Mission To Mars (STEM project ASO)

Inleiding

Een robot in de ruimte werkt slechts een beetje zelfstandig. Alles wat hij moet uitvoeren zal via een signaal vanop aarde de ruimte ingestuurd worden via schotelantennes en satellieten. Wij gaan in het eerste deel van onze sessie ook signalen in de ruimte sturen. Het budget was voor een satelliet te lanceren net onvoldoende dus zullen we een meting doen en iets aanschakelen via Wifi en de cloud. Dit noemen we IOT (Internet Of Things) en is nu de meest hypermoderne techniek die op de markt is. Met de cloudprofessor zal ons signaal de wereld rond gaan en blijven wijzelf in het STEM lokaal.

In het tweede deel van de sessie zullen we ook via een draadloos signaal de Mindstorm robot aansturen om op de planeet Mars de belangrijke satellietschotel te plaatsen zodat ook de robot iets kan terug sturen naar Mars. Omdat enkel een schotel niet sterk genoeg is (de afstand tussen Aarde en Mars is immers zeer groot) zullen we ook nog de satelliet zelf moeten lanceren. De systemen op en rond de Aarde heeft NASA en ESA al voor ons voorzien.

Communicatie in het ideale geval

Laten we er eerste eens van uitgaan dat ons budget om te communiceren met Mars onbeperkt is. Dan zouden we misschien wel met licht (laser) kunnen werken. Laten we dan eens berekenen hoelang het signaal onderweg zal zijn.

De snelheid van licht is afgerond 300 000 000 m/s dit is enorm snel, als onze auto even snel zou kunnen rijden dan zou de snelheidsmeter niet stoppen bij 300 km/h maar bij … ?

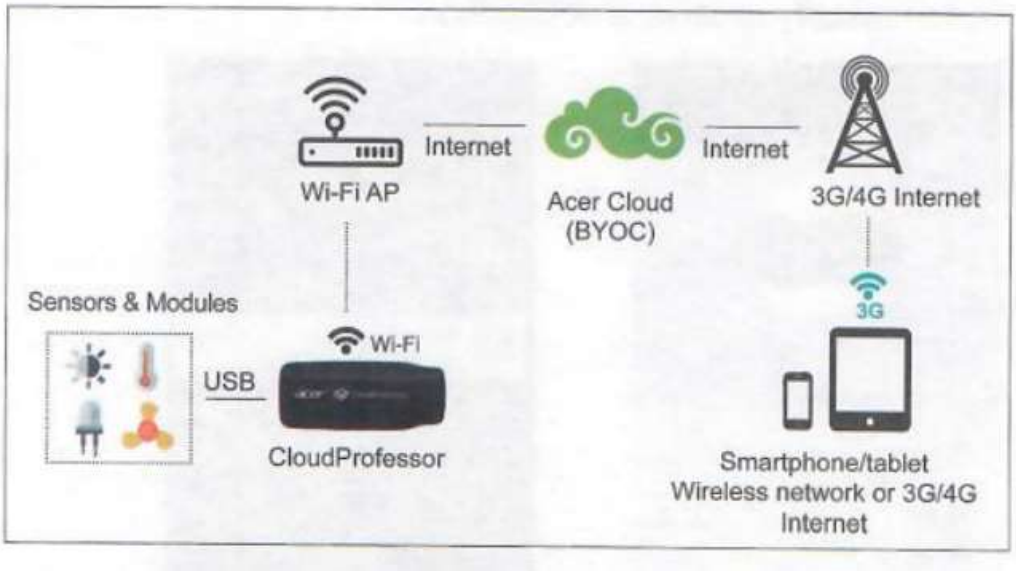
Voor van meter naar km te gaan moeten we het getal delen door 1000, want er gaan nu eenmaal 1000m in 1km. Dit betekend voor de snelheid van het licht dat deze gelijk is aan 300 000 km/s
Als het licht al zoveel km/s aflegt dan zal het een enorm grootte afstand afleggen op een uur en dit klopt ook. Alvorens een uur om is zijn er 3600 seconden verstreken. Elke seconde dat er verstrijkt is het licht 300 000km verder verwijdert. Voor het aantal km/h te weten moet je dus het aantal kilometers dat het licht aflegt per seconde vermenigvuldigen met 3600. Als eindoplossing krijgen we dus dat de snelheid 1 080 000 000 . Dit lezen we als 1 miljard 80 miljoen km/h.

Nu we weten hoeveel km licht kan reizen per uur moeten we de afstand kennen tussen Aarde en Mars. Omdat zowel Aarde als Mars een baan rond de zon beschrijven en deze niet even snel voortbewegen is de afstand niet constant. De kortste afstand is 55 758 006km en de langst mogelijke afstand is afgerond 400 000 000 km. Stel dat we pech hebben en het is de langste afstand dan duurt het signaal… ?

Snelheid is de afstand die je aflegt per tijdseenheid, hieruit kan je afleiden dat voor de tijd te bepalen je de afstand moet delen door de snelheid. Om dus 400 000 000 km af te leggen met een snelheid van 1 080 000 000 km/h heb je 0,37 uur nodig of 1333 seconden. Zoals je ziet is communicatie dus niet eenvoudig. Laten we het daarom vandaag bij communicatie tussen de tablet en de cloudprofessor houden en we laten de cloudprofessor de communicatie met de wifi router en het internet regelen. Zo laten we de communicatie tussen het internet en de tablet ook over aan de app. De communicatie met de EV3 zal nog eenvoudiger zijn, namelijk via bluetooth.

Cloudprofessor

De cloudprofessor is een mini computer die via wifi in staat is om te communiceren met het internet. Hierbij zal hij data ophalen en schrijven naar de cloud. Dankzij dit ben je in staat iets te schakelen via de app of een waarde van de sensor te lezen. De cloudprofessor zorgt voor het aanzetten van outputs en het inlezen van data via de USB verbinding met een Arduino board of met de LED101 module.
In de figuur hieronder zie je hoe de communicatie verloopt.



CloudProfessor en het mobiele apparaat kunnen ook elk via verschillende netwerken verbonden zijn. Bijvoorbeeld kan het mobiele apparaat via 3G netwerk verbonden zijn met het internet en de cloudprofessor via WiFi. Daar alles werkt via het internet kan de lamp zich in Taiwan bevinden en wij deze via de tablet hier in België bedienen.

Alle instellingen zijn op voorhand geconfigureerd en zowel de tablet als de cloudprofessor is verbonden via het wifi netwerk. Alles is dus klaar om te beginnen met het eerste project de BYOC Lamp. [Meer info over configuratie van de cloudprofessor bevindt zich in de bijgesloten handleiding bij de cloudprofessor starterkit.

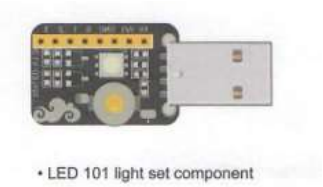
CPF 101 BYOC Lamp

Beschrijving

Deze les is een beginners les voor CloudProfessor. Het zal in eerste instantie 1 component gebruiken zodat je kan ervaren hoe leuk het is om de CloudProfessor te gebruiken als IoT (Internet of Thing) toepassing. Extra componenten die het gebruiken van de cloud nog plezieriger maken zullen in een latere les toegevoegd worden. De component die je in deze les zal gebruiken is de LED 101 verlichting set. Eens deze set via USB met de CPF verbonden is zal je met je mobiele toestel de witte led en de driekleuren LED kunnen aansturen. Met de app kan je dan de witte led, de groene, rode en blauwe LED aan- of uitschakelen. Je kan ook het programma aanpassen zodat je in staat bent om

bijvoorbeeld de witte led te laten knipperen op een bepaalde frequentie. Bovenop dit zal deze eerste les je ook introduceren in de basis programmeer instructies (javascrips) zodat je een beter begrip krijgt van deze code en in een later stadium in staat zal zijn deze aan te passen naar jou eigen IoT (Internet of Thing) creatie via dit leertraject.

Gebruikte componenten

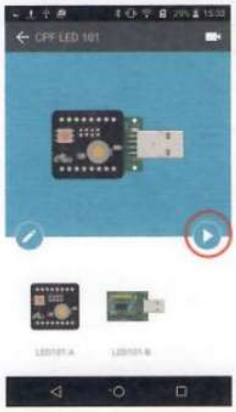


CPF LED 101 App : uit te voeren stappen

1) **Verbind de voeding van de Cloudprofessor en duw de power knop in en houd deze ingedrukt voor minimaal 2 seconden. De blauwe LED die aanduid of de CloudProfessor aan staat zal oplichten. Verbind nu pas de LED 101 licht module met het USB 3.0 slot van de Cloudprofessor.**

2) **Het mobiele toestel zal een notificatie ontvangen, klik op deze notificatie en kies ervoor op de CPF LED 101 app te openen. Als de app geopend is kan je voor Lesson1 : CPF LED 101 kiezen.**
Als de app nog niet geïnstalleerd is op het toestel zal je naar de store geleid worden, volg dan de stappen die op het scherm verschijnen zodat de app op je toestel kan geïnstalleerd worden.

3) **Duw op de play knop om toegang te krijgen tot de controle pagina. Je kan nu eenvoudig en zonder beperkingen de witte, rode, groene en blauwe LED aan of uitschakelen door gebruik te maken van de controle user interface (UI)**



• Execute button

• White LED light control switch

4) **Duw op de edit knop (potloodje) indien je in plaats van de controle pagina toegang wil krijgen tot de editor. Geavanceerde instellingen kunnen uitgevoerd worden vanaf deze edit pagina. Gedetailleerde informatie kan je vinden in het volgende programmeer code hoofdstuk.**

• Edit program code button

• Program editor

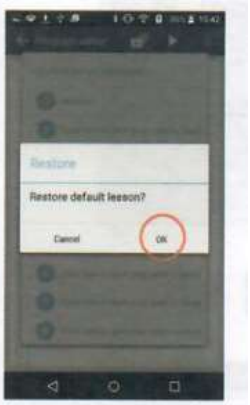
Alle lessen hebben een functie om het originele programma terug te kunnen plaatsen. Dit kan via het indrukken van een knop in de editor. Je kan ook wijzigingen opslaan, de instructies voor herstel of opslaan zijn de volgende.

1) **Na het openen van de programma editor kan je de programma herstel knop vinden aan de bovenkant van de editor pagina. Deze knop laat je toe het programma op te slaan in een nieuw bestand of om het programma te herstellen naar de originele les.**

• Program restore button

2) **Wanneer de programma herstel knop ingedrukt is zal het programma terugkeren naar het origineel, voor de zekerheid vraagt men om deze actie te bevestigen.**

• Program restore button



• Confirm button

3) **Er zijn 8 verschillende locaties beschikbaar in de interface om wijzigingen in op te slaan. De bestandsnaam zal automatisch geselecteerd worden van de eerste rij code die begint met // in de programmeeromgeving. Als deze rij niet bestaat dan zal het programma opgeslagen worden onder een standaard bestandsnaam.**

• Savebutton

• Confirm button

4) **Naast de functie om een programma op te slaan is er uiteraard ook de mogelijkheid o een eerder opgeslagen programma terug te laden.**

• Load button

• Confirm button

Programma code

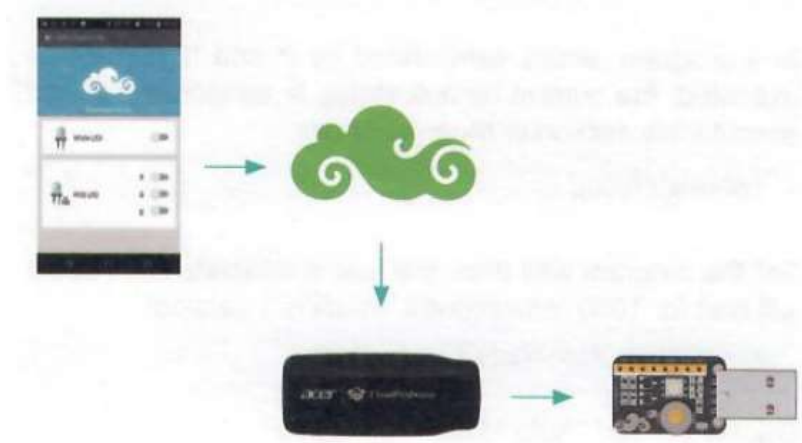
```
// CPF-LED 101 Lesson 1
var value;
value = ui.get("white led");
cpf.set("white led", value);

var r = ui.get("rgb led", "r");
var g = ui.get("rgb led", "g");
var b = ui.get("rgb led", "b");
cpf.set("rgb led", r, g, b);

// blink led
/*
cpf.sleep(1000);
ui.set("title", "Auto Mode")
ui.set("white led", 1);
cpf.set("white led", 1);
cpf.sleep(1000);
ui.set("white led", 0);
cpf.set("white led", 0);
cpf.sleep(1000);
*/
cpf.repeat();
```

Uitleg bij de programma code.

```
// CPF-LED 101 Lesson 1
var value;
value = ui.get("white led");
cpf.set("white led", value);
```

Niet enkel bij dit programma maar bij ALLE programmeertalen heeft // de betekenis commentaar te zijn. Deze lijn code zal dus niet uitgevoerd worden en slaat men over. Ondanks dat dit niet uitgevoerd wordt is het wel handig om te hebben in je programma ter verklaring. Zo weet jij of iemand anders die de code leest wat de betekenis is.

Bij het eigenlijke programma maakt men eerst een variabele (geheugen) aan die men de naam value meegeeft. Dan gebruikt men de ui.get() functie om de toestand van de witte led switch in de app in te lezen (1: slider AAN, 0: slider UIT). Deze toestand slaat men dan op in de variabele value. Dan zal deze waarde via de Cloud verzonden worden naar de Cloudprofessor. Via de cpf.set() functie zal de toestand van deze variabele de LED aanwezig op de printplaat de toestand toekennen. (1: LED AAN, 0: LED UIT). Met andere woorden neemt de LED eenvoudig weg de toestand van de slider in de user interface over en dit via de cloud. Onderstaande afbeelding toont het verloop van het signaal.



```
var r = ui.get("rgb led", "r");
var g = ui.get("rgb led", "g");
var b = ui.get("rgb led", "b");
cpf.set("rgb led", r, g, b);
```

Dezelfde logica is toepasbaar voor het aansturen van de RGB LED. Omdat de user interface deze LED controleert door middel van 3 schakelaars (die overeenkomen met rood, groen en blauw licht). Daarom gebruikt men ui.get() functie voor het lezen van elke switch en deze toestand wordt telkens opgeslagen in het overeenkomstige geheugen. Daarna gebruikt men cpf.set() om deze drie variabelen te versturen naar de CloudProfessor om daar dan de hardware de overeenkomstige toestand te geven (1: LED AAN, 0: LED UIT)

```
cpf.repeat();
```

Deze laatste lijn zal tot slot het programma in een continue lus herhalen.

Programma code 2

```
/*
*/
```

Wanneer in een programma code tussen /* en */ staat dan zal deze code NIET uitgevoerd worden. Dit omdat alle code dan als een blok commentaar zal worden beschouwd.

```
cpf.sleep(1000);
```

De functie cpf.sleep() bepaalt de wachttijd, gedurende deze tijd in ms zal de cloudprofessor niets uitvoeren. In dit geval 1000ms wat dus overeenkomt met 1s.

```
ui.set("title", "Auto Mode");
```

Dit regeltje code zorgt ervoor dat de titel “auto mode” op het scherm van je mobiel apparaat zal verschijnen.

```
ui.set("white led", 1);
cpf.set("white led", 1);
cpf.sleep(1000);
```

Zet de witte LED bij de user interface op 1 (SLIDER AAN) en zet daarna de LED hardwarematig ook op 1 (LED AAN) waarna er 1 seconde gewacht zal worden.

```
ui.set("white led", 0);
cpf.set("white led", 0);
cpf.sleep(1000);
```

Zet dan de witte LED bij de user interface op 0 (SLIDER UIT) en zet daarna de led hardwarematig ook op 0 (LED UIT) en wacht 1 seconde.
Dit tweede gedeelte code zal dus de led doen knipperen door deze een bepaalde tijd aan te zetten en een bepaalde tijd uit. Je kan gerust experimenteren door deze tijden aan te passen, ze hoeven niet gelijk te zijn. Je kan gerust de LED een korte tijd aanschakelen en een langere tijd uitschakelen en vice versa.

Om dit stuk code uit te voeren verwijder je de /* */ code die rond deze code staat en plaats je deze rond het eerste gedeelte zodat dit nu in commentaar komt en niet meer uitgevoerd zal worden.

Als je dit stuk code begrijpt ben je klaar om te beginnen met de Cloudprofessor in combinatie met de Arduino Leonardo.