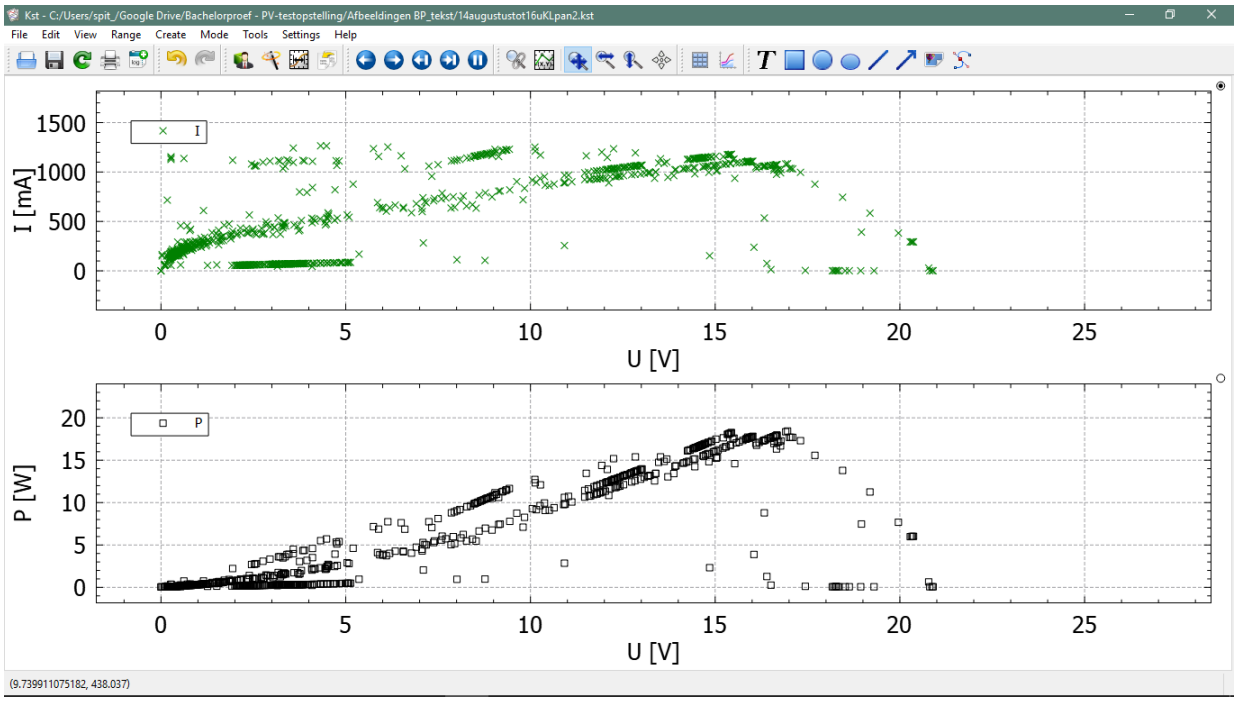
Figuur 68: 14 augustus 2017 - licht bewolkt



Figuur 69: Meting bij heldere hemel (14-08-2017)



Figuur 70: Korte meting bij heldere hemel en variërende belasting (14-08-2017)



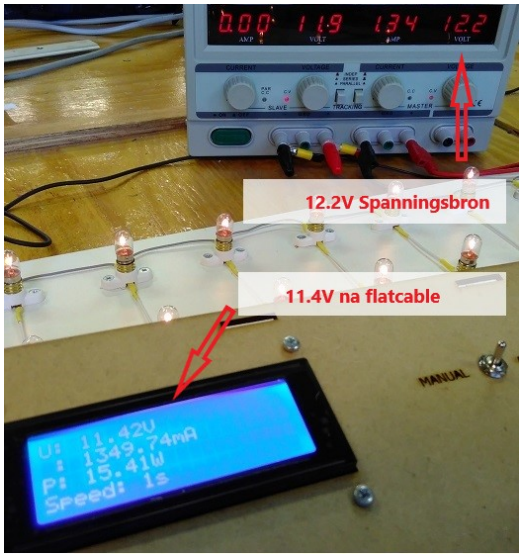
Figuur 71: Meting gedurende 2 uur met afwisseling tussen wolken en heldere hemel en bij variërende belasting (14-08-2017)

Deel III

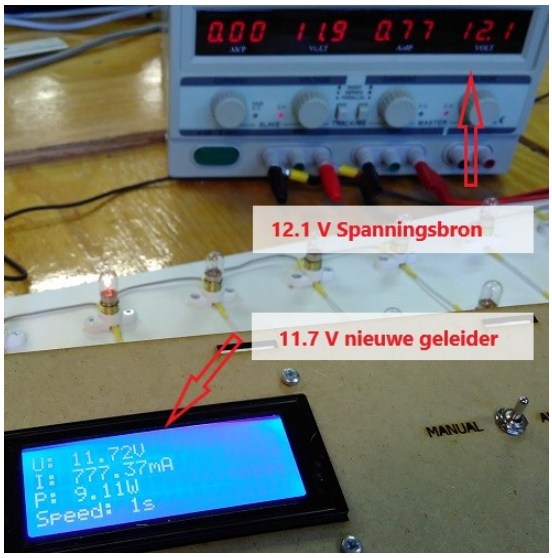
Nabeschouwing

Tijdens de praktische uitvoering werden we geconfronteerd met moeilijkheden of problemen. Er komt ook nog bij dat dit prototype van een solartracker op verschillende vlakken kan geoptimaliseerd worden. Wat hier volgt is een overzicht van mogelijke verbeteringen voor de toekomst.

Het gebruik van lange **kabels** tussen de solartracker en het bedieningspaneel heeft alleen nadelige gevolgen. De spanningsval die de meetresultaten negatief beïnvloedt, is de belangrijkste. Maar daarnaast ontstaan er praktische moeilijkheden. De kabels van het besturingspaneel belemmeren de beweging van de ronddraaiende schijf. Onze eenvoudige oplossing is om deze kabels vast te monteren op het kastje van de tracker en met behulp van een kabelrups de kabels te begeleiden zodat deze de draaibeweging van de tracker niet hinderen. Het zou echter ook mogelijk zijn om de signalen van het besturingspaneel (knoppen voor de manuele beweging) draadloos te sturen via Xbee.
De enige kabels die niet weg te werken zijn, zijn die van de batterij naar de motoren en deze van het zonnepaneel naar de stroom- en spanningsmeeteenheid. Toch zou door een meer efficiënt ontwerp van de solartracker de bekabeling nog geoptimaliseerd kunnen worden.
De overweging die gemaakt moet worden om de spanningsval zoveel mogelijk te reduceren, is de plaats waar de verbruikers (belasting op het paneel) moet komen te staan. Wij hebben ervoor gekozen deze bij de computer te installeren, zodat je bij het wijzigen van de belasting onmiddellijk op het computerscherm de gevolgen op de $I(U)$ en $P(U)$-grafiek kan waarnemen. Een andere mogelijkheid is om de belasting vlak bij de solartracker te plaatsen zodoende de lengte van de kabels te beperken. Het nadeel is dan wanneer de computer niet naast de tracker staat, je pas later, de invloed van de belasting op de stroom, spanning en vermogen ziet.
Om het aantal kabels te reduceren, hadden we aanvankelijk gekozen om gebruik te maken van de flatcable bij de meting van I en U. Maar omdat deze een zeer lange en dunne geleider is, werden we geconfronteerd met een significante spanningsval. De meetresultaten waren dus niet representatief. Dit probleem is gedeeltelijk verholpen door een geleider te gebruiken met een dikkere doorsnede. De nieuwe kabel heeft een doorsnede van 2x 0.91 mm^2 (zie datasheet in bijlage).
Meting ter illustratie: Via de flatcable ontstaat een spanningsval van 0.8 V ten opzichte van een spanningsval van 0.4 V bij de nieuwe geleider. Dit is een ‘verbetering’ van 50%. Door de lengte van de kabel, dus de belasting dichter bij het zonnepaneel te installeren, zou de spanningsval nog verder kunnen afnemen.



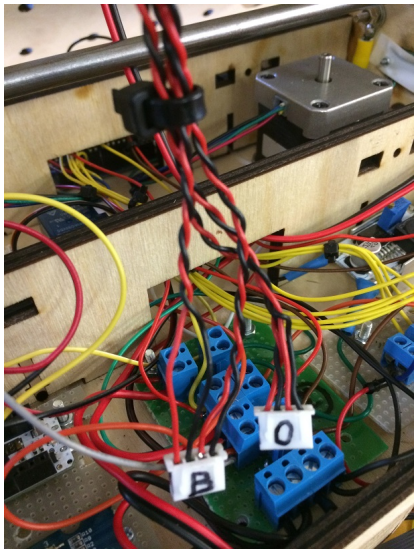
Figuur 72: Flatcable



Figuur 73: Nieuwe geleider

Nauwkeurigheid van de meting met betrekking tot de stroom: Uit bovenstaande foto's valt op dat de lengte van de kabel geen invloed heeft op de stroom. De gemeten stroom via de INA219 en geleverde stroom zijn identiek.

Een tweede punt van verbetering is het ontwerpen van degelijke **PCB's**. Achter de houten behuizing van de tracker en het bedieningspaneel zit een wirwar van draadjes verscholen. De meeste kabeltjes zitten stevig vast met screw-blocks maar toch is het mogelijk dat er door schokken of stoten kabeltjes los komen. Een definitief en bruikbaar toestel vereist dat losse kabeltjes tot een minimum worden gereduceerd.

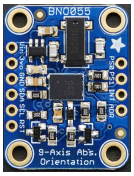


Figuur 74: Nood aan PCB's

Ten derde zou er verder gewerkt kunnen worden aan een meer **gebruiksvriendelijke** manier om de de meetresultaten te plotten. Er wordt nu gebruik gemaakt van Arduino, Processing en KST. Iemand met de nodige vaardigheden zou er volgens ons in slagen om een programma te schrijven dat de bovenstaande programma 's combineert.

Voor het vinden van het Maximum Power Point, wordt nu gebruik gemaakt van een reeks lampjes. Het nadeel van dit systeem is dat er 'sprongsgewijs' gezocht wordt naar het MPP. Een elegantere oplossing is een **schuifweerstand** of potentiometer omdat deze een veel fijnere variatie in de belasting kent. Op die manier kan veel nauwkeuriger het Maximum Power Point gevonden worden. Belangrijk hierbij is dat deze regelbare weerstand het vermogen van het zonnepaneel moet aankunnen. Eventueel moet de potentiometer niet rechtstreeks het totale vermogen dragen, maar stuurt deze een mosfet + Arduino aan.

Tenslotte hebben we ook gezocht naar een manier om de **oriëntatie** (windrichting) van het zonnepaneel te loggen, samen met de meetresultaten. Hiervoor gebruikten we een BNO055 absolute orientating sensor. Het probleem waarmee we geconfronteerd werden, was het magnetisch veld van de motoren die de kompas-module in de war bracht. Momenteel is deze sensor losgekoppeld van de solartracker. Toch moet het mogelijk zijn het probleem van de magnetische interferentie te omzeilen, door het elektrisch veld van de motoren op de juiste manier te isoleren.



Figuur 75: BNO055

Bibliografie

[1] B. Netterfield and R. Chern, "kst - plots scientific data." https://kst-plot.kde.org/. Accessed: 2017-02-19.

[2] S. Andreev, "Car battery voltage." http://carbatteryworld.com/car-battery-voltage/, 01/14/2013. Accessed: 2017-04-05.

[3] AA1Car, "Diagnosing a car battery that runs down." http://www.aa1car.com/library/battery_runs_down.htm, August 9, 2017. Accessed: 2017-04-06.

[4] Velleman, "Velleman vma404 - dc-dc adjustable voltage step down module lm2596s." https://www.velleman.eu/downloads/29/vma404_a4v01.pdf, 2017. Accessed: 2017-07-11.

[5] D. Lindsell, "What is a bypass diode?." http://www.solar-facts.com/panels/panel-diodes.php, 2008. Accessed: 2017-04-10.

[6] L. Bas, "Bocking and by-pass diodes in solar panels." https://www.civicsolar.com/support/installer/questions/what-bypass-diode, 2017. Accessed: 2017-04-10.

[7] TI, "Current sensing - current shunt monitor." http://www.ti.com/paramsearch/docs/parametricsearch.tsp?family=analog&familyId=426&uiTemplateId=NODE_STRY_PGE_T&DCMP=analog_signalchain_mr&HQS=currentshuntport-pr, 2017. Accessed: 2017-03-22.

[8] Adafruit, "Data sheet for the ina219 chip." http://www.adafruit.com/datasheets/ina219.pdf, 2012. Accessed: 2017-03-18.

Lijst van figuren

Appendices

Arduinocode Solartracker

```
#include <Stepper.h>
const int stepsPerRevolution1 =1000;
const int stepsPerRevolution2 = 1000;
Stepper myStepper2(stepsPerRevolution2, 6, 7, 8, 9);
Stepper myStepper1(stepsPerRevolution1, 10, 11, 12, 13);//Schijf
int stepCount1 = 0;
int stepCount2 = 0;



int button1 = 0;
int button2 = 0;
int button3 = 0;
int button4 = 0;

int klikker = 0;
int x = 0;


int sensorLB = 0;
int LB = 0;
int sensorRB = 0;
int RB = 0;
int sensorLO = 0;
int LO = 0;
int sensorRO = 0;
int RO = 0;

int LinksAVG = 0;
int RechtsAVG = 0;
int OnderAVG = 0;
int BovenAVG = 0;


int switch1 = 0;


void setup()
{
  pinMode(2,INPUT);//button1
  pinMode(3,INPUT);//button2
  pinMode(4,INPUT);//button3
  pinMode(5,INPUT);//button4
  pinMode(A4,INPUT);
  pinMode(A5,INPUT);

  Serial.begin(9600);

  myStepper1.setSpeed(3);
  myStepper2.setSpeed(3);

}

void loop() {

  button1 = digitalRead(2);
  button2 = digitalRead(3);
  button3 = digitalRead(4);
  button4 = digitalRead(5);

  klikker = digitalRead(A5);


  sensorLB = analogRead(A0);
```

```
sensorRB = analogRead(A1);
sensorLO = analogRead(A2);
sensorRO = analogRead(A3);

LB = round(sensorLB/10);
RB = round(sensorRB/10);
LO = round(sensorLO/10);
RO = round(sensorRO/10);

LinksAVG = (LB + LO)/2;
RechtsAVG = (RB + RO)/2;
OnderAVG = (RO + LO)/2;
BovenAVG = (RB + LB)/2;

switch1 = digitalRead(A4);



if (x == 0){
   myStepper2.step(1);
   if (klikker == HIGH){
      x = 1;
      stepCount2 = 0;
   }
}
if (x == 1){
  myStepper2.step(-1);
  Serial.print("steps2:");
  Serial.println(stepCount2);
  stepCount2++;
  if ((622 <= stepCount2)&&(stepCount2 <= 625)){
    x = 2;
    Serial.print("steps2:");
    Serial.println(stepCount2);
  }
}

// MANUEEL

if (switch1 == HIGH){
  //beweging schijf
  if (button3 == HIGH){ //tegenwijzerszin
    myStepper1.step(1);
    //Serial.print("steps1:");
    //Serial.println(stepCount1);
    stepCount1++;
  }
  else if (button4 == HIGH){ //wijzerszin
    myStepper1.step(-1);
    //Serial.print("steps1:");
    //Serial.println(stepCount1);
    stepCount1--;
  }

  //beweging cel

  if ((button1 == HIGH)&&(stepCount2>=1)){ //wijzerszin
    myStepper2.step(1);
    //Serial.print("steps2:");
    //Serial.println(stepCount2);
    stepCount2--;
  }
  else if ((button2 == HIGH)&&(stepCount2<=1200)){ //tegenwijzerszin
    myStepper2.step(-1);
    //Serial.print("steps2:");
    //Serial.println(stepCount2);
```

```
      stepCount2++;
    }
  }

  //AUTOMATISCH

  else{

    //Beweging schijf
    if (LinksAVG - RechtsAVG > 1 ) { //moet tegenwijzerszin
      if (stepCount2 <= 623){
        myStepper1.step(-1);
        //Serial.print("steps1:");
        //Serial.println(stepCount1);
        stepCount1--;
      }
      else{
        myStepper1.step(1);
        //Serial.print("steps1:");
        //Serial.println(stepCount1);
        stepCount1++;
      }
    }
    else if(RechtsAVG - LinksAVG > 1){ //moet wijzerszin
      if (stepCount2 <= 623){
        myStepper1.step(1);
        //Serial.print("steps1:");
        //Serial.println(stepCount1);
        stepCount1--;
      }
      else{
        myStepper1.step(-1);
        //Serial.print("steps1:");
        //Serial.println(stepCount1);
        stepCount1++;
      }
    }

    //Beweging cel

    if ((OnderAVG + 1 < BovenAVG)&&(stepCount2<=1200)){ //moet wijzerszin
       myStepper2.step(-1);
       //Serial.print("steps2:");
       //Serial.println(stepCount2);
       stepCount2++;
    }
    else if ((BovenAVG + 1 < OnderAVG)&&(stepCount2>=1)){ //moet
tegenwijzerszin
       myStepper2.step(1);
       //Serial.print("steps2:");
       //Serial.println(stepCount2);
       stepCount2--;
    }

  }
  /*Serial.println(LB);
  Serial.println(RB);
  Serial.println(LO);
  Serial.println(RO);
  Serial.println(" ");*/
  //printFeedback();

}
```

Arduino-code voor relais die de 12V batterij uitschakelt bij te lage spanning

```
int pinRelais = 6;                  // Stuurt signaal of relais moet sluiten

void setup()
{
    Serial.begin(9600);
    pinMode(pinRelais,OUTPUT);
    digitalWrite(pinRelais, LOW);   // Relais in NC-stand, dus geen signaal nodig om
                                    // circuit te sluiten
}

void loop()
{
    long  sensorValue = analogRead(A1);         // Voltagedivider GROVE
    long  sum = 0;
    float Vbat = 0;
    for(int i=0;i<1000;i++)                     // Met deze for-loop wordt er een
                                                // gemiddelde van 1000 samples gemaakt
    {
        sum=sensorValue+sum;
        sensorValue=analogRead(A1);
        delay(2);
    }
    sum = sum/1000;

    Serial.print("if you set the Gain to 3,the input voltage: ");
    Serial.println(3*sum*4.980/1023.00);        // original code: 10*sum*4980/1023.00
                                                // (Gain = 10)
    Vbat = 3*sum*4.980/1023.00;

    Serial.print("Vbat = ");Serial.println(Vbat);

if (Vbat < 11.40){                              //Relais in NC-toestand
     digitalWrite(pinRelais, HIGH);             // High-signaal onderbreekt de arduino.
}
else if (Vbat > 12.60){                         // Low-signaal: relais zou pas opnieuw
                                                // sluiten vanaf 12.6V (= volle batterij)
     digitalWrite(pinRelais, LOW);
}
    delay(1000);
}
```

Arduino-code Meting Stroom/Spanning

```
// I2C:
// ----
#include <Wire.h>

// Date and time functions using a PCF8523 RTC:
// --------------------------------------------
#include "RTClib.h"
RTC_PCF8523 rtc;
char daysOfTheWeek[7][12] = {"Sunday", "Monday", "Tuesday", "Wednesday", "Thursday", "Friday",
"Saturday"};

// BNO055 as Compass:
// ------------------
#include <Adafruit_Sensor.h>
#include <Adafruit_BNO055.h>
#include <utility/imumaths.h>
#include <math.h>
#define BNO055_SAMPLERATE_DELAY_MS (100)
Adafruit_BNO055 bno = Adafruit_BNO055();

// I2C INA219 current sensor:
// --------------------------
#include <Adafruit_INA219.h>
Adafruit_INA219 ina219;
#if defined(ARDUINO_ARCH_SAMD)
#define Serial SerialUSB
#endif

// I2C LCD:
// -------
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x3F, 2, 1, 0, 4, 5, 6, 7, 3, POSITIVE);   // Ons Adres is op 0x3F ipv 0x27

// Parameters:
// -----------
const int buttonPin = 2;                // pin D2 Start/Stop meting
//const int buttonPin2 = 3;      // pin D3 Fast/Slow meting
int pinRelais = 6;                      // Stuurt signaal of relais moet sluiten naar D6
long Sampels = 50.00;                         // Aantal sampels we nemen alvorens het gemiddelde
(Stroom, Spanning) te berekenen
int buttonState = 0;             // variable for reading the pushbutton status
int Start = 0;
int SamplesVD = 50;        // Aantal sampels om de spanning voor de Voltagedivider te bereken
int info = 0;
int i = 0;                 // Wordt gebruikt bij de berekening van de gemiddelde waarde
int sp = 2;
const int buttonPin4 = 4;               // Snelheid meting TRAGER
const int buttonPin5 = 5;               // Snelheid meting SNELLER
int buttonSLOW = 0;                     // current state of the button
int buttonFAST = 0;                     // current state of the button
long SSpeed = 0;
String Vsample = "1s   ";
int lastButtonStateS = 0;               // previous state of the SLOW-button
int lastButtonStateF = 0;               // previous state of the FAST-button
unsigned long previousMillis = 0;
unsigned long CurrentMillis = 0;
//unsigned long CurMilDif = 0;
unsigned long lastDebounceTime = 0;     // the last time the output pin was toggled
unsigned long debounceDelay = 50;       // the debounce time; increase if the output flickers
float totalU = 0;
float totalI = 0;




/*-------------------------VOID SETUP -------------------------
  --------------------< Wordt 1x doorlopen >-------------------*/
void setup() {

  Serial.begin(9600);
  // Start-schakelaar:
  // -----------------
```

```
  pinMode(buttonPin, INPUT);

  // Overspanning:
  // -------------
  pinMode(pinRelais, OUTPUT); // Pin van de relais die onderbreekt wanneer er te veel V wordt
gegenegeerd

  // Speed-schakelaar:
  // -----------------
  pinMode(buttonPin4, INPUT);
  pinMode(buttonPin5, INPUT);
  SSpeed = 20;

  // PCF8523 RTC:
  // ------------
  if (! rtc.begin()) {
    Serial.println("Couldn't find RTC");
    while (1);
  }
  if (! rtc.initialized()) {
    Serial.println("RTC is NOT running!");
  }

  // BNO055 as Compass:
  // ------------------
  //if (!bno.begin())
  {
    /* There was a problem detecting the BNO055 ... check your connections */
    //Serial.print("Ooops, no BNO055 detected ... Check your wiring or I2C ADDR!");
    while (1);
  }
  bno.setExtCrystalUse(true);

  // I2C INA219 current sensor:
  // --------------------------
#ifndef ESP8266
  while (!Serial);     // will pause Zero, Leonardo, etc until serial console opens
#endif
  uint32_t currentFrequency;
  ina219.begin();


  // I2C LCD:
  // --------
  lcd.clear();
  lcd.begin(20, 4);     // LCD van 4x20 karakters
  lcd.setCursor(0, 1);  //3e plaats, 2e rij
  lcd.print("  Photovoltaic Cell");
  lcd.setCursor(0, 2);  //6e plaats, 3e rij
  lcd.print("      Test-Setup");
  delay(3000);
  lcd.clear();

  // TXT-file:
  // ---------
  Serial.println("Photovoltaic Cell Test-setup");
  Serial.println("Measuring voltage and current with INA219 ...");
  Serial.print("Date & time;"); Serial.print("\t"); Serial.print("U;"); Serial.print("I;");
Serial.print("P;"); Serial.print("\t"); Serial.println("Cardinal direction"); // Grootheden
  Serial.print("Day & hour;"); Serial.print("\t"); Serial.print("V;"); Serial.print("mA;");
Serial.print("W;"); Serial.print("\t"); Serial.println(";");                 // Eenheden
} // end setup




/*-------------------------VOID LOOP -------------------------
  -----------------< Wordt continu herhaald >---------------*/
void loop() {


  // Parameters:
  // -----------
  long Vdiv = 0;  // Beveiliging bij te veel spanning
```

```
  // Waarde spanningsdeler (veiligheid):
  // -----------------------------------
  long  sensorValue = analogRead(A0); // Voltagedivi op GROVE-shield
  long  sum = 0;
  for (int i = 0; i < SamplesVD; i++) // Met deze for-loop wordt er een gemiddelde van 100 samples
gemaakt (origineel 1000)
  {
    sum = sensorValue + sum;
    sensorValue = analogRead(A0);
  }
  sum = sum / SamplesVD; // delen door aantal samples
  //Serial.print("if you set the Gain to 10,the input voltage:");
  //Serial.println(10*sum*5.2/1023.00);       // original code: 10*sum*4980/1023.00
  Vdiv = 10 * sum * 5.2 / 1023.00;
  //Serial.print("Vdiv = ");Serial.println(Vdiv);

  // Stand START-schakelaar:
  // -----------------------
  buttonState = digitalRead(buttonPin);
  if (buttonState == HIGH) { // Als de schakelaat in stand 1 wordt gezet, krijgt deze waarde HIGH
    Start = 1;
  } else {
    Start = 0;
    info = 0;
  }

  // Stand SPEED-schakelaar:
  // -----------------------
  int readingSLOW = digitalRead(buttonPin4);
  int readingFAST = digitalRead(buttonPin5);

  if (readingSLOW != lastButtonStateS) {  // if the state has changed, increment the counter
    lastDebounceTime = millis();         // reset the debouncing timer
  }
  if ((millis() - lastDebounceTime) > debounceDelay) {
    if (readingSLOW != buttonSLOW) {
      buttonSLOW = readingSLOW;

      if (buttonSLOW == HIGH) {
        sp = sp - 1; // Als de schakelaar in stand 1 wordt gezet, krijgt deze waarde HIGH

        if (sp == 1) {
          SSpeed = (1000 / 2 / Sampels) / 1.142;
          Vsample = "0.5s ";
        }
        else if (sp == 2) {
          SSpeed = (1000 / Sampels) / 1.142;
          Vsample = "1s   ";
        }
        else if (sp == 3) {
          SSpeed = (1000 * 2 / Sampels) / 1.142;
          Vsample = "2s   ";
        }
        else if (sp == 4) {
          SSpeed = (1000 * 3 / Sampels) / 1.09;
          Vsample = "3s   ";
        }
        else if (sp == 5) {
          SSpeed = (1000 * 5 / Sampels) / 1.05;
          Vsample = "5s   ";
        }
        else if (sp == 6) {
          SSpeed = (1000 * 10 / Sampels) / 1.03;
          Vsample = "10s  ";
        }
        else if (sp == 7) {
          SSpeed = 1000 * 30 / Sampels / 1.03;
          Vsample = "30s  ";
        }
        else if (sp == 8) {
          SSpeed = ((100 * 60 / Sampels) * 10) / 1.03;
          Vsample = "1 min";
        }
        else if (sp == 9) {
          SSpeed = ((100 * 120 / Sampels) * 10) / 1.03;
          Vsample = "2 min";
```

```
      }
      else if (sp == 10) {
        SSpeed = ((100 * 300 / Sampels) * 10) / 1.03;
        Vsample = "5 min";
      }

      lcd.setCursor(0, 3);  // 4e rij
      lcd.print("Speed: ");
      lcd.setCursor(8, 3);
      lcd.print(Vsample);

    }
    if (sp < 1) {
      sp = 1;
    }
  }
}
lastButtonStateS = readingSLOW;

if (readingFAST != lastButtonStateF) {  // if the state has changed, increment the counter
  lastDebounceTime = millis();         // reset the debouncing timer
}
if ((millis() - lastDebounceTime) > debounceDelay) {
  if (readingFAST != buttonFAST) {
    buttonFAST = readingFAST;

    if (buttonFAST == HIGH) {
      sp = sp + 1; // Als de schakelaar in stand 1 wordt gezet, krijgt deze waarde HIGH
      if (sp == 1) {
        SSpeed = (1000 / 2 / Sampels) / 1.142;
        Vsample = "0.5s ";
      }
      else if (sp == 2) {
        SSpeed = (1000 / Sampels) / 1.142;
        Vsample = "1s   ";
      }
      else if (sp == 3) {
        SSpeed = (1000 * 2 / Sampels) / 1.142;
        Vsample = "2s   ";
      }
      else if (sp == 4) {
        SSpeed = (1000 * 3 / Sampels) / 1.09;
        Vsample = "3s   ";
      }
      else if (sp == 5) {
        SSpeed = (1000 * 5 / Sampels) / 1.05;
        Vsample = "5s   ";
      }
      else if (sp == 6) {
        SSpeed = (1000 * 10 / Sampels) / 1.03;
        Vsample = "10s  ";
      }
      else if (sp == 7) {
        SSpeed = 1000 * 30 / Sampels / 1.01;
        Vsample = "30s  ";
      }
      else if (sp == 8) {
        SSpeed = ((100 * 60 / Sampels) * 10) ;
        Vsample = "1 min";
      }
      else if (sp == 9) {
        SSpeed = ((100 * 120 / Sampels) * 10) / 1.01;
        Vsample = "2 min";
      }
      else if (sp == 10) {
        SSpeed = ((100 * 300 / Sampels) * 10) / 1.01;
        Vsample = "5 min";
      }

      //Serial.println(""); Serial.println(Vsample); Serial.println("");
      lcd.setCursor(0, 3);  // 4e rij
      lcd.print("Speed: ");
      lcd.setCursor(8, 3);
      lcd.print(Vsample);
    }
    if (sp > 10) {
```

```
        sp = 10;
      }
    }
  }
  lastButtonStateF = readingFAST;


  /* ---- <START DE METING> ---- */
  if (Start == 0) {
    digitalWrite(pinRelais, LOW);
    // I2C LCD:
    // --------

    lcd.setCursor(0, 0);       //1e plaats, 1e rij
    lcd.print("1. Select:");
    lcd.setCursor(0, 1);       //4e plaats, 2e rij
    lcd.print("   a. Manual / Auto");
    lcd.setCursor(0, 2);       //1e plaats, 3e rij
    lcd.print("2. Start the test");
    lcd.setCursor(0, 3);       //1e plaats, 4e rij
    lcd.print("3. Adapt sample rate");
  }

  else if ((Start == 1) && (10 * sum * 5.2 / 1023.00 < 25.5 )) {

    if (info == 0 ) {
      lcd.setCursor(0, 0);       //1e plaats, 1e rij
      lcd.print("                    ");
      lcd.setCursor(0, 1);       //4e plaats, 1e rij
      lcd.print("    Wait for the    ");
      lcd.setCursor(0, 2);       //4e plaats, 2e rij
      lcd.print(" first measurement  ");
      lcd.setCursor(0, 3);       //1e plaats, 4e rij
      lcd.print("                    ");
    }
    digitalWrite(pinRelais, HIGH);

    if ((millis() - CurrentMillis >= SSpeed) ) {
      // Gemiddelde
      float ArrayU[Sampels];  // type float getallen en Samples =  10 dus onze rij bestaat uit 10
getallen
      float ArrayI[Sampels];  // type float getallen en Samples =  50 dus onze rij bestaat uit 10
getallen

      // INA219
      float shuntvoltage = 0;
      float busvoltage = 0;
      float current_mA = 0;
      float p_current_mA = 0;
      float loadvoltage = 0;

      shuntvoltage = ina219.getShuntVoltage_mV();
      busvoltage = ina219.getBusVoltage_V();
      current_mA = ina219.getCurrent_mA();
      loadvoltage = busvoltage + (shuntvoltage / 1000);

      if (current_mA < 0) {
        p_current_mA = current_mA * -1;
      }
      else if (current_mA > 0) {
        p_current_mA = current_mA * 1;
      }
      // Gemiddelde STROOM
      ArrayI[i] = p_current_mA;
      totalI = totalI + ArrayI[i];
      // Gemiddelde SPANNING
      ArrayU[i] = loadvoltage;
      totalU = totalU + ArrayU[i];
      i = i + 1;
      CurrentMillis = millis();

      if (i >= Sampels) {
        // Average Current - Voltage - Power
        float averageU = totalU / Sampels;
        float averageI = totalI / Sampels;
        float averageP = averageU * averageI / 1000;
```

```
// Cardinal direction
float X_h = 0;
float Y_h = 0;
float teta = 0;
String Wr;
imu::Vector<3> mag   = bno.getVector(Adafruit_BNO055::VECTOR_MAGNETOMETER);
X_h = mag.x();
Y_h = mag.y();

if (X_h != 0.00) {
  teta = atan2 (Y_h, X_h) * 57.2958;   // omzetten radialen naar graden & * -1 omdat de
                                       //BNO055 op zijn kop w gemonteerd

  if ((teta > -11.25) && (teta <= 11.25))
    Wr = "N  ";
  else if ((teta > 11.25) && (teta <= 33.75))
    Wr = "NNE";
  else if ((teta > 33.75) && (teta <= 56.25))
    Wr = "NE ";
  else if ((teta > 56.25) && (teta <= 78.75))
    Wr = "NEE";
  else if ((teta > 78.75) && (teta <= 101.25))
    Wr = "E  ";
  else if ((teta > 101.25) && (teta <= 123.75))
    Wr = "SEE";
  else if ((teta > 123.75) && (teta <= 146.25))
    Wr = "SE ";
  else if ((teta > 146.25) && (teta <= 168.75))
    Wr = "SSE";
  else if ((teta < -11.25) && (teta >= -33.75))
    Wr = "NNW";
  else if ((teta < -33.75) && (teta >= -56.25))
    Wr = "NW ";
  else if ((teta < -56.25) && (teta >= -78.75))
    Wr = "NWW";
  else if ((teta < -78.75) && (teta >= -101.25))
    Wr = "W  ";
  else if ((teta < -101.25) && (teta >= -123.75))
    Wr = "SWW";
  else if ((teta < -123.75) && (teta >= -146.25))
    Wr = "SW ";
  else if ((teta < -146.25) && (teta >= -168.75))
    Wr = "SSW";
  else if ((teta < -168.75) || (teta > 168.75))
    Wr = "S  ";

} // end IF

if ((X_h == 0.00) && (Y_h < 0.00)) {
  Wr = "O  ";
}
else if ((X_h == 0.00) && (Y_h > 0.00)) {
  Wr = "W  ";
}
else if ((X_h == 0.00) && (Y_h == 0.00)) {
  Wr = "Sart";
}

// Date and Time
DateTime now = rtc.now();
Serial.print(daysOfTheWeek[now.dayOfTheWeek()]);
Serial.print(" ");
Serial.print(now.day(), DEC);
Serial.print('/');
Serial.print(now.month(), DEC);
Serial.print('/');
Serial.print(now.year(), DEC);
Serial.print(" ");
Serial.print(now.hour(), DEC);
Serial.print(':');
Serial.print(now.minute(), DEC);
Serial.print(':');
Serial.print(now.second(), DEC);
Serial.print(";\t");
```

```
        Serial.print(averageU); Serial.print(";"); Serial.print(averageI); Serial.print(";");
Serial.print(averageP); Serial.print(";\t"); Serial.println(Wr);

        // LCD
        lcd.setCursor(0, 0);  // 1e rij
        lcd.print("U: ");
        lcd.print(averageU);
        lcd.print("V");

        lcd.setCursor(0, 1);  // 2e rij
        lcd.print("I: ");
        lcd.print(averageI);
        lcd.print("mA           ");

        lcd.setCursor(0, 2);   // 3e rij
        lcd.print("P: ");
        lcd.print(averageP);
        lcd.print("W            ");

        lcd.setCursor(17, 3);
        lcd.print(Wr);

        // Message first measurement must dissapear
        info = info + 1;

        // Reset
        i = 0;
        totalU = 0;
        totalI = 0;

      }

    }

  }


  else if ((Start == 1) && (10 * sum * 5.2 / 1023.00 > 26.00 )) {
    digitalWrite(pinRelais, LOW);
    // I2C LCD: / geen loop - geen LCDclear maar wel lege plaatsen printen
    // --------
    lcd.setCursor(0, 0); //bovenste rij blanco
    lcd.print("                    ");
    lcd.setCursor(0, 1); //4e plaats, 2e rij
    lcd.print("   Stop the test    ");
    lcd.setCursor(0, 2); //2e plaats, 3e rij
    lcd.print(" Voltage is to high ");
    lcd.setCursor(0, 3); //onderste rij blanco
    lcd.print("                    ");

  } // end 2e else if-loop

} // end loop
```

Code Processing voor de aanmaak van het *.txt-bestand

```
import processing.serial.*;

Serial mySerial;

PrintWriter output;

void setup() {
  mySerial = new Serial( this, Serial.list()[0], 9600) ;       // verander de [0] naar 1,2 ... om de juiste poort te selecteren
  String filename = year()+"."+month()+"."+day()+"-"+hour()+"h"+minute()+".txt"     // Maak van het tijdstip waarop de meting
                                                                                    //begint een string 'filename'
  output = createWriter(filename);                                                  // Maak het textbestand aan.
}

void draw() {                                          // Serialprint uit arduino wordt gelezen en opgeslagen in een .txt-bestand
    if (mySerial.available() > 0 ) {
         String value = mySerial.readStringUntil('\n');
         if ( value != null ) {
           value = trim(value);
           String spanning = value+";";
           output.println(spanning);
         }
    }
    output.flush();                                                 // Schrijf de overgebleven data naar het bestand
}

void keyPressed() {
  if (key == CODED) {
    if (keyCode == ESC) {                                           // Het maken van het tekstbestand moet stoppen wanneer op ESC
wordt gedrukt
    output.close();
    exit();                                                         // Stop het programma
    }
  }
}
```

Multicore Unscreened Cable

pro-POWER

RoHS Compliant

Specifications:

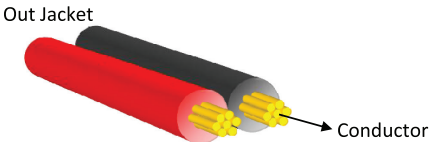
Conductor

Material	: Bare Copper (OFC)
Construction	: 80 × 0.12mm
Cross Section	: 0.91mm^2 × 2

Insulation

Material	: Polyvinylchloride (PVC)
Thickness	: 1mm
Diameter	: 3.4mm × 6.8mm
Operation Temperature	: -20°C to +70°C
Resistance Conductor	: 21.5Ω/kM

Construction:



Part Number Table

Description	Part Number
80/0.12mm Speaker Cable Red/Black, 100m	JY-1280-RB