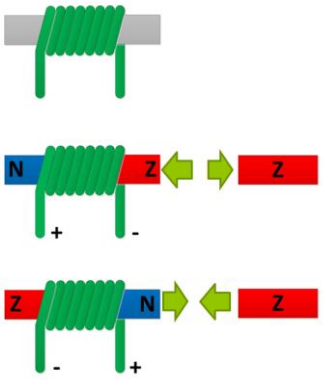
INHOUD

Werking dc motor

Magnetisme



In deze les willen we de werking uitleggen van de kleine DC motortjes, die we dikwijls gebruiken in allerhande projectjes die we met onze microcontroller willen besturen. De werking ervan berust volledig op magnetisme.

We beschouwen hier even een elektromagneet. Een elektromagneet bestaat uit een elektrische wikkeling rond een metalen kern. Als we een elektrische stroom door die wikkeling sturen, dan zal de metalen kern magnetisch worden met een Zuidpool aan de ene kant en een Noordpool aan de andere kant.

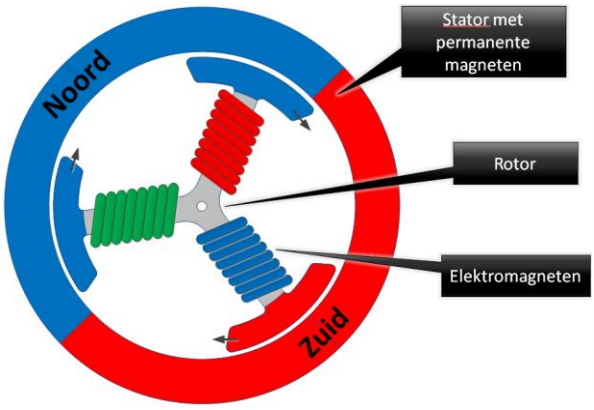
Als we de stroomzin door de elektrische wikkeling omkeren, dan blijft de metalen kern magnetisch, maar dan wisselen de polen.

Een ander basisprincipe van magnetisme is dat gelijke polen – bijvoorbeeld twee zuidpolen – elkaar afstoten en dat tegengestelde polen – bijvoorbeeld een noord en een Zuidpool – elkaar aantrekken.

Als we deze kennis even toepassen op deze tekeningen, dan kunnen we door de elektrische stroomzin om te keren, de ene keer de Zuidpool magneet afstoten en de volgende keer de zuidpoolmagneet aantrekken. We kunnen vanaf nu dus elektrische energie omzetten in een mechanische kracht.

Rotor en Stator

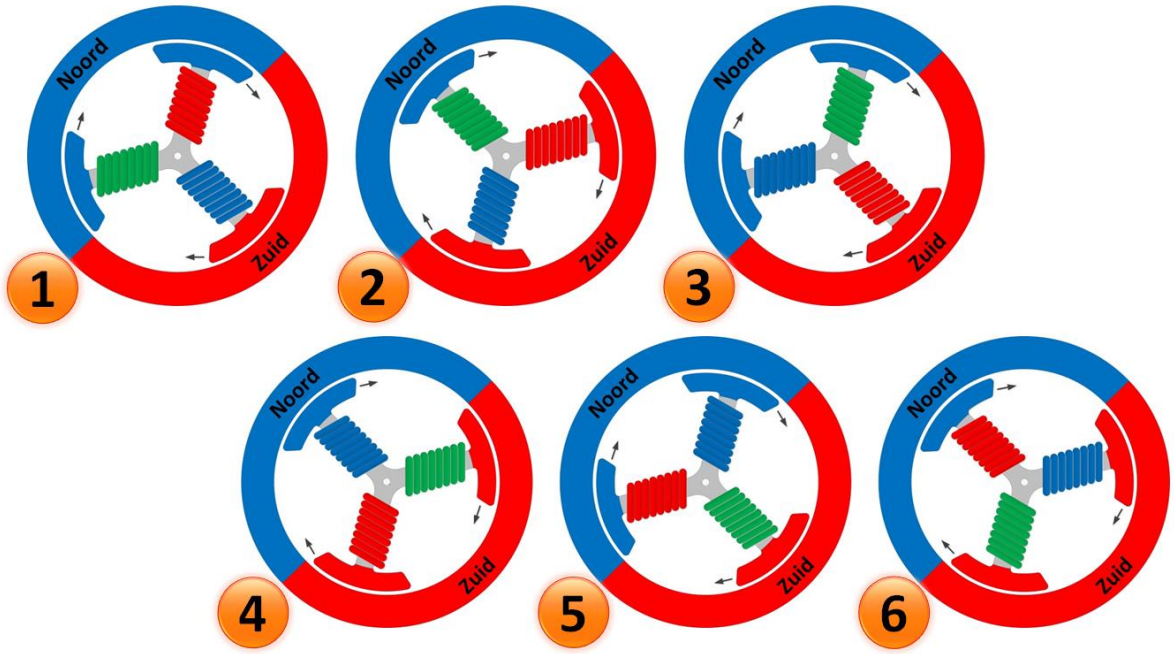
De kleine DC motortjes die wij gewoonlijk slopen uit kleine elektrische toestellen werken typisch met 3 – of een veelvoud van 3 – elektromagneten in de rotor.



De rotor is het gedeelte van de DC motor dat ronddraait. De stator is het statische of vaststaande deel van de motor. Bij kleine DC motortjes bestaat die uit een metalen behuizing met daarin 1 of meer permanente magneten telkens bestaande uit een noord en een Zuidpool.

De 3 elektromagneten vormen de rotor. Afhankelijk van in welke zin de stroom door de respectievelijke spoelen gestuurd wordt zal de ijzeren kern magnetisch worden gemaakt met een zuid of een Noordpool. We hebben onthouden dat gelijke polen elkaar afstoten. De blauwe winding maakt van deze elektromagneet een Zuidpool. Deze Zuidpool stoot zich af tegen de Zuidpool van de permanente magneet van de stator.

De groene en de rode spoel induceren beide een Noordpool. Deze noordpolen worden ook afgestoten door de permanente Noordpool van de stator en aangetrokken door de permanente Zuidpool van de stator. De rotor wordt door al deze afstoot en aantrekkingskrachten in beweging gezet en begint rond te draaien.



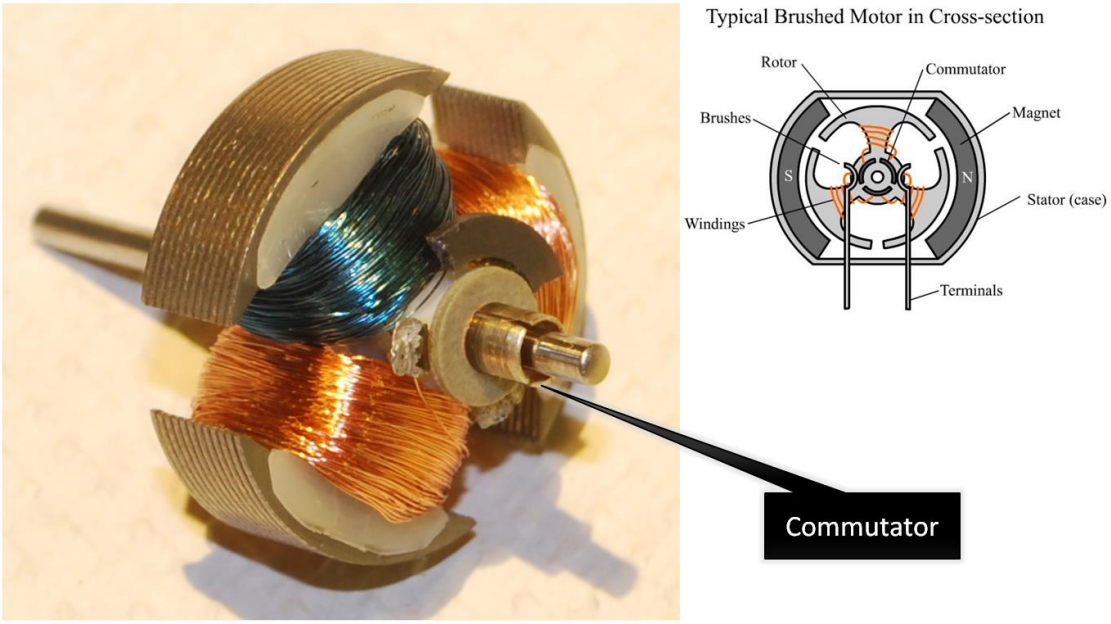
De kunst bestaat er bij een DC motor in om vanaf nu, op het juiste moment de stroomzin door de spoelen om te keren, zodat de rotor blijft ronddraaien.

Je ziet hier dat bij de tweede situatie de groene spoel een Noordpool gebleven is, maar dat de rode spoel nu een Zuidpool geworden is – samen met de blauwe spoel. Deze nieuwe situatie wordt ook hier maximaal afgestoten door de statormagneten, en de motor draait door.

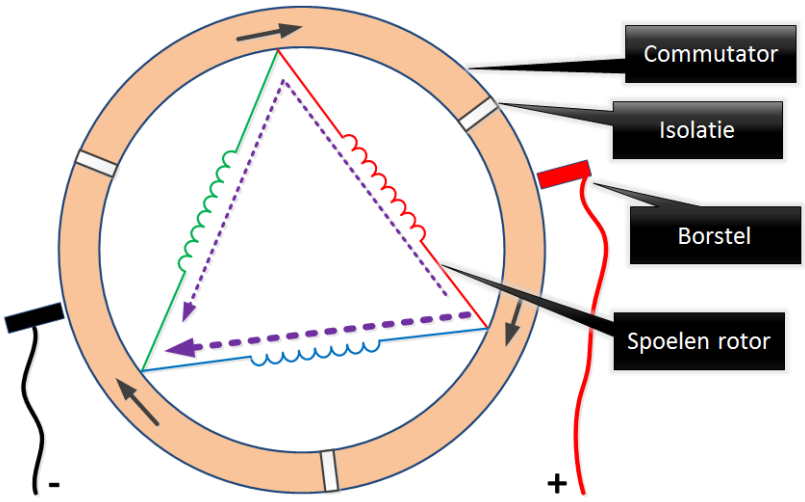
In de volgende situatie krijgen we weerom 2 noordpolen en een Zuidpool magneet. In situatie 4 is het weerom de blauwe spoel die alleen bekrachtigd is – zoals in situatie 1, maar hier veroorzaakt de blauwe spoel een Noordpool. Na de 6^{e} stap beginnen we terug bij stap 1.

Commutator

Tot op dit punt lijkt dit werkingsprincipe nog niet te ingewikkeld, maar je zou er eens over moeten nadenken hoe we telkens op het juiste moment de stroomzin door deze 3 verschillende spoelen gaan omkeren. We voeden de DC motor immers met een gelijkspanning.



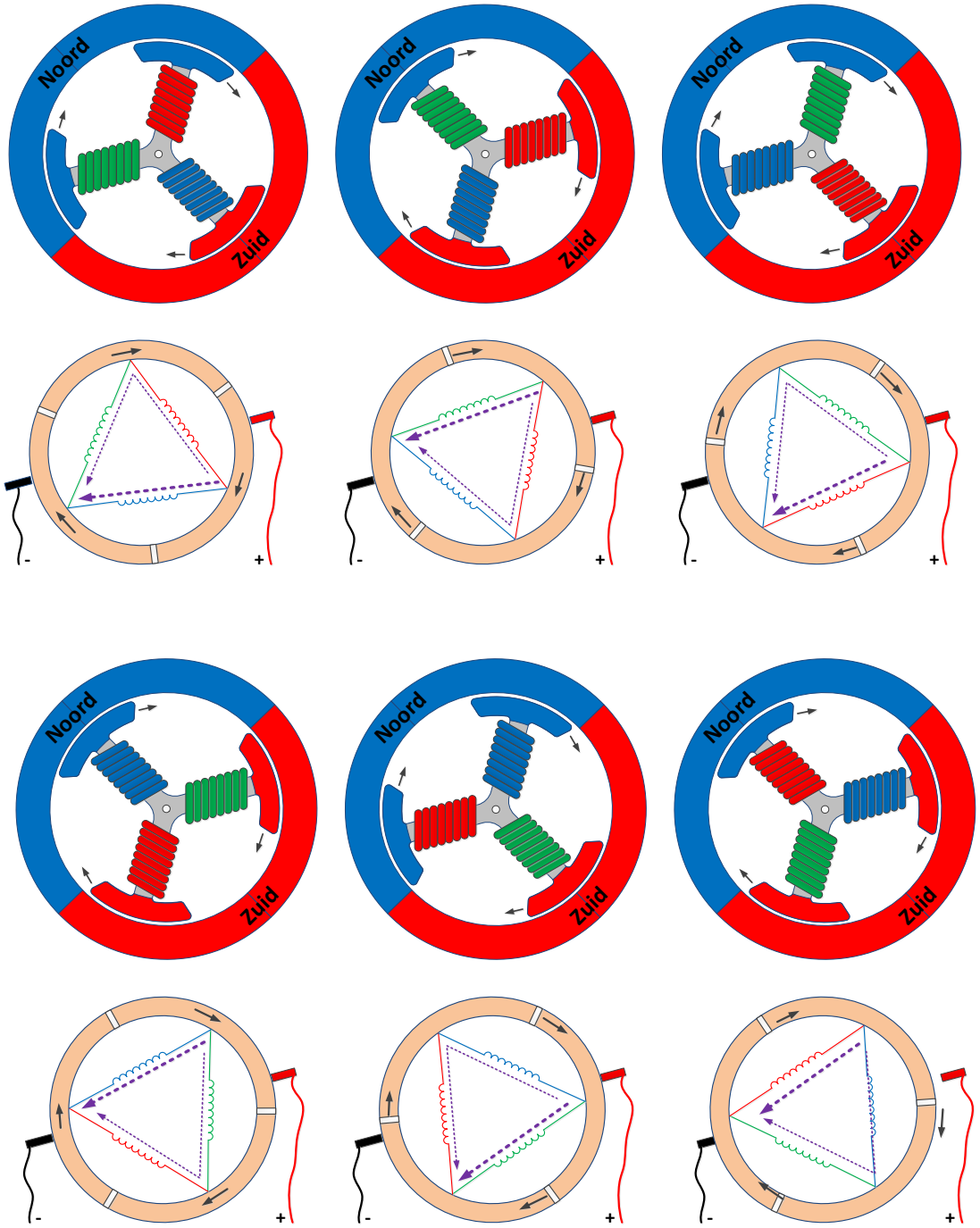
Het antwoord is om gebruik te maken van een commutator en borstels.



Een commutator draait mee rond met de rotor van de motor. Onze commutator bestaat hier uit 3 geleidende delen – ook lamellen genoemd die elektrisch van elkaar gescheiden zijn. De borstels of sleepringen hangen vast aan de stator van de motor en hangen dus stil. Ze wrijven tegen de commutator-lamellen en brengen zo elektrische DC stroom over op de commutator.

De 3 spoelen van de 3 elektromagneten van de rotor zijn vast verbonden met twee van de 3 commutator-lamellen. Het is hier mooi te zien dat de drie spoelen in driehoek staan.

Op deze tekening vloeit er een stroom door de blauwe spoel – van rechts naar links en er vloeit een kleinere stroom door de rode en groene spoel vermits die nu in serie staan. Dit veroorzaakt twee noordpolen in de groene en blauwe spoel en een Zuidpool in de blauwe spoel en de motor draait.



De commutator draait mee en 60° verder vloeit er nu een stroom door de groene spoel en staan de rode en de blauwe spoel in serie.

In stap 4 is het weerom de blauwe spoel die het sterkste bekrachtigd wordt en staan groen en rood weerom in serie. In stap 1 veroorzaakte dit een Zuidpool op de blauwe wikkeling, maar vermits nu de blauwe spoel omgedraaid is, veroorzaakt dit hier een noordpool.

Om de draairichting van de motor om te keren volstaat het om simpelweg de + en – klem om te keren.

Het voordeel aan werken met een commutator is dat het omwisselen van de stroom door de spoelen altijd synchroon verloopt met de draaisnelheid en de stand van de motor – omdat de commutator mechanisch vast hangt aan de rotor. De commutator maakt in dit geval van onze DC spanning in principe een 3 fase wisselspanning die mechanisch gesynchroniseerd wordt met het toerental van de motor…

Het nadeel is dan weer dat borstels verslijten, dat ze een bepaalde wrijving veroorzaken en dat die wrijving niet geheel geruisloos is.

Voordelen:

- Faseomkering spoelen altijd synchroon met toerental

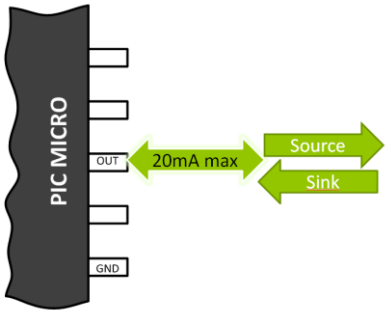
Nadelen:

- Wrijving van borstels – dus
  - Slijtage
  - Minder hoge toerentallen
  - Geluid

Uitdagingen werking dc motor

1. Demonteer een klassieke DC motor (geen brushless DC) uit een afgedankt toestel (CD speler, accuboormachine, ….). Sluit deze DC motor aan op een DC spanning en breng deze aan het draaien.
2. Demonteer deze DC motor voorzichtig, maak foto's van de spoelen, de rotor, de stator, de commutator en de borstels.
   a. Hoeveel spoelen zijn er?
   b. Uit hoeveel N en Z polen bestaat de stator? (met een andere magneet kan je voelen of er wordt afgestoten of aangetrokken)
   c. Uit hoeveel lamellen bestaat de commutator?
3. Monteer alle onderdelen terug en laat de DC motor terug draaien.

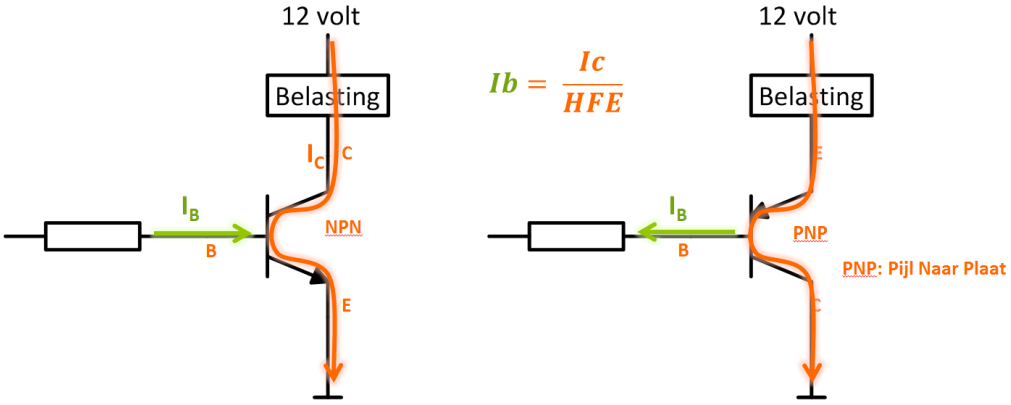
Bipolaire Junctie Transistor, Darlington of Mosfet



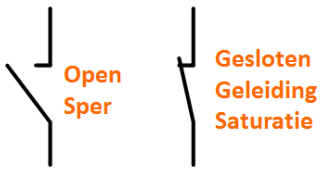
De stroom die één pin van een microcontroller kan leveren –ook sourcen genoemd - of binnen kan trekken -ook sinken genoemd is voor onze microcontroller slechts 20 à 25 mA, maar deze stroom ligt voor modernere microcontrollers zelfs nog een flink stuk lager. Een eenvoudig ledje aan en uit laten gaan lukt dus nog wel, maar zwaardere belastingen kunnen we niet meer rechtstreeks schakelen met een microcontroller. Voor die situaties gebruiken we dan een extra component - een elektronische schakelaar of een transistor die wel voldoende stroom kan schakelen.

Er bestaan verschillende types transistoren die elk hun eigen toepassingsgebied hebben.

Bipolaire Junctie Transistor



Als eerste zie je hier de bipolaire junctie transistoren. Je ziet hier de twee versies van een bipolaire transistor – de NPN en de PNP versie. De meest gebruikte is zonder twijfel de NPN transistor. PNP symbolen kan je herkennen omdat hierbij de Pijl naar de Plaat staat – PNP – Pijl naar plaat. De 3 pinnen van een bipolaire transistor zijn de basis, de collector en de emitter.



Een transistor kan zowel in geleiding als in sper staan. In geleiding – ook saturatie genoemd - gedraagt de transistor zich tussen de collector en emitter als een ‘gesloten schakelaar’ – in sper gedraagt de transistor zich als een open schakelaar.

In geleiding vloeit er een stroom door de transistor van collector naar emitter bij NPN en van emitter naar collector bij PNP.

Pas als er voldoende basisstroom in de transistor gestuurd wordt, dan pas zal een transistor helemaal in geleiding gestuurd worden. Die basisstroom kan je berekenen a.d.h.v. 2 parameters. De eerste is de collectorstroom – de collectorstroom is ook de stroom die we door de belasting willen sturen, die is dus gekend. De tweede parameter is de HFE factor. De HFE factor is terug te vinden in de datasheets. Deze HFE factor bepaalt hoeveel maal kleiner de basisstroom t.o.v. van de collectorstroom mag zijn – om de transistor toch volledig in saturatie te krijgen. Deze basisstroom kan je regelen door een weerstand mee in de basis op te nemen, maar daar komen we later nog op terug.

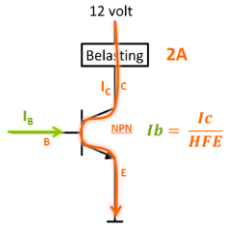
Even kort samenvatten – om een bipolaire transistor in geleiding te krijgen moet je een stroom in de basis sturen. Deze basisstroom mag HFE maal kleiner zijn als de grote stroom die we in de collector willen schakelen.

I_c max = 100mA
H_{FE} = 400

I_c max = 2A
H_{FE} = 100

I_c max = 10A
H_{FE} = 20

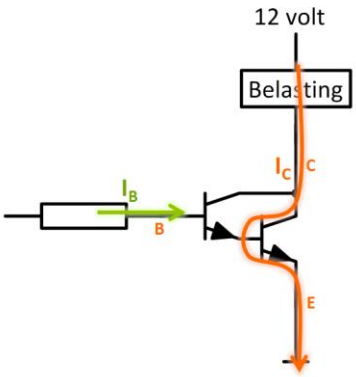
Er zijn verschillende soorten bipolaire transitoren – zo heb je klein vermogen transitoren die typisch stromen van enkele honderden mA kunnen schakelen en die heel grote HFE waardes hebben. Er bestaan ook transistoren die grotere stromen kunnen schakelen – de vermogen transistoren – maar die hebben dan weer kleinere HFE waarden.



We werken even een voorbeeld uit van een situatie waarbij we 2A door een belasting willen laten vloeien. De klein-vermogen transistor komt al niet in aanmerking omdat deze slechts 100mA kan schakelen. De tweede kan nipt de 2A schakelen die we wensen, maar zit dan juist op het randje van wat wij wensen. Met een HFE van 100 komen we dan uit op een minimale stroom in de basis van 20mA. Deze 20mA is ook maar juist op de grens van wat onze uC kan sturen met een pin. Het zou gaan, maar bij een goed ontwerp blijven we liever wat van deze grenzen weg. Dat verbetert de levensduur van de schakeling namelijk aanzienlijk. De derde transistor kan 10A schakelen – daar blijven we met onze 2A ruim binnen, maar met een HFE van slechts 20 komen we hier uit op een minimale basisstroom van 100mA om 2A collectorstroom te schakelen. Deze 100mA ligt ver boven de maximale 20mA die we met een uC kunnen leveren.

Hoe groter het schakelvermogen van de transistor – hoe kleiner de HFE waarde. Zeker als de belastings-stromen wat groter worden, is het moeilijk om binnen het segment van de klassieke bipolaire transistoren een transistor te vinden die de gewenste stroom kan schakelen, en die daarvoor slechts een basisstroom van beneden de 20mA nodig heeft.

De Darlington Transistor

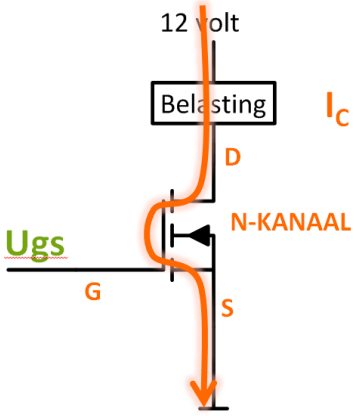


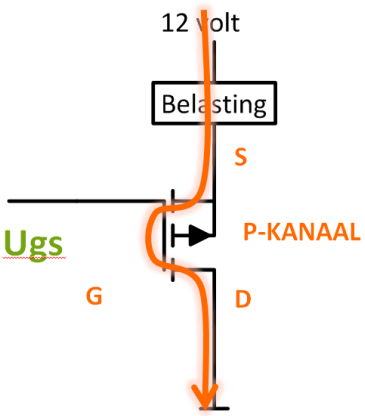
BDX33B

Ic max = 10A
HFE = 750 !!!

Een darlingtontransistor kan hier echter wel een oplossing bieden. Een darlingtontransistor ziet er uitwendig juist hetzelfde uit als een gewone transistor, met een basis, een collector en een emitter, maar inwendig bestaat deze uit een combinatie van een laagvermogen transistor met een hoge HFE en een power transistor met een lage HFE. Echter samen geschakeld zoals hier in een Darlington-configuratie – mogen deze twee HFE's met elkaar vermenigvuldigd worden. We verkrijgen zo een power transistor met een hoge HFE, die tot gevolg heeft dat we met zeer kleine basisstromen toch hele grote collectorstromen kunnen schakelen.

De Mosfet





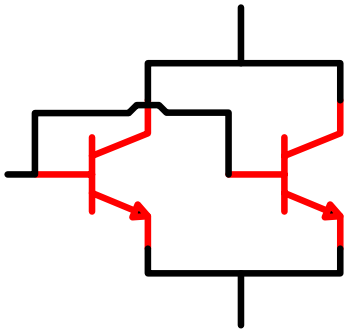
Naast de bipolaire transistor en de darlingtontransistor is ook de verrijkingsmosfet een elektronische schakelaar die je mee moet overwegen.

Mosfets bestaan nog niet zo lang als de bipolaire transistoren en waren vooral in hun beginjaren niet zo populair omdat ze héél gevoelig waren voor statische elektriciteit – door de mosfet verkeerd vast te nemen kon deze al stuk zijn – maar die problemen zijn nu volledig opgelost.

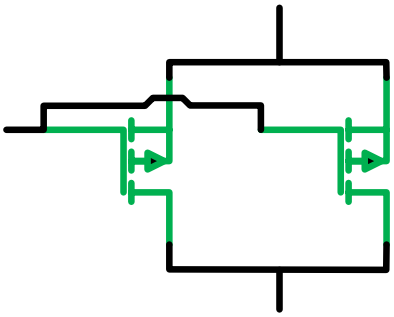
Ook bij de mosfets hebben we N-kanaals mosfets en P kanaals mosfets en ook hier zijn de N-kanaals de meest gebruikte. De 3 pinnen noemen nu Drain, Source en Gate. Als de mosfet in geleiding staat, dan kan er – bij een N-kanaals - een grote belastingsstroom vloeien van drain naar source. Bij een P kanaals is dit juist omgekeerd. Een mosfet heeft echter een zeer groot voordeel t.o.v. een bipolaire transistor. In de gate van een mosfet moet er geen stroom vloeien om de mosfet in geleiding te zetten – er moet enkel een spanning worden aangelegd. De mosfet trekt dus helemaal geen stroom uit de pin van de uC. In de datasheets kan je terugvinden hoeveel spanning er tussen de gate en de source moet worden aangelegd om de mosfet in geleiding te brengen. De zogenaamde LOGIC

mosfets kunnen al met 5 volt in geleiding gestuurd worden, wat ideaal is voor ons omdat onze uC ook mooi 5 volt uit geeft. Een bijkomend voordeel is dat er geen basisweerstand moet berekend en geplaatst worden. De mosfet heeft nog wel wat voordelen, maar die bespreken we in een volgende les.

Parallel Schakelen van Darlington of Mosfet



BJT = NTC werking
Eén T trekt bij begin net iets meer stroom
IC ↗ => Temp↗ => Rce↘ =>
IC↗ => Temp↗…..
Transistor trekt nog meer stroom en gaat stuk



MOSFET = PTC werking
Eén T trekt bij begin net iets meer stroom
ID ↗ => Temp↗ => Rce↗ =>
ID↘ => Temp↘…..
Stroom verdeelt zich mooi over beide mosfets

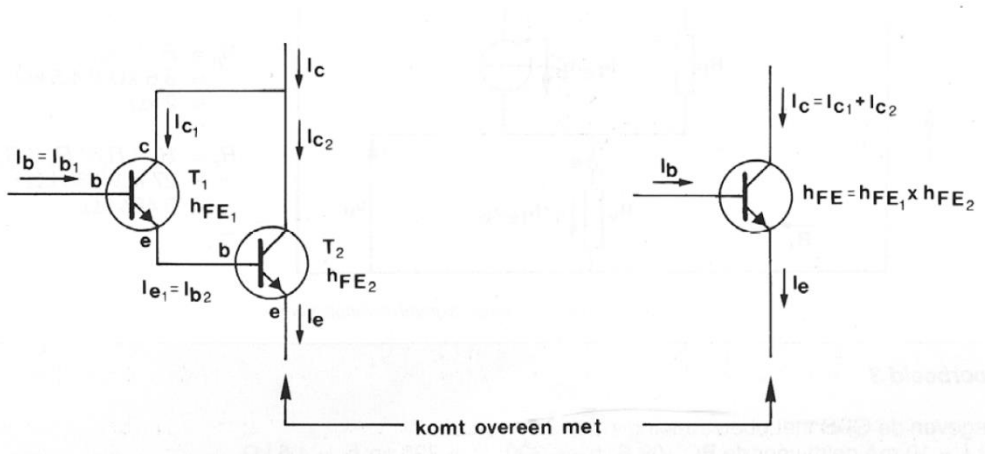
Je kunt in een situatie komen waarin je met de transistor of mosfet die je hebt toch niet de stroom kan schakelen die je wenst. Dan zou je er aan kunnen denken om twee of meerdere parallel te plaatsen om zo de stroom te verdelen over twee elektronische schakelaars.

BJT en Darlingtons hebben echter een NTC werking. Als je twee transistoren parallel zou plaatsen dan is er bij de start altijd wel één die net iets meer stroom schakelt als de andere. Deze wordt dan iets warmer en de NTC werking zorgt ervoor dat de weerstand tussen C en E daalt waardoor Ic nog meer stijgt en de temperatuur dus nog meer stijgt tot na een korte tijd één transistor alle stroom voor zich neemt en stuk gaat. Er zijn manieren om dit tegen te gaan, maar het parallel schakelen van BJT is niet eenvoudig.

Bij mosfets is dit wel eenvoudig. Mosfets hebben namelijk een zelfregelende PTC werking. Als de temperatuur stijgt, dan stijgt ook de weerstand, waardoor de stroom zal dalen en de temperatuur ook terug zal dalen. Mosfets die in parallel geschakeld worden verdelen dus zelf de stroom zodat alle mosfets even veel stroom voor zich nemen.

Uitdagingen BJT, Darlington Of Mosfet

1. Zoek de datasheet op van een BC547
    a. Is dit een bipolaire, een Darlington of een Mosfet?
    b. Is dit een NPN of PNP?
    c. Teken de behuizing en duid aan wat de collector, emitter en basis zijn.
    d. Wat is de maximale collectorstroom die continu door de transistor mag vloeien als de transistor in geleiding staat?
    e. Wat is de minimale HFE van een BC547B?
    f. Hoeveel spanning staat er nog tussen collector en emitter als de transistor volledig in geleiding staat en er 100mA collectorstroom vloeit – dit wordt aangegeven met Uce(sat)?
    g. Hoeveel spanning staat er tussen basis en emitter als de transistor volledig in geleiding staat en er 100mA collectorstroom vloeit – dit wordt aangegeven met Vbe(sat)?
2. Zoek de datasheet op van een 2N3055
    a. Is dit een bipolaire, een Darlington of een Mosfet?
    b. Is dit een NPN of PNP?
    c. Teken de behuizing en duid aan wat de collector, emitter en basis zijn.
    d. Wat is de maximale collectorstroom die continu door de transistor mag vloeien als de transistor in geleiding staat?
    e. Wat is de minimale HFE van een 2N3055 – bij 4A Ic?
    f. Hoeveel spanning staat er nog tussen collector en emitter als de transistor volledig in geleiding staat en er 4A collectorstroom vloeit – dit wordt aangegeven met UCE(sat)?
    g. Hoeveel spanning staat er tussen basis en emitter als de transistor volledig in geleiding staat en er 4A collectorstroom vloeit – dit wordt aangegeven met VBE(on)?
3. Zoek de datasheet op van een BDX33B
    a. Is dit een bipolaire, een Darlington of een Mosfet?
    b. Is dit een NPN of PNP?
    c. Teken de behuizing en duid aan wat de collector, emitter en basis zijn.
    d. Wat is de maximale collectorstroom die continu door de transistor mag vloeien als de transistor in geleiding staat?
    e. Wat is de minimale HFE van een BDX33B – bij 3A Ic?
    f. Hoeveel spanning staat er nog tussen collector en emitter als de transistor volledig in geleiding staat en er 3A collectorstroom vloeit – dit wordt aangegeven met UCE(sat)?
    g. Hoeveel spanning staat er tussen basis en emitter als de transistor volledig in geleiding staat en er 4A collectorstroom vloeit – dit wordt aangegeven met VBE?
4. Zoek de datasheet op van een RFP12N10L
    a. Is dit een bipolaire, een Darlington of een Mosfet?
    b. Is dit een N-kanaals of P-kanaals?
    c. Teken de behuizing en duid aan wat de drain-source en gate zijn.
    d. Wat is de maximale collectorstroom die continu door de transistor mag vloeien als de transistor in geleiding staat?
    e. Bekijk de Vds i.f.v. Id grafiek. Wat is de maximale drain stroom die je kan schakelen als je slechts 3 volt Ugs spanning aan de gate zou leggen?
    f. Hoe groot is de weerstand tussen drain en source als de mosfet in saturatie staat – dit wordt aangegeven met Rds(on)
    g. Hoeveel spanning staat er dan tussen drain en source als er een belastingsstroom van 2A vloeit – dit moet je zelf berekenen met de Rds(on) waarde?
5. Hieronder wordt wiskundig bewezen dat de totale HFE van een Darlingtontransistor gelijk is aan het product van beide HFE's. Bestudeer dit bewijs!!



De collectorstroom van een darlington:

	Ic = Ic1 + Ic2		
Omdat	Ic1 « Ic2	is	Ic ≈ Ic2
of	Ic2 = hFE2 . Ib2		
Nu is:	Ib2 = Ie1 ≈ hFE1 . Ib		
Vandaar:	Ic = hFE1 . hFE2. Ib		

De totale statische of DC-stroomversterking wordt dan:

hFE = hFE1 . hFE2

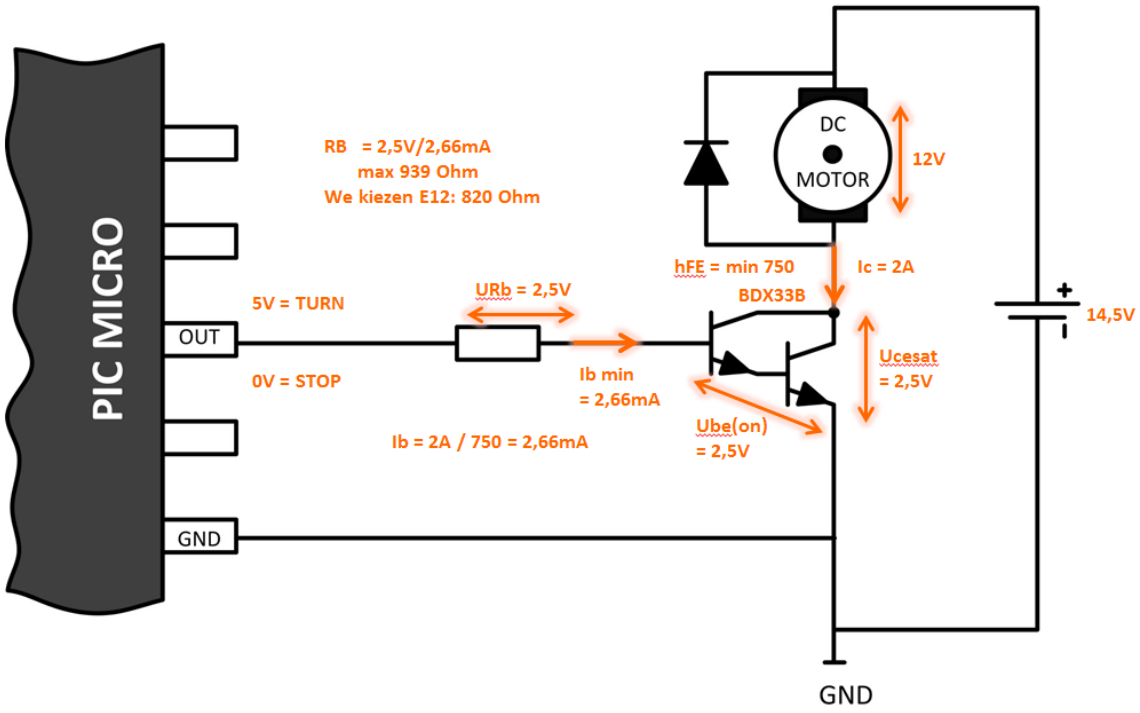
6. Wat is het wezenlijke verschil tussen een NPN en een PNP transistor. Leg dit uit a.d.h.v. enkele tekeningen.
7. Bekijk op het internet enkele filmpjes die de werking van een transistor uitleggen. Kies er één uit dat je het duidelijkste vindt.
8. Teken de opbouw van een N-kanaals verrijkingsmosfet en leg aan de hand hiervan de werking uit.
9. Bekijk op het internet filmpjes die de werking van een mosfet uitleggen. Kies er één uit dat je het duidelijkste vindt.

Oplossingen Uitdagingen BJT, Darlington Of Mosfet

1. Zoek de datasheet op van een BC547
    a. Is dit een bipolaire, een darlington of een Mosfet? Bipolaire
    b. Is dit een NPN of PNP? NPN
    c. Teken de behuizing en duid aan wat de collector, emitter en basis zijn.
    d. Wat is de maximale collectorstroom die continu door de transistor mag vloeien als de transistor in geleiding staat? 100mA
    e. Wat is de minimale HFE van een BC547B? 200
    f. Hoeveel spanning staat er nog tussen collector en emitter als de transistor volledig in geleiding staat en er 100mA collectorstroom vloeit – dit wordt aangegeven met UCE(sat)? 100mA
    g. Hoeveel spanning staat er tussen basis en emitter als de transistor volledig in geleiding staat en er 100mA collectorstroom vloeit – dit wordt aangegeven met VBE(sat)? 700mV
2. Zoek de datasheet op van een 2N3055
    a. Is dit een bipolaire, een Darlington of een Mosfet? Bipolaire
    b. Is dit een NPN of PNP? NPN
    c. Teken de behuizing en duid aan wat de collector, emitter en basis zijn.
    d. Wat is de maximale collectorstroom die continu door de transistor mag vloeien als de transistor in geleiding staat? 15A
    e. Wat is de minimale HFE van een 2N3055 – bij 4A Ic? 20
    f. Hoeveel spanning staat er nog tussen collector en emitter als de transistor volledig in geleiding staat en er 4A collectorstroom vloeit – dit wordt aangegeven met UCE(sat)? 1.1V
    g. Hoeveel spanning staat er tussen basis en emitter als de transistor volledig in geleiding staat en er 4A collectorstroom vloeit – dit wordt aangegeven met VBE(on)? 1.5V
3. Zoek de datasheet op van een BDX33B
    a. Is dit een bipolaire, een Darlington of een Mosfet? Darlington
    b. Is dit een NPN of PNP? NPN
    c. Teken de behuizing en duid aan wat de collector, emitter en basis zijn.
    d. Wat is de maximale collectorstroom die continu door de transistor mag vloeien als de transistor in geleiding staat? 10A
    e. Wat is de minimale HFE van een BDX33B – bij 3A Ic? 750
    f. Hoeveel spanning staat er nog tussen collector en emitter als de transistor volledig in geleiding staat en er 3A collectorstroom vloeit – dit wordt aangegeven met UCE(sat)? 2.5V
    g. Hoeveel spanning staat er tussen basis en emitter als de transistor volledig in geleiding staat en er 4A collectorstroom vloeit – dit wordt aangegeven met VBE? 2.5V
4. Zoek de datasheet op van een RFP12N10L
    a. Is dit een bipolaire, een Darlington of een Mosfet? MOSFET
    b. Is dit een N-kanaals of P-kanaals? N-kanaals
    c. Teken de behuizing en duid aan wat de drain-source en gate zijn.
    d. Wat is de maximale collectorstroom die continu door de transistor mag vloeien als de transistor in geleiding staat? 12A
    e. Bekijk de VDS ifv ID grafiek. Wat is de maximale drain stroom die je kunt schakelen als je slechts 3 volt Ugs spanning aan de gate zou leggen? +/-7A – geen 10A
    f. Hoe groot is de weerstand tussen drain en source als de mosfet in saturatie staat – dit wordt aangegeven met Rds(on) 0.2 ohm
    g. Hoeveel spanning staat er dan tussen drain en source als er een belastingsstroom van 2A vloeit – dit moet je zelf berekenen met de Rdson waarde? 0.4 volt

DARLINGTONTRANSISTOR ALS SCHAKELAAR

Als eerste werken we een voorbeeld uit waarbij we een klassieke bipolaire darlingtontransistor gebruiken als elektronische schakelaar.

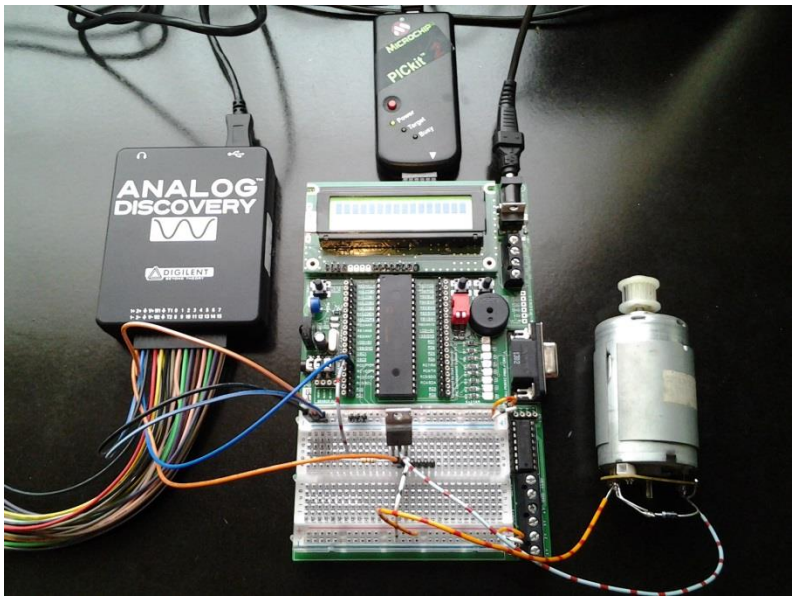


De BDX33B Darlingtontransistor die we voor deze toepassing gekozen hebben moet deze motor aan en uit kunnen schakelen. Om een bipolaire transistor als schakelaar in te stellen worden er steeds een aantal vaste stappen doorlopen.

- In de datasheet vinden we terug dat “Ucesat” – dat is de spanning die tussen collector en emitter staat als de transistor volledig in geleiding is – 2.5Volt bedraagt. De motor werkt op 12 Volt, dus de voeding moet 14,5 volt leveren.
- Vergeet ook zeker niet om de gnd van de voeding aan de gnd van de microcontroller te hangen!!
- Deze DC motor trekt in volle belasting ongeveer 2 ampère, dus de collectorstroom is 2A. Dit is ruim onder de 10A die de BDX33 kan schakelen.
- In de datasheet vinden we ook terug dat de hFE minimaal 750 is wat betekent dat we deze transistor voor deze collectorstroom van 2A volledig in geleding kunnen sturen met een basisstroom die 750 maal kleiner is. In dit geval is dit dus minimaal 2.66mA om deze transistor volledig in geleiding of saturatie te sturen. Onze PIC kan 20mA leveren per pin, dus deze 2.66mA ligt binnen de grenzen.
- Het sturen zelf willen we met een PIC microcontroller doen. Een uitgangspin van de uC kan ofwel 0 volt uitsturen – waarmee we de motor willen stoppen, of 5 volt uitsturen, waarmee we de motor willen laten draaien.
- In de datasheet vinden we ook nog terug dat de spanning tussen basis en emitter 2.5Volt is als de transistor in geleiding is. Dat betekent dat er over RB nog 2,5 volt blijft staan.
- We weten nu de spanning over en de stroom door RB en komen zo voor RB een maximale waarde van 939 Ohm uit. Uit de E12 reeks kiezen we de dichtstbijzijnde waarde van 820 Ohm, maar nog een stap kleiner naar 680 Ohm zou ook niet verkeerd zijn. We willen er namelijk zeker van zijn dat de transistor volledig in geleiding is.

Tot slot mag je ook zeker de vrijloopdiode over de motor niet vergeten. DC motors zijn inductieve belastingen. In de spoelen zit energie opgeslagen die er, nadat de transistor de stroom naar de motor heeft afgesloten, toch voor probeert te zorgen dat er stroom blijft vloeien . Deze diode zorgt ervoor dat deze stroom blijft ronddraaien in de motor en zo rustig uitdooft, en vermijdt dat deze stroom de transistor in sper stuk zou maken.

Testopstelling Darlington



Dit is de testopstelling met de Darlington. U ziet de DC motor, De BDX33B en als u goed kijkt ziet u ook de basisweerstand. De Darlington wordt aangestuurd met pin C0 van de uC.

Met de oscilloscoop meten we de spanning op pin C0 en de spanning over UCE.

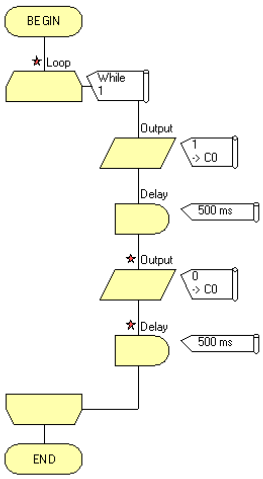
Programma 1Hz in C

Dit is het programma in C. De uC draait tegen 19.660800Hz via een Extern Xtal. Alle pins van poort C zijn output pins en pin C0 wordt afwisselend hoog en laag gemaakt met een delay van 500msec wat samen een frequentie van 1Hz maakt.

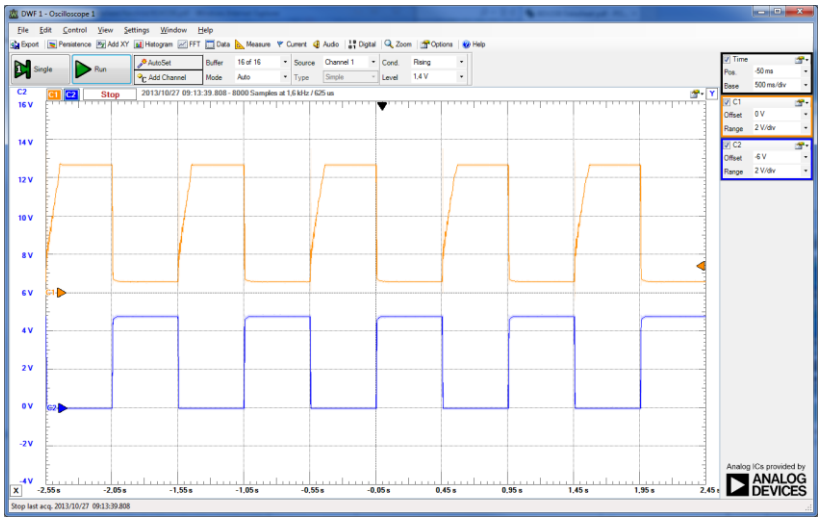
```
/*********************************************************************
Videolessen Deel 4 - Bart Huyskens - RTC Antwerpen - 2013
PIN 0 van PORTC afwisselend hoog en laag met freq van 1Hz
Extern Xtal - 19660800Hz
*********************************************************************/

#include <htc.h>
#define _XTAL_FREQ 19660800

__CONFIG(0x23E2);   //Extern HS Xtal, WDT DIS, PWRT EN, MCLR EN, BOR EN,
INT/EXT DIS, LVP DIS, NO Code/data protect
__CONFIG(0x3FFF);   //BOR 4,0V, Write protect off

void main()         // hoofdloop van het C programma
{
TRISC = 0x00;       // alle pins PORTC - Output

    while(1)
    {
    PORTC = 0b00000001;         // bit 0 port C
    __delay_ms(500);            //delay 500msec
    PORTC = 0b00000000;         // bit 0 port C
    __delay_ms(500);            //delay 500msec
    }
}
```

Programma 1Hz In Flowcode

Dit is het programma in Flowcode. De uC draait tegen 19.660800Hz via een Extern Xtal. Output Pin C0 wordt afwisselend hoog en laag gemaakt met een delay van 500msec wat samen een frequentie van 1Hz maakt.

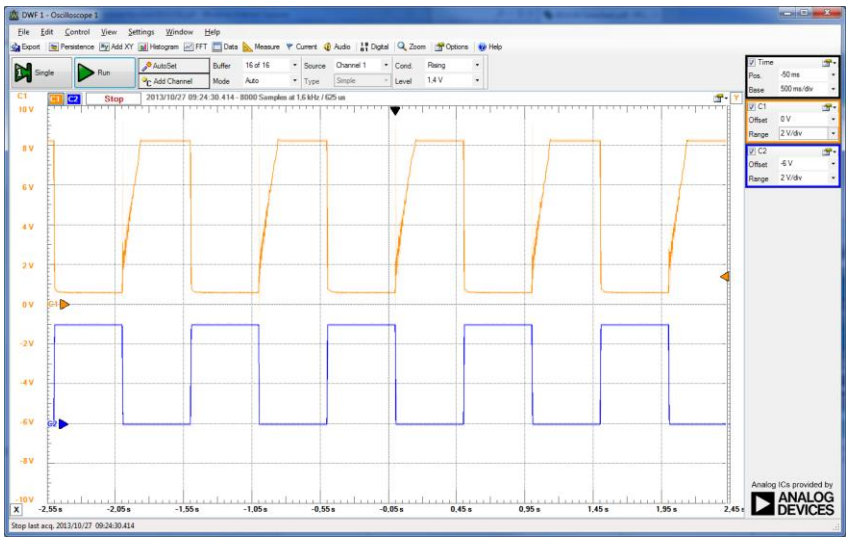


Controle of transistor volledig in geleiding gaat



Het oranje signaal van kanaal 1 is UCE over de transistor. Het blauwe signaal is het stuursignaal van de microcontroller. (ook hier zien we duidelijk het tragere uit als inschakelen van de transistor)

Als je deze schakeling opbouwt en test, dan is het heel belangrijk dat je – als de transistor in geleiding gestuurd wordt – en de motor dus draait, UCE sat nameet. Die moet beneden de 2.5V liggen zoals aangegeven in de datasheet van de BDX33B. Als die hoger is als 2,5Volt, dan staat de transistor niet volledig in geleiding en dan zal die opwarmen of stuk gaan. We zoomen dus best in op het scoopbeeld om UC sat na te meten.



We zien hier dat UCE sat ver beneden de 2.5Volt ligt – wat heel goed is – we meten hier zelfs maar 0,6 volt. De 2.5V waarover sprake in de datasheet was dan ook een maximum waarde.

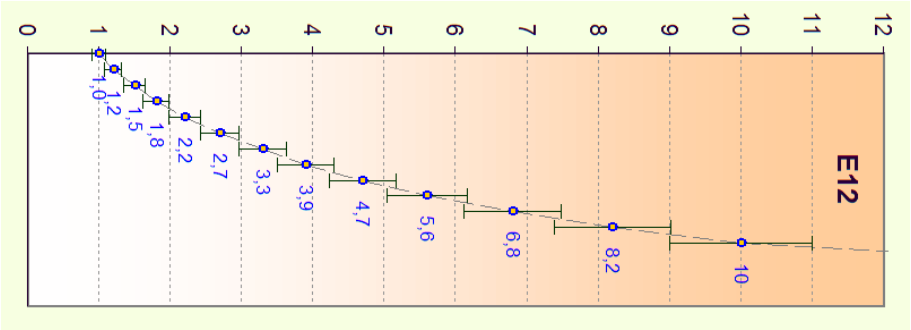
Stel dat UCE sat toch in de buurt – of zelfs groter dan de voorgeschreven 2.5V zou zijn, dan is er iets mis.

- Zijn je berekeningen correct?
- Meet Rb eens na – heeft die wel de waarde die je berekend hebt.
- Probeer eens om Rb te verkleinen en daarmee Ib te verhogen
- Als dit niet lukt, dan is er misschien iets mis met de transistor

Uitdagingen Darlington als schakelaar

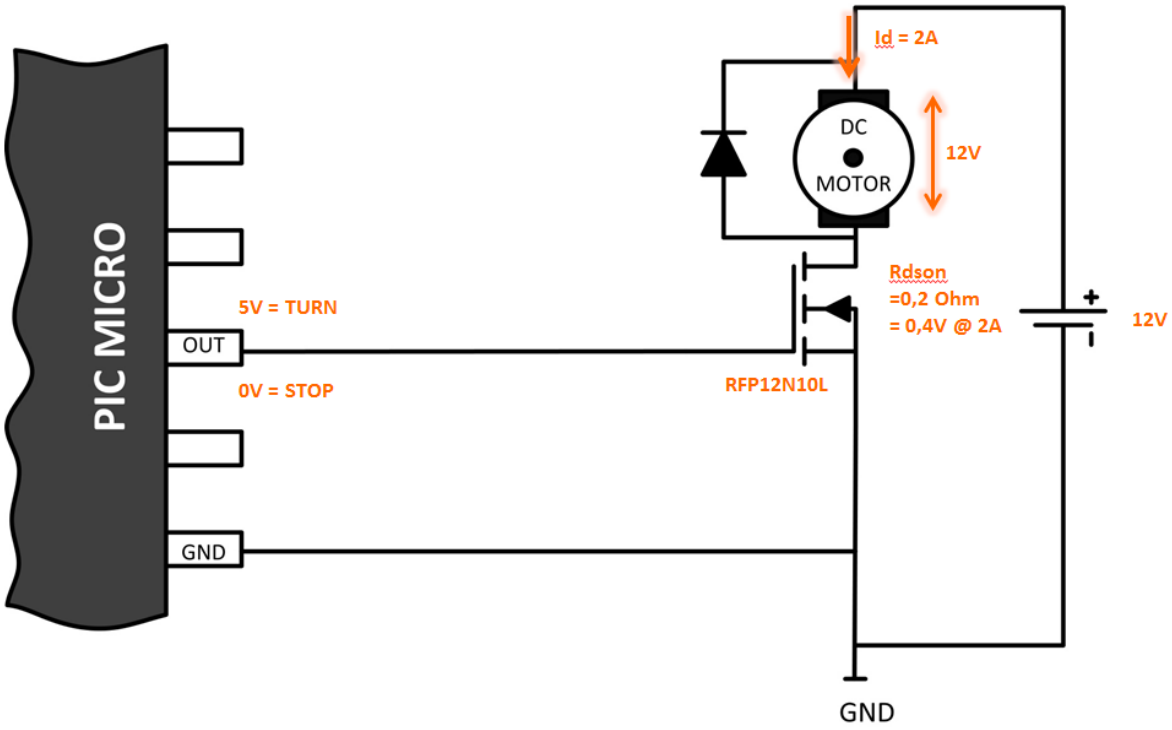
- Bouw de schakeling met een BDX33B op zoals in het schema is aangegeven. Vergeet zeker de vrijloopdiode (1N4001 is ok) niet. De uitgangspin die je gebruikt in pin C0.
- Gebruik een DC motor naar keuze, meet de belastingsstroom – zorg dat deze beneden de 3A blijft en herbereken Rb voor deze belastingsstroom.
- Schrijf een programma dat de motor start met een drukknop op B1 en stopt met de drukknop op B2.
- Meet UCE na als de motor niet draait – verklaar deze meting.
- Meet UCE na als de motor wel draait. Ligt deze meetwaarde beneden de 2,5V UCE SAT zoals die is aangegeven in de datasheet.
- Wat zou er mis kunnen zijn als deze UCE SAT niet beneden de 2,5 volt zou liggen.

Bijlage: E12 reeks weerstanden:



Mosfet als Schakelaar

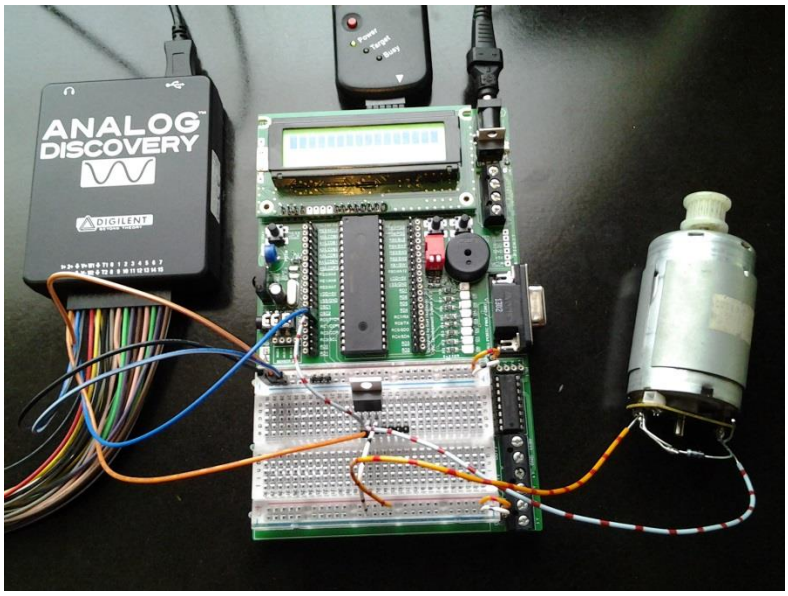
In ons tweede voorbeeld willen we demonstreren dat een mosfet een veel eenvoudiger component is om als schakelaar te gebruiken.



De mosfet die wij gebruiken is de RFP12N10L. 12 staat voor de 12A die deze mosfet kan schakelen. N staat voor N kanaals, wat betekent dat deze mosfet in geleiding kan gestuurd worden met een positieve spanning op de gate. 10 staat voor de 100Volt die deze mosfet kan schakelen tussen drain en source en L staat voor Logic. Hiermee wordt aangeduid dat 5 volt aan de gate voldoende is om de mosfet helemaal in geleiding te sturen, die we met TTL logica rechtstreeks kunnen leveren. De meeste power mosfets worden namelijk met spanningen – hoger als 5V – geschakeld.

- De datasheet van deze mosfet leert ons dat deze een RDSon heeft van 0.2 Ohm. Dat betekent dat de weerstand tussen drain en source slechts 0.2 Ohm bedraagt als de mosfet in geleiding staat. Bij 2A belastingsstroom veroorzaakt dat een spanningsval over de mosfet van slechts 0.4 Volt wat we hier dus kunnen verwaarlozen. Als voedingsspanning voor een 12V motor nemen we dus ook gewoon 12V.
- Voor mosfets moeten er geen basisweerstanden berekend worden. Een mosfet is een spanningsgestuurd component. Zet 5 volt op de gate en de mosfet gaat in geleiding – ongeacht welke belasting er aan hangt.
- Een mosfet trekt ook geen stroom uit de PIC, enkel een stuurspanning van 5V. Daarom is een mosfet ideaal om met een microcontroller aan te sturen.
- De vrijloopdiode beveiligt ook hier de Mosfet tegen gevaarlijke inductie- spanningen van de motor.
- Mosfets zijn nogal gevoelig voor statische elektriciteit en zwevende spanningen. Let er dus extra goed op dat de gnd van de voeding steeds aan de gnd van de microcontroller gekoppeld is. Dit is namelijk in vele gevallen destructief voor de mosfet.

Testopstelling Mosfet



Bij deze testopstelling herkent u de motor en de mosfet. De gate van de mosfet hangt rechtstreeks aan pin C0 van de uC.

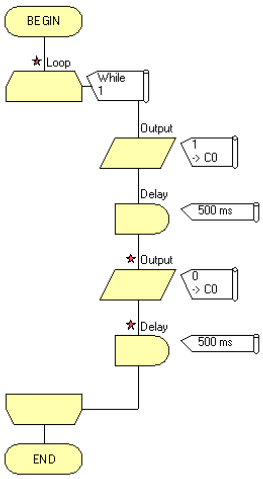
Programma 1Hz in C

Dit is het programma in C. De uC draait tegen 19.660800Hz via een Extern Xtal. Alle pins van poort C zijn output pins en pin C0 wordt afwisselend hoog en laag gemaakt met een delay van 500msec wat samen een frequentie van 1Hz maakt.

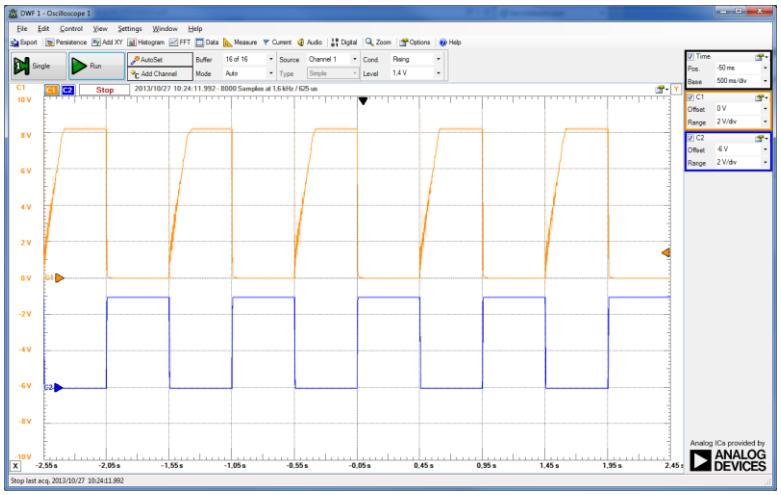
```
/*********************************************************************
Videolessen Deel 4 - Bart Huyskens - RTC Antwerpen - 2013
PIN 0 van PORTC afwisselend hoog en laag met freq van 1Hz
Extern Xtal - 19660800Hz
*********************************************************************/

#include <htc.h>
#define _XTAL_FREQ 19660800 // 4MHz

__CONFIG(0x23E2);   //Extern HS Xtal, WDT DIS, PWRT EN, MCLR EN, BOR EN,
INT/EXT DIS, LVP DIS, NO Code/data protect
__CONFIG(0x3FFF);   //BOR 4,0V, Write protect off

void main()         // hoofdloop van het C programma
{
TRISC = 0x00;       // alle pins PORTC - Output

    while(1)
    {
    PORTC = 0b00000001;         // bit 0 port C
    __delay_ms(500);            //delay 500msec
    PORTC = 0b00000000;         // bit 0 port C
    __delay_ms(500);            //delay 500msec
    }
}
```

Programma 1Hz in Flowcode

Dit is het programma in Flowcode. De uC draait tegen 19.660800Hz via een Extern Xtal. Output Pin C0 wordt afwisselend hoog en laag gemaakt met een delay van 500msec wat samen een frequentie van 1Hz maakt.



Controle of Mosfet volledig in saturatie gaat



Het oranje signaal van kanaal 1 is UDS over de Mosfet. Het blauwe signaal is het stuursignaal van de microcontroller.

Als je deze schakeling opbouwt en test, dan is het heel belangrijk dat je – als de transistor in geleiding gestuurd wordt – en de motor dus draait, UDSsat nameet. UDS kan je berekenen door De RDSon uit de datasheet te vermenigvuldigen met de stroom die je door de belasting stuurt.

RDSon = 0.2Ohm x ID = 2A = 0,4 volt

Deze 0,4 volt is een maximale waarde. De UDS sat spanning die we hier meten ligt hier onder. Voor zover wij hier kunnen zien ligt deze zelfs op 0 volt wat heel goed is.

Stel dat UDSsat toch in de buurt – of zelfs groter dan de berekende 0,4 V zou zijn, dan is er iets mis.

- Heb je een juiste mosfet – type RFP12N10L
- Meet de UGS spanning na – is deze 5 volt. Bij spanningen onder de 4 volt gaat deze mosfet niet meer zo goed in geleiding.
- Is de GND doorverbonden aan de GND van de mosfet

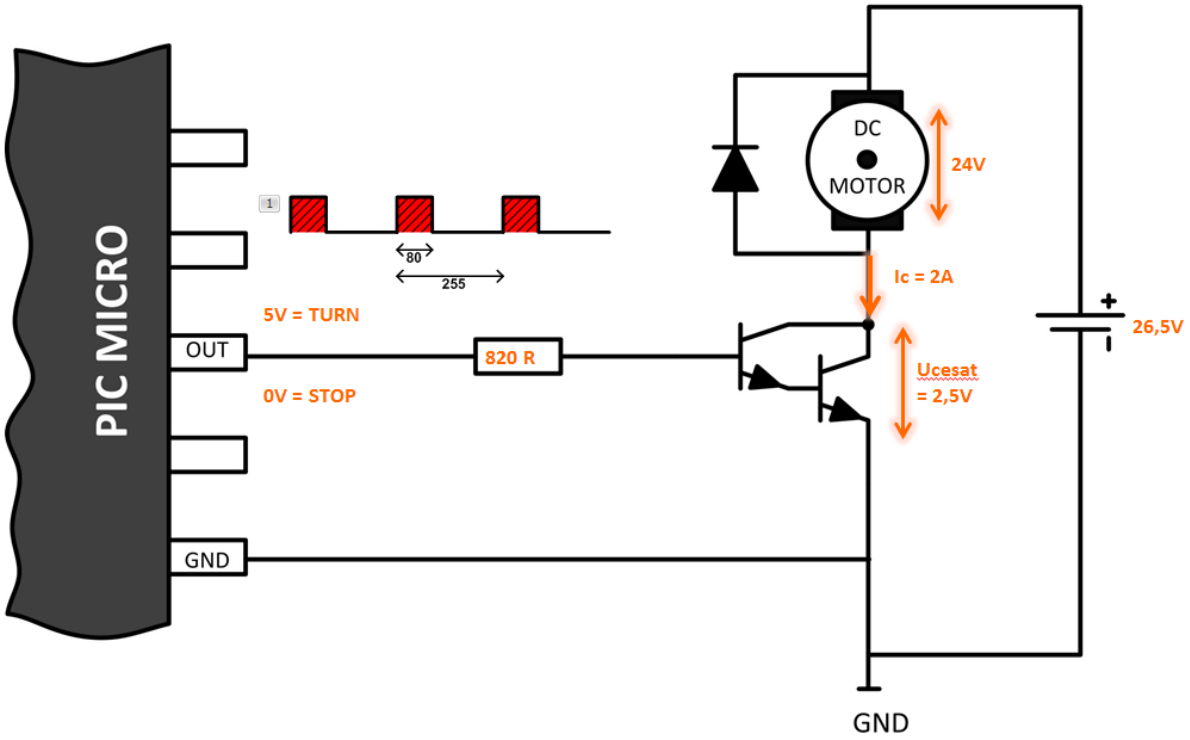
Uitdagingen Mosfet als Schakelaar

- Bouw de schakeling met een RFP12N10L op zoals in het schema is aangegeven. Vergeet zeker de vrijloopdiode niet. De uitgangspin die je gebruikt in pin C0.
- Gebruik een DC motor naar keuze, meet de belastingsstroom – zorg dat deze beneden de 3A blijft en herbereken Rb voor deze belastingsstroom.
- Schrijf een programma dat de motor start met een drukknop op B1 en stopt met dezelfde drukknop B1. (START-STOP met zelfde drukknop!!)
- Meet UDS na als de motor niet draait – verklaar deze meting.
- Meet UDS na als de motor wel draait. Bereken zelf de UDS a.d.h.v. de RDSon en de stroom die uw motor trekt. Is je meting gelijk aan je berekening?
- Wat zou er mis kunnen zijn als deze UDSon veel groter zou zijn als wat je berekend hebt met de RDS on.

Snel schakelen met Darlingtontransistor en Mosfet

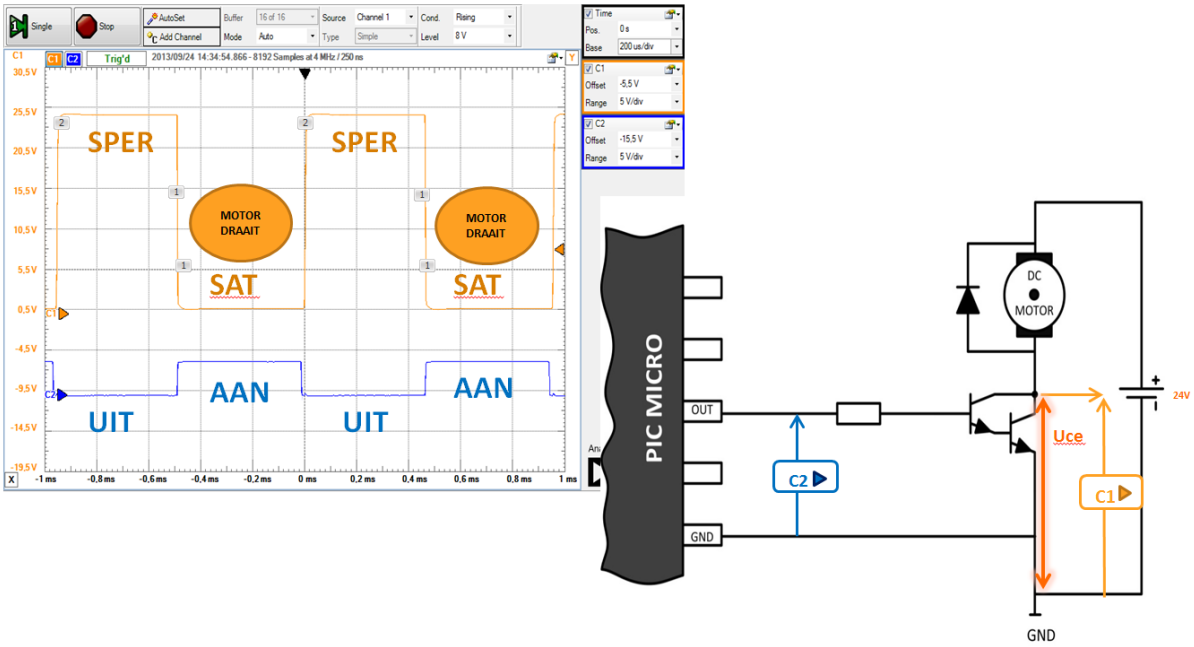
Om de snelheid van de DC motor te kunnen regelen, moeten we het vermogen naar de motor kunnen regelen. Hiervoor wordt meestal Puls Breedte Modulatie of PWM (Pulse Width Modulation) gebruikt, maar dat vereist dat de elektronische schakelaar enkele duizenden malen per seconde aan en uit kan schakelen. We onderzoeken in deze les even het aan en uitschakelgedrag van de BJT en de Mosfet.

Snel schakelen met Darlingtontransistor



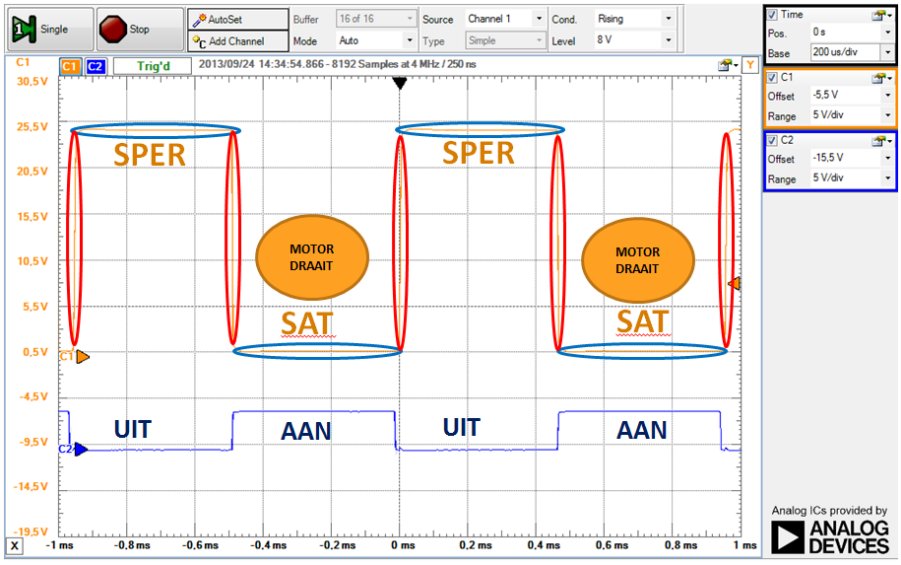
Als eerste nemen we hier het voorbeeld van de BJT, in dit geval een Darlingtontransistor BDX33B die we qua schakelvermogen kunnen vergelijken met onze Power mosfet – de RFP12N10L.

Met kanaal C2 van de oscilloscoop meten we de stuurspanning van 0 en 5 volt die uit de microcontroller komt. Met kanaal C1 meten we de spanning tussen collector en emitter van de Darlingtontransistor. Let op dat je deze meting goed interpreteert. Als de transistor in geleiding is, dan staat er over die transistor nog slechts een spanning die ongeveer en bij voorkeur kleiner is als de saturatiespanning van 2,5Volt in dit geval. In dit geval vloeit er een stroom door de motor en transistor en nu kan de motor draaien. Als de transistor geen stuursignaal krijgt, dan spert deze en dan staat de volledige voedingsspanning van 14,5 volt over de transistor. De sperrende transistor is immers te vergelijken met een open schakelaar.



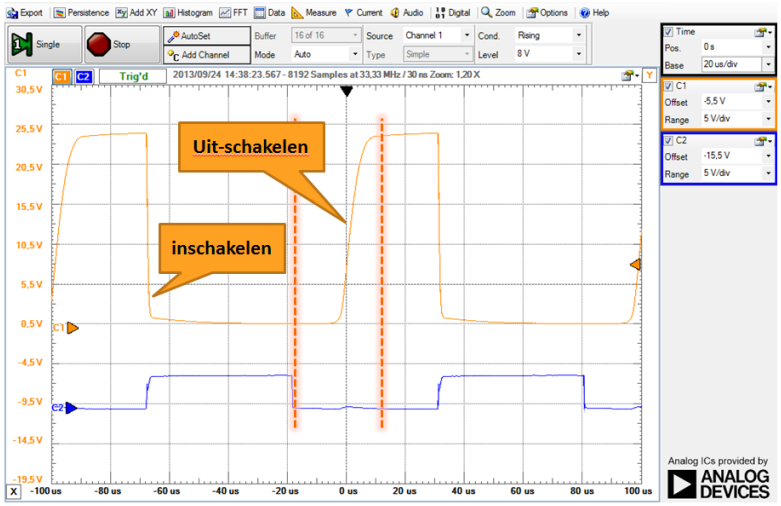
Op dit scoopsignaal zien we dit ook duidelijk. Het blauwe C2 signaal stelt de stuurspanning van de uC voor. Het is duidelijk te zien wanneer de uC de transistor uit, aan, terug uit en weer terug aan stuurt.

Als de transistor uit wordt gestuurd, dan gaat deze in sper. Dit is mooi te zien op het oranje signaal, want de volledige voedingsspanning staat nu over de transistor. Als de transistor wordt “aan”-gestuurd, dan gaat de transistor in geleiding of in saturatie. Op dit moment staat er nog slechts de saturatiespanning over de transistor. De stroom kan op dit moment door de transistor vloeien en de motor draait. Laat je dus niet in de war brengen door deze scoopbeelden – je moet hier op redeneren.

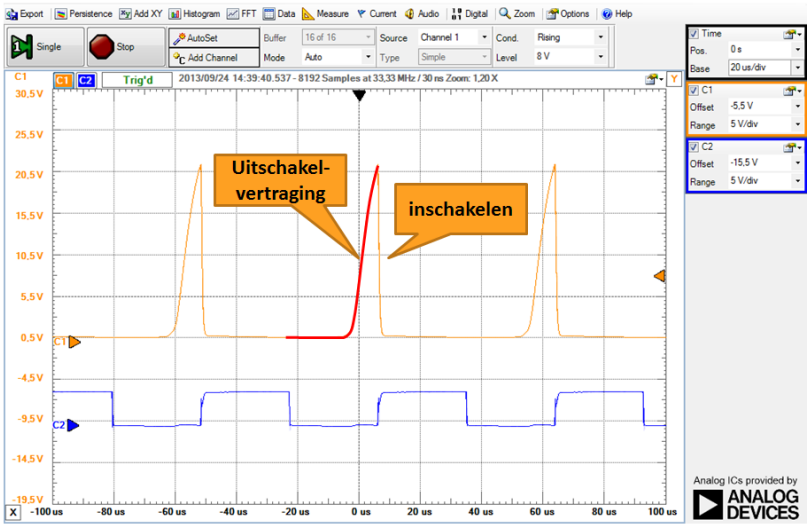


Op dit scoopbeeld – waarbij we de transistor aan en “uit”-schakelen met een frequentie van ongeveer 1Khz, zien we mooi dat wanneer het stuursignaal hoog wordt, de transistor nagenoeg meteen in geleiding is – de spanning over Uce wordt meteen laag, en wanneer het stuursignaal laag wordt, de transistor meteen weer terug in sper gaat.

In een transistor die volledig in geleiding, of volledig in sper is, wordt nagenoeg geen warmte gedissipeerd, enkel op de overgangsmomenten tussen sper en geleiding wordt er in principe warmte gedissipeerd, maar die momenten zijn op deze lage schakelfrequentie te verwaarlozen.



Bij amper 10 KHz (zie boven) zien we bij de darlingtonstransistor al een heel ander verschijnsel. Het inschakelen gebeurt nog steeds zo goed als meteen. U ziet dit hier duidelijk. Maar het duurt een hele tijd nadat het stuursignaal van de microcontroller laag geworden is, voordat de transistor begint te reageren en uiteindelijk ook helemaal spert. De warmte die gedurende dit trage uit schakelen gedissipeerd wordt is nu niet meer te verwaarlozen.

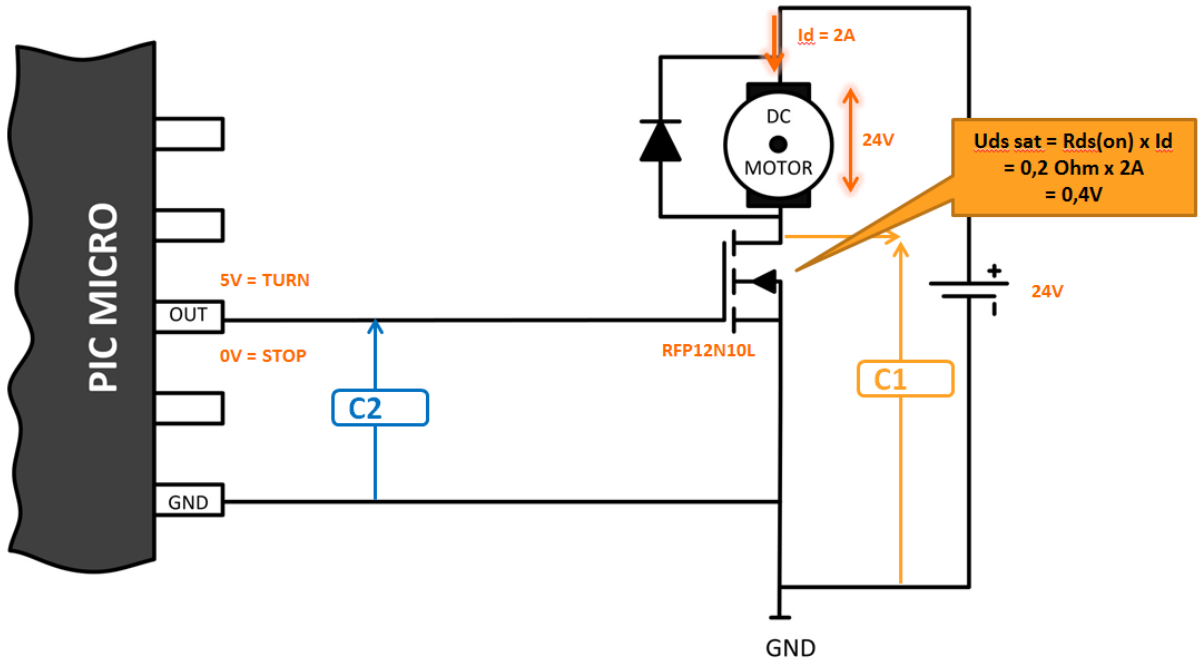


Bij 17 KHz gaat de uitschakelvertraging zo'n groot deel uitmaken dat de transistor nog amper een fractie uitgeschakeld is voordat die weer terug in geleiding gezet wordt. Nog steeds gebeurt het in geleiding gaan nagenoeg direct.

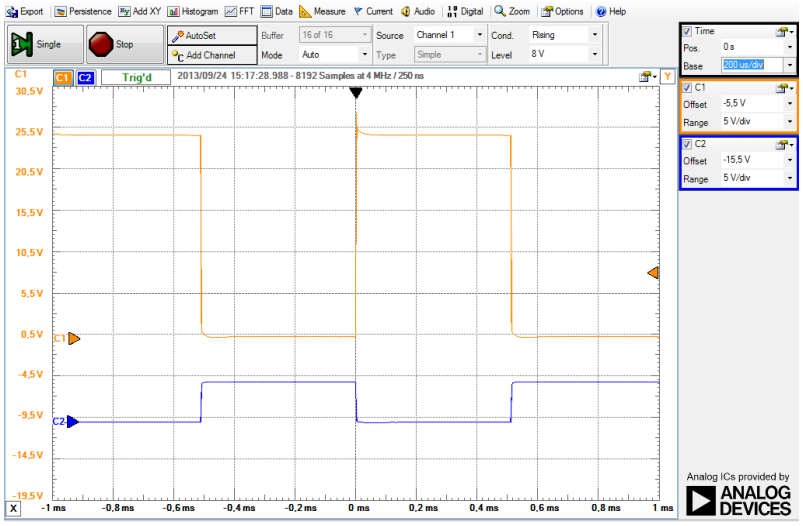
En bij 30 KHz is de uitschakelvertraging zo groot in verhouding tot de schakelperiode dat de transistor de hele tijd in saturatie blijft – de transistor schakelt hier niet meer uit en de motor draait constant.

Klassieke BJT's hebben dus vooral een slecht uitschakelgedrag en dat begint bij hogere frequenties zeker een rol te spelen. Dat uitschakelgedrag kan verbeterd worden door kortstondig een negatieve stuurspanning i.p.v. een 0V signaal aan te leggen, maar dat zou ons hier te ver brengen.

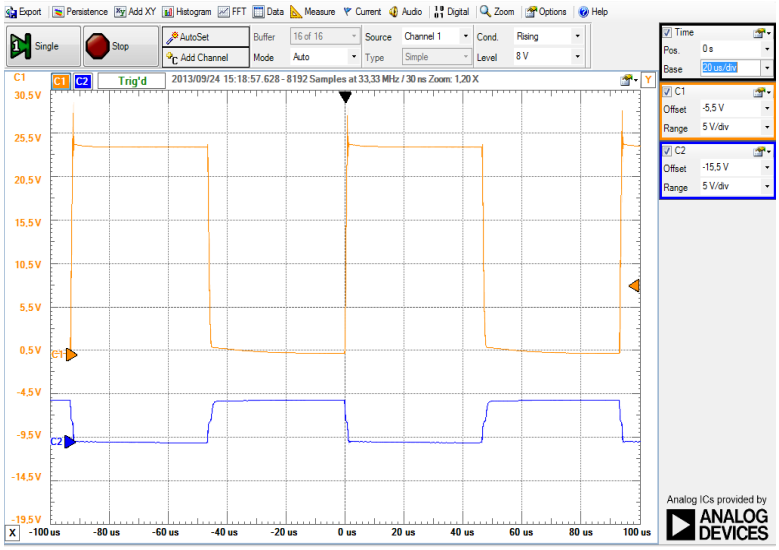
Snel Schakelen met Mosfet



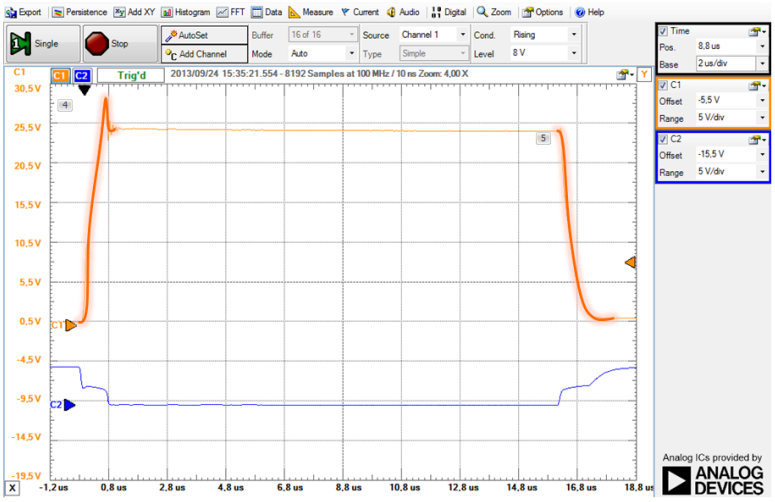
Ook hier meten we met kanaal C1 de spanning over de mosfet en met kanaal C2 - in het blauw - de stuurspanning van de microcontroller. Het is een N kanaals mosfet, dus een positief stuursignaal zal de mosfet in geleiding zetten en een 0 volt signaal zal de mosfet laten sperren. Over een mosfet die in geleiding staat – staat enkel een spanning die kan berekend worden door de stroom door de mosfet te vermenigvuldigen met de Rdson waarde uit de datasheet. In ons geval is dit 2A x 0.2Ohm – dat maakt 0,4 volt. Over een mosfet in sper staat de volledige voedingsspanning.



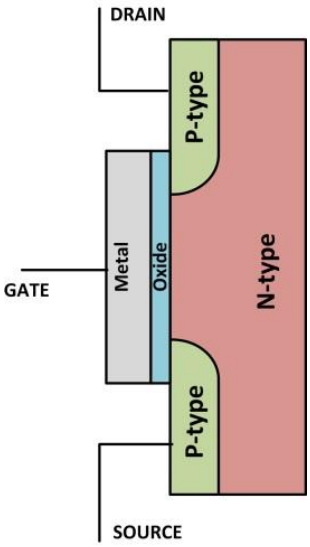
Bij deze eerste meting – opgemeten bij 1Khz – zien we geen probleem. De mosfet volgt mooi het stuursignaal voor zover wij hier kunnen zien.

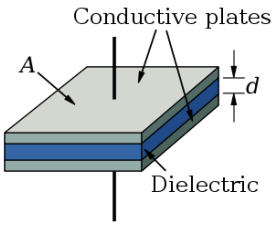


Bij 10Khz zien we toch al een beetje dat de flanken wat minder steil worden, maar dit is nog steeds enorm veel beter dan de Darlingtonstransistor tegen 10KHz. Wat ook begint op te vallen is dat de flanken van het stuursignaal van de uC minder steil beginnen worden.



Dit fenomeen zien we nog veel duidelijker als we tegen 30Khz beginnen schakelen. Let wel op – bij de klassieke transistor was het bij 30Khz al lang gedaan met schakelen.

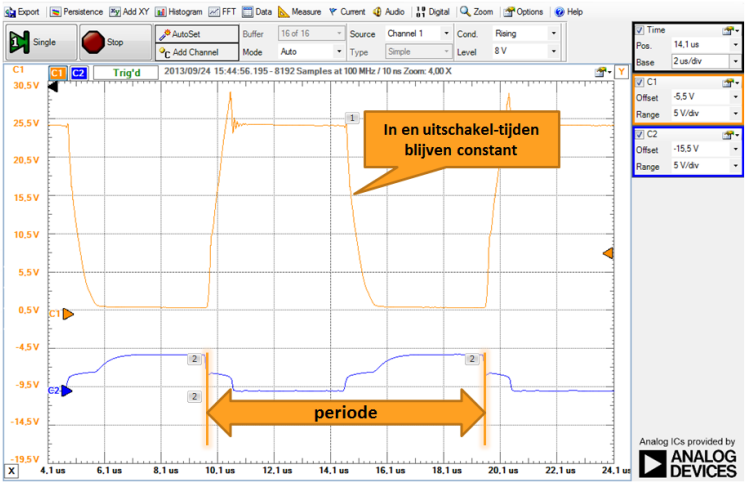




De inschakel en uitschakelvertraging is vooral te wijten aan de opbouw van een MOSFET. Tussen de geleidende metalen plaat aan de gate en de geleidende delen P en N silicium zit een niet geleidend oxidelaagje. Twee geleidende platen en een niet geleidende middenstof vormen samen een condensator en het is het opladen en ontladen van deze parasitaire gate capacitor dat we hier ook in het stuursignaal terugzien. De waarde van deze 'input capacitance' vinden we ook in de datasheet terug.

Input Capacitance	C_{ISS}	V_{GS} = 0V, V_{DS} = 25V, f = 1MHz (Figure 8)	-	-	900	pF
Output Capacitance	C_{OSS}		-	-	325	pF

Het is duidelijk dat in en uitschakelen niet oneindig snel gebeurt en vermits er vooral vermogen gedissipeerd wordt tijdens dit schakelen, zal deze mosfet hier opwarmen. In dit voorbeeld werd de mosfet bijvoorbeeld 40°C.



Als we nu de mosfet testen om te schakelen tegen 100 KHz, dan zijn de in en uitschakelvertragingen niet groter geworden, alleen is de tijd van in en uitschakelen t.o.v. de totale periode nu wel groter geworden, wat zal resulteren in een hogere temperatuur. We meten hier al 70°C. Dit komt al in de buurt van de maximale 150°C.

De in en uitschakelvertragingen van een MOSFET worden meestal in de datasheet vermeld. De turn on delay time is de tijd, gemeten van de start van de flank van het stuursignaal, tot het moment dat de spanning over de mosfet tot 90% gedaald is. De daaltijd van 90% tot 10% is de Rise time. Beide tijden geven aan hoe snel de mosfet in geleiding kan komen. Het terug sperren wordt ook getypeerd met twee tijden – de turn off delay time en de Fall

Time. Als we deze gegevens bekijken dan zouden we kunnen stellen dat ook deze mosfet trager uitschakelt dan inschakelt, maar die 80nsec zijn eigenlijk te verwaarlozen.

Met deze 4 tijden kunnen we wel de maximale schakelfrequentie van deze mosfet bepalen. Zo komen we – als we telkens de langste tijden nemen uit de datasheet – op een totale periode van 480ns wat resulteert in een maximale schakelfrequentie van juist iets meer als 2MHz.

Turn-On Delay Time	$t_{d(ON)}$	I_D = 6A, V_{DD} = 50V, R_G = 6.25Ω, V_{GS} = 5V (Figures 9, 10, 11)	-	15	50	ns
Rise Time	t_r		-	70	150	ns
Turn-Off Delay Time	$t_{d(OFF)}$		-	100	130	ns
Fall Time	t_f		-	80	150	ns

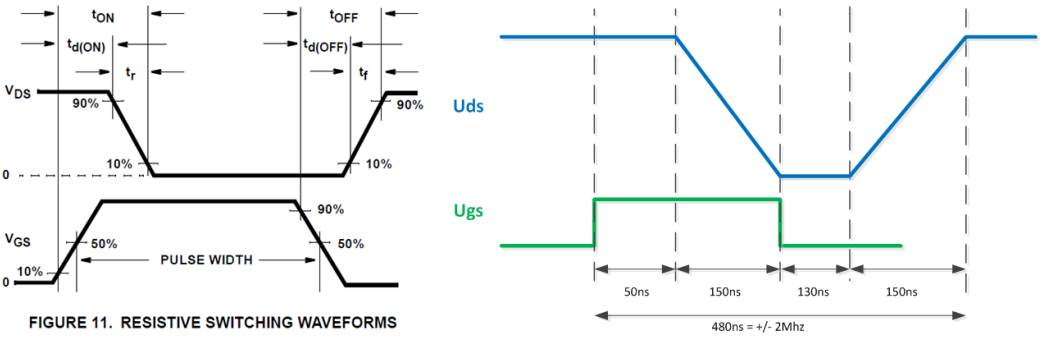


FIGURE 11. RESISTIVE SWITCHING WAVEFORMS

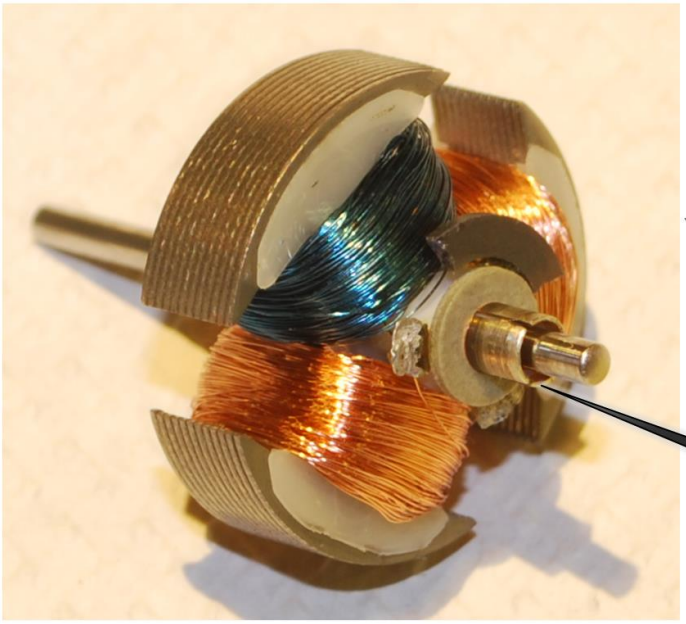
We mogen dus besluiten dat mosfets sneller kunnen schakelen als klassieke BJT, maar dat ook bij mosfets een hogere schakelsnelheid een hogere vermogendissipatie, temperatuur en dus schakelverlies tot gevolg zal hebben. Een zeer belangrijke reden dus om de schakelsnelheid – ook bij PWM – zo laag mogelijk te houden.

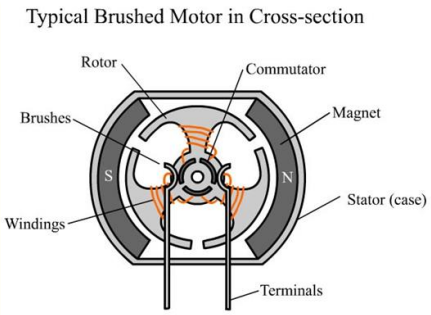
Uitdagingen Snel Schakelen met Darlington en Mosfet

Labo: Vergelijk het schakelgedrag van de BDX33B met dat van de RFP12N10L mosfet.

- Als belasting neem je een motor of een lamp of een vermogen weerstand die meer dan 1A trekt. De voedingsspanning mag je zelf kiezen (<50Volt DC!!)
- Vergeet niet om de GND van de elektronische schakelaar aan de GND van de uC te hangen.
- Vergeet de vrijloopdiode niet als je een inductieve belasting schakelt.
- Je stuurt de elektronische schakelaars aan en uit met een bepaalde frequentie via een eenvoudig programma in je uC – sluit deze aan op poort C0
- Meet de stuurspanning die van de uC komt en de spanning over de elektronische schakelaar.
- Verhoog telkens de frequentie waarmee je schakelt (door je software aan te passen) en neem een 6-tal zinvolle scoopbeelden binnen (3 van de Darlington en 3 van de Mosfet)
- Verklaar bij elk scoopbeeld duidelijk wat je ziet en wat er gebeurt, eventueel wat de reden is en verwijs waar mogelijk naar gegevens uit de datasheet.
- Maak een duidelijk laboverslag met schema's, berekeningen, programma, scoopbeelden, verklaringen, relevante screenshots uit je datasheet, …

Tegen-EMK en Vrijloopdiode



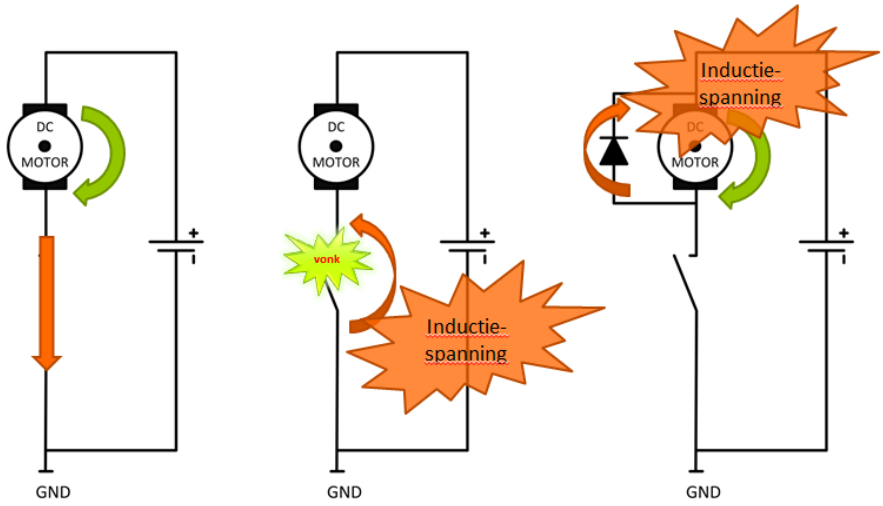


Zoals we eerder al hebben besproken werkt een DC motor als volgt. De rotor bestaat uit één of meerdere spoelen waar we via de borstels en de commutator een stroom door laten vloeien. Die stroom veroorzaakt een magnetisch veld. Dit magnetisch veld wordt afwisselend aangetrokken en afgestoten door het magnetisch veld van de stator en zo begint de motor te bewegen. De stroomzin in de rotorspoelen wordt door de commutator steeds omgedraaid zodat we een draaiende beweging krijgen.

Als we stoppen met stroom door de spoel van de rotor te sturen, dan worden er twee spanningen opgewekt:

De eerste is de gegenereerde spanning. De DC motor zal zich nog even als generator gedragen. De rotorspoel draait namelijk nog steeds en beweegt zich in het magnetisch veld van de statormagneten en dit veroorzaakt een stroom en spanning in de rotorspoel. Dit wordt de gegenereerde spanning genoemd maar deze spanning is altijd een beetje lager dan de oorspronkelijke bronspanning en daardoor dus vrij ongevaarlijk en deze spanning dooft ook vrij snel uit omdat het toerental ook snel vermindert.

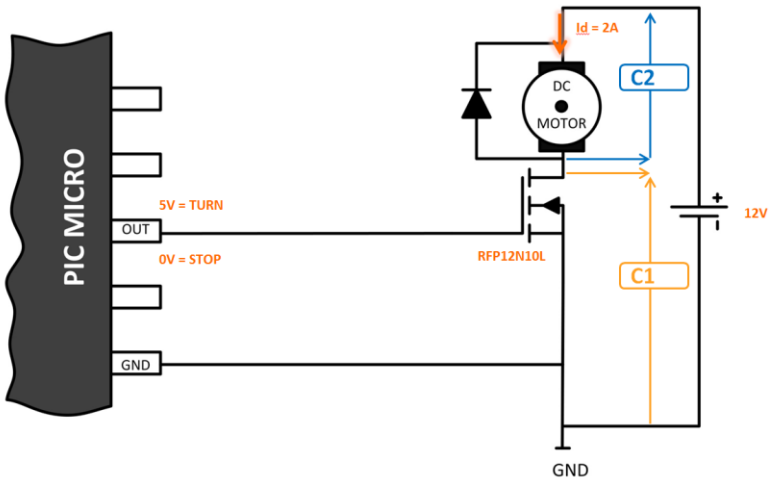
De tweede is de geïnduceerde spanning. Een motor bestaat uit spoelen. Spoelen hebben de eigenschap dat ze , na het uitschakelen van de bron, toch nog stroom willen laten vloeien in de kring. Deze inductiestroom kan zo een zeer hoge geïnduceerde spanning veroorzaken. Tot honderden – soms duizenden volt zodat er zelfs vonken kunnen ontstaan over open schakelaars. Als de schakelaar een elektronische schakelaar is zoals een transistor – dan kan die stuk gaan door deze hoge inductiespanning.



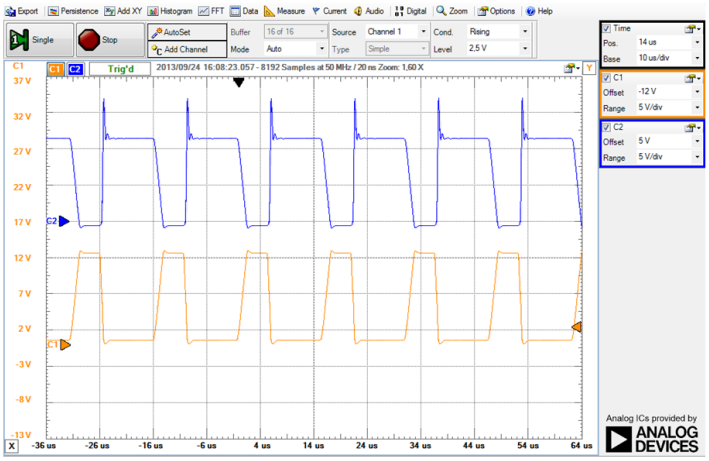
In dit schema zien we een DC motor die ronddraait omdat de elektrische stroomkring gesloten is.

Als we de schakelaar openen, dan word er in de spoelen van de motor een inductiespanning gegenereerd. Deze elektrische energie veroorzaakt over de schakelaar een zodanige grote spanning dat er zelfs vonken kunnen ontstaan. Als we de schakelaar zouden vervangen door een transistor of mosfet, dan zou deze opgewekte inductiespanning deze transistor of mosfet kunnen vernietigen. Een inductiespanning kan al snel oplopen tot enkele 100-en of zelfs 100-en volts en daar kunnen onze componenten niet tegen

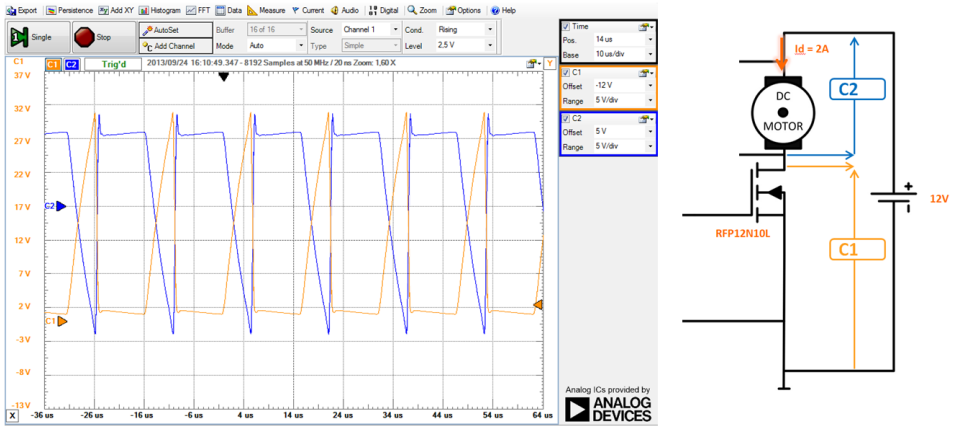
De eenvoudigste oplossing is om over de motor een diode te plaatsen in antiparallel. Op de momenten dat de elektronische schakelaar de stroom spert en dat er in de motor dus een inductiespanning wordt opgewekt, ontstaat er een kleine stroomkring met enkel de diode en de motor in de kring. De inductiestroom vloeit via de diode terug de motor in, wordt omgezet in warmte en dooft zo vrij snel uit zonder de mosfet of transistor te beschadigen.



Mosfets en Transistors worden met een vrijloopdiode dus op een goedkope en eenvoudige manier beveiligd tegen de inductiespanningen die destructief zouden kunnen zijn.



Op dit scoopbeeld hebben we zowel de spanning over de motor – in het blauw – als de spanning over de transistor – in het oranje – gemeten. Let op, om de spanning over de motor te kunnen meten heb je een differentiële probe nodig omdat deze spanning niet t.o.v. de massa gemeten is. We zien hier mooi dat wanneer de transistor in geleiding is, de volledige voedingsspanning over de motor staat, en dat wanneer de transistor spert, de voedingsspanning volledig over de transistor staat. Op dit moment gaat de spanning over de motor even een beetje negatief. Dit is de spanning die we meten over de vrijloopdiode. De mosfet heeft in dit geval, door de vrijloopdiode, geen enkele last van de inductiespanning.



Voor deze metingen werd de vrijloopdiode weggelaten. U ziet dat de inductiespanning over de transistor in sper nu spanningen veroorzaakt die meer dan twee maal hoger zijn als de voedingsspanning. Vermits onze mosfet 100Volt kan schakelen is deze inductiespanning in onze situatie niet destructief. De blauwe grafieken verduidelijken mooi dat de inductiespanning wel degelijk van de motor afkomstig is.

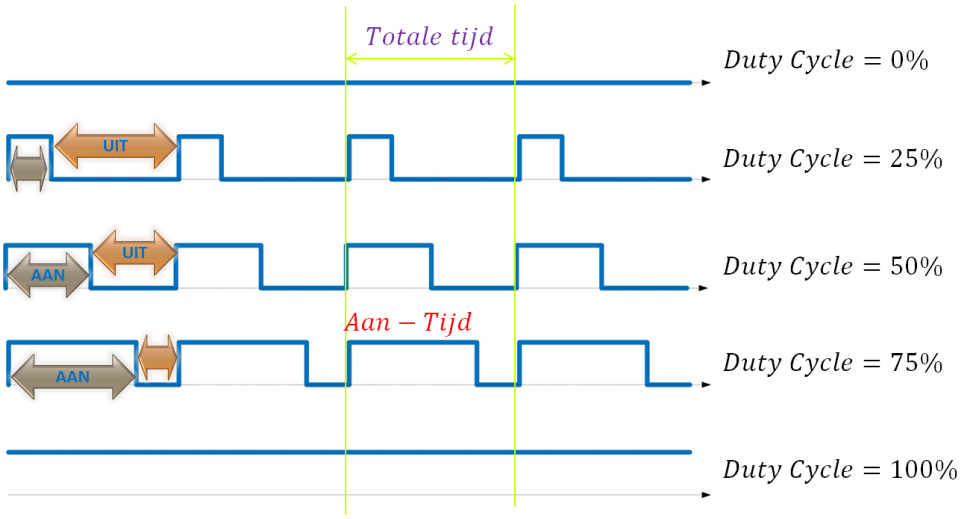
We kunnen dus besluiten dat we bij het schakelen van inductieve belastingen steeds een vrijloopdiode als beveiliging zullen moeten gebruiken.

Uitdagingen bij Tegen-EMK en Vrijloopdiode

1. Geef 3 voorbeelden van resistieve of zuiver ohmse belastingen.
2. Geef 3 voorbeelden van inductieve belastingen
3. Geef een voorbeeld van een capacitieve belasting.

PWM – Pulse Width Modulation

In deze les bespreken we stap voor stap wat PWM is en hoe we een PWM signaal kunnen genereren met onze microcontroller.



$$Duty\ Cycle = \frac{Aan - Tijd}{Totale\ tijd}$$

Laten we dus eerst maar eens beginnen met uit te leggen wat PWM juist is. PWM is de afkorting van Pulse Width Modulation. Dit is dus een gemoduleerd signaal zoals u hier zien met een bepaalde aan tijd en een bepaalde uit tijd. De aan tijd kan ook korter zijn zoals bij dit PWM signaal, maar dan zal de uit tijd langer worden. De aan tijd kan ook langer worden, maar dan wordt de uit tijd weer korter. De twee uitersten zijn een signaal dat constant 0 Volt is en een signaal dat contant hoog is.

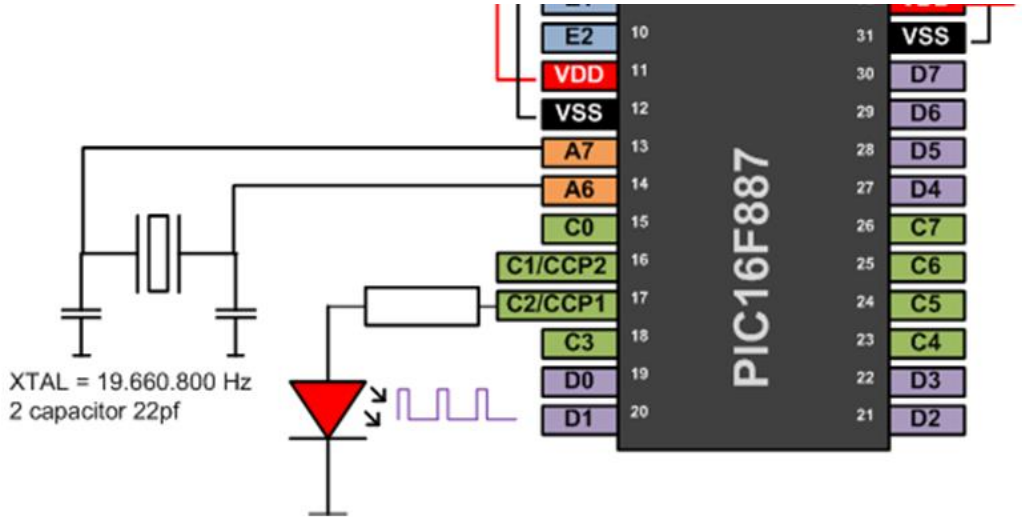
Een PWM signaal wordt altijd omschreven met een duty cycle. De dutycycle is de verhouding van de aan-tijd op de totale tijd of de periode. De aan tijd is de tijd dat het signaal hoog is en de totale tijd is de periode van het signaal. De periode of frequentie van een PWM signaal blijft constant – enkel de duty cycle verandert. Het eerste signaal heeft een duty cycle van 0%, het tweede 25%, dan 50 en 75% en het laatste signaal tenslotte heeft een Duty cycle van 100%.

PWM wordt veelvuldig gebruikt om lichten te dimmen – zowel klassieke verlichting als led verlichting en kleurmenging met RGB leds. PWM wordt ook toegepast om het toerental van DC motoren te regelen, in geschakelde voedingen of om klasse D eindtrappen van audioversterkers aan te sturen.

Voor het dimmen van lichten is een PWM frequentie van 120Hz voldoende, voor het regelen van toerentallen worden frequenties tussen 1 en 30 KHz gebruikt en voor geschakelde voedingen en audioversterkers schakelen we al snel tegen 100Khz.

Er bestaat geen vuistregel om te bepalen wat de ideale PWM schakelfrequentie is als het gaat om het aansturen van DC motoren. Er zijn wel een aantal richtlijnen:

- Niet te snel, want hoe sneller je een mosfet laat schakelen, hoe warmer de mosfet wordt – dat hebben we gezien in één van de vorige videolessen.
- Niet te traag – de motor moet zonder schokken blijven draaien – anders zouden er tijdens elke periode ook terug opstart-piekstromen optreden.
- Soms wordt er gesteld dat je moet schakelen met een frequentie boven de 17kHz, boven het gehoor gebied van mensen om geen vervelende pieptonen te horen in de motor, maar zeker niet iedereen houdt zich hieraan. Test dit gerust uit – in een aantal gevallen kan je de schakelfrequentie horen in de motor.



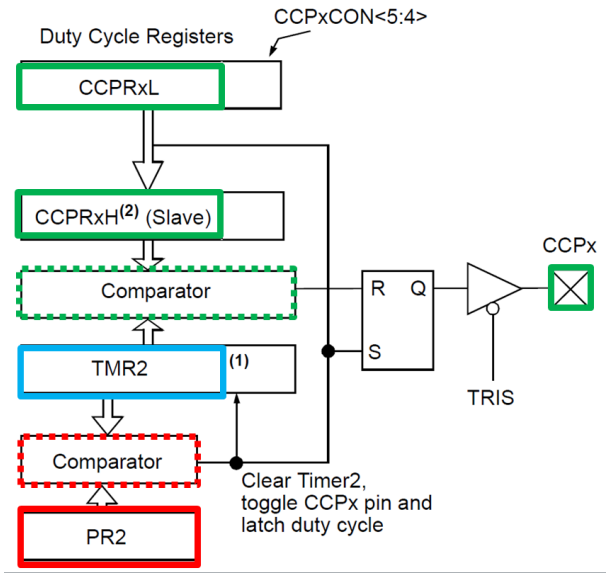
Met onze PIC16F887 kunnen we twee PWM signalen genereren. Hiervoor gebruiken we de CCP1 en CCP2 of Capture-Compare-PWM modules die in onze PIC zitten. De PWM signalen kunnen enkel gegenereerd worden op pins C2 die gekoppeld is aan CCP1 en pin C1 die gekoppeld is aan CCP2.

De CCP1 module heeft een Capture mode en een Compare mode die beiden een samenwerking hebben met 16 bit Timer1 en een enhanced PWM mode die samenwerkt met 8 bit Timer 2. De CCP2 module is identiek aan de CCP1 module, enkel CCP2 heeft geen enhanced PWM mode. Een enhanced PWM mode is een handige extra feature die kan gebruikt worden om H-bruggen aan te sturen, maar wij gaan deze in ons voorbeeld niet gebruiken.

Wij bespreken hier enkel de PWM mode van de CCP1 en CCP2 modules.

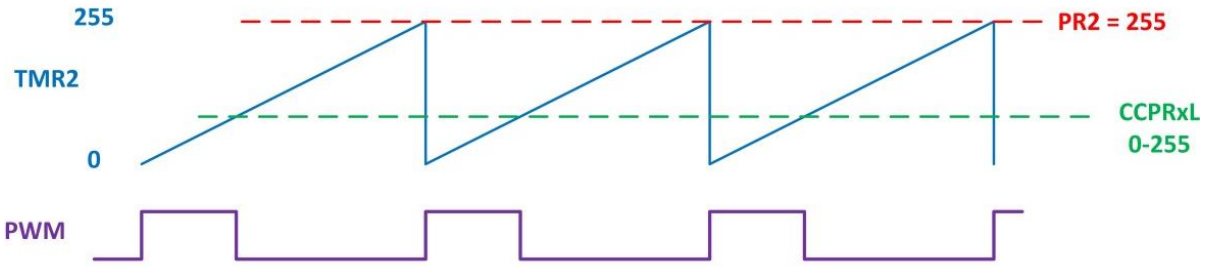
In deze PWM mode zien we dat de CCP pin als uitgang geschakeld is – hierop komt het PWM signaal te staan. Dit PWM signaal kan er zo uit zien.

Deze PWM module kan zowel in 10bit mode als in 8 bit mode werken. Wij werken in 8 bit mode.



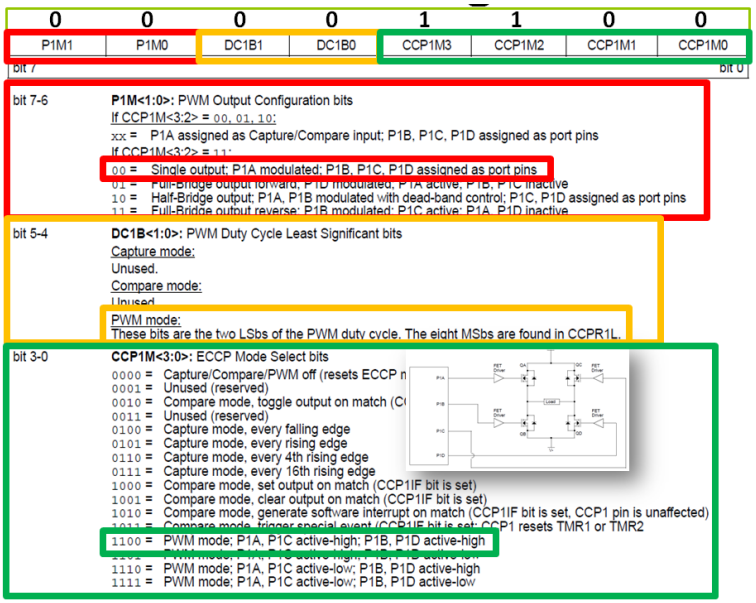
Het centrale deel van de PWM module is Timer 2 met daar rond 2 comparators. Timer 2 is ingesteld om altijd te blijven tellen van 0 tot 255, terug te starten bij 0 en zo te blijven ronddraaien. Via de prescaler van TMR2 kan je de snelheid en dus de frequentie van het PWM signaal bepalen. Je kunt de klok met de prescaler delen door 1, door 4 en door 16. Héél veel keuze om een bepaalde PWM frequentie te kiezen heb je dus niet.

Door in het PR2 register 255 te laden, laten we TMR2 telkens volledig tot 255 tellen alvorens het PWM signaal terug hoog wordt gemaakt. Als TMR2 gelijk wordt met PR2 zal de rode comparator de CCP pin hoog maken en zal TMR2 terug met 0 geladen worden. Door dit PR2 register kleiner als 255 te maken, forceren we de comparator om de CCP pin al vroeger hoog te maken en TMR2 al terug op 0 te zetten alvorens 255 bereikt is. Hiermee kunnen we eveneens de PWM frequentie verhogen, maar dat gaat wel ten koste van de resolutie. Wij houden bij voorkeur het PR2 register op 255 voor de hoogste resolutie.



In het CCPRxL register laden we de duty cycle – een getal van 0 tot 255. Dit getal wordt automatisch naar het CCPRxH register gekopieerd op het moment TMR2 gelijk is aan PR2. Als TMR2 dan uiteindelijk de waarde in het CCPRxH register bereikt heeft zal de groene comparator de CCPx pin resetten. Zo kunnen we met de waarde in het CCPRxL register de duty cycle laten variëren tussen 0 en 255 of 0 en 100%.

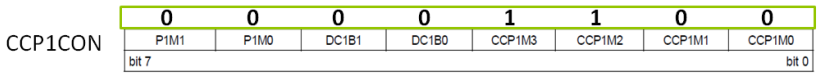
Registers:



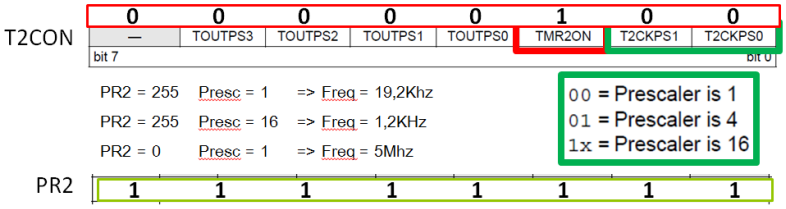
We bekijken even de registers die nodig zijn om een PWM signaal te genereren op pin CCP1. In de datasheet staat dit op deze manier weergegeven. De 8 bits van het CCP1CON register zijn allemaal van het read write type en na een totale reset zijn ze allemaal 0. We zullen dit register in ons programma laden met 0b00001100. CCP1 heeft als extra de enhanced PWM mode die het aansturen van H-bruggen sterk vereenvoudigt. Wij willen hier in ons programma geen gebruik van maken en schakelen de 3 extra pins als gewone IO pins.

De volgend twee bits zijn in PWM mode de twee LSB's om het Duty cycle register in 10 bit mode te laten werken. Wij werken in 8 bit mode dus deze bits mogen 0 zijn.

De volgende 4 bits bepalen of we in capture – compare of PWM mode werken. Voor de PWM mode hebben we door de Enhanced H-brug mode nog eens 4 keuzes welke voor ons programma alle 4 goed zouden zijn. Wij kiezen de eerste.



Periode / Frequentie



Duty Cycle

De periode of frequentie van ons PWM signaal wordt bepaald door de prescaler in TMR2 en door het PR2 register. Wij zetten hier de prescaler in het T2CON register op 00 en vergeten ook zeker niet om TMR2 aan te schakelen via de TMR2ON bit.

Het PR2 register laden we met 255 – hoe groter deze waarde – hoe lager de frequentie van het PWM signaal. In onze situatie – met PR2 op 255, de prescaler op 16 en een Xtal van 19660800 zal dit resulteren in een PWM frequentie van 1.2 KHz

De duty cycle van ons PWM signaal wordt tenslotte bepaald door de waarde die we in het CCPR1L register laden. Dit is voor ons voorbeeld een waarde tussen 0 en 255.

Programma PWM In C

We bekijken even ons programma met de klassieke config regels.

```
/*********************************************************************
Videolessen Deel 4 - Bart Huyskens - RTC Antwerpen - 2013
PWM op PIN C2-CCP1 van PORTC
Extern Xtal - 19660800Hz
*********************************************************************/

#include <htc.h>
#define _XTAL_FREQ 19660800

__CONFIG(0x23E2);   //Extern HS Xtal, WDT DIS, PWRT EN, MCLR EN, BOR EN,
INT/EXT DIS, LVP DIS, NO Code/data protect
__CONFIG(0x3FFF);   //BOR 4,0V, Write protect off

void main()
{
unsigned char Duty_Cycle = 0;       // variabele voor Duty Cycle
TRISC   = 0x00;                     // leds aan PORTC - Output
PORTC   = 0x00;                     // alle pins PORTC laag maken
CCP1CON = 0b00001100;               //CCP1 in PWM mode zetten
T2CON   = 0b00000110;               // TMR2 ON - presc op 16
PR2     = 255;                      // periode register op 255
while (1)
{
CCPR1L = Duty_Cycle;                // copy var Duty_Cycle naar CCPR1L
Duty_Cycle++;                       // increment Duty_Cycle
__delay_ms(20);                     // wacht 20 msec - 255x20msec=5.1sec
}
}
```

We declareren een variabele DutyCycle.

Alle pins van PORTC worden output en we maken deze pins allemaal laag.

Het CCP1CON register vullen we met 00001100 om de CCP1 module in PWM mode te zetten.

Het T2CON register laden we met 00000110 om TMR2 aan , en de prescaler op 16 te zetten.

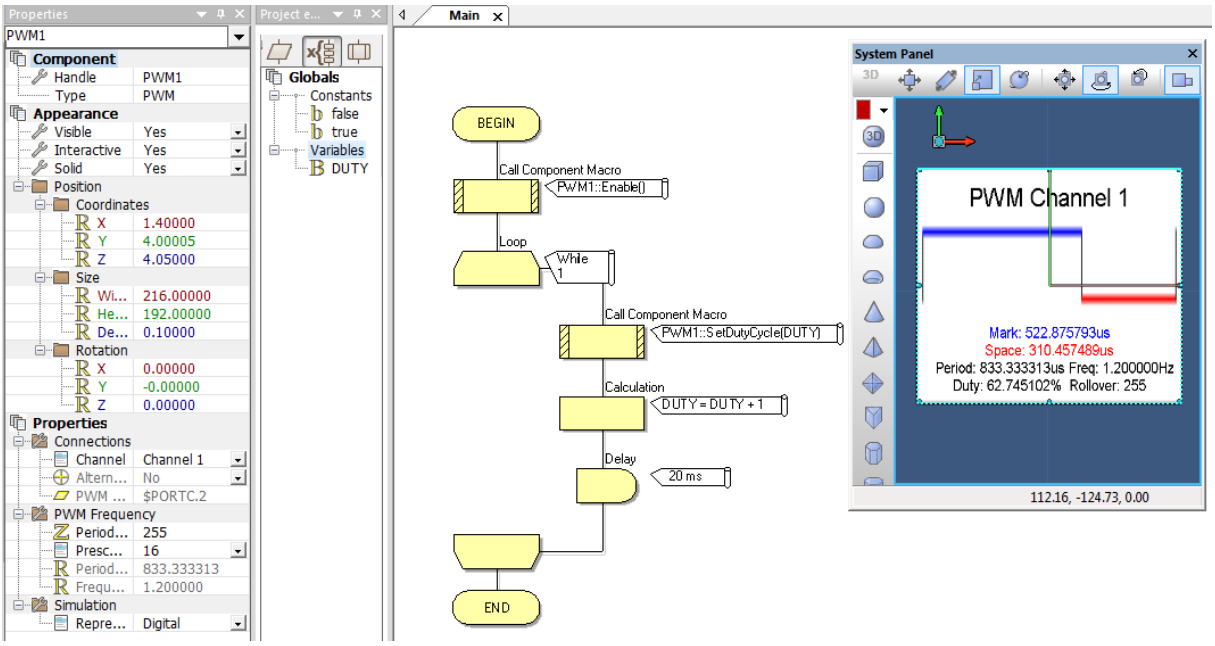
En het PR2 register laden we zoals afgesproken met 255 om de volledige PWM resolutie te behouden.

In een while 1 loop kopiëren we de inhoud van de variabele duty cycle naar het duty cycle register van CCP1.

We verhogen vervolgens de variabele Duty cycle met één en we wachten even 20 msec. In deze loop wordt dus elke 20msec het CCPR1L register met één verhoogd tot 255 om zo het de aan-tijd van het PWM signaal ook te verhogen van 0 tot maximum.

Als we nu – zoals in dit schema – een led hangen aan pin C2 , dan zal mooi te zien zijn dat deze led telkens uit gaat, dan zacht begint te branden en in een periode van 5 seconden harder en harder gaat branden.

Programma PWM in Flowcode

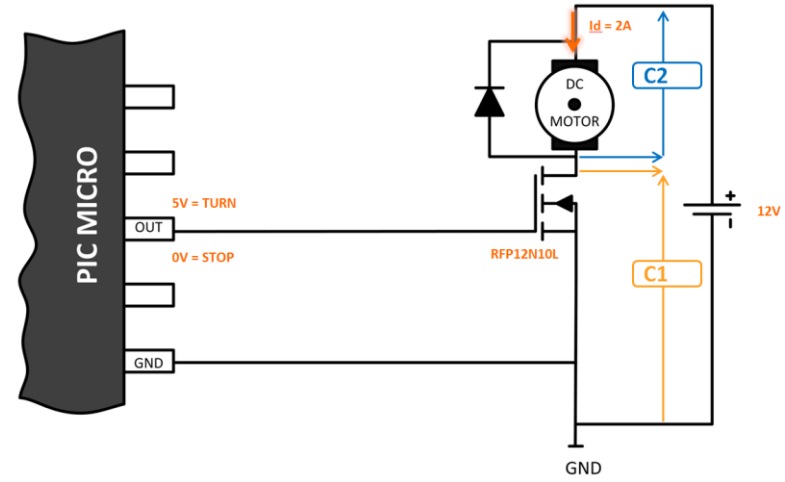


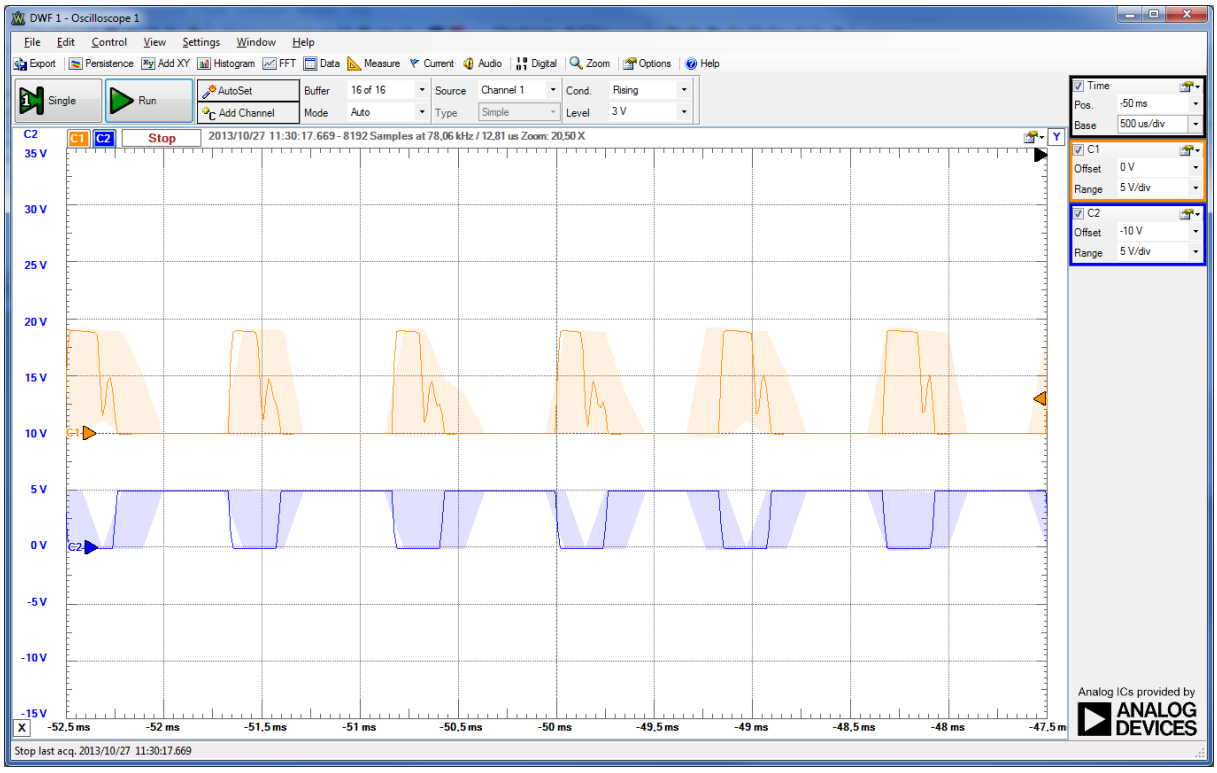
We plaatsen de PWM output component op het System panel, we veranderen het kanaal naar Channel 1 – dus pin C2 en zetten de prescaler op 16 om de PWM frequentie te verlagen naar 1.2Khz. We benoemen ook een 8 bit variabele met de naam DUTY.

In het programma gaan we als eerste PWM enabelen. Vervolgens herhalen we in een eeuwige loop dat de duty cycle van PWM1 de waarde moet krijgen die op dat moment in de variabele DUTY zit. Vervolgens verhogen we Duty met 1 en bouwen we een delay van 20msec in.

In het simulatievenster kan je nu mooi het PWM signaal op pin C2 volgen.

PWM en Dc Motor



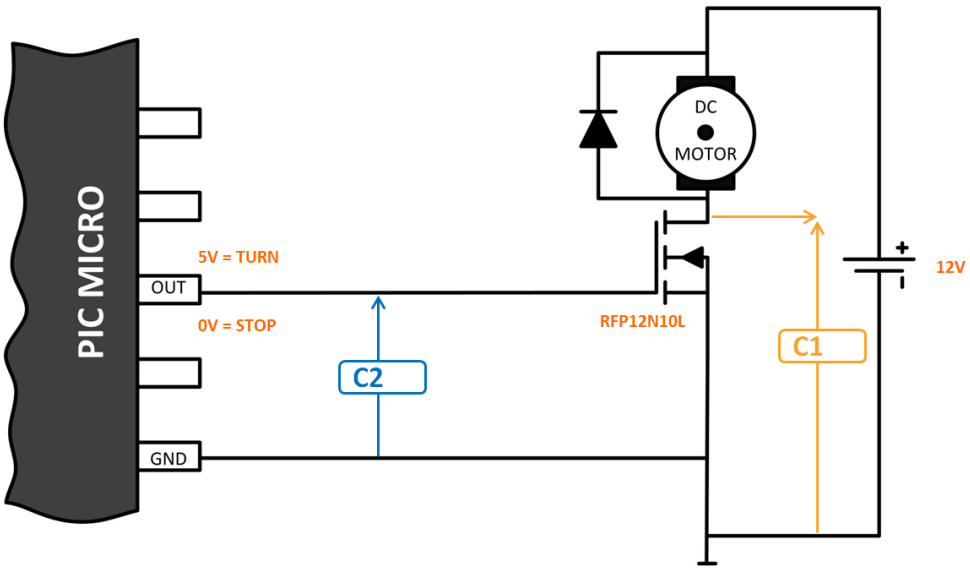


Als we de led vervangen door een Mosfet die een DC motor aanstuurt zoals in het reeds gebruikt schema, dan zien we dat het motortoerental rustig hoger wordt en dan terug nul wordt om weer terug te beginnen stijgen. De periode dat de DC motor nog niet draait, betekent niet dat het PWM signaal dan 0 is, het betekent enkele dat de Duty cycle nog te klein is om voldoende kracht aan de motor te kunnen leveren om de motor effectief aan het draaien te brengen.

Hier ziet u een scoopbeeld met in het blauw de PWM spanning op pin C2 en in het oranje de UDS spanning over de mosfet. De gemeten frequentie is 1.2Khz.

Uitdagingen PWM

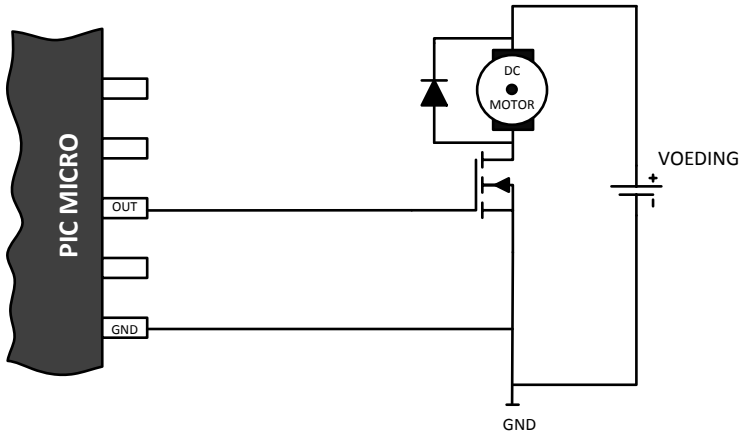
1. Test het programma van deze les uit. Meet het PWM signaal na met de oscilloscoop.
2. Vul het programma aan zodat ook op CCP2 het zelfde PWM signaal komt. Gebruik de datasheet om dit klaar te krijgen.
3. Pas het programma aan zodat het signaal op CCP2 steeds het inverse is van het signaal op CCP1.
4. Via twee drukknoppen moet je het PWM signaal kunnen regelen. RB1 is PWM verhogen – RB2 is PWM verlagen.
5. Verklaar waarom wanneer je het PR2 register met een waarde vult die kleiner is als 255 dat je dan een lagere PWM resolutie hebt – je kan dan de PWM waarde niet meer regelen tussen 0 en 255.
6. Maak een testopstelling zoals in het schema hieronder is aangegeven. De OUT pin van de microcontroller is pin C1. Vergeet de vrijloopdiode niet. Meet C1 en C2 met de oscilloscoop.



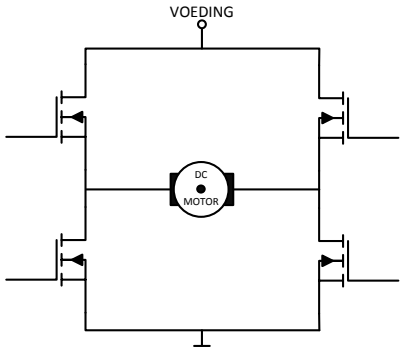
7. Schrijf een programma waarbij je met twee drukknoppen de snelheid van de DC motor kan regelen.
8. Schrijf een programma waarbij je met een potentiometer aan een analoge ingang van de uC de snelheid van de DC motor kan regelen.
9. Schrijf een programma waarmee je met een schakelaar de motor in of uitschakelt, maar waarbij je met PWM het toerental rustig laat aanlopen bij inschakelen en rustig laat uitbollen bij uitschakelen.

De H-Brug

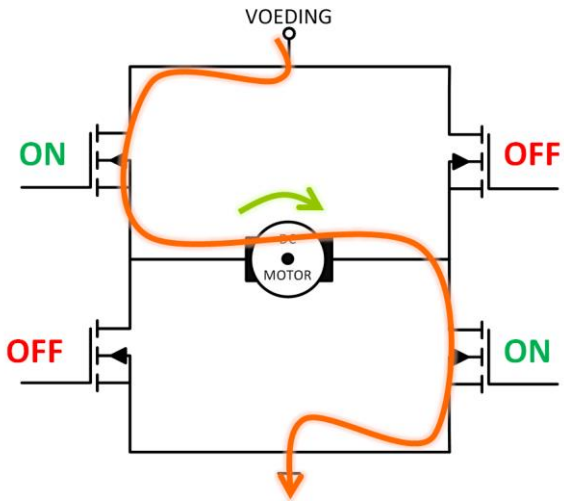
Werkingsprincipe H-Brug



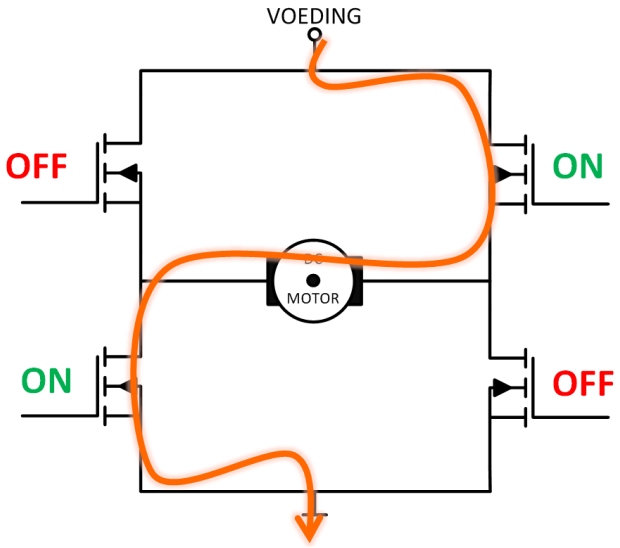
Tot hiertoe hebben we een motor steeds aangestuurd met één bipolaire transistor of mosfet. We kunnen hiermee een DC motor starten, stoppen en via PWM ook in snelheid regelen, maar wat we nog niet kunnen is de DC motor van draairichting laten veranderen. Om de draairichting van een DC motor om te keren moet je de stroomzin door de DC motor omkeren. Dat is niet mogelijk met slechts 1 mosfet. Je zou een schakeling kunnen bedenken waarbij je met een relais de stroomzin zou omkeren, maar wij verkiezen een H-brug – volledig opgebouwd met elektronische componenten.



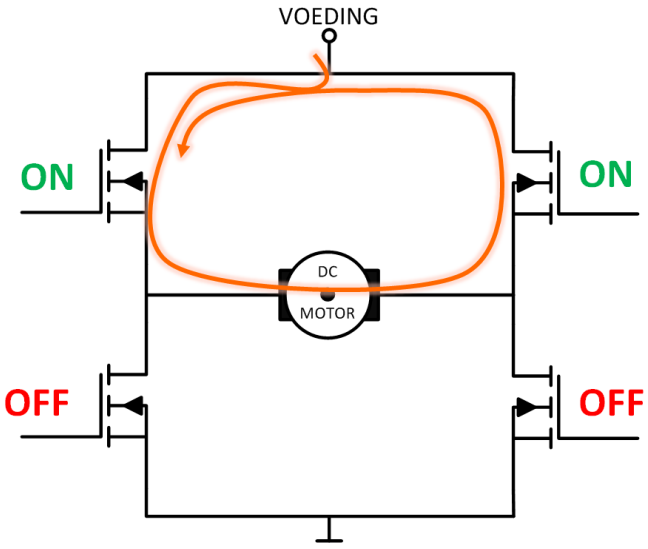
Een H-brug bestaat uit 4 elektronische schakelaars – In ons voorbeeld 4 verrijkings-mosfets van het N type .



Het werkingsprincipe van een H-brug stelt op zich weinig voor. We sturen de mosfet links boven en rechts beneden in geleiding – en zorgen ervoor dat beide andere mosfets niet in geleiding staan. De stroom van de voeding vloeit nu van links naar rechts door de DC Motor. De DC motor draait nu in een bepaalde richting.



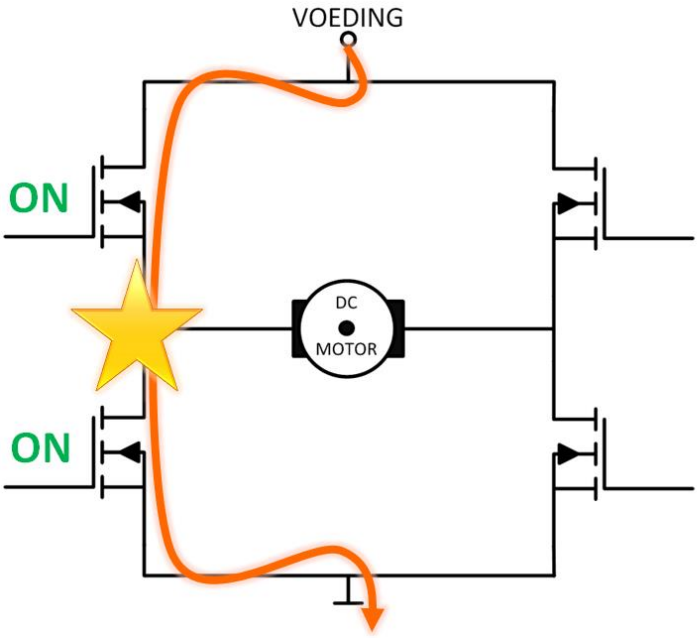
Om de motor in de andere richting te laten draaien moeten we de mosfet rechts boven en links onder in geleiding sturen. De stroom vloeit nu van rechts naar links door de DC motor wat tot gevolg zal hebben dat de motor nu in de andere richting draait.



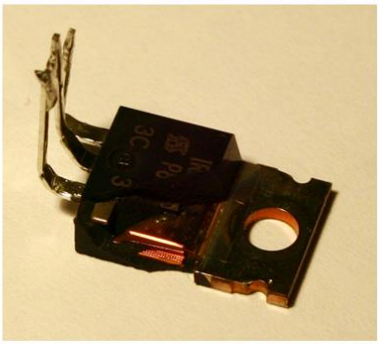
Door de beide bovenste of beide onderste mosfets in geleiding te zetten, kan de DC motor zelfs worden afgeremd.

H-bruggen lijken zo héél eenvoudig, maar dat zijn ze absoluut niet – er zijn heel wat problemen waar er een oplossing voor moet komen.

H-Brug Probleem Kortsluiting

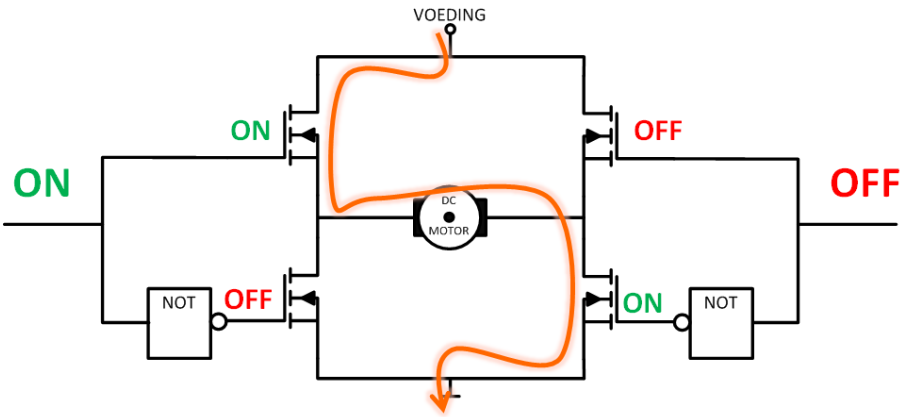


Ge-ëxplodeerde mosfet



Het gevaar bestaat dat je 'per ongeluk' beide mosfets aan de linker – of aan de rechter kant - in geleiding zet. Vermits de motor nu niet meer mee in de kring staat, en er dus geen belasting meer is, is dit een zuivere kortsluiting en zal dit resulteren in minimaal 1 mosfet die stuk is. Een mooie afbeelding van een mosfet die te veel stroom te verwerken kreeg zie je aan de rechterkant.

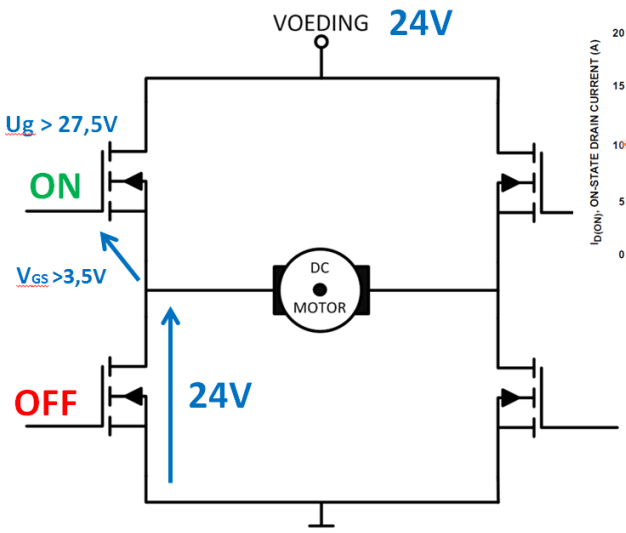
In deze context moeten we hier ook rekening houden met iets wat we in één van de vorige lessen hebben behandeld. We hebben namelijk gezien dat elektronische schakelaars niet oneindig snel aan en uitschakelen en dat vooral het uit geleiding komen meer tijd in beslag kan nemen als het in geleiding komen. Dat betekent dat je – als je eerst de onderste mosfet laat sperren, je even moet wachten tot die helemaal spert alvorens je de bovenste mosfet in geleiding mag zetten.

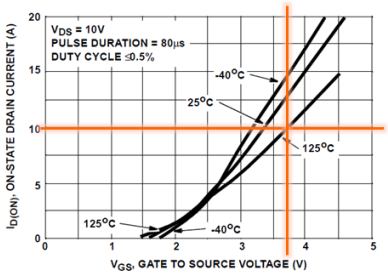


Een oplossing die hier soms wordt gebruikt is een invertor. De invertor zorgt ervoor dat de twee mosfets die onder elkaar staan nooit samen in geleiding kunnen gezet worden. De invertor zorgt ook voor voldoende vertraging zodat de uitschakelvertraging van de mosfet gecompenseerd wordt.

H-Brug Probleem – High Side Mosfets

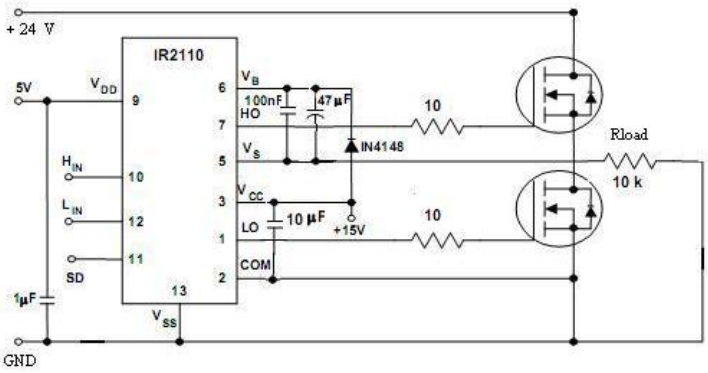
Een tweede – nog complexer probleem is dat van de high side mosfets.





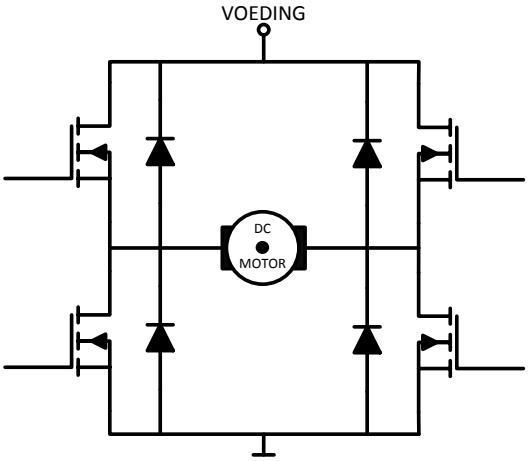
We bekijken hiervoor even terug het schema van de H brug en veronderstellen dat de mosfet links onder niet in geleiding is en de mosfet links boven wel in geleiding staat. De 24 volt zal dan nagenoeg volledig over de onderste sperrende mosfet staan (en over de motor die 'parallel' met deze mosfet staat als de mosfet rechts onder ook in geleiding komt) .

In de datasheet van onze mosfet – de RFP12N10L vinden we terug dat de spanning die we tussen de gate en de source moeten aanleggen afhankelijk is van de belastingsstroom, maar toch minimaal 3.5 volt is om de mosfet volledig in geleiding te brengen. Vermits de source van de bovenste mosfet door het sperren van de onderste mosfet reeds op 24 volt staat, moeten we aan de gate van de high side mosfet nu een spanning van 24V +3.5V = 27,5 volt aanleggen om deze in geleiding te brengen. Onze microcontroller levert slechts 5 volt en ook aan de 24 volt die we van de voeding krijgen – hebben we hier niet voldoende.



In deze schakeling wordt dit probleem opgelost door gebruik te maken van een high en low side driver. De IR2110 die hier gebruikt wordt is maar één van de vele mogelijkheden. Deze driver maakt via een externe diode en condensator een opslingercircuit (een charge pump of bootstrap circuit) waarmee de spanning van 24 volt wordt opgeslingerd naar een hogere spanning – voldoende om de high side mosfet in geleiding te krijgen. Er bestaan zelfs drivers die ook beveiligen tegen kortsluiten en het eventueel te trage uitschakelen van de mosfets. Dit type drivers noemt met 'High side en low side mosfet drivers'.

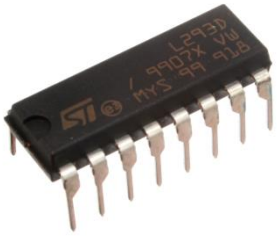
H-Brug Probleem Vrijloopdiodes

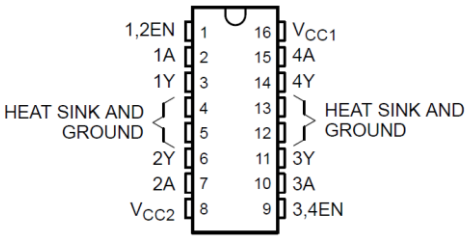


Als we zelf een H-brug zouden bouwen, dan moeten we ook hier de mosfets beveiligen tegen tegen-emk spanningen. 4 Diodes moeten deze tegen-emk spanningen op een veilige manier afleiden. Deze 4 diodes moeten echter voldoende snel zijn en voldoende vermogen kunnen dissiperen.

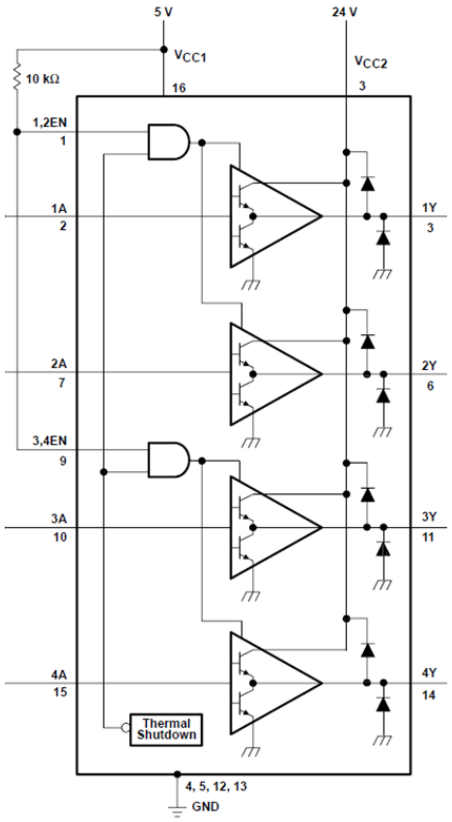
L293D – H-Brug 0,6A

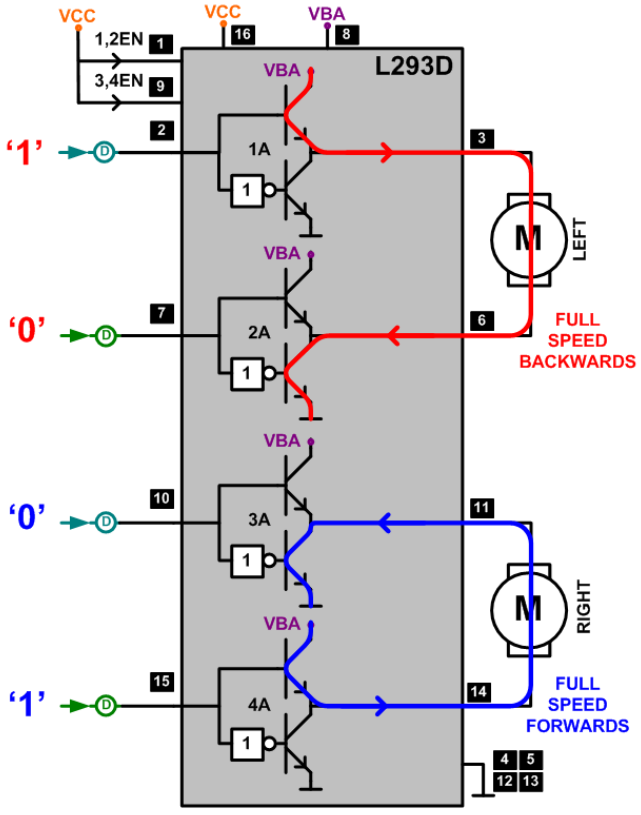
Zelf zo maar even een werkende H-brug bouwen voor uw specifieke toepassing blijkt dus al snel vrij complex te worden – zeker als de stromen en spanningen wat groter worden. Gelukkig bestaan er kant en klare oplossingen die voor ons in de meeste situaties een goede oplossing bieden.





De goedkoopste oplossing is de L293D IC. In deze 16 pin IC zit een dubbele H-brug die 600mA per brug kan leveren aan een belasting en voor de meeste kleine DC motortjes volstaat dit. De noodzakelijke vrijloopdiodes zijn mee in deze IC ingewerkt, dus daar moet u hier geen extra tijd, ruimte en kosten voor voorzien.



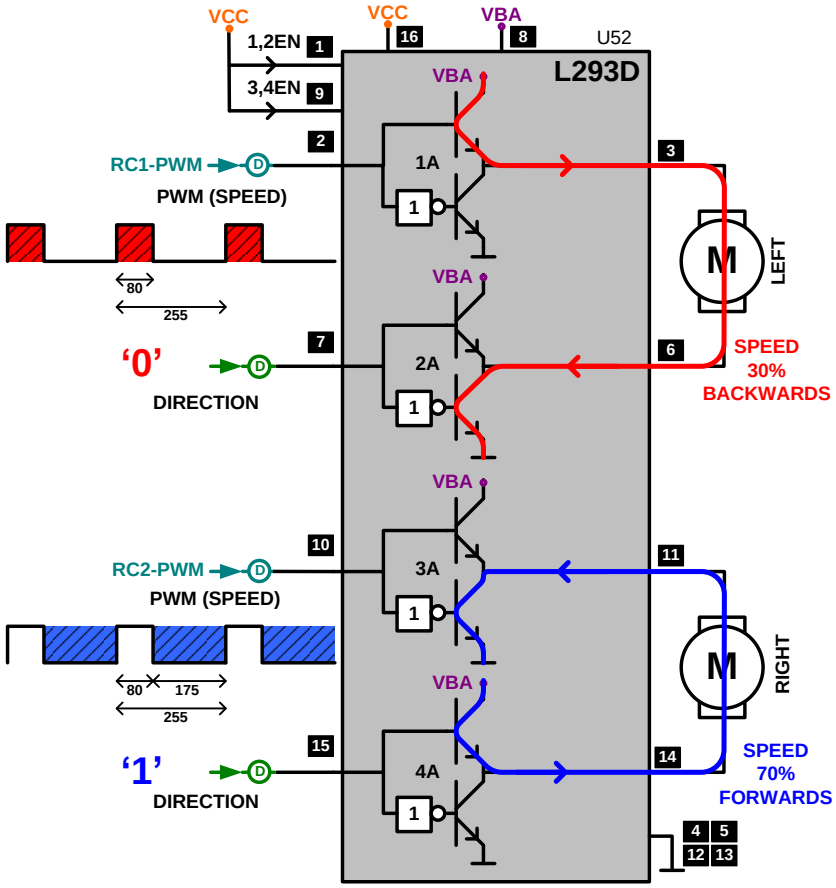


We bekijken de L293D even van wat dichterbij. Links ziet u een schema uit de datasheet – rechts een zelfgemaakte tekening. We gebruiken het rechtste schema. De bovenste 4 transistoren vormen de ene H-brug – de onderste 4 vormen de andere H-brug. We sluiten 2 DC motoren aan tussen pinnen 3 en 6 en tussen 11 en 14. De voeding voor de motoren – die tot wel 36Volt mag oplopen – sluiten we aan – aan de VBA pin. Door pin 2 hoog – of 5 volt te maken – sturen we de bovenste transistor in geleiding. De 0 volt op pin 7 wordt door de invertor omgezet in een 1 waarmee de onderste transistor in geleiding wordt gebracht. Er kan vanaf nu een stroom vloeien van de VBA

voeding door de bovenste transistor – door de motor en door de onderste transistor terug naar de VBA voeding. De VBA voeding is de voeding die de motoren zal voeden.

De blauwe pijl stelt de situatie voor waarmee we met de onderste H-brug de motor in de andere richting laten draaien.

Om naast het omdraaien van de draaizin van de motor ook het toerental van de motor te kunnen sturen, moeten we gebruik maken van PWM.



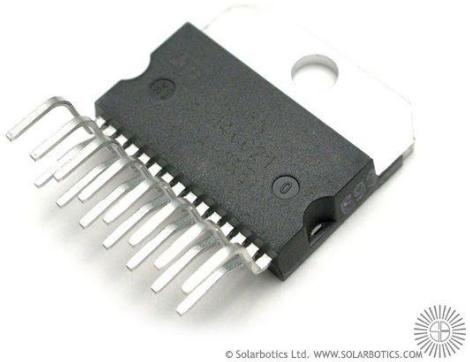
De onderste transistorparen van de respectievelijke H-bruggen blijven we aansturen met gewone digitale uitgangen van de microcontroller. Hiermee controleren we de draairichting – 0 is vooruit – 1 is achteruit.

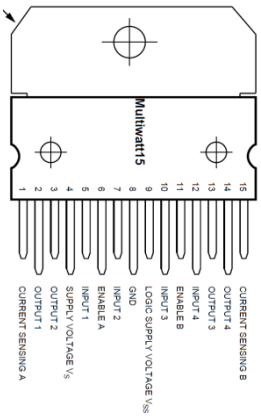
De twee mogelijke PWM uitgangen hangen bij onze uC aan PIN C1 en PIN C2 en deze worden aan de bovenste transistorparen van de respectievelijke H-bruggen gekoppeld.

We bekijken de bovenste H-brug. Op de momenten dat het PWM signaal hoog is zal er stroom vloeien door de motor. Hoe langer de hoog-tijd van het PWM signaal – hoe sneller de motor zal draaien.

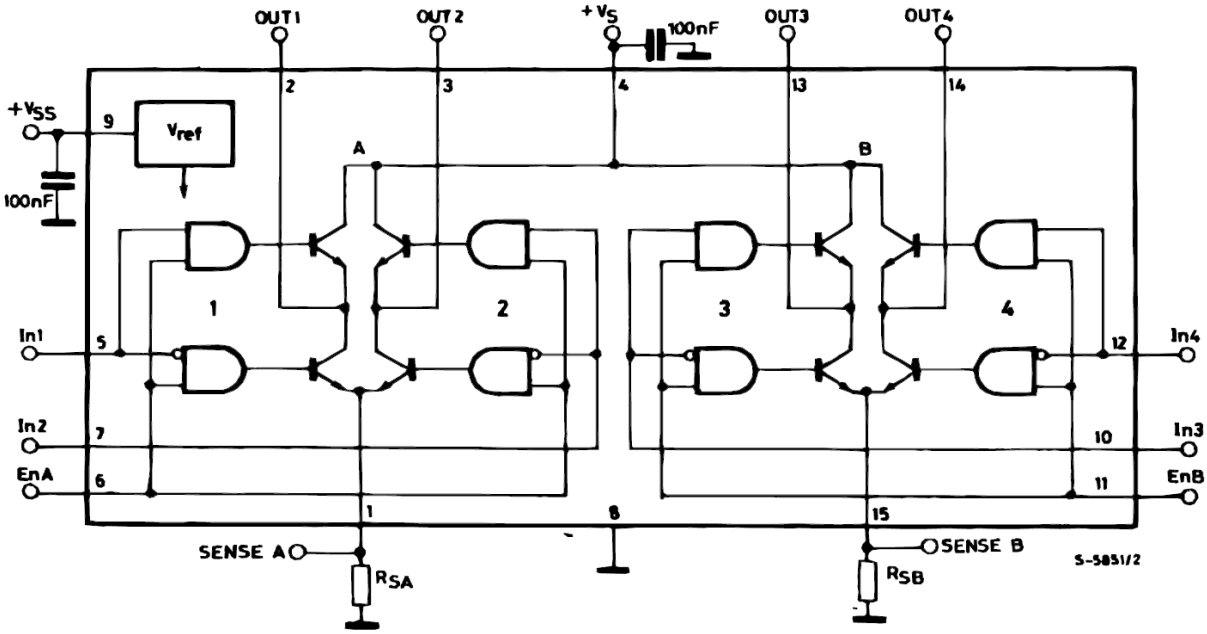
Bij de onderste H-bug moeten we toch even opletten. Om de draairichting van deze DC motor om te keren hebben we pin 15 hoog gemaakt. Dat betekent dat er nu stroom vloeit door de DC motor op de momenten dat het PWM signaal LAAG is. Hoe hoger de duty cycle van dit PWM signaal – hoe trager dat de motor in dit geval draait. Opletten dus!! – Je Duty Cycle moet in je software geïnverteerd worden als de draairichting verandert.

L298 H-Brug 2A





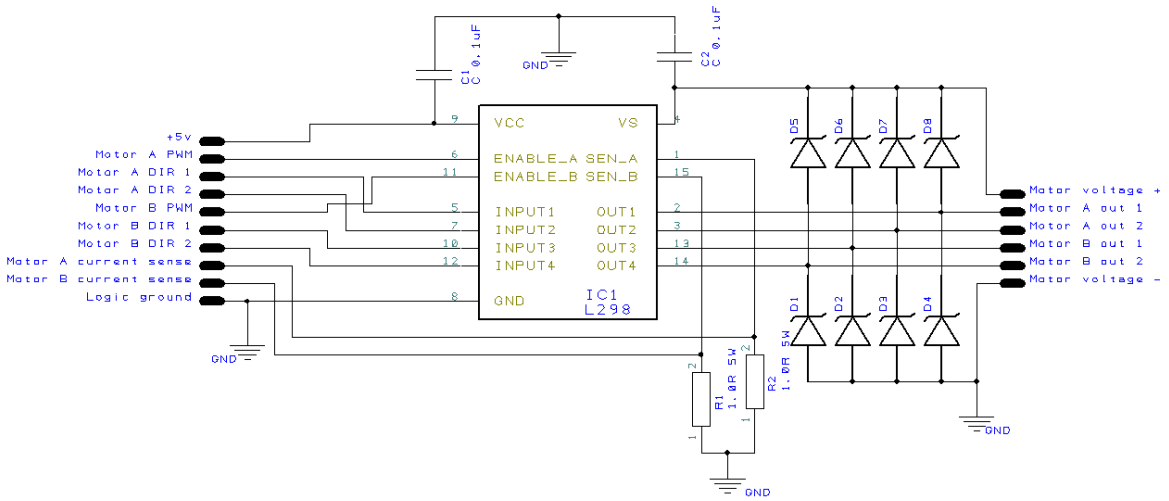
Voor het zwaardere werk kiezen we dikwijls voor de L298. Ook deze bezit een dubbele H-brug, maar kan tot 2A per kant aan de belasting leveren . Het is zelfs mogelijk om beide H-bruggen parallel te gebruiken zodat de totale uitgangsstroom op 4A komt. Deze H-brug moet echter nog wel extern van vrijloopdiodes voorzien worden.



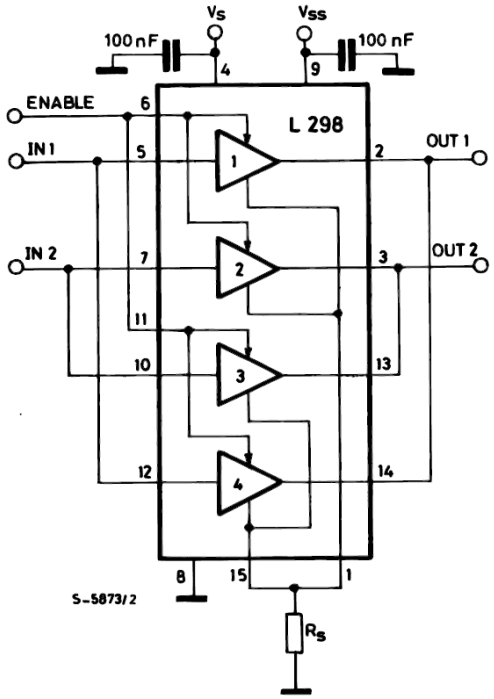
Op het blokschema van deze L298 herkent u duidelijk de twee H-bruggen – de pinnen om de twee DC motoren op aan te sluiten en de stuurpinnen van de H-buggen. Ook de pinnen om de H-bruggen te enabelen en de pinnen om de voeding voor de motoren aan te leggen zijn heel gelijkaardig aan die bij de L293D.

Het enige verschil zijn hier misschien de sense weerstanden wel. Deze zijn optionele kleine vermogen-weerstanden die je onder de H-buggen kan plaatsen. De spanning over deze weerstanden is recht evenredig met de stroom door de motoren en deze zou door een analoge ingang in de microcontroller kunnen worden binnengelezen om zo closed loop motor control toe te passen.

Zoals eerder gezegd moeten we er bij de L298 wel rekening mee houden dat we zelf extern nog vrijloopdiodes moeten zetten om de H-brug te beveiligen tegen inductiespanningen. Op deze tekening ziet u duidelijk hoe dit moet worden uitgevoerd.



De twee H-bruggen van de L298 kunnen ook parallel gezet worden om tot 4A belastingsstroom te gaan zoals hieronder te zien is.

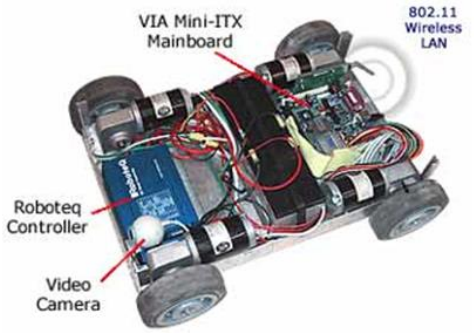


Als we dan nog grotere stromen willen schakelen, dan zouden we kunnen overwegen om zelf een volledige H-brug op te bouwen uit discrete componenten, maar als je even rondkijkt op het internet, dan zie je dat anderen deze complexe ontwikkeling al voor u gedaan hebben.

MD22
5A-24V

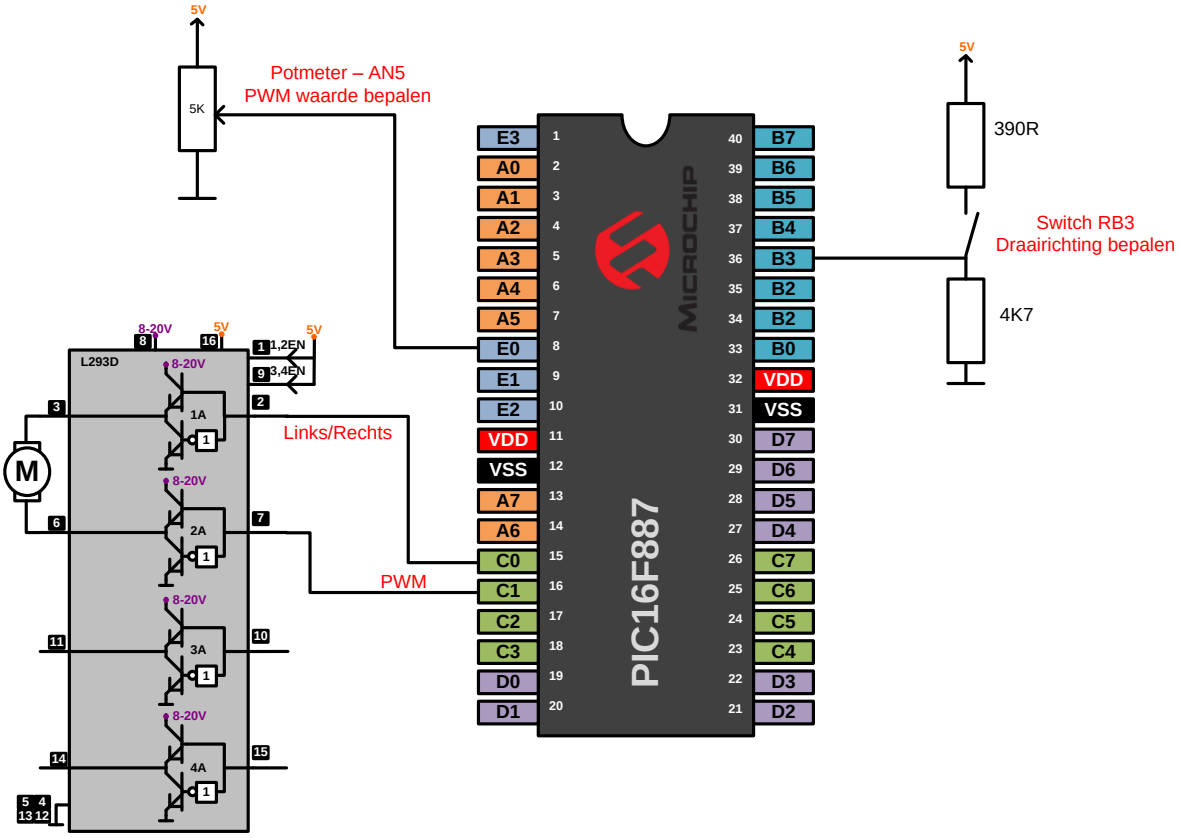
MD03
20A-50V

ROBOTEQ – AX2550
120A – 50V



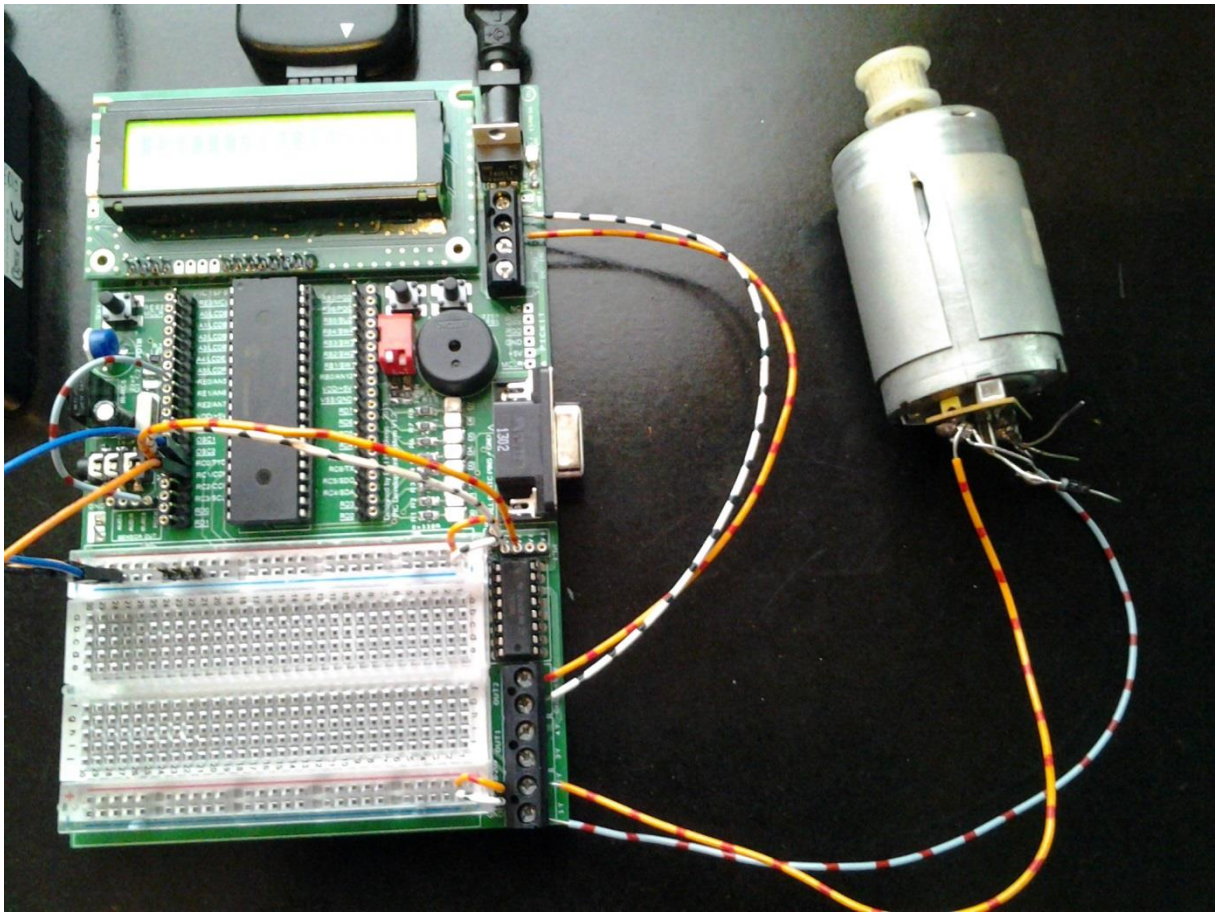
We vinden meteen een 5A, een 20 A en zelfs een 120A H-brug. Deze zware H-bruggen worden typisch gebruikt om elektrische brommers en grote robots te besturen…

Meetopstelling met L293D H-Brug



We sluiten één van de 2 H-bruggen van de L293D aan op pin C0 en C1 van de microcontroller. Met pin C0 zullen we de draairichting van de motor bepalen door deze pin hoog of laag te maken. Met pin C1 bepalen we de draaisnelheid door op deze pin een regelbaar PWM signaal te genereren.

Met de schakelaar aan ingangspin B3 bepalen we of pin C0 hoog of laag is en dus ook wat de draairichting van de DC motor is. Met de potmeter aan de analoge ingang RE0 of AN5 bepalen we dan de duty cycle van het PWM signaal.



Op deze foto ziet u duidelijk de L293D H-brug. We verbinden de voeding van de 12 volt adapter door naar de voeding van de H-brug. De motor sluiten we aan via lustersteen 1Y en 2Y. Uitgangspinnen C0 en C1 van de uC worden doorverbonden naar ingangen 1A en 2A van de H-brug. SWB3 is vast verbonden met PIN B3 en we mogen niet vergeten om de potmeter met een draadje te verbinden met ingang RE0/AN5

Programma H-Brug in C

```
/*********************************************************************
Videolessen Deel 4 - Bart Huyskens - RTC Antwerpen - 2013
Extern Xtal - 19660800Hz
L293D H-brug - C0 aan 1A, C1(PWM) aan 2A
DC motor tussen 1Y en 2Y
Potmeter aan RE0/AN5
Switch aan RB3
*********************************************************************/

#include <htc.h>
#define _XTAL_FREQ 19660800

__CONFIG(0x23E2);   //Extern HS Xtal, WDT DIS, PWRT EN, MCLR EN, BOR EN, INT/EXT DIS,
LVP DIS, NO Code/data protect
__CONFIG(0x3FFF);   //BOR 4,0V, Write protect off

void main()
{
unsigned char Duty_Cycle = 0;        // variabele voor Duty Cycle
TRISC   = 0x00;                      // leds aan PORTC - Output
TRISD   = 0x00;                      // alle pins PORTD - Output - leds
TRISB   = 0b00001000;                // RB3 = input -- switch
TRISE   = 0b00000001;                // PINE0 = input --A/D potmeter

ANSEL   = 0b00100000;                // AN5 (RE0)= AD input andere normal IO
ANSELH  = 0b00000000;                // alle andere AD pins normal IO

CCP2CON = 0b00001100;                //CCP2 in PWM mode zetten
T2CON   = 0b00000110;                // TMR2 ON - presc op 16
PR2     = 255;                       // periode register op 255

ADCON0  = 0b01010101;                //Fosc/8-AN5-ADON (Go/DONE blijft 0)
ADCON1  = 0b00000000;                //Meting tuss VDD en VSS- L justified

while(1)
    {
    RC0 = RB3;                          // bepaal draairicht.(C0) met switch aan RB3
    __delay_us(20);                     //Acquisition time om C op te laden
    ADCON0 = ADCON0 | 0b00000010;       // GO/DONE = 1 met masker
        while (ADCON0 & 0b00000010){}   // doe niets zolang AD bezig is
    CCPR2L = ADRESH;                    // 8 bit resultaat AD nr PWM register
    PORTD = ADRESH;                     //zet AD waarde op leds aan PORTD
    }
}
```

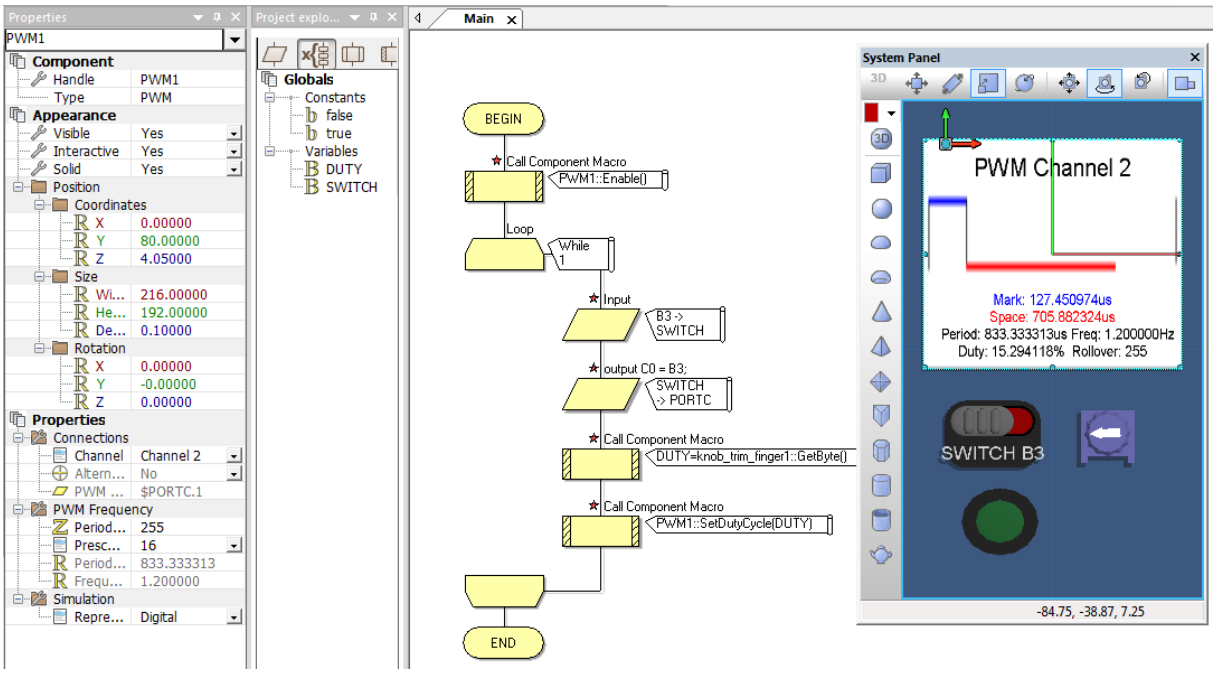
We configureren als eerste de juiste pins als ingangen en de juiste pins als uitgangen. De ANSEL registers mag je ook zeker niet vergeten. Alle pins die je als AD ingang wil configureren moeten op 1 staan, alle andere moeten op 0 staan. Standaard staan alle ANSEL bits op 1. Wij zetten enkel de bit voor AN5 op 1, alle andere pins zijn normale IO pins.

Omdat we hier een PWM signaal willen genereren op pin C1, moet hier het CCP2CON register geconfigureerd worden. TMR2 wordt gedeeld tussen CCP1 en CCP2, dus hier verandert niets aan.

Om de potmeter als analoge waarde in te lezen moeten we via het ADCON0 register een aantal instellingen doen. Bekijk de datasheet als je juist wil weten welke instellingen.

In een eeuwige loop zetten we vervolgens de toestand van Switch aan RB3 op pin CO waarmee we de draairichting bepalen. Vervolgens wordt de AD omzetting gestart en er wordt gewacht tot de DONE bit terug laag wordt om aan te geven dat de AD omzetting klaar is. Het resultaat staat nu in het ADRESH register en dit wordt gekopieerd naar het CCPR2L register om de duty cycle van het PWM signaal te veranderen. Het wordt ook weergegeven op de leds aan PORTD.

Programma H-Brug Flowcode



Om het PWM signaal op pin C1 te krijgen activeren we nu PWM kanaal 2. De prescaler wordt op 16 gezet om de PWM frequentie op 1.2Khz te houden. Een switch aan pin B3 stelt onze schakelaar voor, en een potmeter op kanaal AN5 stelt onze potmeter voor. De groene led geeft de toestand van pin C0 weer.

De eerste twee instructies in de while loop doen niets meer dan de toestand van de switch op B3 op de uitgang van C0 zetten – hiermee bepalen we de draairichting van de motor.

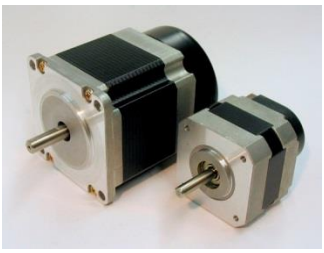
De andere twee instructies doen niet meer dan de analoge waarde van de potmeter inlezen en omzetten naar een binaire waarde tussen 0 en 255. Vervolgens wordt deze waarde gebruikt om de duty-cycle van de motor te bepalen.

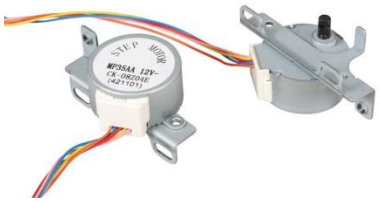
Let er wel op dat wanneer je de draairichting omkeert, dat dan ook de duty cycle waarde omkeert zoals je gezien hebt in de theorie over de H-brug.

Uitdagingen H-Brug

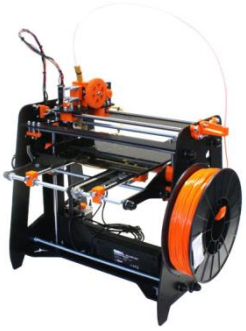
1. Verander het huidige demo H-brug programma zodat de potmeter vanaf nu steeds naar rechts moet gedraaid worden om te versnellen en naar links om te vertragen – ongeacht de stand van SWITCH aan PORTB die de draairichting bepaalt.
2. Pas het programma aan zodat je vanaf nu twee DC motoren in snelheid kan regelen – als de ene versnelt , dan moet de andere vertragen en omgekeerd. Met SWB3 en SWB2 bepaal je de respectievelijke draairichtingen. Gebruik de uitgangen C0 en C1(PWM) voor de ene motor en C2(PWM) en C3 voor de tweede motor.

WERKING STAPPENMOTOR

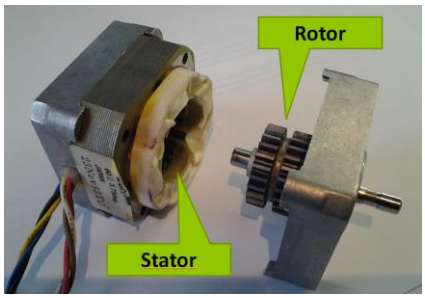




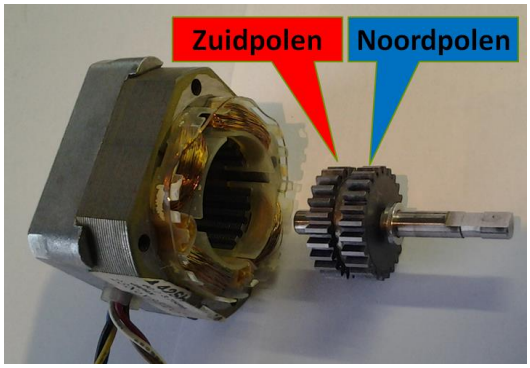
Stappenmotoren zijn DC motoren die kunnen ronddraaien in kleine stappen. Ze worden veel gebruikt in allerlei toepassingen. In kleine en in grote printers om de printkop juist te positioneren, in CNC freesmachines en printfreesmachines om de freeskop exact te positioneren in de X en Y as en in 3D printers om de printkop exact te positioneren in de X, Y en Z richting, en zo zijn er nog heel veel toepassingen van stappenmotoren.



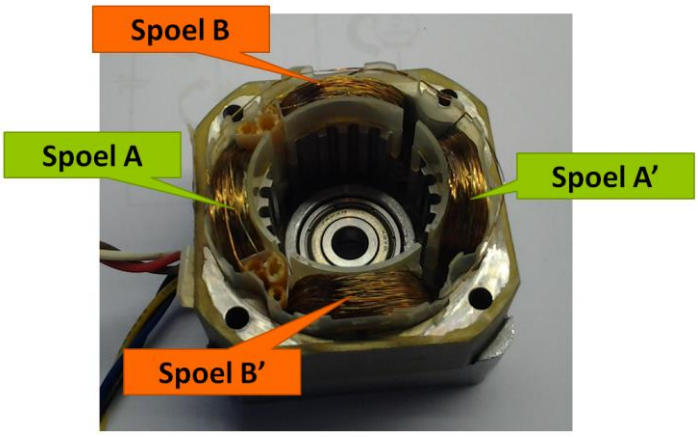
Werking Stappenmotor

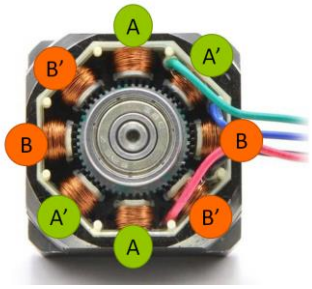


We demonteren even deze stappenmotor en herkennen zo meteen de rotor – die ronddraait en de stator die vast aan de behuizing zit. Er komen 6 draden uit deze stappenmotor.



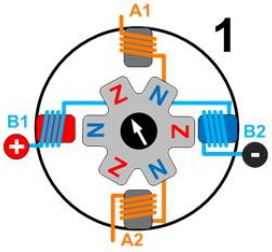
De rotor lijkt te bestaan uit twee tandwielen. Dit is echter 1 grote permanente magneet waarvan het ene tandwiel de Zuidpool vormt en het andere tandwiel de Noordpool. Als je heel nauwkeurig kijkt dan zie je dat de tanden van de noordpolen juist tussen de tanden van de zuidpolen vallen.



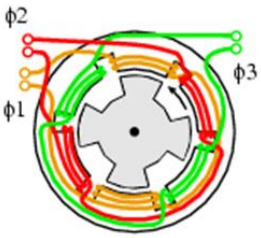


In de stator herkennen we 4 spoelen – elk met hun getande spoelkernen. Spoel A, spoel A' en spoel B en spoel B'. Afhankelijk van in welke zin de stroom door zo'n spoel gestuurd wordt, wordt de getande metalen spoelkern gemagnetiseerd als noordpool of als zuidpool.

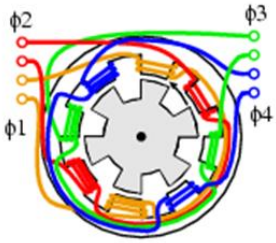
Zoals je in de rechtse afbeelding kan zien bestaan er ook stappenmotoren met 8 of zelfs nog meer statorspoelen. De 8 spoelen staan dan per twee in serie om een betere magnetische koppeling met de rotor te bekomen. Qua aansturing moeten we hier echter totaal geen rekening mee houden.



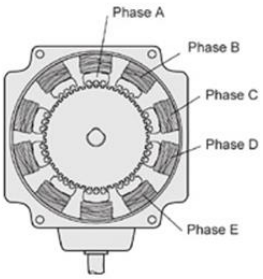
2 fasen



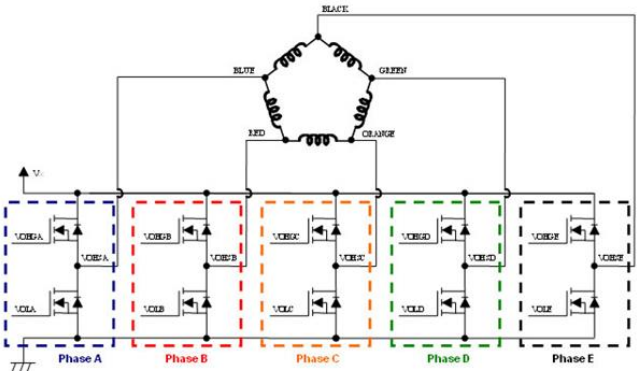
3 fasen



4 fasen

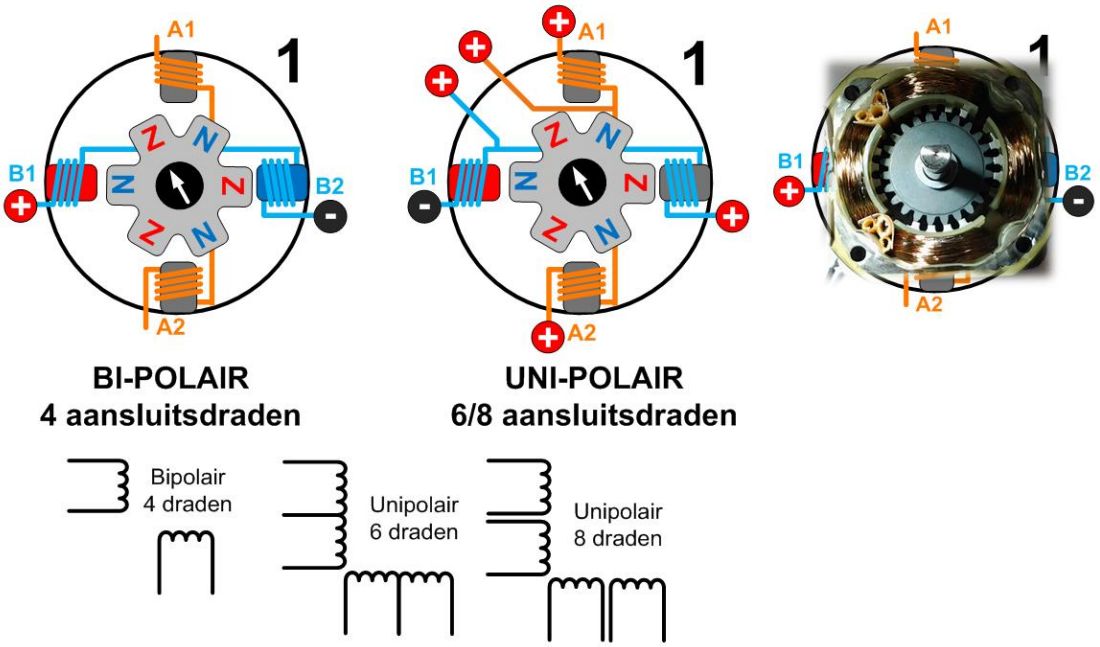


5 fasen



De stappenmotoren die wij meestal tegenkomen zijn 2 fasig – ze hebben een A en een B spoel, maar om nog grotere nauwkeurigheden te bekomen wordt het aantal fasen soms verhoogd naar 3, 4 of 5 fasen. De aansturing wordt er dan zeker niet eenvoudiger op.

Verschil Bipolair en Unipolair



Op de tekening van deze vereenvoudigde stappenmotor herken je de 4 statorspoelen – identiek aan een echte stappenmotor. De vereenvoudiging is gebeurd met het aantal rotor-polen. Deze rotor heeft slechts 6 polen – 3 zuid en 3 noord, terwijl onze stappenmotor 24 zuid en 24 noordpolen heeft.

Stappenmotoren worden onderverdeeld in twee grote groepen – de bipolaire en de unipolaire stappenmotoren. Bipolaire stappenmotoren kan je gemakkelijk herkennen omdat ze slechts 4 aansluitdraden hebben. Unipolaire stappenmotoren hebben meestal 6 maar soms ook 8 aansluitdraden.

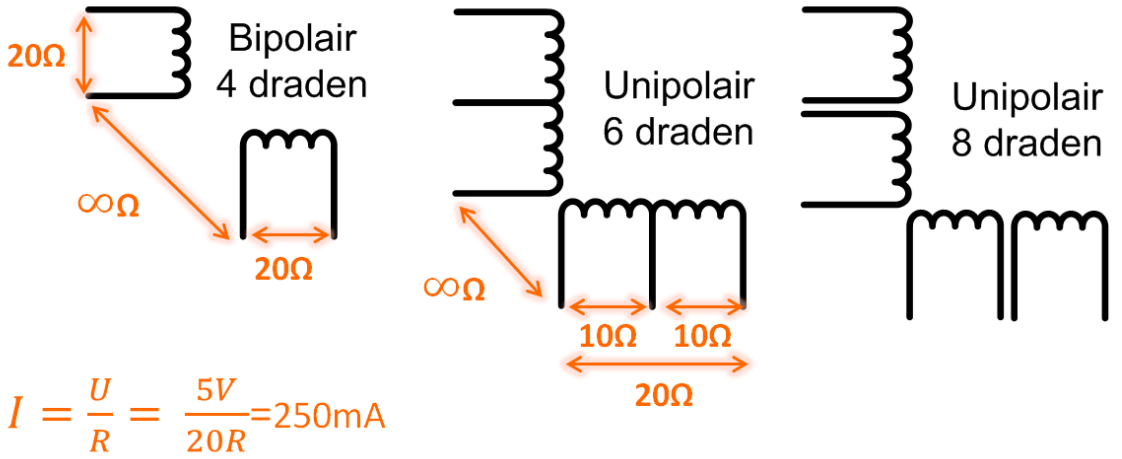
Zoals je weet bestaat de stator van een stappenmotor uit 4 spoelen. Bij een bipolaire stappenmotor staan die spoelen zonder meer per twee in serie. U ziet mooi in de linkse tekening dat er 4 draden in de motor lopen. Draad A1 gaat eerst naar de bovenste spoel, dan naar de onderste spoel en zo weer naar buiten via draad A2.

B1 gaat via de linkse spoel, naar de rechtse spoel en ook weer naar buiten via de B2 draad.

Bij unipolaire stappenmotoren is er een extra aftakking die naar buiten gebracht wordt tussen de twee spoelen. 2 extra draden maakt 6 draden in totaal. Soms is het zelfs zo dat van elke spoel twee afzonderlijke draden naar buiten gebracht worden, die je dan zelf extern nog aan elkaar moet doorverbinden om er een unipolaire stappenmotor van te maken. 4 spoelen met elk 2 draden maakt 8 draden in totaal. Was het je misschien al opgevallen dat je van een unipolaire stappenmotor steeds een bipolaire stappenmotor kan maken door gewoon de middenaftakkingen niet te gebruiken.

Spoelen en Stroomverbruik Stappenmotor Bepalen

Van stappenmotoren uit afbraak zijn meestal niet alle gegevens bekend – je kunt de gegevens op het kentekenplaatje wel eens googelen, maar dikwijls vind je niets. Je zult zelf moeten uitzoeken hoe de spoelen aangesloten zijn en hoeveel stroom de stappenmotor zal trekken.



$$I = \frac{U}{R} = \frac{5V}{20R} = 250\text{mA}$$

Je kunt al veel zien aan het aantal draden die naar buiten komen. Bij 4 draden is het een bipolaire stappenmotor, bij 6 of 8 draden is het een unipolaire stappenmotor.

Bij een bipolaire is het eenvoudig. Je gebruikt de ohm-meter en meet tussen alle 4 de draden de ohmse weerstand. Dit zal 2 x een bepaalde waarde geven en 2 maal oneindig. Je weet nu tussen welke draden de spoelen zitten en hoeveel ohm de spoelen zijn. Als op het kentekenplaatje een spanning staat, dan werk je hiermee, anders start je met 5 volt om zo de stroom door de spoelen te berekenen. Deze stroom is vooral belangrijk om te bepalen hoeveel stroom de H-brug driver moet leveren.

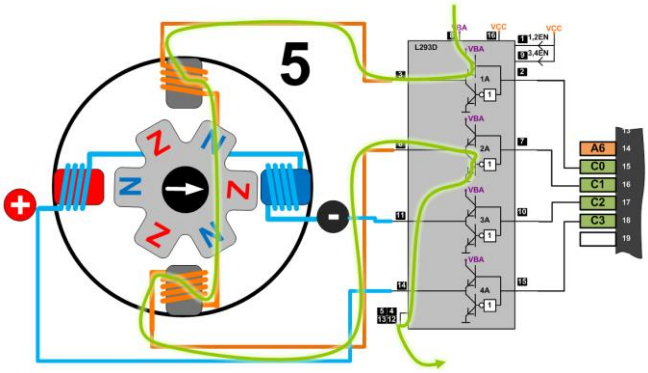
Bij een unipolaire stappenmotor met 6 draden moet je ook nog eens uitzoeken wat de middenaftakking van elke spoel is. Als je zoals hier 1 x 20 Ohm en 2 x 10 Ohm meet tussen 3 draden, dan kan je hier uit afleiden wat de middenaftakking is.

Bij 8 draden kan je de ohm-meter gebruiken om te bepalen tussen welke draden de spoelen zitten, maar om zonder enige extra informatie te weten te komen welke 2 spoelen er in serie moeten geschakeld worden, en hoe ze in serie geschakeld moeten worden, bestaat er slechts één methode: 'Trial and Error'. Sluit de stappenmotor aan op een driver en bestuur de driver met een stappenmotor programma waarvan je zeker bent dat het werkt – meet de 4 stappen na met een oscilloscoop of logic analyzer om zeker te zijn. Het komt er op aan om nu alle verschillende combinaties te testen tot de stappenmotor vlot in één richting draait.

Stappenmotor Driver-Methodes

Stappenmotoren trekken te veel stroom om rechtstreeks met de 20mA pins van een microcontroller aan te sturen. We moeten hier dus een vermogen trap of driver tussen zetten.

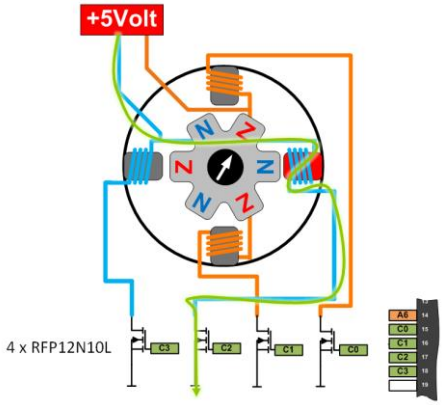
Bipolair Met H-Brug



Bij bipolaire stappenmotoren heb je geen keuze – hier moet je een dubbele H-brug voor gebruiken vermits je de stroom door 2 spoelen telkens van richting moet laten veranderen. Als de stroom door de spoelen onder de 600mA blijft, dan is de L293D een ideale component. Bij meer stroom zou de L298 een oplossing kunnen bieden. We hebben deze hier aangesloten op 4 pins van poort C van de uC. De 4 uitgangen van de 2 H-bruggen worden aangesloten aan de 4 draden van de stappenmotor. Met pin C0 hoog en C1 laag vloeit er een stroom door de oranje spoel zoals hier aangegeven is. Met C0 laag en C1 hoog kunnen we deze stroomzin omkeren – voor de blauwe spoel geldt het zelfde principe, enkel hier aangestuurd met de pinnen C2 en C3.

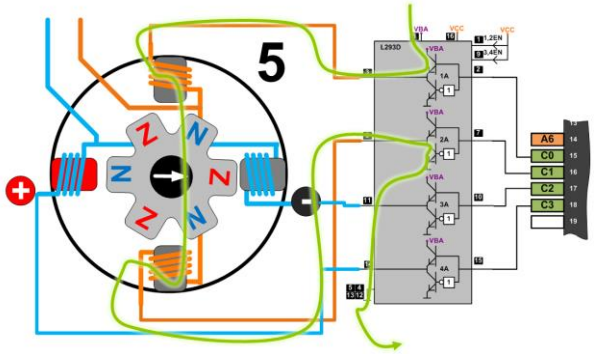
Bij unipolaire stappenmotoren kan je wel keuzes maken.

Unipolair met Mosfets



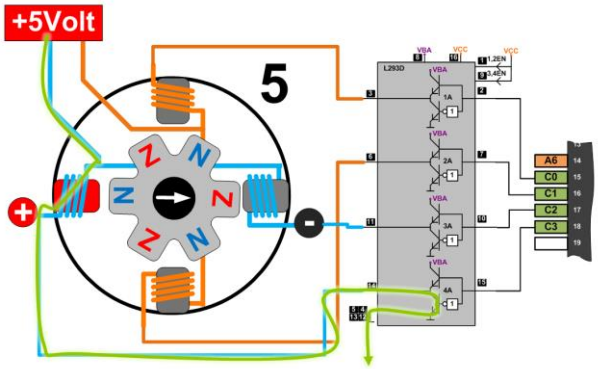
Je sluit de middenaftakkingen aan, aan een bron – in dit geval 5 volt, en je gebruikt 4 mosfets als driver. Als we hier pin C2 hoog maken, dan zal de stroom vloeien zoals hier aangegeven wordt. De mosfets die we hier gebruiken kunnen 100 volt aan, en kunnen daarmee redelijke tegen-EMK spanningen verwerken, maar eigenlijk moeten je deze mosfets ook beveiligen met vrijloopdiodes.

Unipolair als Bipolair met H-Brug



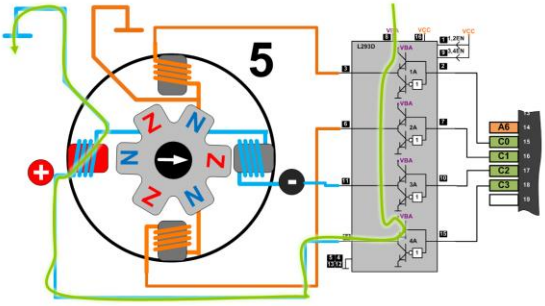
We kunnen een unipolaire stappenmotor ook altijd gebruiken als een bipolaire stappenmotor. Niets houdt ons namelijk tegen om de middenaftakkingen open te laten hangen en de aansturing identiek als bij een bipolaire met een H-brug te doen.

Unipolair met H-Brug – Middenaftakking aan +



Bij de derde optie hang je de middenaftakkingen van de unipolaire toch aan de voeding, maar gebruik je de 4 onderste transitoren van de H-bruggen op een zelfde manier als de aansturing met de mosfets. Het voordeel hiervan is dat alles in één behuizing zit en dat de L293D intern vrijloopdiodes heeft die beschermen tegen tegen-EMK.

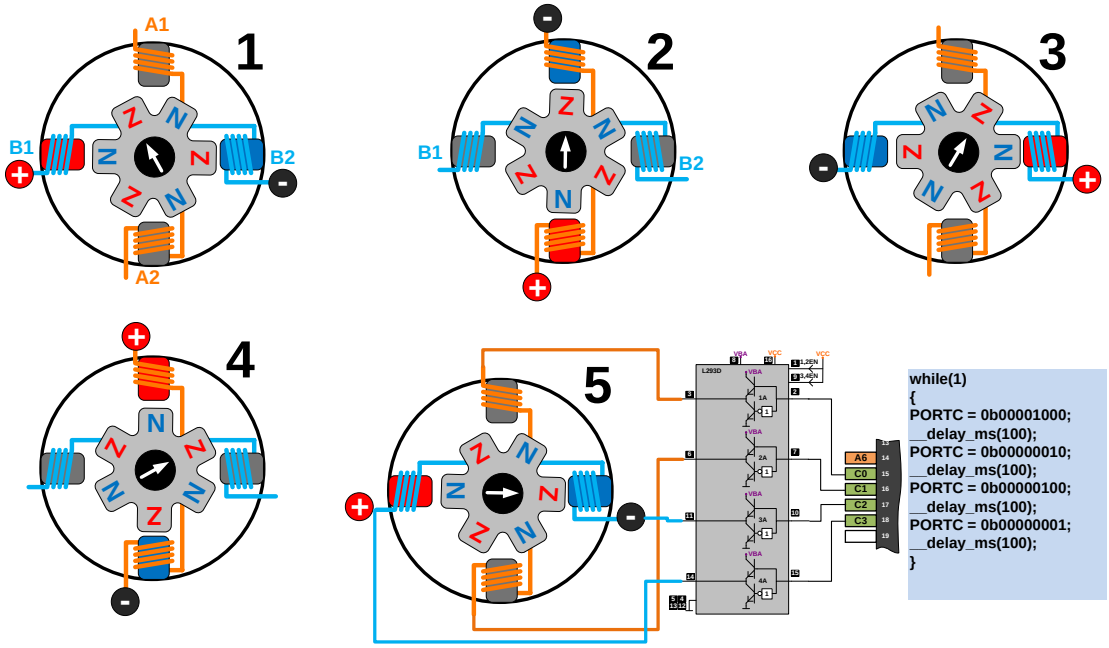
Unipolair met H-Brug – Middenaftakking aan Gnd



Bij de 4e optie hang je de middenaftakkingen aan GND en gebruik je nu de 4 bovenste transistoren van de H-brug om al dan niet stroom door de spoelen van de stappenmotor te laten vloeien.

Aansturing Bipolair Wave Step

Er zijn verschillende methodes om stappenmotoren aan te sturen. De ene methode verbruikt wat minder stroom, dan andere is dan weer sterker of nauwkeuriger. We bespreken er hier enkele.



Als eerste sturen we de bipolaire stappenmotor aan in Wave step. Hier onderscheiden we 4 stappen die we in een eeuwige loop blijven herhalen. Als eerste stap sturen we de stroom van B1 door de horizontale spoelen naar B2. De linkse statorspoel wordt een Zuidpool en trekt de dichtstbijzijnde Noordpool van de rotor aan – de rechtse spoel wordt een Noordpool en trekt de dichtstbijzijnde Zuidpool van de rotor aan. De rotor staat nu vast in deze positie tot we overgaan naar stap 2.

Bij stap twee sturen we een stroom van onder naar boven door de verticale spoel. We sturen geen stroom meer door de horizontale spoel. De verticale spoel induceert ook hier een noord en een Zuidpool en de rotor draait 30° omdat de dichtstbijzijnde noord en Zuidpool van de rotor hierdoor wordt aangetrokken.

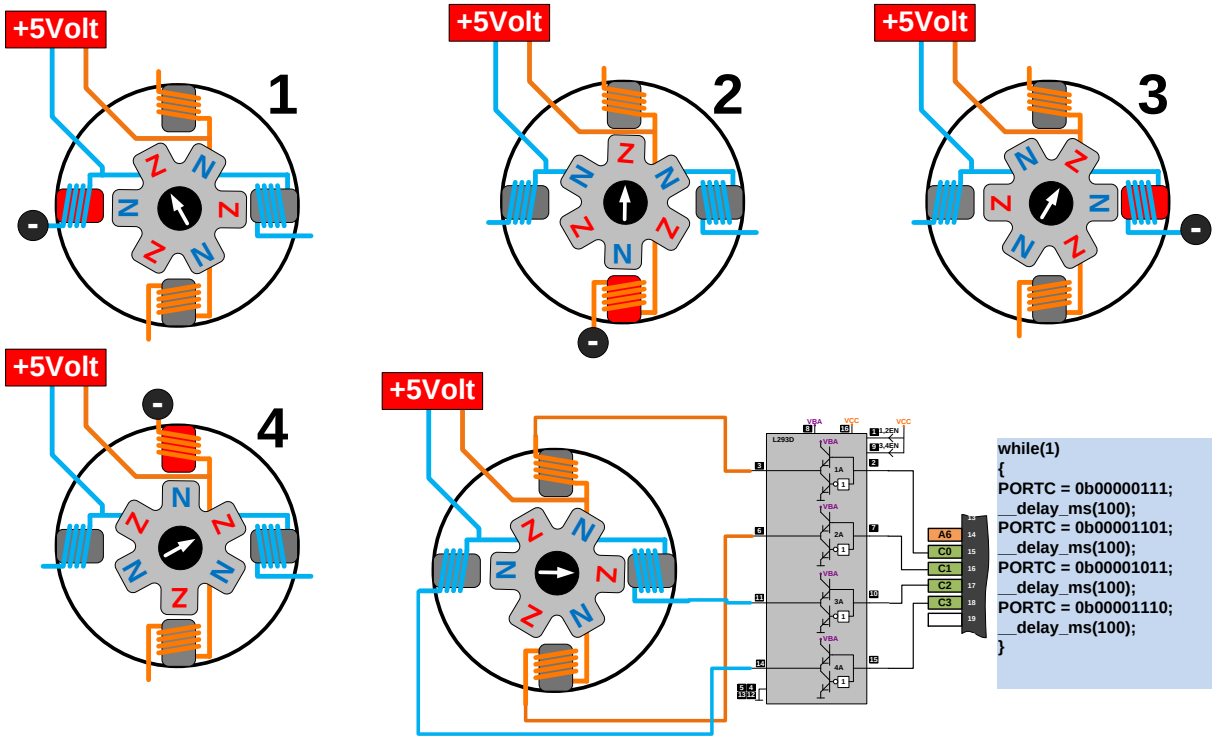
Bij stap 3 wordt terug de horizontale spoel van stap 1 bekrachtigd, maar nu wordt de stroom er via de H-brug in de andere richting door gestuurd. De rechterkant van de stator wordt nu Zuidpool en de linkerkant wordt Noordpool. De rotor draait weerom 30°.

En bij de 4^{e} stap wordt deze keer de verticale spoel omgekeerd aangestuurd t.o.v. de 2^{e} stap. Ook hier verdraait de rotor weer 30°. Na 4 stappen is deze motor met slechts 6 rotormagneten 120° verdraaid. Na de 4^{e} stap beginnen we terug bij stap 1.

Er wordt telkens maar 1 van de twee fasen aangestuurd – dat heeft als gevolg dat we half zoveel stroom verbruiken, maar dat de motor ook maar half zo krachtig is in vergelijking met een aansturing waarbij we beide spoelen gelijktijdig bekrachtigen.

Als we de stappenmotor aansluiten zoals hier getekend is. De verticale spoel op de bovenste H-brug van de L293 en de horizontale spoel op de onderste H-brug, dan zou de aansturing ervan via dit programma kunnen werken. Er wordt telkens slechts 1 uitgang hoog gemaakt. Tussen de stappen wachten we 100 msec om de motor de tijd te geven om te verdraaien. Deze delay kan je verkleinen om de draaisnelheid te vergroten. De maximale draaisnelheid en dus minimale delay is voor elke motor en elke belasting verschillend.

Aansturing Unipolair Wave Step



Als tweede bespreken we de wave step methode voor unipolaire stappenmotoren. We hebben gezien dat we unipolaire stappenmotoren op veel verschillende manieren kunnen aansluiten, maar hier hangen we de middenaftakkingen aan de 5 volt en we schakelen de 4 spoelen wel of niet naar gnd met de L293D IC.

In stap 1 wordt de linkse spoel aan gnd gelegd – er vloeit een stroom van de 5 volt voeding door de spoel naar de gnd – deze statorspoel wordt een Zuidpool en de dichtstbijzijnde Noordpool van de rotor wordt hierdoor aangetrokken.

Bij stap 2 wordt de onderste spoel als Zuidpool bekrachtigd – de rotor draait 30°.

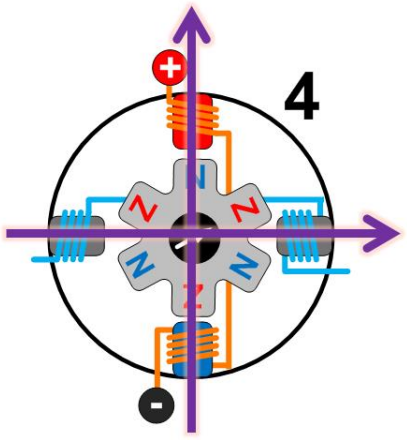
Bij stap 3 is de rechtse spoel een Zuidpool en de rotor draait weerom 30°.

En bij stap 4 wordt tenslotte de bovenste spoel als Zuidpool geïnduceerd en weerom draait de rotor hier 30°.

Vermits we hier telkens een 0 naar de L293D H-brug moeten sturen om de onderste transistor in geleiding te krijgen om zo de betreffende spoel naar gnd te kunnen koppelen en er een stroom door te laten vloeien, moeten we 1-en naar de spoelen sturen die we niet willen bekrachtigen. Ons programma lijkt zo heel sterk op het programma van de bipolaire in wave step, maar hier zijn de 1-en vervangen door 0-en en omgekeerd.

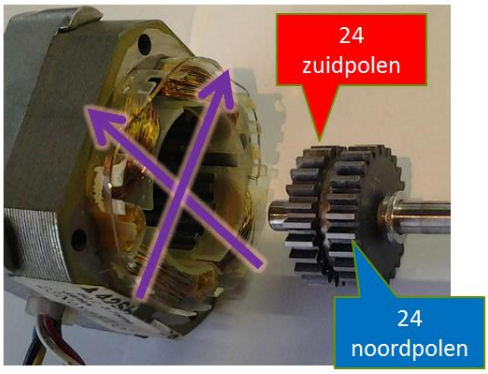
Aantal Graden Per Stap

Blijf wel in acht nemen dat de rotor in onze vereenvoudigde tekening slechts 6 rotorpolen heeft en dat de stapgrootte daardoor onrealistisch groot is.



- 2 stator-assen
- 6 rotor-polen

$$\frac{360}{2 \; x \; 6} = 30°/\text{stap}$$

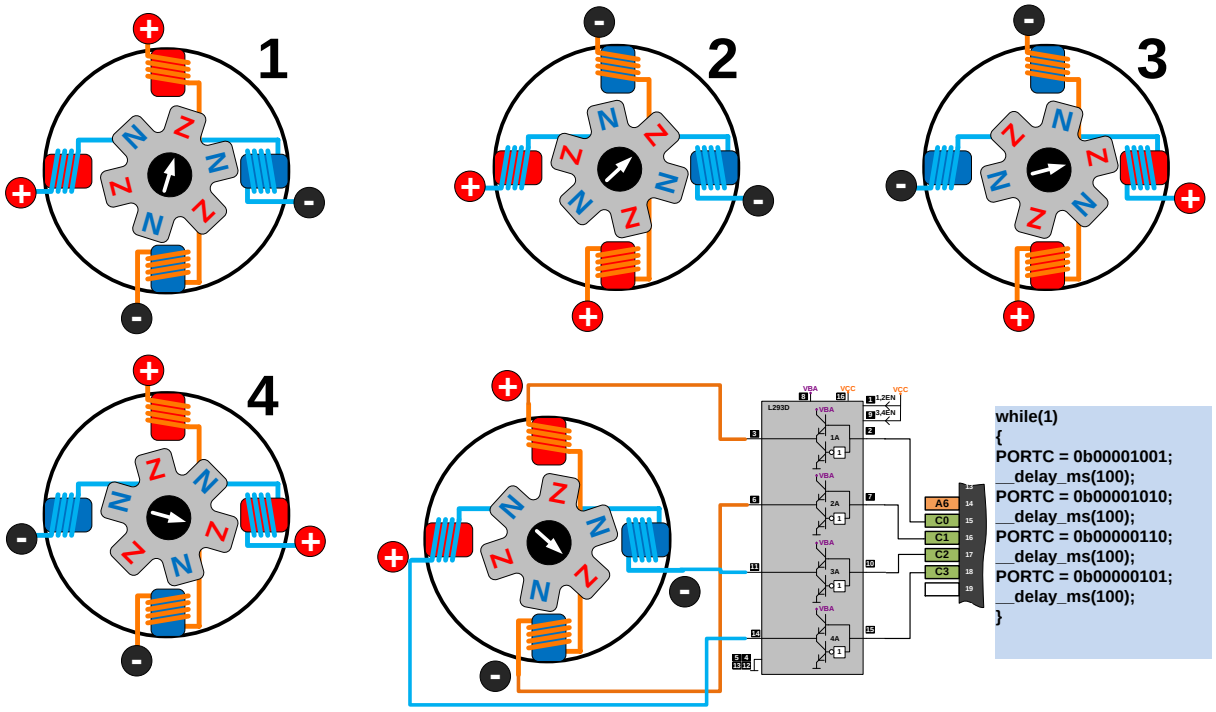


- 2 stator-assen
- 48 rotor-polen

$$\frac{360}{2 \; x \; 48} = 3{,}75°/\text{stap}$$

De stappenmotor die we uit elkaar gehaald hebben heeft bijvoorbeeld 24 zuidpolen en 24 noordpolen. Dat maakt samen 48 rotorpolen. Onze stappenmotor heeft 4 statorspoelen – samen in paren van 2 stator-assen, wat resulteert in een stapgrootte van slechts 3,75°/stap. 1,8°/stap is zo ook een veelvoorkomende waarde.

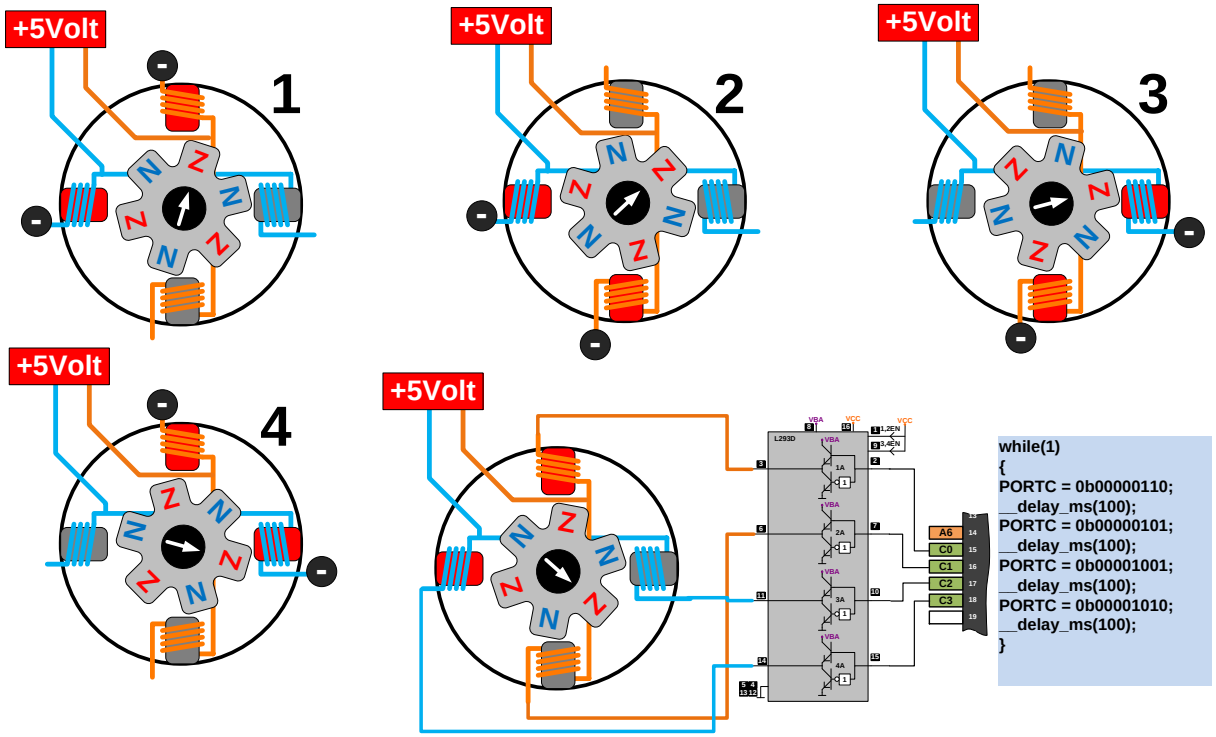
Aansturing Bipolair Full Step



In de eerste stap van de Full step aansturing valt al meteen op dat we nu beide spoelen van de stappenmotor bekrachtigen. Dat resulteert in twee zuidpolen en in twee noordpolen in de statorwikkelingen. De dichtstbijzijnde Noordpool van de rotor richt zich nu tussen de twee zuidpolen en de dichtstbijzijnde Zuidpool van de rotor richt zich tussen de twee noordpolen van de stator. Dubbel zoveel spoelen die nu moeten bekrachtigd worden betekent dubbel zoveel stroomverbruik als nadeel maar ook wel een dubbel zo krachtige positionering als voordeel. De stapgrootte blijft hier ook 30°/stap

Ook hier herkennen we 4 stappen, waarbij we duidelijk zien dat er telkens twee naast elkaar liggende statormagneten als Zuidpool fungeren en de andere twee als Noordpool. In ons programma moeten we dan ook telkens in 4 stappen en in de juiste volgorde twee uitgangspinnen hoog maken en de andere twee laag.

Aansturing Unipolair Full Step

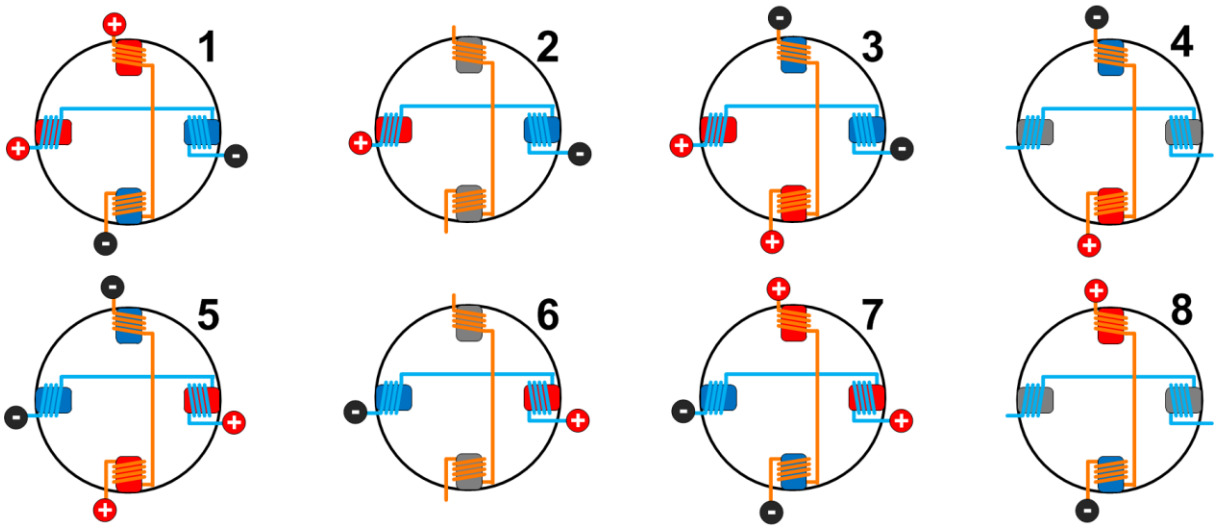


Ook unipolaire stappenmotoren kunnen in full step worden aangestuurd. Doordat de spoelen gevoed worden vanuit de middenaftakking kunnen er hier enkel halve spoelen bekrachtigd worden. Unipolaire stappenmotoren zijn dus minder sterk als gelijkaardige bipolaire stappenmotoren. Het voordeel is dan weer dat het bekrachtigen van halve windingen slechts de helft van het vermogen vraagt dan wanneer je de hele winding moet bekrachtigen.

Bij een unipolaire full step aansturing worden telkens 2 van de 4 ‘halve’ stator spoelen bekrachtigd zodat dit zuidpolen worden. De dichtstbijzijnde Noordpool van de rotor zal zich hier dan snel tussen richten. Ook hier worden er 4 stappen doorlopen.

Als we het programma bekijken, dan sturen we in deze situatie – met de unipolaire stappenmotor – met de middenaftakkingen aan de 5volt – en aangestuurd via een H-brug – een “0” naar de fasen die we willen bekrachtigen en een “1” naar de fasen die we niet willen bekrachtigen.

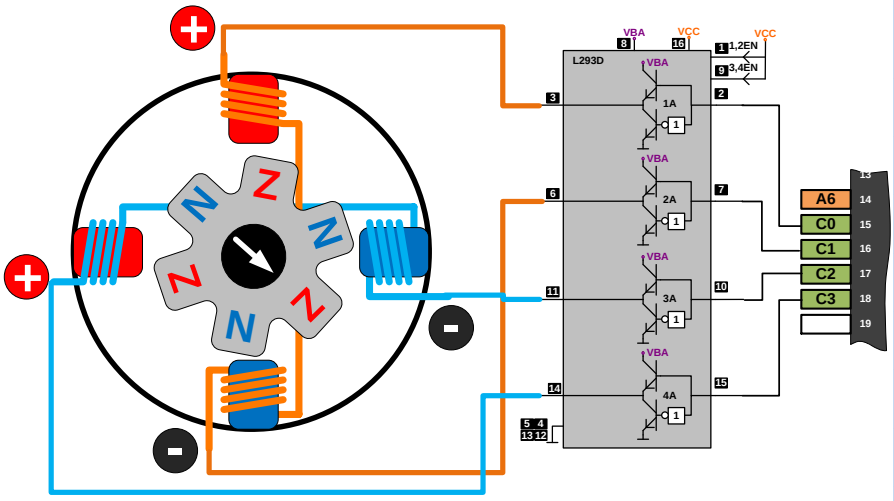
Aansturing Bipolair Half Step



Half step is een aanstuurmethode waarbij full step en wave step worden afgewisseld. Bij full step richt de rotormagneet zich tussen de twee statorspoelen in en bij wave step richt de rotor zich naar de statorspoel. Door beide methoden af te wisselen in 8 verschillende stappen kunnen we nu de stapgrootte halveren en de nauwkeurigheid van de stappenmotor zo verdubbelen.

Doordat er nu afwisselend 1 of twee fasen bekrachtigd worden, varieert ook het stroomverbruik en de kracht die de motor heeft.

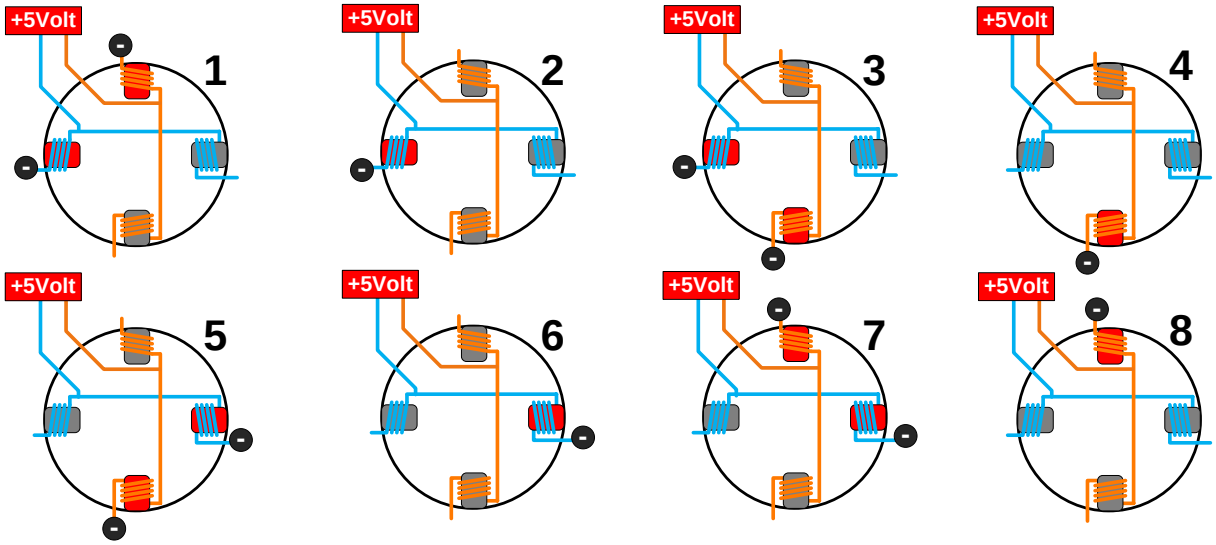
We kunnen dit echter niet meer demonstreren met onze rotor met 6 polen – dit werkt enkel met een rotor met veel meer polen – zoals dat in echte stappenmotoren het geval is, maar de 8 stappen om de statormagneten te induceren zijn hier wel correct voorgesteld.



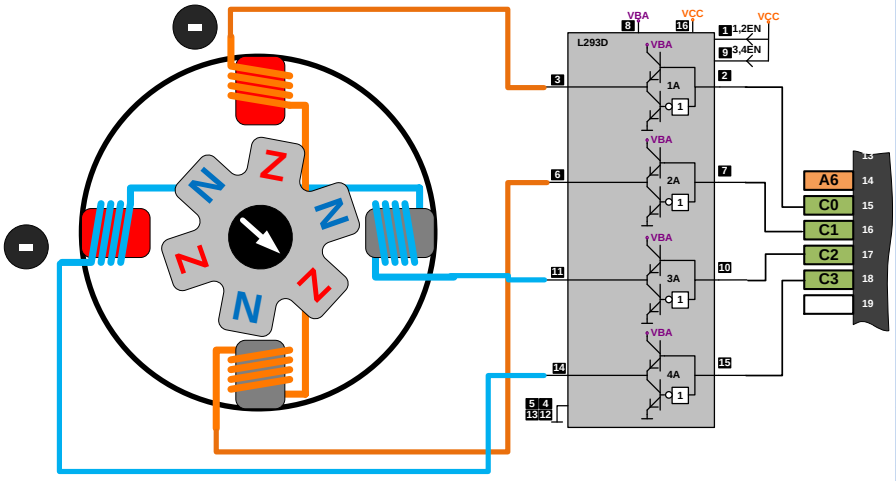

```
while(1)
{
PORTC = 0b00001001;
__delay_ms(100);
PORTC = 0b00001000;
__delay_ms(100);
PORTC = 0b00001010;
__delay_ms(100);
PORTC = 0b00000010;
__delay_ms(100);
PORTC = 0b00000110;
__delay_ms(100);
PORTC = 0b00000100;
__delay_ms(100);
PORTC = 0b00000101;
__delay_ms(100);
PORTC = 0b00000001;
__delay_ms(100);
}
```

Het programma bestaat nu ook uit 8 stappen i.p.v. uit 4 stappen. Afwisselend worden er 1 en dan weer twee spoelen bekrachtigd. De volgorde is ook hier heel belangrijk.

Aansturing Unipolair Half Step



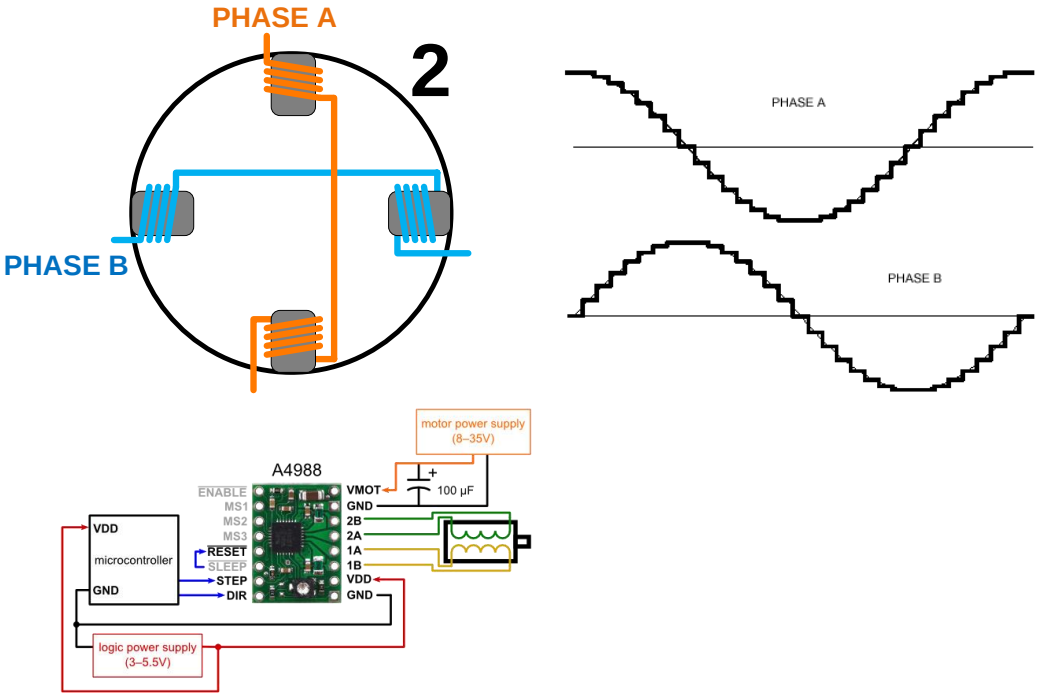
Ook unipolaire stappenmotoren kunnen in half step worden aangestuurd. Afwisselend wordt ook hier in 8 opeenvolgende stappen telkens één en dan weer twee halve fasen bekrachtigd. De fase of fasen die je wilt bekrachtigen moeten aan de gnd gekoppeld worden via de transistoren van de L293D H-brug. Deze onderste transistoren worden met een 0 in geleiding gezet.




```
while(1)
{
PORTC = 0b00000110;
__delay_ms(100);
PORTC = 0b00000111;
__delay_ms(100);
PORTC = 0b00000101;
__delay_ms(100);
PORTC = 0b00001101;
__delay_ms(100);
PORTC = 0b00001001;
__delay_ms(100);
PORTC = 0b00001011;
__delay_ms(100);
PORTC = 0b00001010;
__delay_ms(100);
PORTC = 0b00001110;
__delay_ms(100);
}
```

In het programma ziet u dan ook afwisselend één en twee 0-en in de 4 laagste bits van poort C die verbonden zijn met de L293 driver.

Aansturing via Microstepping



Microstepping is een manier van aansturen die gebruikt wordt om stappenmotoren nog nauwkeuriger te laten stappen. Bij voorgaande methodes was een fase ofwel wel, ofwel niet bekrachtigd – 1 of 0. Bij microstepping gaat men de statorspoelen bekrachtigen met analoge sinusvormige signalen. Zo neemt de ene fase het rustig over van de volgende fase en zal de rotor zich veel soepeler bewegen tussen de twee stappen. Op deze manier kunnen veel hogere resoluties gehaald worden met dezelfde stappenmotor – je maakt immers heel wat stapjes bij tussen de twee statorwikkelingen. De kracht is constant, het stroomverbruik is constant en de motor beweegt veel minder met schokjes en zorgt ook voor veel minder trillingen en de daarbij horende geluiden.

Deze methode kan enkel worden toegepast met speciaal daarvoor bestemde drivers – deze microstep drivers moeten immers door een microcontroller gestuurd kunnen worden met digitale signalen, maar moeten de stappenmotor wel kunnen besturen met analoge signalen – eigenlijk een soort DA omvormer.

Pololu onder andere heeft zo'n microstepping driver die heel veel gebruikt wordt in allerhande hobby en school-projecten. Ook veel Reprap 3D printers gebruiken deze Pololu microstep drivers. De aansluiting kan u hier boven zien. De aansturing is eenvoudig gemaakt – met 1 pin bepaal je de snelheid waarmee er ge-micro-stept moet worden, met de andere pin bepaal je de draairichting.

Samenvatting Stappenmotoren

Stappenmotoren zijn ideaal om een bepaalde positionering te doen

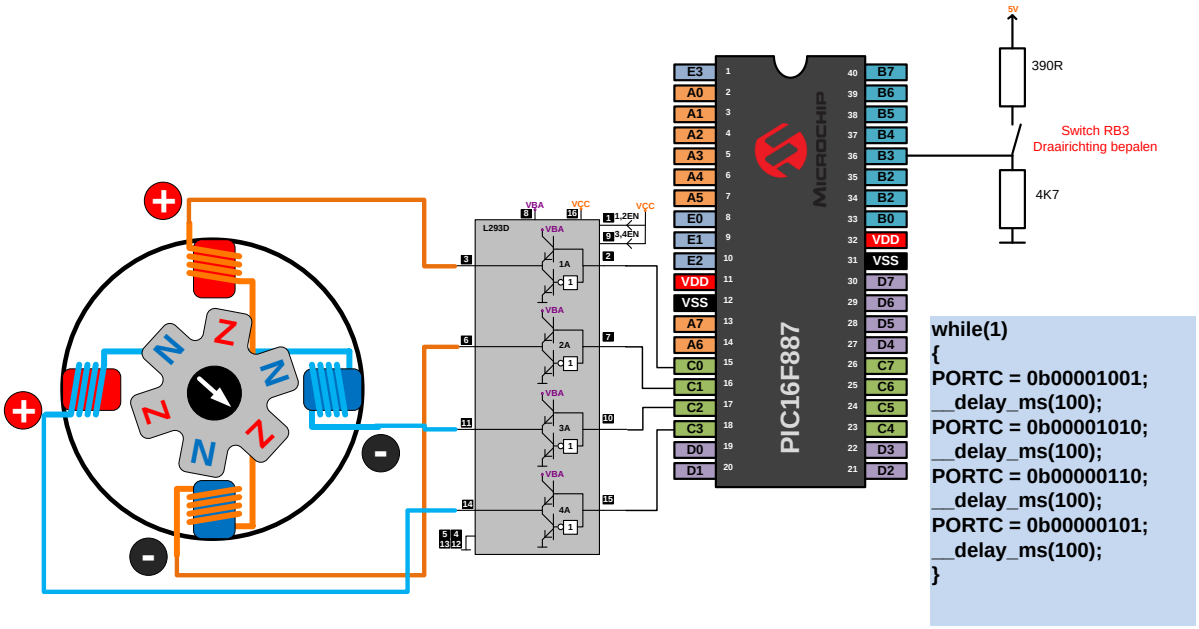
Stappenmotoren kunnen een bepaalde positie vasthouden

Stappenmotoren trekken relatief veel stroom en zijn daarom niet ideaal in batterijgevoede apparaten

Unipolaire stappenmotoren verbruiken slechts de helft van de stroom, maar zijn dan ook maar half zo krachtig als bipolaire stappenmotoren

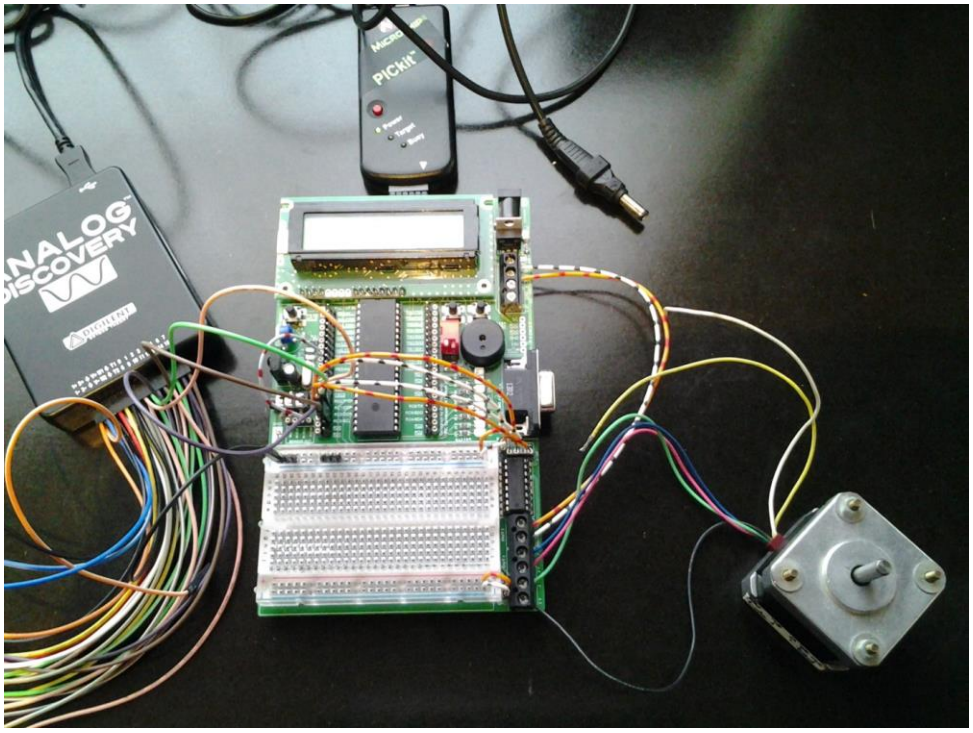
Elke aanstuurmethode: wave step, full step, half step en microstep hebben hun specifieke voor en nadelen

Meetopstelling Stappenmotor




```
while(1)
{
PORTC = 0b00001001;
__delay_ms(100);
PORTC = 0b00001010;
__delay_ms(100);
PORTC = 0b00000110;
__delay_ms(100);
PORTC = 0b00000101;
__delay_ms(100);
}
```

We gebruiken voor deze oefening een unipolaire stappenmotor die we als bipolaire zullen aansturen in Full Step mode. Met een schakelaar aan RB3 bepalen we de draairichting. Uitgangen C0 tot C3 worden versterkt door de L293 dubbele H-brug driver.



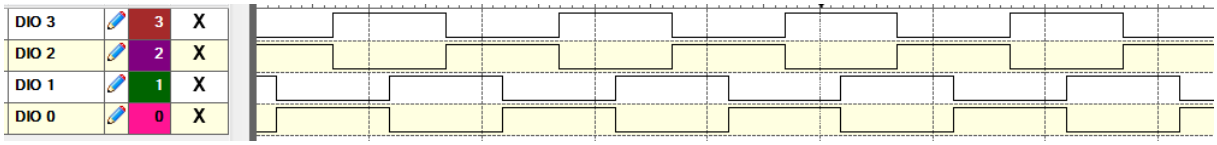
Op deze foto ziet u mooi hoe de stappenmotor is aangesloten op de H-brug. De H-brug wordt gevoed via de 12V adapter. De 4 pinnen C0-C3 van de uC besturen de 4 ingangen van de dubbele H-brug. Met SWB3 kan de draairichting bepaald worden.

Programma Stappenmotor Bipolair Full Step in C

```
/*********************************************************************
Videolessen Deel 4 - Bart Huyskens - RTC Antwerpen - 2013
Extern Xtal - 19660800Hz
L293D Stepper - C0-1A, C1-2A, C2-3A, C3-4A
Bipol stepper motor tussen 1Y en 2Y, 3Y en 4Y
Switch aan RB3 voor draairichting
*********************************************************************/

#include <htc.h>
#define _XTAL_FREQ 19660800

__CONFIG(0x23E2);   //Extern HS Xtal, WDT DIS, PWRT EN, MCLR EN, BOR EN,
INT/EXT DIS, LVP DIS, NO Code/data protect
__CONFIG(0x3FFF);   //BOR 4,0V, Write protect off

void main()
{
TRISC   = 0x00;                     // leds aan PORTC - Output
TRISB   = 0b00001000;               // RB3 = input -- switch

ANSEL   = 0b00000000;               // alle pins normal IO
ANSELH  = 0b00000000;               // alle andere AD pins normal IO

    while(1)
    {
        if (RB3)                    // als SWITCH aan RB3 is ingedrukt
            {
            PORTC = 0b00001001;     // STAP A
            __delay_ms(100);
            PORTC = 0b00001010;     // STAP B
            __delay_ms(100);
            PORTC = 0b00000110;     // STAP C
            __delay_ms(100);
            PORTC = 0b00000101;     // STAP D
            __delay_ms(100);
            }
        else                        // als SWITCH aan RB3 niet is ingedrukt
            {
            PORTC = 0b00000101;     // STAP D
            __delay_ms(100);
            PORTC = 0b00000110;     // STAP C
            __delay_ms(100);
            PORTC = 0b00001010;     // STAP B
            __delay_ms(100);
            PORTC = 0b00001001;     // STAP A
            __delay_ms(100);
            }
    }
}
```

Softwarematig houdt het aansturen van stappenmotoren weinig in. We controleren hier in een eeuwige loop of de schakelaar aan RB3 is ingedrukt. Als dat zo is, dan worden de 4 stappen van de FULL STEP mode in de volgorde ABCD uitgevoerd.

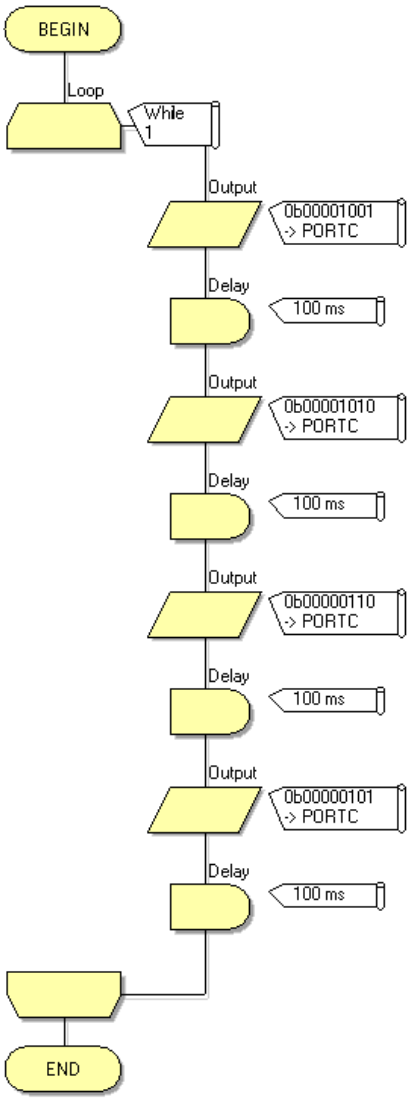
Als de schakelaar niet is ingedrukt dan worden de 4 stappen in de omgekeerde volgorde uitgevoerd. Tussen elke stap wordt er hier een delay van 100msec gebruikt. Deze delay kan veel kleiner om de snelheid op te drijven, maar de minimale delay hangt af van de stappenmotor en de belasting ervan.



Programma Stappenmotor Bipolair Full Step met Flowcode

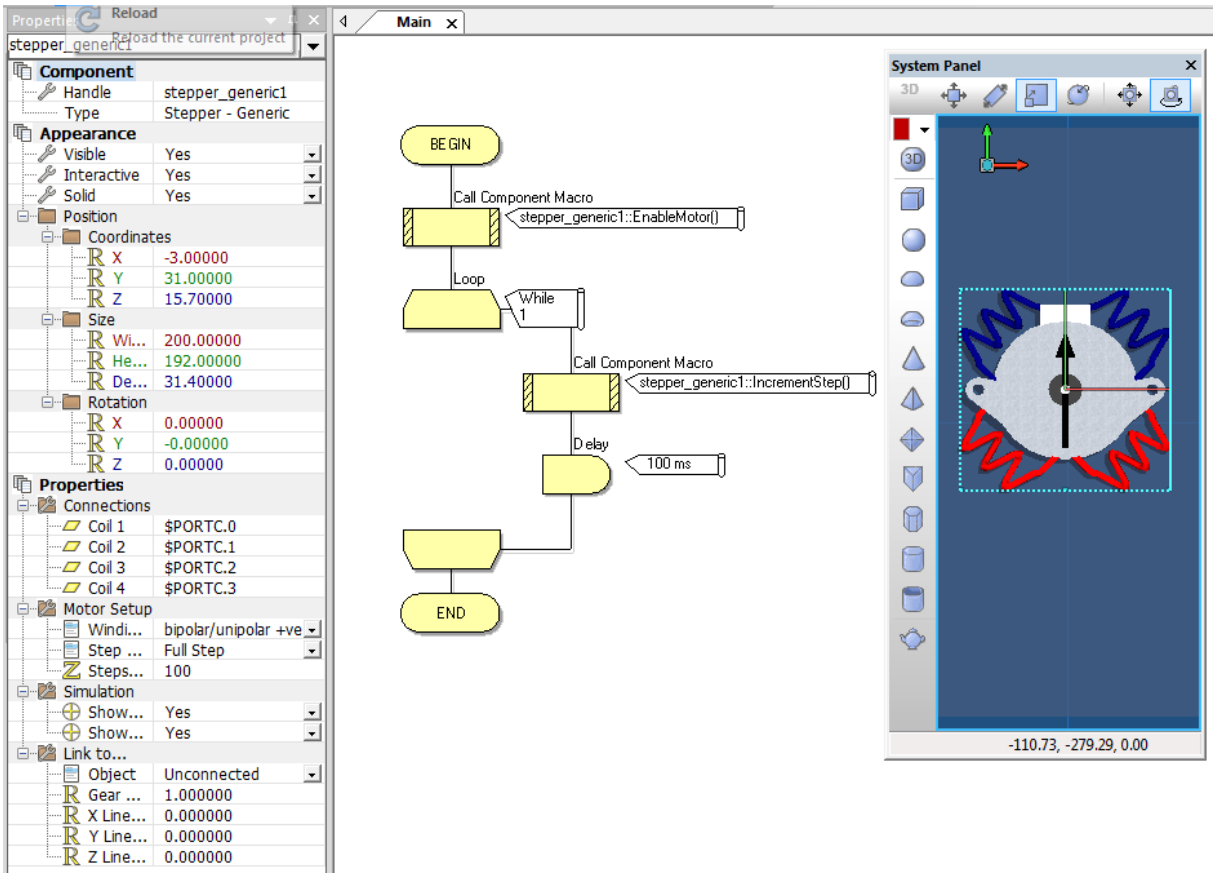
Om een stappenmotor aan te sturen met Flowcode heb je de keuze uit twee methodes.

Methode 1:



Om het programma kort te houden is er hier geen keuze van draairichting. Poort C wordt rechtstreeks aangestuurd – u herkent mooi de 4 stappen van de FULL STEP mode.

Methode 2:



Flowcode heeft een stappenmotor component aangemaakt. Deze kan je terugvinden onder het “mechatronics” menu. We hebben deze stappenmotor hier op het system panel geplaatst.

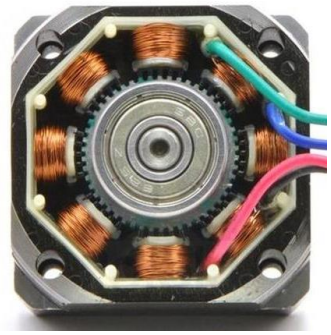
We passen als eerste de properties aan. Bij de motor setup kies je voor bipolair/unipolair +ve, Full step en je geeft bij steps het aantal stappen in dat uw stappenmotor nodig heeft voor een volledige omwenteling.

Bij connections selecteer je de pinnen van de uC die de stappenmotor besturen – voor ons is dat C0 tot C3.

In het programma zelf moeten we eerst de stepper component enabelen. Vervolgens komen we in een eeuwige loop waarin we telkens een “ 1 stap voorwaarts” of “increment step” instructie geven en de noodzakelijke vertraging van 100msec uitvoeren.

Uitdagingen Aansturing Stappenmotor

1. Neem een stappenmotor (reeds in bezit, gekocht, gedemonteerd uit afgedankt toestel) en bepaal hiervan de volgende zaken:
    - Is het een bipolaire of een unipolaire?
    - Hoeveel aansluitdraden zijn er?
    - Hoeveel stroom zal deze stappenmotor trekken uit de driver?
    - Welke drivermethode verkies je?
    - Teken het volledige aansluitschema – inclusief stappenmotor, vermeld de kleur van de draden, teken de driver en teken de microcontroller. Vergeet zeker niet om de gnd's van alle componenten aan elkaar te hangen.
2. Bereken de stapgrootte van deze stappenmotor. Houd er wel rekening mee dat je hier slechts de zuidpolen van de rotor kan zien – de tanden met de noordpolen zitten hier nog achter, maar zien er juist hetzelfde uit.



3. Test hoe snel je uw stappenmotor kan laten stappen – m.a.w. wat is de kleinste delay die je tussen twee stappen kan plaatsten – zonder stappen over te slaan.
4. Neem een stappenmotor naar keuze, meet alles na, sluit aan op een driver naar keuze en programmeer de microcontroller om er iets origineels mee te doen.

    Inspiratie:

    - Maak een analoge klok met secondewijzer
    - Laat de stappenmotor de stand van een analoge potmeter volgen (hiermee maak je een servo volgsysteem)
    - Bepaal met twee drukknoppen de positie van een stappenmotor
    - …

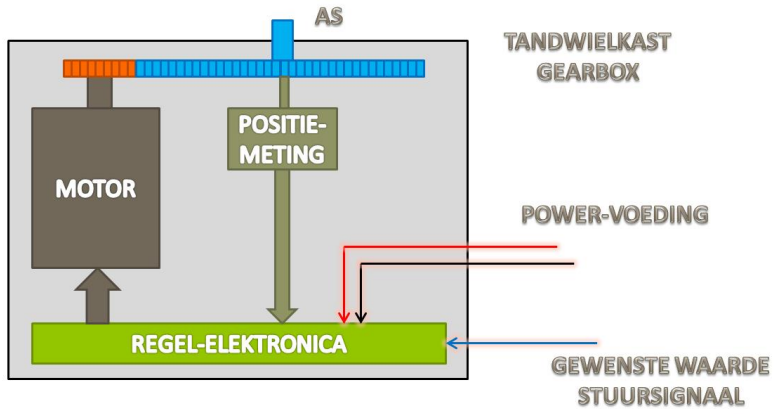
Servo Motoren

Werkingsprincipe Servo-Motoren

We willen hierbij meteen al duidelijk stellen dat servo-motoren geen afzonderlijk type motoren zijn, zoals DC motoren of stappenmotoren dat wel zijn. Een servo motor moet je eerder bekijken als een systeem dat gebruik maakt van een motor – zij het een DC, BLDC of een AC motor – en dat daarnaast onder andere ook nog bestaat uit een encoder en elektronica.

Servo motoren moet je ook niet bekijken als motoren die veel toeren ronddraaien. De motor stuurt namelijk meestal een tandwielkast aan die op haar beurt sterk vertraagd de uitgangsas aanstuurt. Dikwijls zal die uitgangsas maximaal 360° kunnen ronddraaien.

We leggen hier stap voor stap de principiële werking van een servomotor uit.



Als eerste heb je de motor. Bij goedkopere servo's is dit een gewone DC motor, bij duurdere een BLDC motor en bij zeer krachtige servo motoren kan dit een AC motor met frequentiesturing zijn.

Deze motor stuurt een tandwielkast of gearbox aan. Meestal bestaat die gearbox uit meer dan de twee tandwielen die hier getekend staan. De as die naar buiten komt is de as waar uiteindelijk de belasting aan wordt gekoppeld.

De positie van de uitgangsas wordt gemeten. Dit kan met een goedkope potentiometer gebeuren, maar voor nauwkeurigere metingen worden er optische encoders gebruikt. Bij duurdere servo systemen wordt ook de snelheid gemeten waarmee de uitgangsas beweegt.

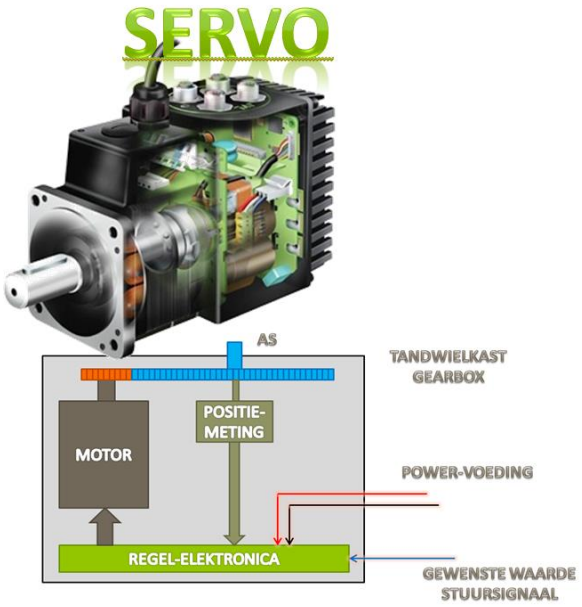
Het resultaat van de positiemeting wordt naar de regel-elektronica gestuurd. De elektronica vergelijkt de positiemeting met de gewenste waarde – ook het stuursignaal genoemd. Als de gewenste waarde afwijkt van de gemeten waarde, dan wordt de motor bijgestuurd in de juiste richting tot de as de gewenste positie bereikt.

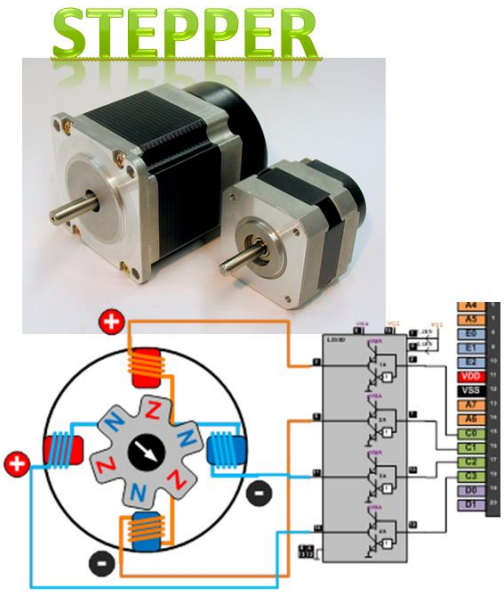
Vanzelfsprekend moet het geheel ook gevoed worden. Al deze componenten zitten samen in één behuizing.

Dit type van bijsturing of regeling wordt een closed loop controle genoemd omdat alle componenten – de motor, de tandwielkast, de positiemeting en de elektronica in een gesloten kring constant op elkaar blijven inwerken.

Toepassing Servomotoren

Het toepassingsgebied van servomotoren is positionering. Daarmee komen servomotoren in het vaarwater van de stappenmotoren die eveneens gebuikt worden voor positionering.





Doordat servomotoren bestaan uit een samenstelling van een motor, gearbox, sensor en elektronica zijn deze toch vrij duur.

Stappenmotoren hebben dan weer als voordeel dat ze zonder toepassing van extra elektronica en sensoren toch een nauwkeurige positionering kunnen leveren, maar dat is ook net hun zwakte. Een stappenmotor die – als is het maar heel kortstondig – te zwaar belast wordt, zal stappen verliezen, zonder dat de stuurelektronica daar ook maar iets van weet. Alle volgende stappen vanaf dat moment zijn dus fout. Je zou dit kunnen oplossen door de stappenmotor te over-dimensioneren – door deze dus veel zwaarder te maken dan eigenlijk noodzakelijk om zo een mogelijke overbelasting aan te kunnen, maar dan begint de meerprijs van de over-dimensionering dikwijls zwaarder te wegen als de meerprijs voor een servo systeem. Je zou de stappenmotor ook kunnen uitrusten met een encoder, en zo een closed loop controle maken, maar dan heb je in principe zelf een servo systeem gemaakt.

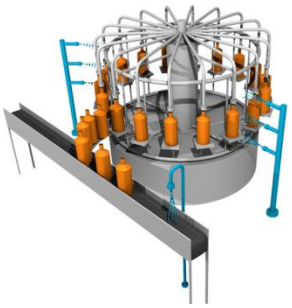
Vooral wanneer de belastingen groter worden en er een grote nauwkeurigheid en snelheid vereist is kiest men voor de closed loop zekerheid van servo motoren.

Industriële Servomotoren

Industriële servomotoren zijn, zoals reeds gesteld, niet goedkoop. Ze worden toegepast in machines waar er een grote nauwkeurigheid en een grote snelheid vereist wordt voor wisselende belastingen.

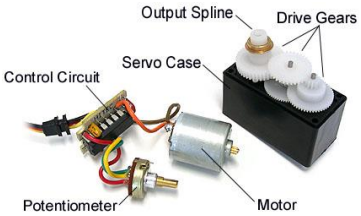


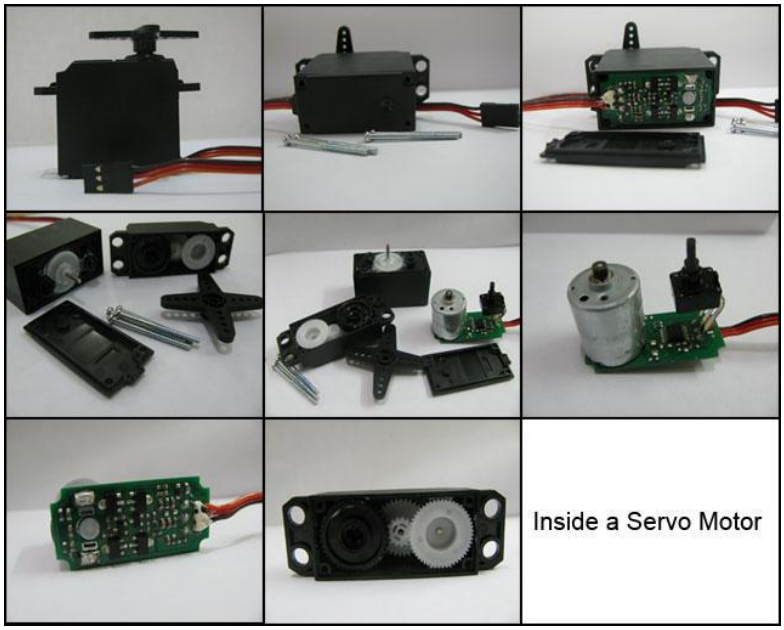
Ze worden daarom veel toegepast bij de aansturing van de assen in CNC machines en onder andere ook bij transportbanden waar er een nauwkeurige positionering vereist is.

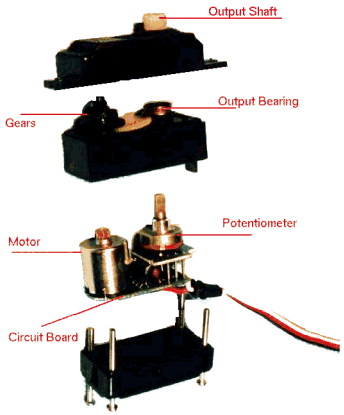



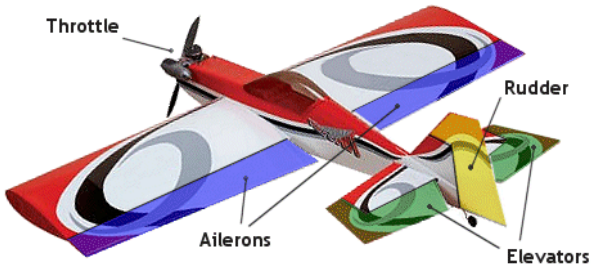
Modelbouw Servomotoren

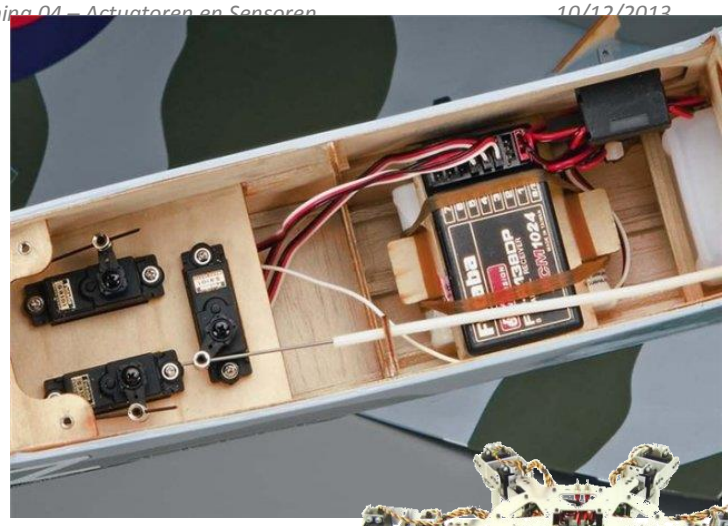
In modelbouw wordt er ook veelvuldig gebruik gemaakt van servomotoren. Deze servomotoren zijn veel kleiner, maar werken volgens exact hetzelfde principe als de grote industriële broertjes. Voor maximaal 15€ heb je al een goede hobby-servomotor. Als motor wordt er meestal een goedkope DC motor gebruikt, de tandwielkasten zijn van kunststof en de encoder is een gewone potentiometer om de prijs laag te houden.

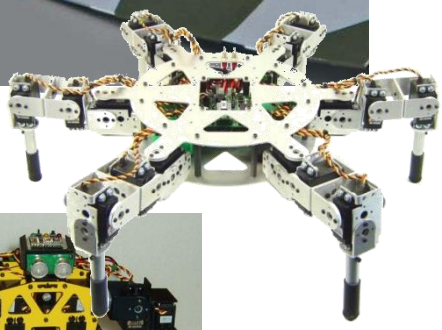










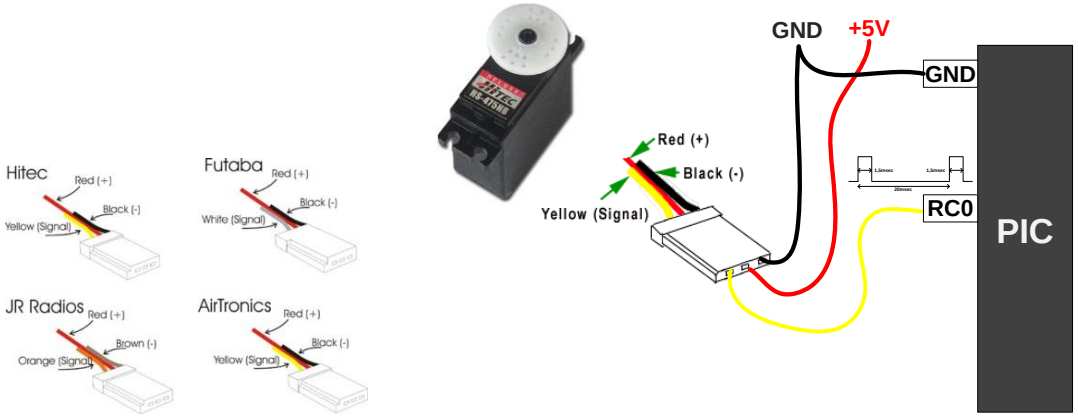


Zo worden servo motoren onder andere gebruikt om bij tele-geleide vliegtuigen de hoogte en richtingsroeren te bewegen, ze worden gebruikt om de wielen van auto's te besturen en in allerlei wandelende robots zoals je hier de biped en de hexapot kan zien.

De lage inkoopprijs en de relatief goede nauwkeurigheid en kracht maken deze modelbouw servo's uitermate geschikt om te gebruiken in allerhande schoolprojecten.

Aansturing Modelbouw Servo's

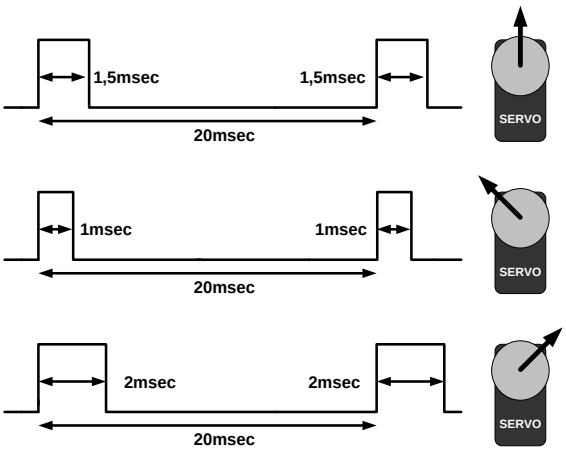
Uit een modelbouw servomotor komen meestal 3 draden. Afhankelijk van het merk kunnen de kleuren van de draden verschillen.



De rode draad is meestal de + van de voeding. Meestal worden servo's gevoed met 6volt, maar 5 volt volstaat ook. De rode draad is meestal de middelste van de 3 om te beveiligen tegen ompoling. Het stroomverbruik is afhankelijk van de belasting en het type servo. Wij rekenen meestal 500mA/servo.

Een van de andere draden – meestal de zwarte of bruine- is de massa. Let er op dat de massa van uw voeding ook aan de massa van de microcontroller MOET hangen.

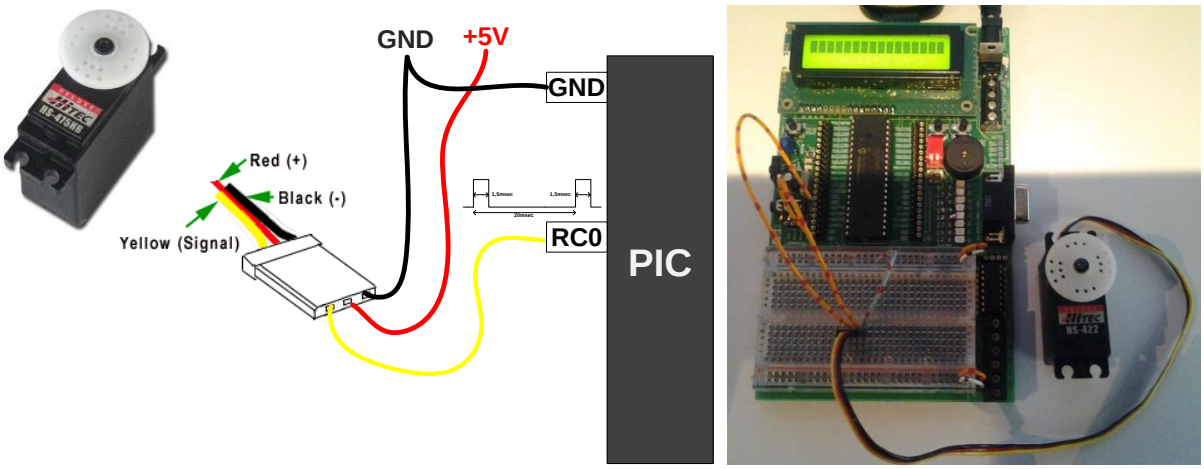
De derde draad is dan de stuurdraad. Hierop wordt het stuursignaal met de gewenste positie naar de servo gestuurd. Meestal wordt dit stuursignaal gegenereerd door een microcontroller.



Het signaal om een servo-motor aan te sturen bestaat uit een puls van 5 volt die tussen de 1 à 2 msec duurt en dan een laag-signaal van tussen de 18 à 19 msec zodat de totale periode altijd 20msec duurt. Als de aan-tijd 1,5msec duurt, en de uit tijd dus 18,5msec, dan positioneert de servo zich in de centrale stand. Met een puls van 1 msec positioneert de servo zich in de uiterst linkse positie en met een puls van 2 msec positioneert de servo zich in de uiterst rechtse positie. Vanzelfsprekend zijn alle andere posities ook te bereiken door alle mogelijke tussenwaarden tussen deze 1 en 2msec.

De 1 en 2 msec als uitersten zijn richtwaarden. Afhankelijk van merk en type kan dit ook tussen de 0.8msec en 2.2 msec liggen als je de absolute uitersten wil bereiken.

Testopstelling Modelbouw-servomotoren



We sluiten de signaaldraad van de servo aan op pin RC0 van de microcontroller. De gnd van de servo wordt aan de gnd van de voeding gehangen EN aan de gnd van de uC. Tenslotte hangen we de rode draad van de servo aan de 5V van de voeding.

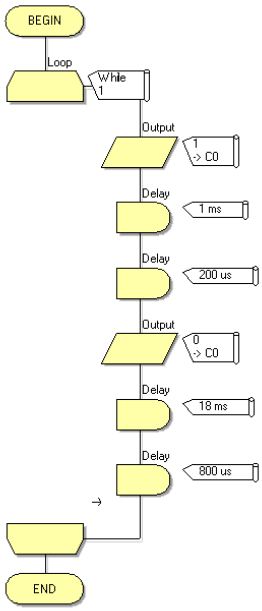
Testprogramma Servo in C

In dit éénvoudige testprogramma maken we pin C0 hoog voor 1,2msec en laag voor de overige 18,8msec om aan een totale periodetijd van exact 20msec te komen. De delays zijn hier samentellingen van msec delays en usec delays. Deze pulsen moeten wel blijven komen in een eeuwige loop om de servo op deze positie vast te houden. Test dit programma uit en probeer de servo-as maar eens te verdraaien – je zal merken dat er hier redelijk wat kracht achter zit.
Als je de reset knop ingedrukt houdt, dan kan je de servo-as wel verdraaien.

```
/*********************************************************************
Videolessen Deel 4 - Bart Huyskens - RTC Antwerpen - 2013
Extern Xtal - 19660800Hz
servomotor gevoed tussen GND en +5V
signaal servo 1,2msec hoog, 18,8msec laag op pin C0
*********************************************************************/

#include <htc.h>
#define _XTAL_FREQ 19660800

__CONFIG(0x23E2);   //Extern HS Xtal, WDT DIS, PWRT EN, MCLR EN, BOR EN,
INT/EXT DIS, LVP DIS, NO Code/data protect
__CONFIG(0x3FFF);   //BOR 4,0V, Write protect off

void main()
{
TRISC   = 0x00;          // PORTC - Output
while(1)
    {
    RC0 = 1;             // C0 hoog
    __delay_ms(1);
    __delay_us(200);     // 1,2msec
    RC0 = 0;             // C0 hlaag
    __delay_ms(18);
    __delay_us(800);     // 18,8msec

    }
}
```

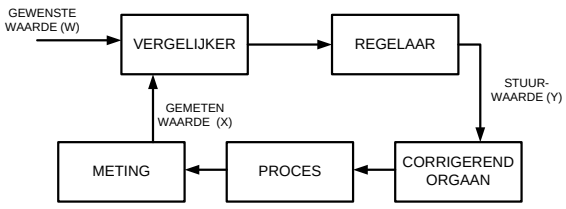
Testprogramma Servo in Flowcode

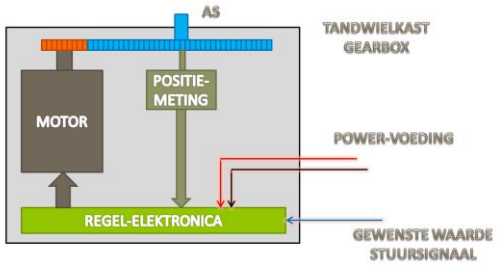
Het Flowcode programma verloopt identiek hetzelfde als het C programma – C0 hoog voor 1.2msec en laag voor 18.8msec.



Uitdagingen Servomotor

1. Hier ziet u het blokschema van een standaard regelkring. Duid alle onderdelen van de standaard regelkring aan op het blokschema van de servo-motor.

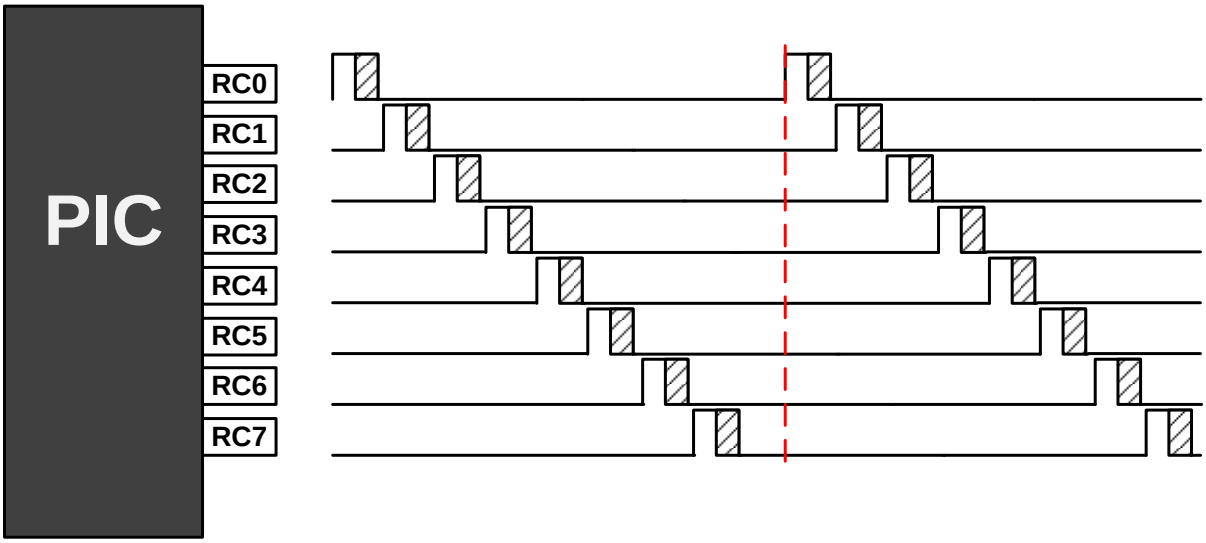




2. Zoals eerder vermeld liggen de uitersten van positieve pulsen tussen de 1 en 2msec niet vast voor elk type of merk van servomotor. Pas het demo programma aan om deze grenzen voor uw type servomotor te bepalen. M.a.w. wat is de waarde voor uiterst links en wat is de waarde voor uiterst rechts?

Programma Servomotor – Maximaal 8 Servo's

Het programma dat we in de vorige les gebruikt hebben om een servo motor aan te sturen was niet echt volgens de regels van de programmeerkunst. De uC was zo constant bezig met het genereren van 1 servo signaal en kon daarnaast niets anders meer doen. Door gebruik te maken van Timers en interrupts kan je wel meerdere servo's gelijktijdig aansturen en daarnaast kan de uC nog andere taken blijven uitvoeren. We bespreken in deze les de seriële methode om maximaal 8 servomotoren aan te sturen met 1 uC. We koppelen in dit geval de 8 servo's aan de 8 pinnen van poort C.

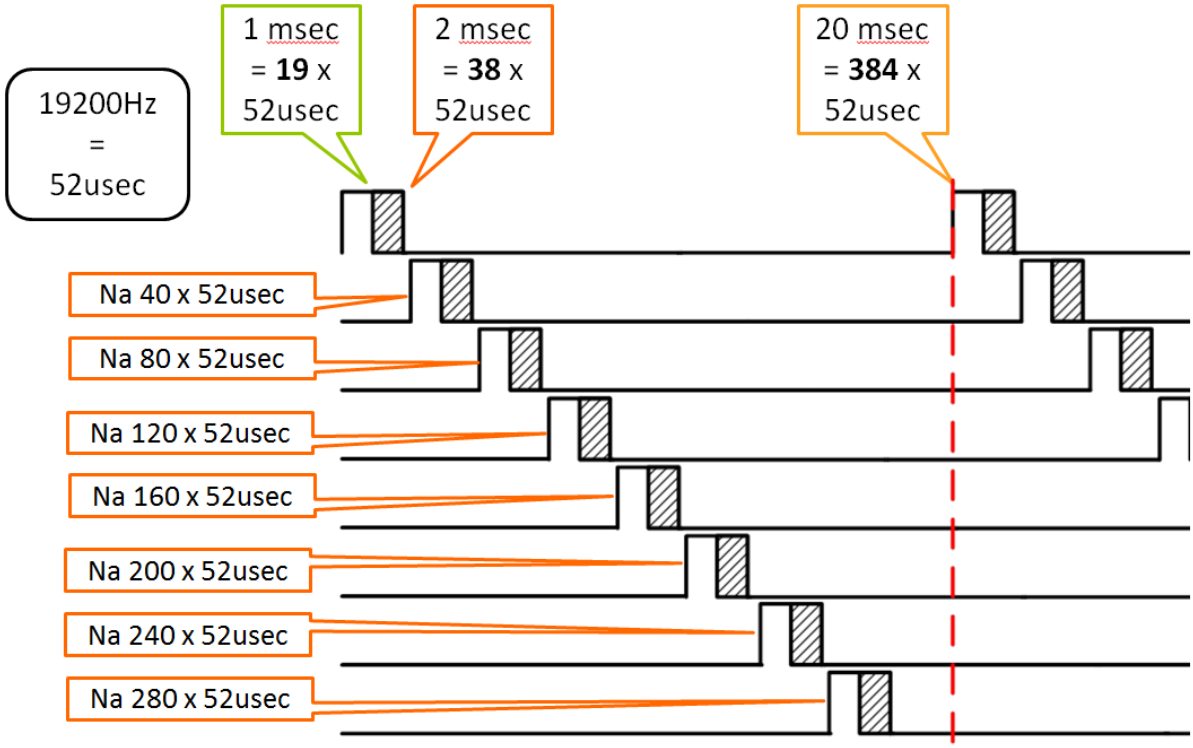


Een positieve puls duurt steeds minimaal 1 msec, met een maximum van ongeveer 2 msec. De totale periode is altijd 20msec (50Hz). Als we het op deze manier aanpakken, dan kunnen we de volgende positieve puls pas beginnen als de vorige is afgelopen. Dat geeft ons theoretisch de tijd om 10 pulsen van 2msec binnen de 20 msec te krijgen alvorens we terug aan de eerste puls moeten beginnen. Praktisch is 8 servo's meestal het hoogst haalbare op deze manier. Omdat alle servo signalen na elkaar worden gegenereerd wordt deze methode ook wel eens de seriële methode genoemd.

Programma 8 Servo's Flowcode

Zie programma FC (te groot voor schermafdruk)

In ons programma stellen we TMR0 in om interrupts te genereren tegen 19200Hz. Dat betekent dat er elke 52usec een interrupt het hoofdprogramma onderbreekt en dat de interruptroutine "SERVO" dus ook elke 52 usec wordt uitgevoerd.



Als we met deze 52usec blijven rekenen, dan wordt het 1msec moment gepasseerd na 19 interrupts en het 2msec punt na 38 interrupts. Wij kunnen met dit programma de servo motor dus laten bewegen tussen de waarde 19 en 38, maar we nemen meteen al 18 en 39 omdat we weten dat de meeste servo's wel wat buiten de 1 en 2msec hun grenzen hebben. We hebben hier dus een resolutie van 22 stappen waarmee we de servo op 22 verschillende hoekstanden kunnen fixeren.

Het 20msec punt wordt dan gepasseerd nadat de interrupt routine 384 maal doorlopen is. Van dit punt begint alles terug opnieuw.

Elke volgende servo mag pas starten nadat de vorige is afgelopen. De maximale waarde van puls 0 is 39. Puls 1 kan dus starten vanaf 40 en loopt tot 40 + 39 = 79. Puls 2 kan starten vanaf het moment dat de interrupt routine 80 maal doorlopen is, puls 3 na 120, puls 4 na 160, puls 5 na 200, puls 6 na 240 en puls 7 na 280.

We bekijken even het Flowcode programma en zien dat TMR0 wordt ingesteld om interrupts te genereren op de interne klok met een prescaler rate van 1:1 wat maakt dat deze timer tegen een frequentie van 19200Hz interrupts zal genereren – 1 interrupt om de 52usec dus. De interrupt routine die elke 52usec wordt aangeroepen heeft als naam SERVO gekregen.

Via een array van 8 bytes – de array ON_TIME[8] kunnen we de 8 posities van de 8 verschillende servo's instellen tussen de waarden 18 en 39. In het calculation symbol bepalen we deze waarden. Servo[0] positioneren we zo in de uiterst linkse positie en servo [7] in de uiterst rechtse positie. Alle andere servo's liggen hier tussen.

De interruptroutine SERVO wordt elke 52usec aangeroepen en uitgevoerd. In de servo routine wordt er eerst gecontroleerd of de teller- variabele TWENTY_MS reeds de waarde 384 bereikt heeft wat betekent dat de 20msec gepasseerd zijn. 52usec x 384 is namelijk 20msec. Als dat zo is, dan moet pin C0 hoog worden. En dan wordt de teller- variabele TWENTY_MS terug op 0 gezet. Let op dat de teller variabele hier als unsigned integer gedeclareerd is, anders zou deze variabele slechts waarden tot 255 kunnen bevatten.

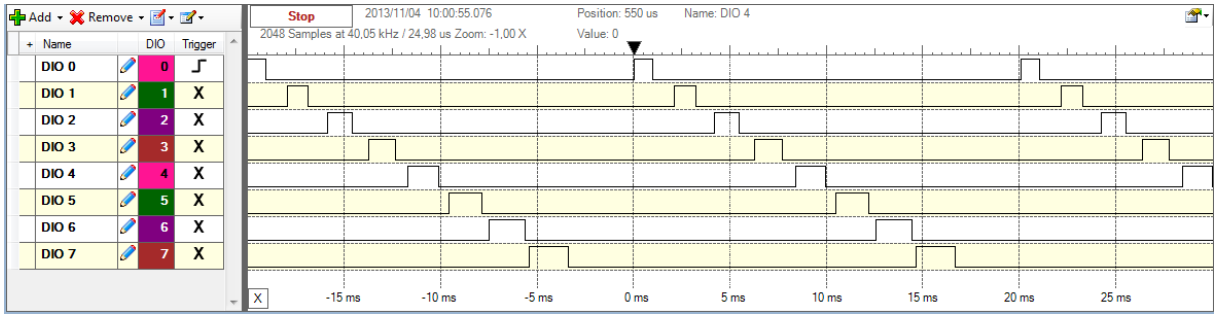
Telkens als de interrupt routine wordt aangeroepen – elke 52usec dus, wordt de teller- variabele TWENTY_MS met één verhoogd.

Er wordt gecontroleerd of TWENTY_MS reeds gelijk is aan ON_TIME[0] - dit is de waarde die we hebben ingesteld op 18 waarmee we servo 0 in de uiterst linkse positie willen zetten. Als dat zo is, na 18 maal de interrupt routine doorlopen te hebben, dan wordt pin C0 terug laag gemaakt.

Als de interruptroutine 40 maal doorlopen is wordt pin C1 hoog gemaakt en na (ON_TIME[1] + 40) wordt deze terug laag gemaakt om servo 1 te positioneren.

Het servo 2 signaal wordt dan hoog gemaakt na 80 maal 52usec en wordt terug laag na (ON_TIME[2] + 80) en zo blijft dit doorgaan tot alle 8 servo's bestuurd zijn en tot de interruptroutine uiteindelijk 384 maal doorlopen is. Van dat moment start alles terug van voor af aan.

Hier ziet u de meting op de pinnen C0 t/m C7. U ziet mooi hoe alle servo signalen worden opgebouwd. De periode is mooi 20msec voor alle servo's. De aan tijd van servo 0 is 1 msec en de aan-tijd van servo 7 is 2msec – alle andere waarden liggen hier tussenin.



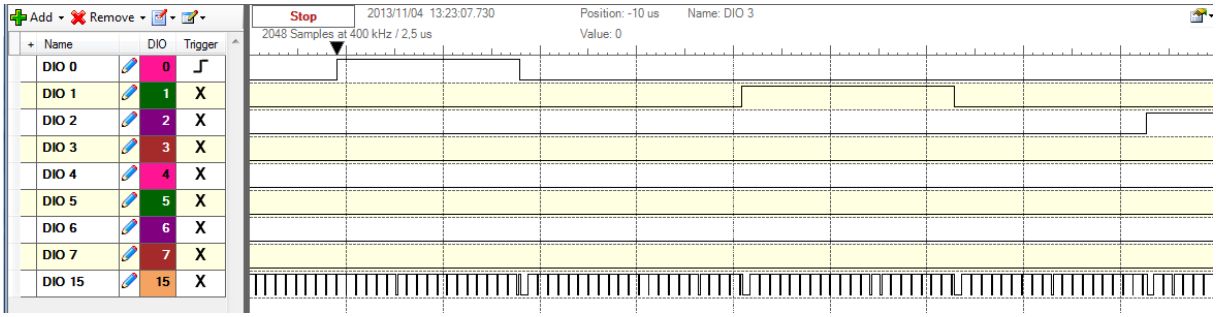
Programma 8 Servo's in C

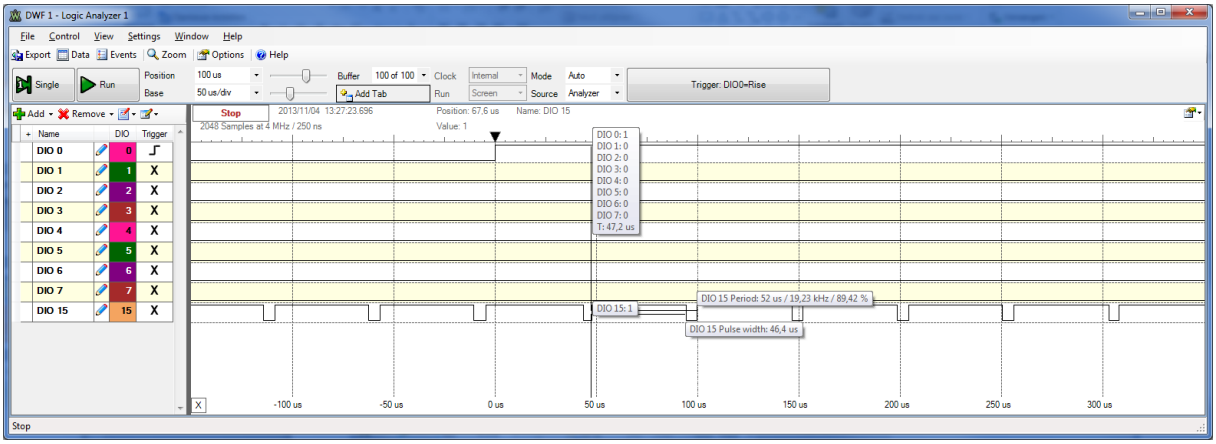
```
/*********************************************************************
Videolessen Deel 4 - Bart Huyskens - RTC Antwerpen - 2013
Extern Xtal - 19660800Hz
8 servo's aan 8 pinnen van PORTC
int freq TMR0 op 19200Hz - 52usec
*********************************************************************/

#include <htc.h>
#define _XTAL_FREQ 19660800

__CONFIG(0x23E2);   //Extern HS Xtal, WDT DIS, PWRT EN, MCLR EN, BOR EN, INT/EXT DIS, LVP DIS, NO Code/data protect
__CONFIG(0x3FFF);   //BOR 4,0V, Write protect off

unsigned char ON_TIME[8]={18,21,24,27,30,33,36,39};      // 8 byte array - positie 8 servo's tussen 18 en 39

static unsigned int TWENTY_MS = 0;                       // counts timer0 overflows (19200 x / second)
void main()
{
TRISC = 0x00;               // configure PORTC as output
OPTION_REG =  0b00001000;   // PSA op 1 - prescaler TMR0 op 000 - int freq = 19200Hz - 52usec
T0IE = 1;                   // Timer 0 interrupt enable
GIE = 1;                    // Global interrupt enable
while(1) { }
}

/***** INTERRUPT SERVICE ROUTINE *****/
void interrupt isr(void)
{
T0IF = 0;                           //  clear TOIF (TMR0 int Flag)
TWENTY_MS = TWENTY_MS+1;            // increment variable

if (TWENTY_MS == 384)               // 20msec passed
{
TWENTY_MS = 0;
RC0 = 1;                                        // Servo 0
}
if (TWENTY_MS == ON_TIME[0]){RC0 = 0;}

if (TWENTY_MS == 40){RC1 = 1;}                  // Servo 1
if (TWENTY_MS == (ON_TIME[1]+40)){RC1 = 0;}

if (TWENTY_MS == 80){RC2 = 1;}                  // Servo 2
if (TWENTY_MS == (ON_TIME[2]+80)){RC2 = 0;}

if (TWENTY_MS == 120){RC3 = 1;}                 // Servo 3
if (TWENTY_MS == (ON_TIME[3]+120)){RC3 = 0;}

if (TWENTY_MS == 160){RC4 = 1;}                 // Servo 4
if (TWENTY_MS == (ON_TIME[4]+160)){RC4 = 0;}

if (TWENTY_MS == 200){RC5 = 1;}                 // Servo 5
if (TWENTY_MS == (ON_TIME[5]+200)){RC5 = 0;}

if (TWENTY_MS == 240){RC6 = 1;}                 // Servo 6
if (TWENTY_MS == (ON_TIME[6]+240)){RC6 = 0;}

if (TWENTY_MS == 280){RC7 = 1;}                 // Servo 7
if (TWENTY_MS == (ON_TIME[7]+280)){RC7 = 0;}
}
```

Hierboven ziet u exact hetzelfde programma in C. TMR0 wordt hier ook ingesteld voor een interrupt frequentie van 19200Hz en in de interrupt routine worden alle 8 servo signalen stap voor stap en na elkaar opgebouwd. Na 20msec of 384 interrupts start alles terug van het begin.

Nadeel Seriële Sturing

Op de manier dat we het hier geprogrammeerd hebben duurt de interruptroutine 46,4usec – dat hebben we gemeten met de logic analyzer door aan het begin van de interrupt routine RD0 hoog te maken en deze op het einde terug laag te maken. We meten RD0 op kanaal 15 van de logic analyzer. We zien dat RD0 het grootste deel van de tijd hoog is wat er op wijst dat de interruptroutine bijna zo lang duurt als de interrupt-periode. Dit is een gevaarlijke situatie want wat gebeurt er als een interrupt ge-interrupt wordt door een interrupt…..

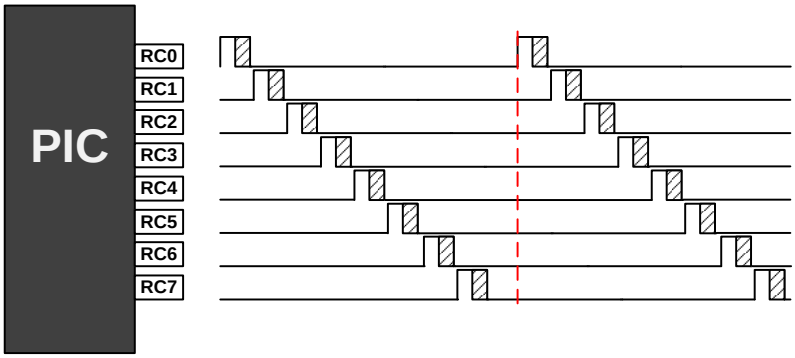




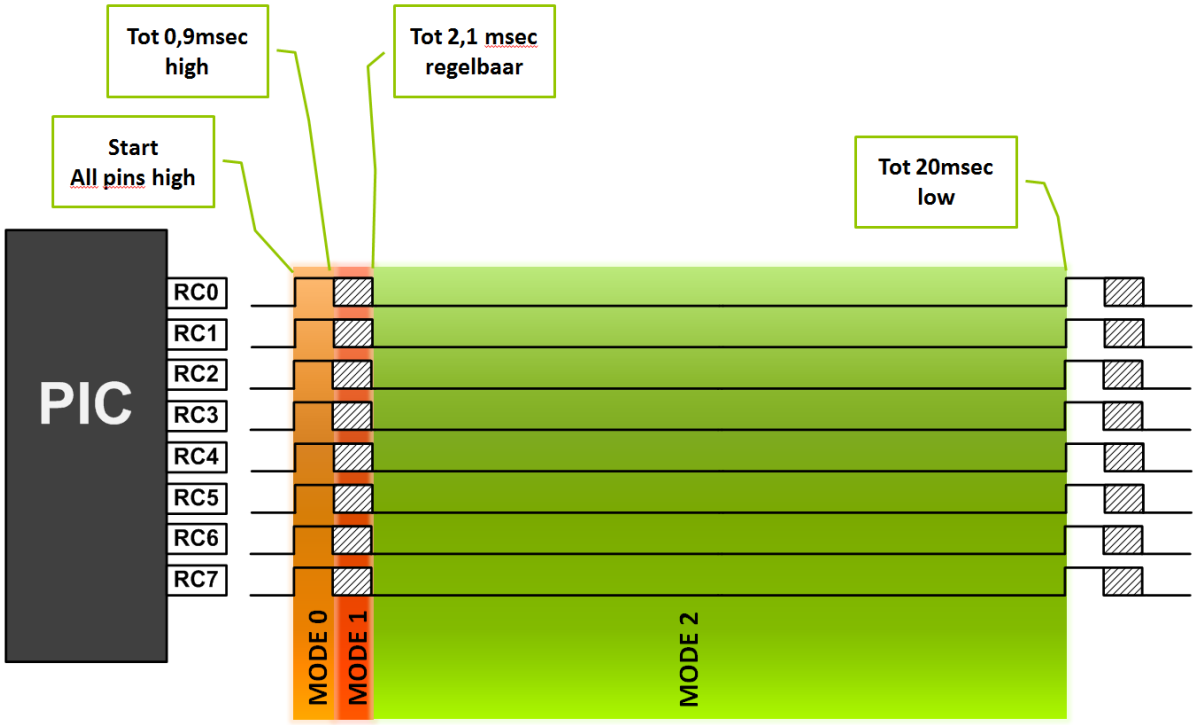
Uitdagingen <=8 Servo's

1. Test het volledige programma uit (C of FC)
2. Schrijf een extra routine die er voor zorgt dat als je waarden voor ON_TIME[] ingeeft die kleiner zijn dan 18 of die groter zijn als 39, dat de waarden dan beperkt blijven tot 18 en 39.
3. Laat alle 8 servo motoren een wave maken (zoals in stadions gedaan wordt)
4. Gebruik een oscilloscoop of logic analyzer en meet – door een bepaalde pin hoog en laag te maken – hoe lang de interrupt routine duurt. Kan jij de code van de interruptroutine verbeteren zodat die minder lang duurt?

Programma Servomotor - Meer dan 8 Servo's



Onze vorige methode – ook wel eens de seriële methode genoemd omdat de signalen na elkaar worden opgebouwd – was beperkt tot het aansturen van een 8-tal servomotoren. De voortdurende en lange interrupt-routines vormden ook een zware belasting op het totale programma.

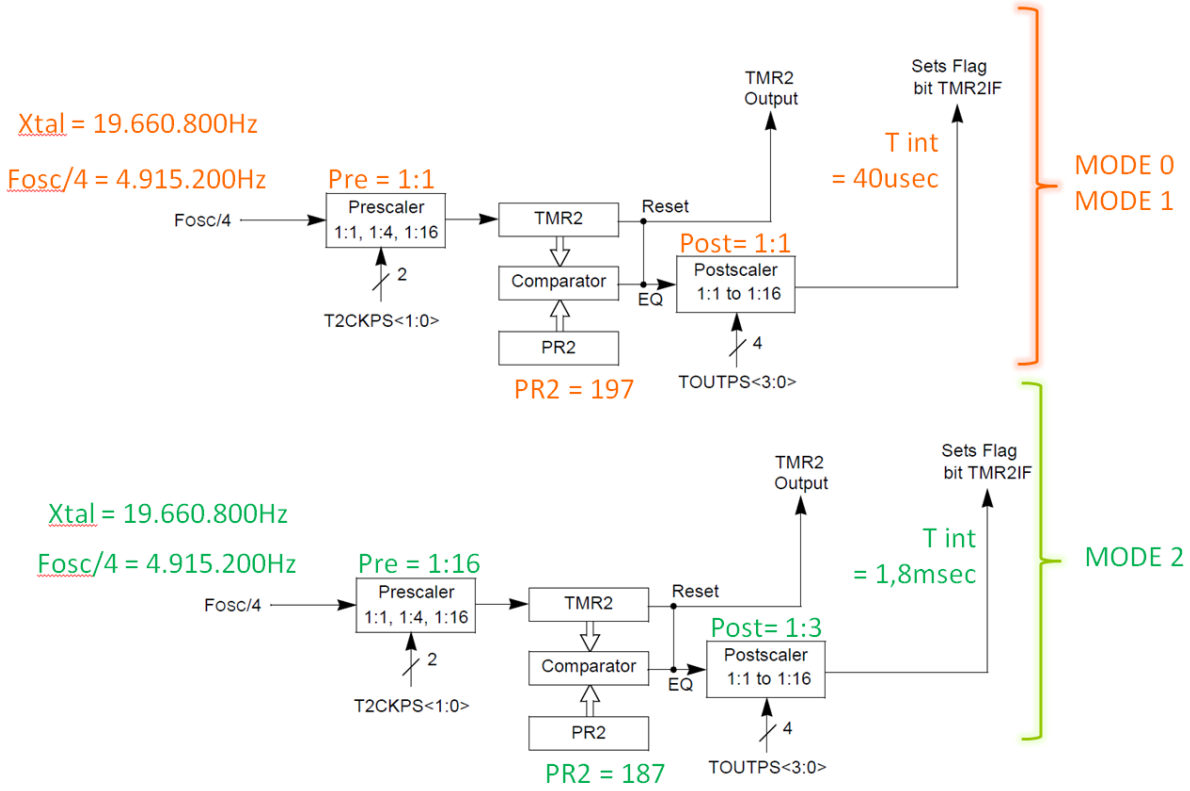


De methode die we in deze les willen voorstellen is de parallelle methode. Hierbij is het aantal aan te sturen servo's in principe onbeperkt. Zoals u op de tekening kan zien worden alle servopulsen gelijktijdig hoog gemaakt. Onze servo is regelbaar tussen 0,9 en 2.1msec. Tot aan de 0.9msec blijven alle signalen hoog. Tussen 0,9msec en 2.1msec moeten alle servosignalen afzonderlijk regelbaar zijn. Al deze servosignalen worden hier op één of ander moment laag. Tot aan de 20msec blijven alle servosignalen laag om vanaf dan weer allemaal samen gelijktijdig terug hoog te worden.

Daarom verdelen we deze sturing in 3 modes. In mode 0 zijn alle servosignalen altijd hoog. In mode 1 moet de microcontroller hard werken – alle servosignalen moeten op het juiste moment laag worden. In mode 2 tenslotte moet het signaal 17,9msec laag blijven. 17,9 msec is een eeuwigheid voor de microcontroller en deze tijd is ideaal voor de microcontroller om met andere zaken bezig te zijn of om in energiebesparende sleepmode te gaan.

We hebben er daarom voor gekozen om tijdens mode 0 en 1 een zeer snelle timer interrupt te gebruiken, om zo een nauwkeurige servo-positionering te doen en tijdens mode 2 vertragen we de interruptfrequentie zodat het hoofdprogramma niet nodeloos door interrupts onderbroken wordt gedurende de 17,9msec low time.

We gebruiken hier TMR2 vermits deze de hoogste interruptfrequentie kan genereren en ook omdat de interruptfrequentie dankzij de vele instelmogelijkheden van TMR2 vrij exact in te stellen is.



De interrupt-frequentie van 40usec die we tijdens mode 0 en 1 gebruiken bereiken we door bij TMR2 de prescaler en de postscaler op 1 te zetten en het PR2 register te laden met 197. TMR2 telt nu op tegen een frequentie van 4.915.200Hz. Als TMR2 197 bereikt, dan wordt er een interrupt gegenereerd. Deze interrupt komt dan om de 40usec.

De veel tragere interrupt van 1.8msec bekomen we door de prescaler op 16 in te stellen. Van elke 16 klokpulsen die van de 4.915.200Hz klok komen wordt er nu slechts één puls doorgelaten naar TMR2. Als TMR2 de waarde 187 bereikt zal die een compare match naar de postscaler doorgeven. De postscaler hebben we op 3 gezet dus per 3 compare-matches wordt er een interrupt gegenereerd. De interrupts tijdens mode 2 komen nu om de 1.8msec. Na 10 interrupts bereiken we de gewenste 18msec laag -tijd.

Programma meer dan 8 Servomotoren in C

```
/*****************************************************************************************************************
Videolessen Deel 4 - Bart Huyskens - RTC Antwerpen - 2013
Extern Xtal - 19660800Hz
16 servo's - 8 aan PORTC en 8 aan PORTD

        MODE 0      MODE 1                                  MODE 2          MODE 0      MODE 1
        0.9msec     0-1.2msec                               17.8msec        0.9msec     0-1.2msec
        *********|***************                                         *********|***************
        *                       *                                         *
        *                       *                                         *
*********                       *******************************////*************
/*****************************************************************************************************************///

#include <htc.h>
#define _XTAL_FREQ 19660800 // 4MHz

__CONFIG(0x23E2);   //Extern HS Xtal, WDT DIS, PWRT EN, MCLR EN, BOR EN, INT/EXT DIS, LVP DIS, NO Code/data protect
__CONFIG(0x3FFF);   //BOR 4,0V, Write protect off

static unsigned char TELLER_TMR2 = 0;           // counts timer2 overflows
static unsigned char MODE = 0;                  // 3 modes om servo signaal op te bouwen
static unsigned char SERVO[16] = {2,4,6,8,10,12,14,16,18,20,22,24,26,28,29,30};     // getal tussen  2 en 30 geeft positie servo

void main()
{
TRISC = 0x00;                       // configure PORTC as output - 8 servo's
TRISD = 0x00;                       // configure PORTD as output - 8 more servo's
ANSEL = 0x00;                       // make all analog inputs digital outputs
ANSELH = 0x00;                      // make all analog inputs digital outputs
TMR2IE = 1;                         // laat int van TMR2 toe
PEIE = 1;                           // laat peripheral interrupts toe
GIE = 1;                            // laat global interrupts toe

T2CON = 0b00000100;                 //post 1:1, pre 1:1; TMR2 on// fast 40usec TMR2 int freq setting
PR2 = 197;                          // 197 maakt int freq van 40usec

    while(1)
    { // later kan hier code worden uitgevoerd, maar best enkel als mode 2 actief is.
    }
}

void interrupt isr(void) /***** INTERRUPT SERVICE ROUTINE *****/
{
TMR2IF = 0;         // clear TMR2 int Flag

if (MODE ==0)                   // eerste 0.9msec hoog
    {
        if (TELLER_TMR2 < 23) { }           // doe niets zolang er geen 0.9msec gepasseerd zijn
        else                                // 0.9mesc passed - go to mode 1
        {
        MODE = 1;
        TELLER_TMR2 = 0;                    // teller terug op 0 zetten
        }
    TELLER_TMR2++;                          // verhoog teller met 1
    }

else if (MODE ==1)              // regelbare hoog tijd tussen 0.9 en 2.1 msec
    {
    TELLER_TMR2++;
        if (TELLER_TMR2 == SERVO[0]) {RC0= 0; }         // maak servo[x] signaal terug laag
        if (TELLER_TMR2 == SERVO[1]) {RC1= 0; }         // maak servo[x] signaal terug laag
        if (TELLER_TMR2 == SERVO[2]) {RC2= 0; }         // maak servo[x] signaal terug laag
        if (TELLER_TMR2 == SERVO[3]) {RC3= 0; }         // maak servo[x] signaal terug laag
        if (TELLER_TMR2 == SERVO[4]) {RC4= 0; }         // maak servo[x] signaal terug laag
        if (TELLER_TMR2 == SERVO[5]) {RC5= 0; }         // maak servo[x] signaal terug laag
        if (TELLER_TMR2 == SERVO[6]) {RC6= 0; }         // maak servo[x] signaal terug laag
        if (TELLER_TMR2 == SERVO[7]) {RC7= 0; }         // maak servo[x] signaal terug laag
        if (TELLER_TMR2 == SERVO[8]) {RD0= 0; }         // maak servo[x] signaal terug laag
        if (TELLER_TMR2 == SERVO[9]) {RD1= 0; }         // maak servo[x] signaal terug laag
        if (TELLER_TMR2 == SERVO[10]) {RD2= 0; }        // maak servo[x] signaal terug laag
        if (TELLER_TMR2 == SERVO[11]) {RD3= 0; }        // maak servo[x] signaal terug laag
        if (TELLER_TMR2 == SERVO[12]) {RD4= 0; }        // maak servo[x] signaal terug laag
        if (TELLER_TMR2 == SERVO[13]) {RD5= 0; }        // maak servo[x] signaal terug laag
        if (TELLER_TMR2 == SERVO[14]) {RD6= 0; }        // maak servo[x] signaal terug laag
        if (TELLER_TMR2 == SERVO[15]) {RD7= 0; }        // maak servo[x] signaal terug laag

        if (TELLER_TMR2 == 30)          // geeft einde van mode 1 aan
            {
            MODE = 2;                       // ga naar mode 2 (18msec laag)
            TELLER_TMR2 = 0;                // teller terug op 0 zetten
            T2CON = 0b00010110;     //post 1:3, pre 1:16; TMR2 on  // SLOW 1.79msec  TMR2 int freq setting
            PR2 = 187;              // interrupt freq op 1.79msec na 10x = 1.8msec
            }
    }

else if (MODE ==2)              // altijd 17.9msec laag - andere bewerkingen zijn nu wel mogelijk
        {
            TELLER_TMR2++;
            if (TELLER_TMR2 < 10) { }   // not yet equal to servo_0 delay (0-1.2msec)
            else                    // 0.9mesc passed - go to mode 1
            {
            MODE = 0;                   // terug naar mode 0
            TELLER_TMR2 = 0;            // reset teller
            PORTC = 0xFF;               // make all servo signals High
            PORTD = 0xFF;               // make all servo signals High


            T2CON = 0b00000100;     //post 1:1, pre 1:1; TMR2 on// fast 4.679usec TMR2 int freq setting for mode 0 and 1
            PR2 = 197;              // zet klok terug in fast mode - int freq = 40usec
            }
        }
}
```

SERVO[0]	2
SERVO[1]	4
SERVO[2]	6
SERVO[3]	8
SERVO[4]	10
SERVO[5]	12
SERVO[6]	14
SERVO[7]	16
SERVO[8]	18
SERVO[9]	20
SERVO[10]	22
SERVO[11]	24
SERVO[12]	26
SERVO[13]	28
SERVO[14]	30
SERVO[15]	32

De variabele 'TELLER_TMR2' en 'MODE' zijn gewone 8 bit variabelen – MODE wordt geladen met 0 omdat we in deze mode starten. De variabele SERVO[16] is een array van 16 variabelen die allemaal de naam SERVO hebben, maar allemaal een verschillende index hebben van 0 tot 15. Zo kan je in 1 regel 16 variabelen declareren en vullen met waarden tussen 2 en 30. Het gebruik van een array i.v.m. 16 afzonderlijke variabelen heeft als bijkomend voordeel dat het manipuleren van de posities van de 16 servo's ook veel sneller en krachtiger kan gebeuren. Deze

16 variabelen bepalen de positie van de 16 servomotoren die we met dit programma gelijktijdig kunnen besturen. De waarde 2 staat voor uiterst links, de waarde 30 voor uiterst rechts.

Nota: Ideaal had geweest als we hier een waarde tussen 0 en 255 hadden kunnen ingeven om zo de servo in 255 verschillende posities te zetten, maar dan moet de interruptfrequentie zo hoog worden dat we tussen 2 interrupts niet meer voldoende tijd hebben om de interruptcode uit te voeren. Tijdens mode 1 zitten we echt aan de grens van de mogelijkheden van onze uC.

We maken hier van alle pins van PORTC en PORTD uitgangen en met de ANSEL registers zetten we alle analoge functies van alle pins uit.

We laten interrupts van TMR2, Peripheral interrupts en Global interrupts toe.

In het T2CON register zetten we de presclaer en de postscaler van TMR2 beide op 1 en we vullen het PR2 register van TMR2 met 197 om de interrupt-periode van TMR2 op 40usec in te stellen.

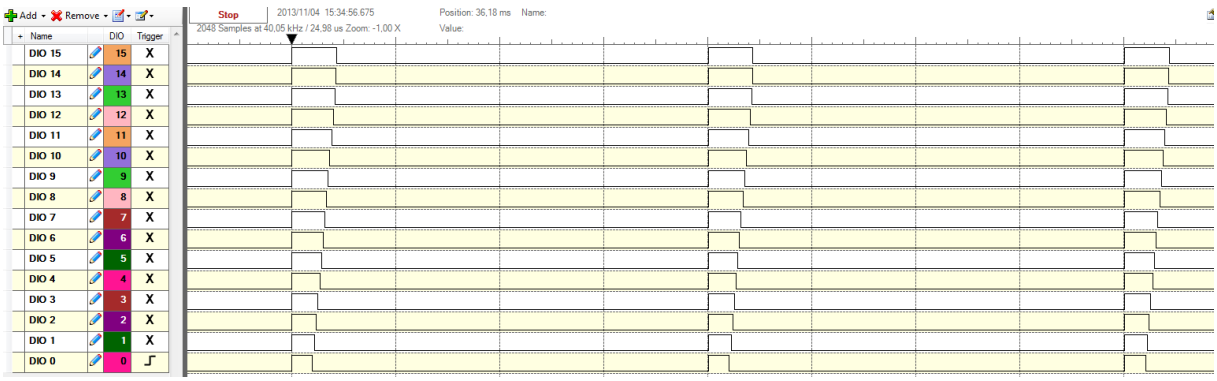
In de while 1 loop doen we hier voorlopig nog niets. Hier kan uw code komen te staan.

In de interruptroutine wordt meteen de TMR2 interrupt flag gecleared om komende interrupts toe te laten.

We komen als eerste in mode 0 en wachten hier totdat de interrupt routine 23 maal doorlopen is. 23 x 40usec = 0.9msec. Vanaf dat moment wordt de TELLER_TMR2 variabele terug op 0 gezet en we activeren mode 1.

In mode 1 wordt er telkens als de interrupt wordt aangeroepen – elke 40usec dus – gecontroleerd welke servo-signalen laag moeten worden. Als de teller gelijk wordt aan de waarde in de respectievelijke SERVO[] variabele, dan wordt de overeenstemmende pin van PORTC of PORTD laag gemaakt. Als de waarde 30 bereikt wordt, dan is het punt van 2.1msec bereikt en dat betekent dat we kunnen overgaan op mode 2. Daarvoor wordt eerst de interruptfrequentie van TMR2 sterk verlaagd naar 1 interrupt per 1.8msec.

In mode 2 worden deze trage 1.8msec interrupts geteld. We verlagen hier de interrupt frequentie zodat het hoofdprogramma gedurende die lange 18msec niet nodeloos elke 40usec zou onderbroken worden. Na 10 interrupts van 1.8msec is de resterende 18msec laag-tijd bereikt en mogen alle pins van PORTD en PORTC terug hoog gemaakt worden. De interrupt frequentie wordt nu ook terug opgedreven naar 1 per 40usec voor mode 0 en 1.



U ziet op dit logic analyser beeld mooi hoe de 16 servo signalen ‘parallel’ worden opgebouwd. Tijdens de eerste 2 msec waarin al de hoog-signalen worden opgebouwd is dit inderdaad een zware belasting voor de uC, maar tijdens de 18msec laag tijd heeft de uC alle ‘tijd’ om andere zaken te doen. 90% van de tijd kan de uC dus bezig zijn met andere taken dan de het servo signaal.

Uitdagingen meer dan 8 Servomotoren

1. Pas dit programma van 16 servo's aan , aan het aantal servo's dat u wenst te gebruiken.
2. Test het programma uit en verander de positie van een aantal servo's eens.
3. Bestudeer het stukje code hieronder zeer aandachtig en loop het stap voor stap door.

```
while(1)
    {
        for (char y= 2; y<30; y++)
        {
        for (char x= 0; x<15; x++)
            {
            SERVO[x] = y;
            }
            __delay_ms(50);
        }

        for (char y= 30; y>2; y--)
        {
        for (char x= 0; x<15; x++)
            {
            SERVO[x] = y;
            }
            __delay_ms(50);
        }
    }
```

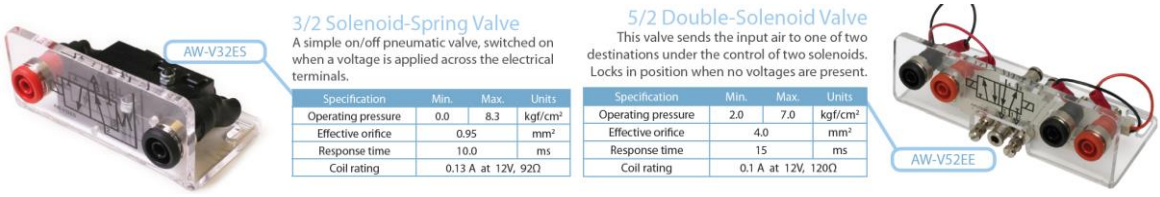
Probeer de waarde van de servo-array hieronder maar eens in te vullen voor de eerste 6 stappen. Dat is de ideale manier om deze code en de kracht van arrays te leren begrijpen. Test daarna de code uit in het echte programma.

SERVO[0]	SERVO[0]	SERVO[0]	SERVO[0]	SERVO[0]	SERVO[0]
SERVO[1]	SERVO[1]	SERVO[1]	SERVO[1]	SERVO[1]	SERVO[1]
SERVO[2]	SERVO[2]	SERVO[2]	SERVO[2]	SERVO[2]	SERVO[2]
SERVO[3]	SERVO[3]	SERVO[3]	SERVO[3]	SERVO[3]	SERVO[3]
SERVO[4]	SERVO[4]	SERVO[4]	SERVO[4]	SERVO[4]	SERVO[4]
SERVO[5]	SERVO[5]	SERVO[5]	SERVO[5]	SERVO[5]	SERVO[5]
SERVO[6]	SERVO[6]	SERVO[6]	SERVO[6]	SERVO[6]	SERVO[6]
SERVO[7]	SERVO[7]	SERVO[7]	SERVO[7]	SERVO[7]	SERVO[7]
SERVO[8]	SERVO[8]	SERVO[8]	SERVO[8]	SERVO[8]	SERVO[8]
SERVO[9]	SERVO[9]	SERVO[9]	SERVO[9]	SERVO[9]	SERVO[9]
SERVO[10]	SERVO[10]	SERVO[10]	SERVO[10]	SERVO[10]	SERVO[10]
SERVO[11]	SERVO[11]	SERVO[11]	SERVO[11]	SERVO[11]	SERVO[11]
SERVO[12]	SERVO[12]	SERVO[12]	SERVO[12]	SERVO[12]	SERVO[12]
SERVO[13]	SERVO[13]	SERVO[13]	SERVO[13]	SERVO[13]	SERVO[13]
SERVO[14]	SERVO[14]	SERVO[14]	SERVO[14]	SERVO[14]	SERVO[14]
SERVO[15]	SERVO[15]	SERVO[15]	SERVO[15]	SERVO[15]	SERVO[15]

4. Schrijf nu – zoals in de opgave hierboven – je eigen code om de servo's een wave te laten maken. De ene loopt dus altijd iets achter op de andere…
5. Schrijf een stukje code zodat je wel waarden tussen 0 en 255 kan ingeven voor de servo posities, maar dat we waarden die naar de servo-interrupt worden doorgegeven nog steeds tussen 2 en 30 liggen. Je zal hierbij delingen e.d. moeten gebruiken. Let op: een deling is een zeer zware instructie voor een uC. Deze code zal dus zeker niet mee in de interrupt routine mogen komen.

De Link met Pneumatica

Elektrische motoren zijn een manier om een elektrische energie om te zetten in een mechanische beweging. Om allerhande redenen wordt er bij automatisaties ook dikwijls gekozen voor pneumatische energie – of luchtdrukenergie om een mechanische beweging te veroorzaken.



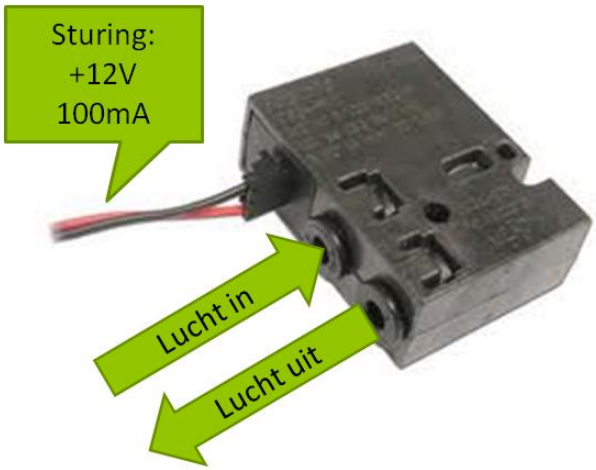
Om deze pneumatische energie te controleren bestaat er een heel gamma van elektronische kleppen of valves. Deze valves laten de lucht wel of niet door in een bepaalde richting. Er bestaan valves die met 12 of 24Volt kunnen worden aangestuurd. Vermits het stroomverbruik hiervan binnen de perken blijft is het ook mogelijk om deze valves ook via onze microcontroller aan te sturen – zij het met een elektronische schakelaar. Op deze manier kunnen we de hele wereld van de pneumatica met alle valves en cilinders ook aan onze microcontroller koppelen.

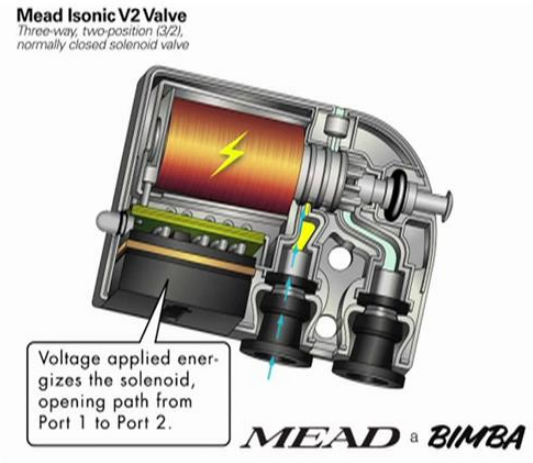
Elektrische Actuatoren i.v.m. Pneumatische Actuatoren

Over de vergelijking tussen pneumatische en elektrische sturingen bestaan er veel meningen, maar over het algemeen wordt er aangenomen dat elektrische sturingen nauwkeuriger zijn en veel flexibeler. Flexibeler omdat wanneer machines bijvoorbeeld één maal per week moeten worden omgebouwd om een nieuw product te maken, dit met elektrische actuatoren meestal op te lossen is door het stuurprogramma te veranderen. Een stappenmotor die enkele stappen meer of minder moet doen moet niet mechanisch versteld worden. Daardoor hebben elektrische actuatoren een lagere onderhoudskost.

Pneumatische systemen hebben dan weer het voordeel dat ze grote krachten en grote snelheden kunnen leveren voor een relatief lage kost en dat ze relatief kleiner zijn dan vergelijkbare elektrische actuatoren. Je moet hier wel een extra installatiekost van de perslucht en het onderhoud van de generator bijtellen en je moet in rekening brengen dat elke omstelling van een machine veel werkuren en dus een grote onderhoudskost met zich meebrengt.

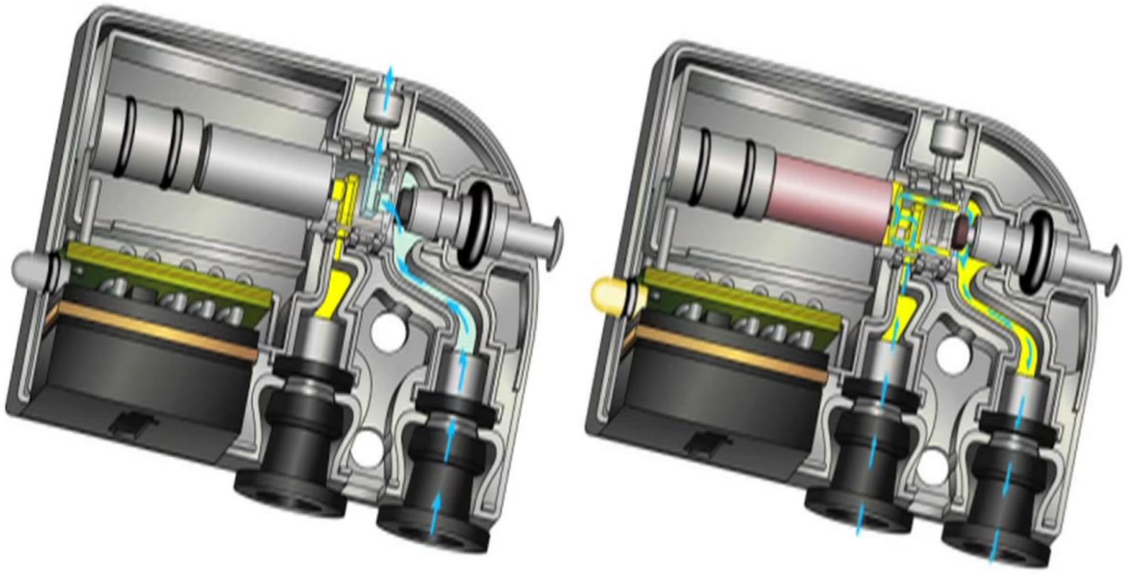
Werkingsprincipe Elektronische Valve





In de linkse afbeelding ziet u hoe zo'n elektronische valve er kan uitzien. Je herkent een luchtinlaat en een connectie waar de lucht er al dan niet uit zal komen. Deze specifieke valve werkt op 12V, maar je hebt er ook die op 24V of zelfs meer werken. Zonder aangelegde spanning staat deze valve in de geblokkeerde toestand, er zal dus geen lucht uit stromen – als je er wel 12 Volt aanlegt dan wordt de valve geopend en kan er wel lucht uit vloeien.

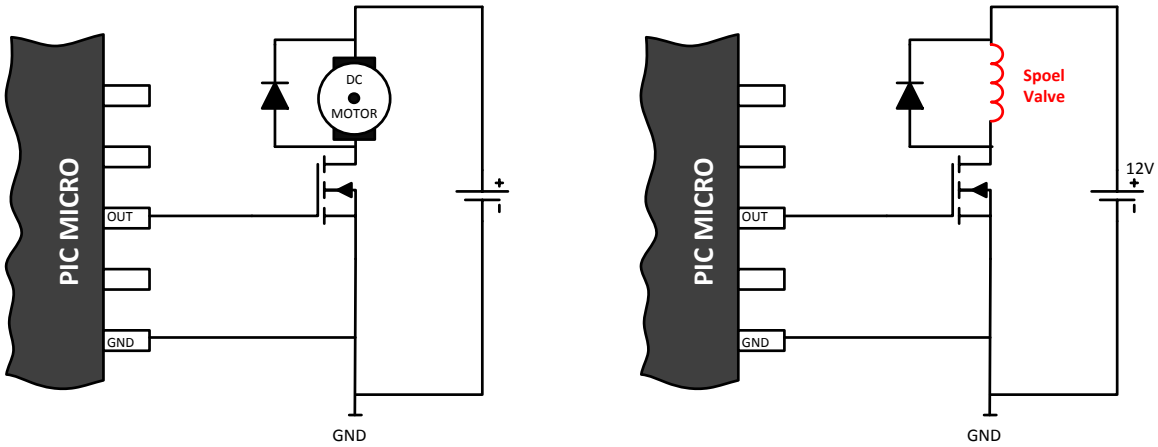
In de interne tekening zien we een spoel zitten. Wanneer we de 12V aanleggen zal er een stroom door de spoel vloeien. Dit veroorzaakt een magnetische energie die voldoende groot is op de klep te openen. Eigenlijk werkt dit deel van de valve identiek hetzelfde als een relais.



In de linkse tekening is de valve gesloten – er is geen stuurspanning aangesloten. De overdruk kan ontsnappen via de uitlaatopening aan de achterzijde.

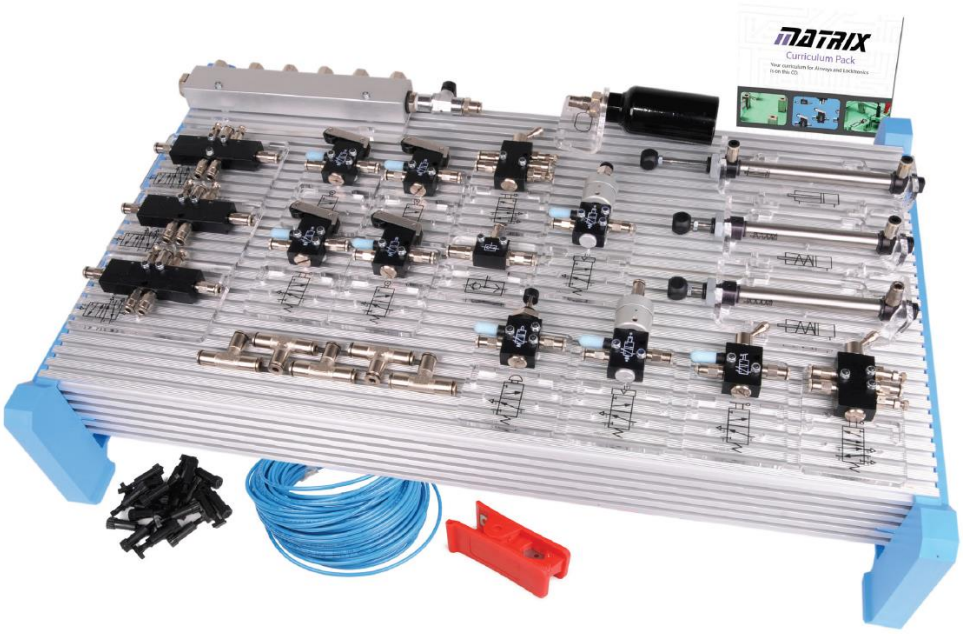
In de rechtse tekening is er wel een stuurspanning van 12V aangelegd. De lucht wordt omgeleid naar de uitgang. Hiermee kan bijvoorbeeld een cilinder worden aangestuurd.

Aansturing Elektronische Valve



We hebben gezien dat de aansturing van een elektronische valve langs de elektrische kant niet meer is dan de aansturing van een spoel. In principe is dit dus heel gelijkaardig als de aansturing van een DC motor welke bestaat uit een set van spoelen. We kunnen identiek hetzelfde schema gebruiken en dienen ook hier de mosfet met een vrijloopdiode te beschermen tegen te grote inductiespanningen – veroorzaakt door de spoel van de valve.

Pneumatisch Materiaal



Vanzelfsprekend heb je héél wat meer nodig dan alleen maar een valve om een pneumatisch systeem op te bouwen en er iets nuttigs mee te doen. Je hebt kleppen nodig, voeding, schakelaars, verdelers, cilinders en nog veel meer. Er bestaan veel verdelers van dit materiaal maar o.a. Matrixmultimedia heeft een pakket met pneumatische componenten samengesteld om de meeste bedenkbare projecten uit te kunnen werken in een klassituatie. Alle componenten zijn ook afzonderlijk te bestellen en het meegeleverde cursusmateriaal is van zeer goede kwaliteit.

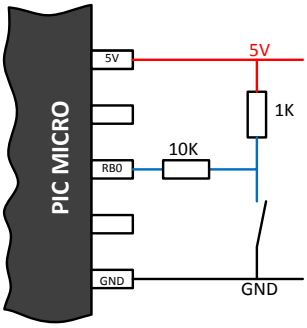
Uitdagingen Pneumatica

1. Gebruik een elektronische valve en bedenk een project om d.m.v. luchtdruk iets leuks te doen met een pingpongballetje… gebruik gerust het internet voor inspiratie….

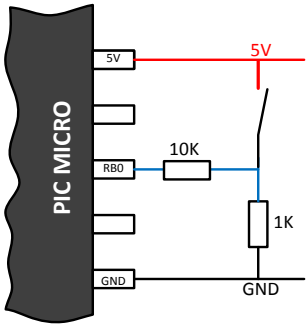
Sensoren

Inlezen van Sensoren

Digitale Sensoren Inlezen



Schakelaar open : pull-up naar 5V RB0 = hoog
Schakelaar dicht : RB0 hangt aan gnd RB0 = laag

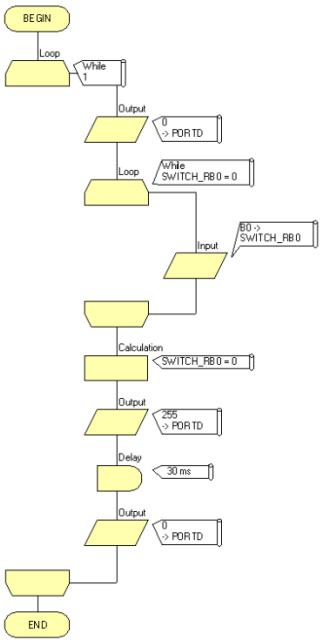


Schakelaar open : pull-down naar gnd RB0 = laag
Schakelaar dicht : RB0 hangt aan 5V RB0 = hoog

In deze eerste les over sensoren delen we de sensoren kort in in 3 groepen volgens hun uitleesprincipe. Zo hebben we de digitale sensoren, de analoge sensoren en de bus-sensoren. Digitale sensoren zijn sensoren die enkel maar aan of uit kunnen zijn. Schakelaars vallen hier in principe ook onder. Ze worden meestal als volgt aangesloten. In het linkse schema is de RB0 pin hoog als de schakelaar open is en wordt de RB0 pin naar GND getrokken als de schakelaar gesloten is.

De 10K weerstand staat hier enkel als beveiliging en doet voor de rest niets aan deze schakeling. Als pin RB0 per ongeluk toch als uitgang zou geprogrammeerd worden dan zal de 10K weerstand de stroom beperken als de schakelaar gesloten wordt.

Het rechtse schema is iets meer toegepast omdat hier – als de schakelaar open is – het signaal aan pin RB0 laag is en wanneer de schakelaar gesloten wordt – het signaal aan pin RB0 hoog is.

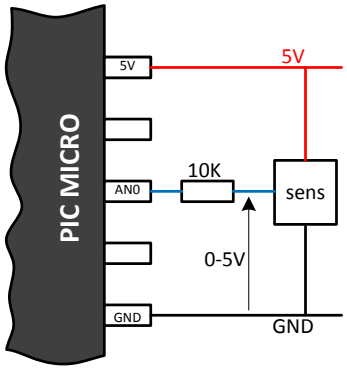


Dit is maar één van de vele mogelijke manieren om een digitale input in te lezen en hierop te reageren. Er hangen 8 leds aan alle 8 pins van PORTD. Die willen we kort laten oplichten als de RB0 ingang hoog wordt. In het begin van het programma worden alle leds uit gezet. In de daarop volgende loop blijven we hangen zolang RB0 laag is. Van het moment RB0 hoog wordt worden we uit de loop gegooid en worden de instructies na de loop uitgevoerd. Als eerste instructie wordt de RB0 variabele terug 0 gemaakt om te vermijden dat we de volgende keer meteen terug uit de loop gegooid worden voordat RB0 getest kan worden. We maken alle pins van PORTD even hoog en dan weer terug laag om alle leds even te laten oplichten. Vervolgens gaan we terug wachten tot RB0 weer wordt ingedrukt.

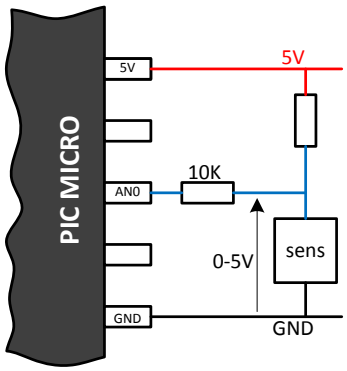
```
/*****************************************************************

Videolessen Deel 4 - Bart Huyskens - RTC Antwerpen - 2013
Extern Xtal - 19660800Hz
digitale input inlezen op poort RB0
*****************************************************************/

#include <htc.h>
#define  XTAL FREQ 19660800

__CONFIG(0x23E2);   //Extern HS Xtal, WDT DIS, PWRT EN, MCLR EN, BOR EN, INT/EXT DIS, LVP DIS, NO
Code/data protect
__CONFIG(0x3FFF);   //BOR 4,0V, Write protect off

void main()
{
unsigned char SWITCH_RB0 = 0;   // variabele voor Duty Cycle
TRISB   = 0b00000001;           // RB0 = input
TRISD = 0b00000000;             // PORTD = output (LEDS)
ANSEL = 0x00;                   // alle AD inputs uitzetten
ANSELH = 0x00;                  // alle AD inputs uitzetten
    while (1)
    {
        PORTD = 0x00;           //alle leds uit
        while (SWITCH_RB0 == 0) // wacht tot RB0 hoog wordt
            {
            SWITCH_RB0 = RB0;   // lees RB0 in
            }
        SWITCH_RB0 = 0;         // zet variabele terug op 0
        PORTD = 0xFF;           // alle leds aan
        __delay_ms(30);
        PORTD = 0x00;           // alle leds uit
    }
}
```

Het C programma verloopt identiek aan het Flowcode programma. Belangrijk is wel dat je via de ANSEL registers de analoge functie - die nogal wat pinnen van deze microcontroller heeft – uitschakelt. Nogmaals – er zijn vele manieren om om te gaan met het inlezen van digitale sensors en dikwijls moet je ook wat hysteresis inbouwen om ongewenste dendereffecten uit te filteren.

Analoge Sensoren Inlezen



Sensoren die rechtstreeks een analoge spanning uit geven (met interne spanningsdeler)



Sensoren die nog in een spanningsdeler moeten worden geplaatst

Bij de meeste moderne microcontrollers kan je verschillende van de IO pinnen ook als Analoge ingangspin configureren. Onze PIC16F887 heeft zo 14 analoge ingangspinnen die allemaal spanningen kunnen meten tussen 0 Volt en de voedingsspanning – in ons geval dus 5 Volt.

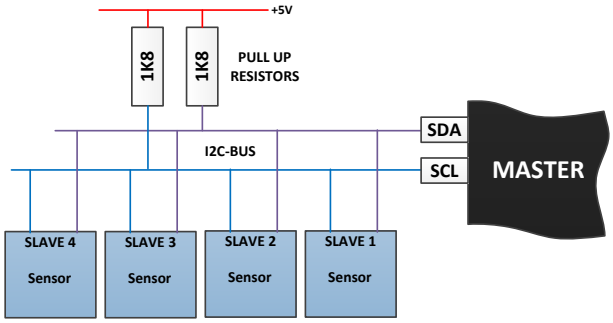
In een aantal gevallen geven analoge sensoren als uitgangssignaal rechtsreeks een spanning tussen 0V en VCC uit. Die spanning kan je dan zonder probleem rechtsreeks inlezen op een analoge pin van je uC.

Een aantal andere analoge sensoren moet je eerst zelf nog in een spanningsdeler zetten. Meestal zijn dit sensoren waarvan de weerstand of de geleiding verandert onder invloed van een bepaalde meetwaarde. Om van een weerstand een spanning te maken gebruiken we een spanningsdeler zoals in het rechtse schema te zien is.

Ook hier dient de 10K weerstand om de pin van de microcontroller te beveiligen tegen te grote stromen die kunnen voorkomen als je in je programma deze pin foutief als uitgang zou schakelen.

Voor de juiste manier om zowel in FC als in C deze analoge ingangspinnen in te lezen verwijs ik u graag door naar videolessen deel 1 en 2.

Bus Sensor - I2C Of SPI Sensoren Inlezen



Nogal wat van de meer complexe sensoren worden aangestuurd en uitgelezen via de I2C bus en in een aantal gevallen ook via de SPI Bus. De volledige werking – inclusief voorbeeldprogramma's, cursusmateriaal en videolessen – van zowel I2C als SPI kan u terug vinden in videolessen deel 3.

Uitdagingen Inlezen Sensoren

1. Bekijk van videolessen deel 1 de video les 14 over Digitale inputs inlezen in FC
2. Bekijk van videolessen deel 1 de video les 23,24 en 25 over Analoge inputs inlezen in FC
3. Bekijk van videolessen deel 2 de video les 12 over Digitale inputs inlezen in C
4. Bekijk van videolessen deel 2 de video les 22over Analoge inputs inlezen in C
5. Bekijk van videolessen deel 3 zeer aandachtig de videolessen, het cursusmateriaal en de voorbeeldprogramma's over SPI en I2C

Voorbeelden van Digitale Sensoren

In de lessen over sensoren worden er regelmatig merken vernoemd – dat is voornamelijk gedaan om u op weg te helpen, maar dat betekent niet dat gelijkaardige sensoren niet ook van andere merken verkrijgbaar zijn.

De meer gespecialiseerde robotica sensoren bestellen wij dikwijls via robotica – websites waarvan we er hier een aantal laten zien, maar kijk ook hiervoor gerust verder want deze wereld evolueert zeer snel.

http://www.active-robots.com/ is misschien wel de grootste robotica website. Ze zijn hier heel vriendelijk en helpen je altijd verder, maar je kunt hier enkel met ponden betalen dus de stand van de pond t.o.v. de Euro is bepalend en ook de overschrijving van ponden kan soms extra kosten met zich meebrengen. Laat je vooraf goed informeren.

http://www.robot-electronics.co.uk/ is de website van Devantech. Naast verkoop ontwerpen zij ook zelf een heel gamma sensoren en actuatoren die je heel veel tegenkomt in robotica-projecten. Ook hier is het zelfde ponden probleem van toepassing.

http://www.robot-italy.com/en/ is een zeer goed alternatief – zeker omdat je rechtstreeks in Euro kan betalen. De site kan ook in het Engels bekeken worden.

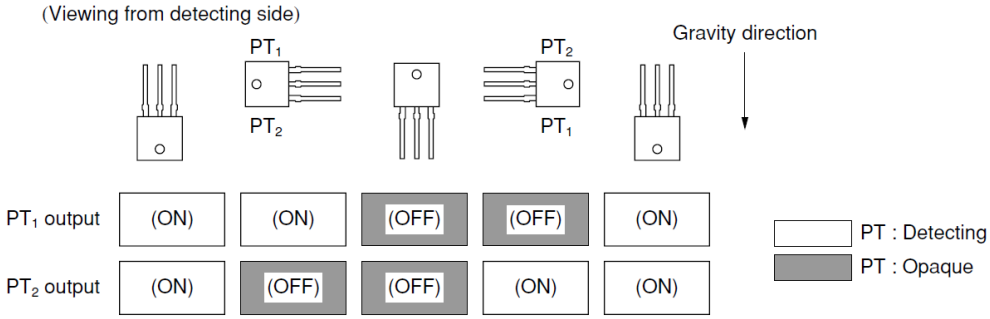
Microswitch

Microswitches zijn schakelaars die zo zijn opgebouwd dat ze kunnen dienen als eindeloopschakelaars en positieschakelaars voor allerhande toepassingen.

Tilt Sensor

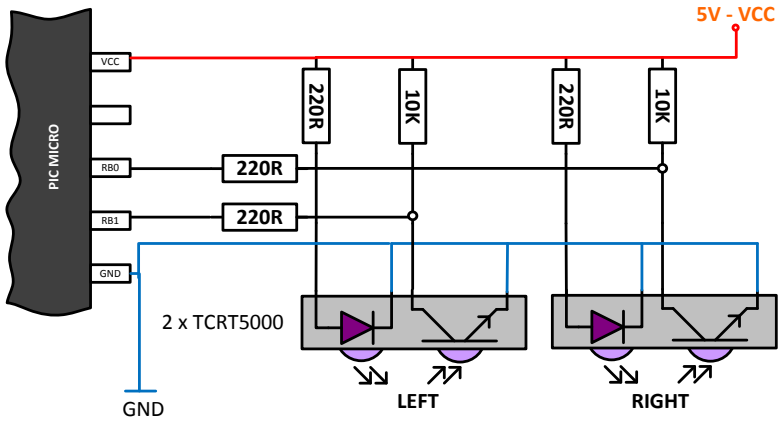
CW1200-1 van COMUS (ASSEMTECH) (Farnell: 540614)

Tilt schakelaars zijn ook gewone schakelaars die onder een bepaalde hoek 'uit' staan en onder een andere hoek 'aan' staan. In de goedkopere versies zit er een metalen 'balletje' – in de duurdere zit dikwijls kwikzilver wat een vloeibaar metaal is. Daarom dat tilt sensors ook wel eens mercury switches worden genoemd.



De GP1S036HEZ tilt sensor van Sharp is een dubbele tilt sensor die aan kan geven in welke richting iets begint over te hellen.

Lijnvolger



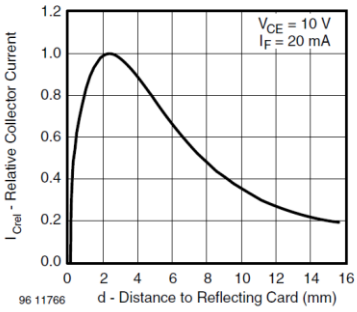
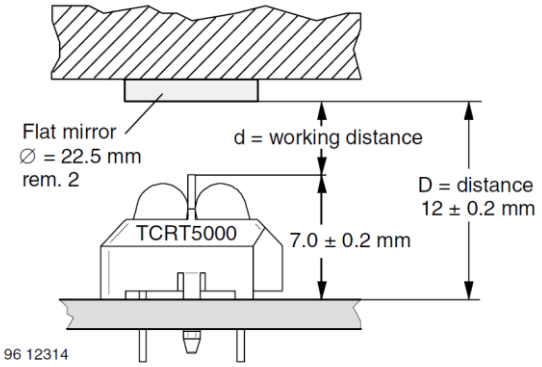


Fig. 9 - Relative Collector Current vs. Distance

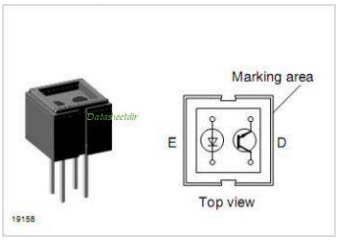


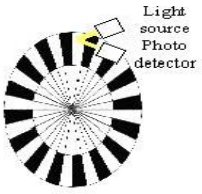
DE TCRT5000 sensor wordt dikwijls gebruikt op robotica projecten om de robot een bepaalde zwarte lijn op een witte achtergrond te laten volgen. Deze sensor bevat een zender die infrarood licht uitzendt en een ontvanger die enkel infrarood licht binnenlaat. Als het infrarode licht weerkaatst op een witte ondergrond dan ziet de ontvanger licht, dan wordt de fototransistor in geleiding gestuurd en dan zal het gemeten signaal aan de microcontroller pin laag zijn. Als de sensor boven een zwart voorwerp staat dat het licht niet weerkaatst dan staat de fototransistor in sper en dan is het signaal aan de pin van de uC hoog.

Uit de datasheet leren we ook dat de ideale afstand tussen de sensor en het zwarte of witte oppervlak 2mm is en dat we als we onder de 1mm of boven de 6mm gaan – dat het signaal dan te zwak wordt.

Toerentalmeting

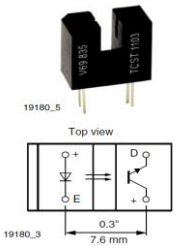
CNY70

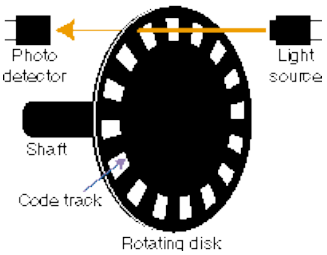


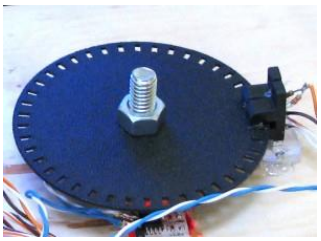


De CNY70 sensor van Vishay werkt volgens het reflectie principe. Identiek eigenlijk als de TRCT5000 sensor. Een IR led belicht het oppervlak van een zwart-wit gestreepte encoder schijf. Zwart weerkaatst niet – wit weerkaatst wel en zal de fototransistor sensor wel in geleiding zetten. De aansluiting kan ook identiek gebeuren als de TCRT5000 lijnvolgsensor.

TCST1300

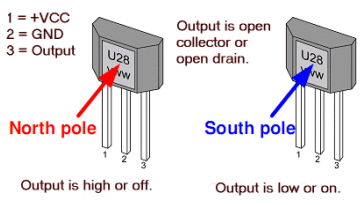


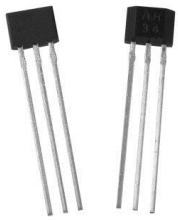




Bij de TCST1300 sensor van Vishay staan de zend-led en de ontvangs-transistor niet meer naast – maar nu wel tegenover elkaar. Als de zend led en de ontvangst transistor elkaar kunnen zien, dan staat de transistor in geleiding – als er een voorwerp – bijvoorbeeld een schijf met perforaties – tussen komt, dan kan deze lichtstraal dus wel of niet onderbroken worden. De frequentie van de pulstrein die we zo meten is recht evenredig met het toerental.

HALL SENSOR





Een hall sensor meet magnetisme. Als je bijvoorbeeld een magneet aan één van de spaken van een fietswiel hangt, dan zal de hall-sensor telkens als die magneet vlak naast deze hall sensor passeert – een hoog of een laag signaal uit geven.

Er zijn twee types – de mono-stabiele hall sensor die aan schakelt als er een magnetisch veld vlakbij is en automatisch terug uitschakelt als dat veld verdwijnt en de bi stabiele die aan schakelt als er een Zuidpool dichtbij is en uit-schakelt als er een Noordpool van een magneet vlakbij is. De Hall sensor op deze tekening is duidelijk een bistabiele.

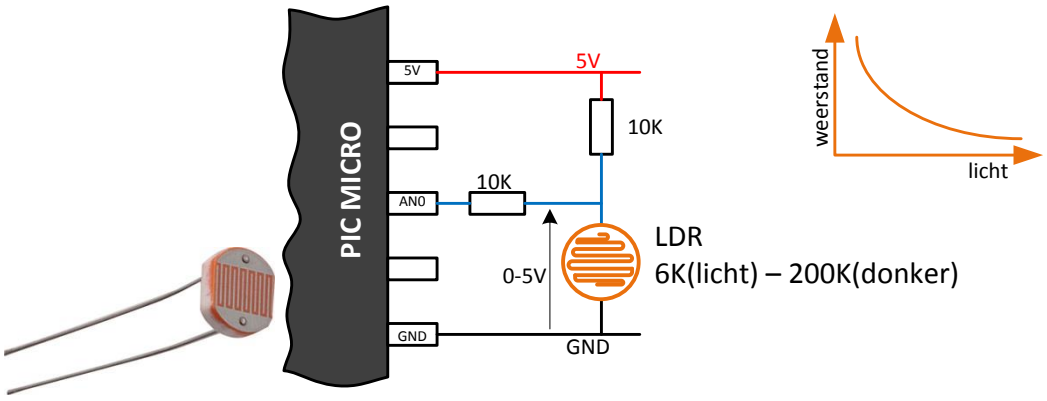
Uitdagingen Digitale Sensoren

- Selecteer één van deze 'digitale' sensoren – sluit deze correct aan op je microcontroller en gebruik je inspiratie om er iets zinvols mee te doen.

Voorbeelden van Analoge Sensoren

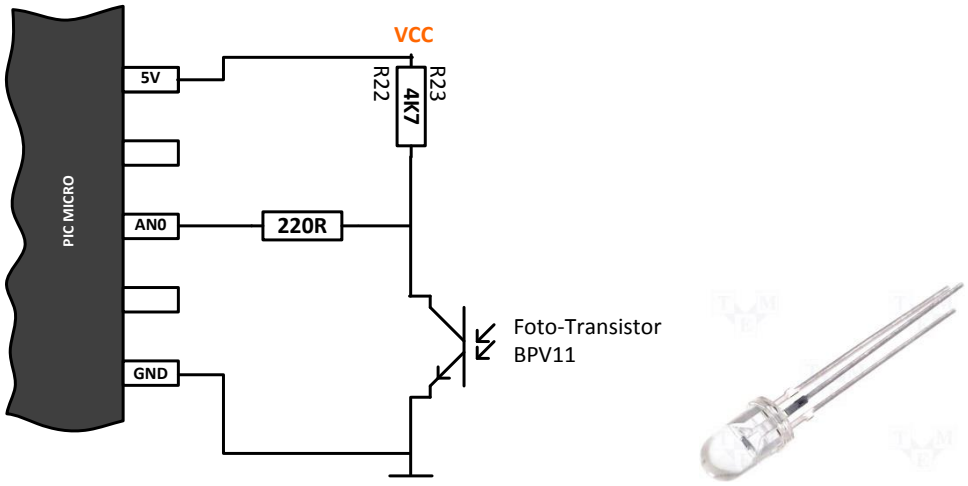
Licht

LDR



De **LDR** of "Light Dependent Resistor" is een sensor waarvan de weerstand lager wordt als er meer licht op valt. Deze sensor moet nog in een spanningsdeler gezet worden om deze met een microcontroller uit te kunnen lezen. Er bestaan verschillende types LDR's die elk hun specifieke weerstandswaarden uitgeven.

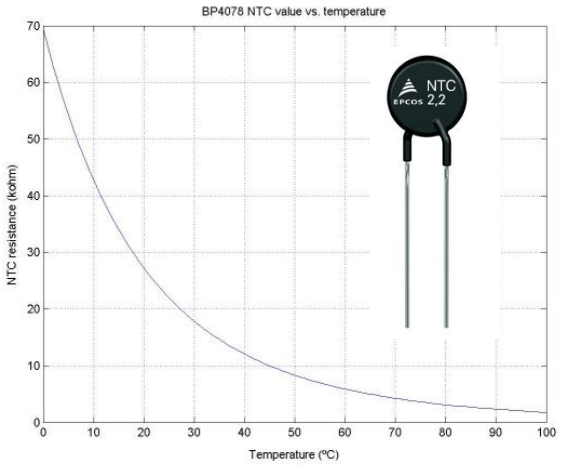
FOTOTRANSISTOR

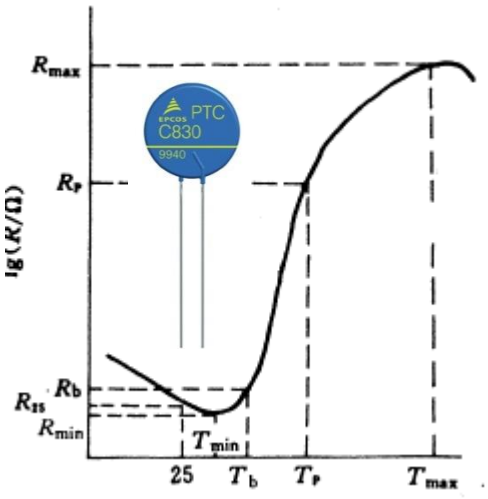


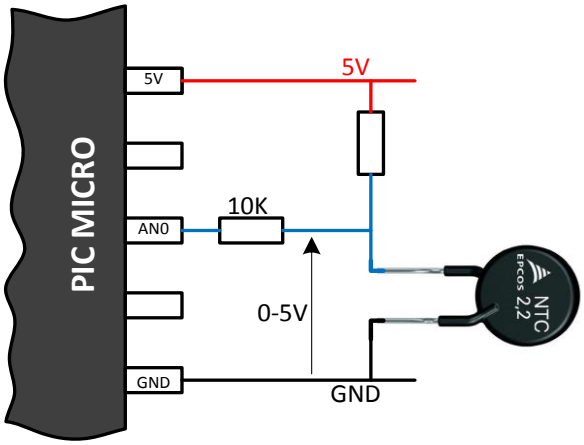
Een fototransistor – zoals deze BPV11 van Vishay doet in principe hetzelfde als een LDR. Hoe meer licht er op valt, hoe beter deze in geleiding gaat, hoe minder spanning er over staat en hoe kleiner het signaal aan de analoge ingangspin van de uC. Een fototransistor heeft het voordeel dat hij veel sneller reageert dan een LDR en je kunt fototransistors kopen die enkel gevoelig zijn voor een bepaald lichtspectrum – bijvoorbeeld enkel voor IR straling of slechts een beperkte kijk-hoek hebben terwijl LDR's heel breed kijken. Merk ook op dat fototransistors 3 pinnen hebben, maar dat de basis meestal niet wordt aangesloten. Deze basis wordt enkel aangesloten als je de transistor vooraf al een bepaalde voorinstelling wilt geven.

Temperatuur

NTC - PTC

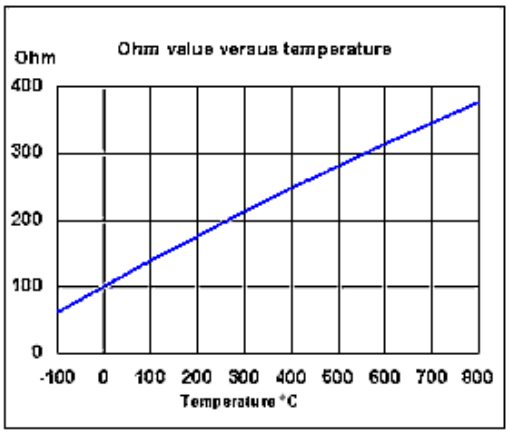


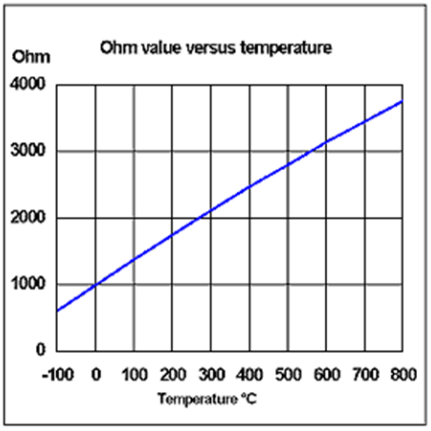


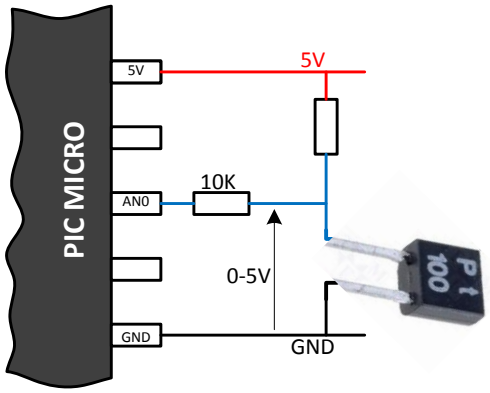


NTC's zijn weerstanden met een Negatieve Temperatuurs-Coëfficiënt – de weerstand daalt als de temperatuur stijgt. PTC hebben een Positieve Temperatuurs-Coëfficiënt – de weerstand stijgt als de temperatuur stijgt. De standaard NTC's en PTC's zijn relatief goedkoop maar vermits beide – zowel de NTC als de PTC geen lineaire grafiek hebben worden ze nog zelden gebruikt om een temperatuurmeting over een breed bereik te doen – wel om bijvoorbeeld een proces aan of uit te schakelen bij één bepaalde temperatuur.

PT100 – PT1000



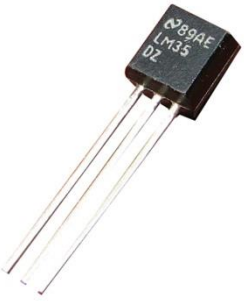


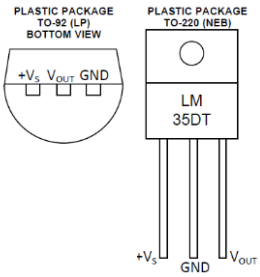


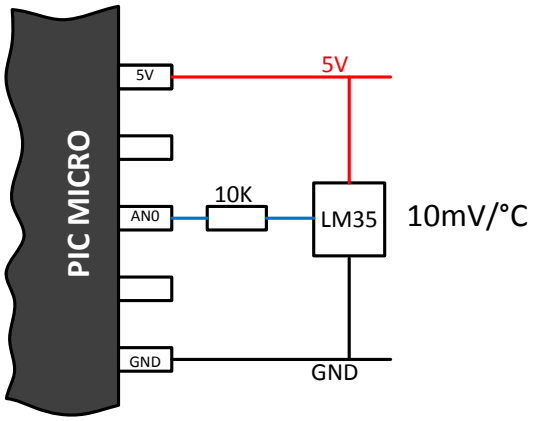
De PT100 en PT1000 zijn in principe ook PTC weerstanden, maar de PT staat hier voor Platina – het dure materiaal waarvan ze gemaakt zijn. Deze PTC's hebben hierdoor wel een weerstand die bijna perfect lineair stijgt i.f.v. de temperatuur. De weerstand van een PT100 is exact 100 Ohm bij 0°C, een PT1000 is exact 1000 Ohm bij 0°C. Er bestaat ook nog een PT500 en een PT2000.

De PT100 e.d. zijn verkrijgbaar als zeer kleine componenten, maar ook als industriële modellen die in tanks e.d. kunnen worden ingebouwd. De werking blijft echter exact dezelfde.

LM35



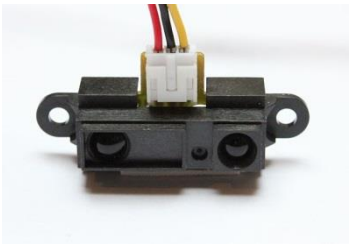


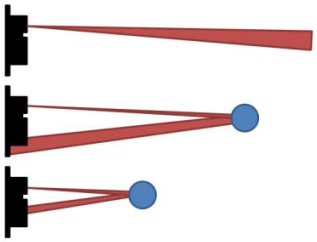


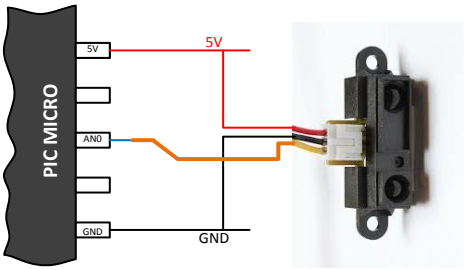
De LM35 is een goedkope component die standaard temperaturen kan meten tussen +2°C en +150°C, maar met een kleine aanpassing ook temperaturen tussen -55°C en +150°C kan meten. Bij elke graad meer wordt de uitgangsspanning met 10mV verhoogd. Doordat het héél eenvoudig is om deze temperatuursensor aan te sluiten op een uC wordt deze zeer veel toegepast. Meestal wordt de kleine TO92 versie gebruikt, maar de LM25DT in TO220 behuizing is in het bijzonder handig omdat deze op het te meten object kan worden vastgeschroefd. De LM34 is de tegenhanger die de temperatuur in ° Fahrenheit uitgeeft.

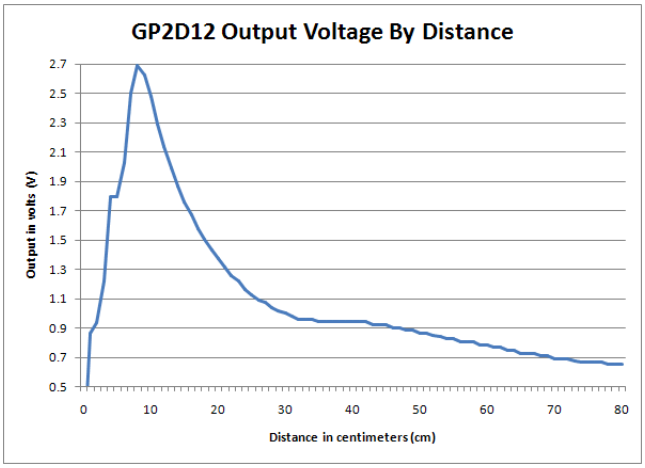
Afstand

Afstand met Gp2d12 Sharp





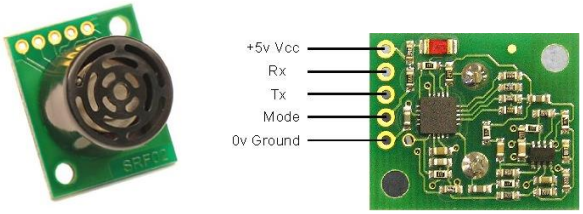




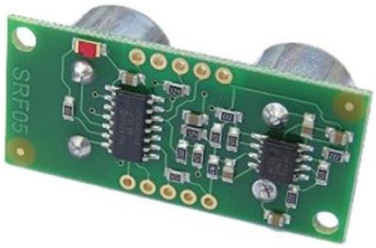
Sharp heeft verschillende afstand sensoren in het gamma die allemaal de moeite zijn om eens te bekijken. Deze GP2D12 sensor meet zeer nauwkeurig de afstand tot objecten tussen 10cm en 80 cm. Het belangrijkste waar je bij deze sensor op moet letten is dat je er voor moet zorgen dat deze sensor nooit afstanden kan meten die kleiner zijn als 10cm, want de waarden die hij hiervoor teruggeeft zijn identiek dezelfde als die boven de 10cm. De aansluiting is heel eenvoudig – 5V, gnd en de analoge uitgangsspanning inlezen via de analoge ingangspin van de uC. Deze sensor 'kijkt' ook heel smal – je kunt deze bijvoorbeeld gebruiken om de exacte positie van een blikje te bepalen dat 60cm verderop staat.

Ultrasone Afstandmeting met de SRF Reeks

Afstandmeting kan ook ultrasoon. Een geluidspuls wordt weggestuurd – de puls weerkaatst op een object en komt terug aan bij de ontvanger met een snelheid van ongeveer 320 meter per seconde. Ultrasoon-meting 'kijkt' vrij breed – dat kan zowel een nadeel als een voordeel zijn. Devantech heeft een aantal Ultrasoon sensors die betaalbaar zijn voor zelfbouw projecten.



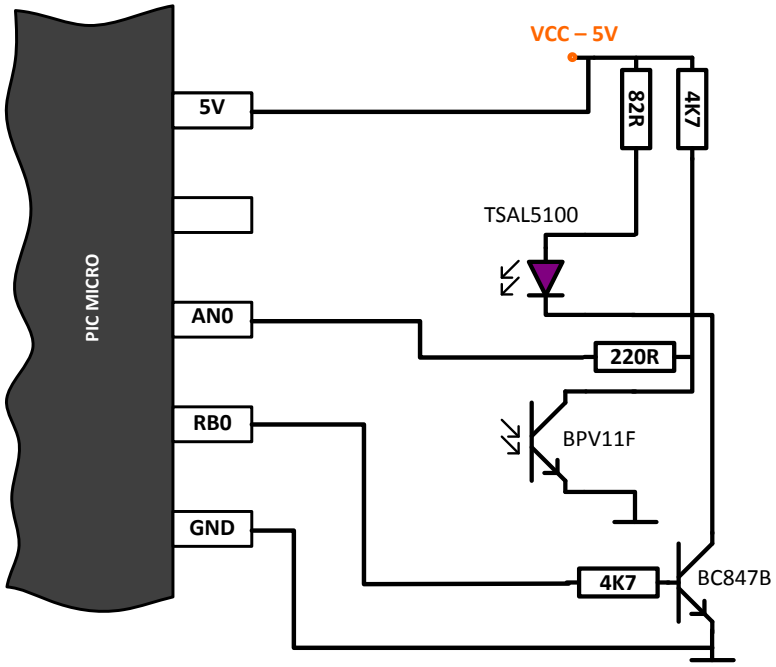
SRF02 meet van 15cm tot 6m maar werkt wel enkel via I2C.



De SRF05 meet van 1cm tot 4meter, maar geeft de afstand weer via eenvoudige TTL pulsen. Geen nood dus om het I2C protocol te gebruiken.

De SRF08 is de duurste sensor uit de reeks. Deze meet tussen 3cm-8m – via het I2C protocol, maar heeft daarnaast heel wat extra functies die je ook bij professionele US sensors kan terugvinden zoals het uitfilteren van ongewenste reflecties ed.

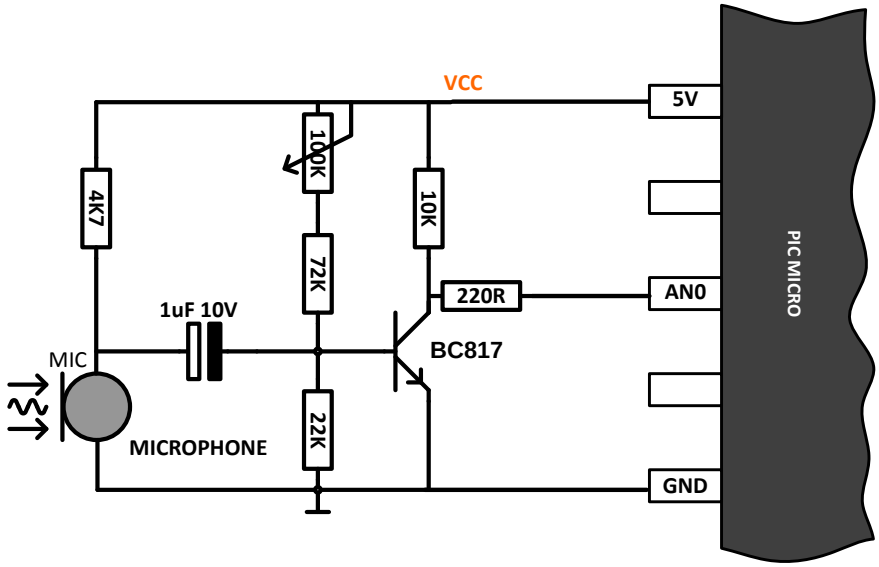
IR AFSTANDMETING – ZELFBOUW

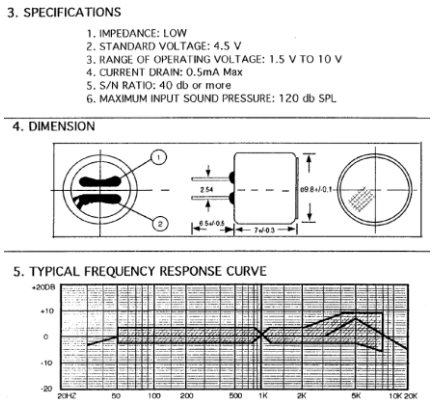


Het is niet zo moeilijk om zelf een IR afstandmeting te maken. De TSAL5100 is een led van Vishay die vooral IR licht uitstraalt. Vermits deze led meer dan 20mA nodig heeft en omdat het voor andere IR afstandmeters storend zou kunnen werken, wordt deze led via een transistor uit en aangestuurd als deze sensor gebruikt wordt.

De BPV11F is dezelfde fototransistor als de BPV11, maar deze heeft een extra donkere filter die enkel IR licht doorlaat. Zo wordt deze sensor niet of veel minder gestoord door omgevingslicht. Met deze sensor meten we het weerkaatste licht. Hoe meer licht er weerkaatst – hoe dichter bij het voorwerp staat. Deze sensor is in de huidige configuratie te gebruiken tot +/-20cm, maar die afstand is, door de IR led wat meer stroom te geven, zeker nog op te drijven.

Audio

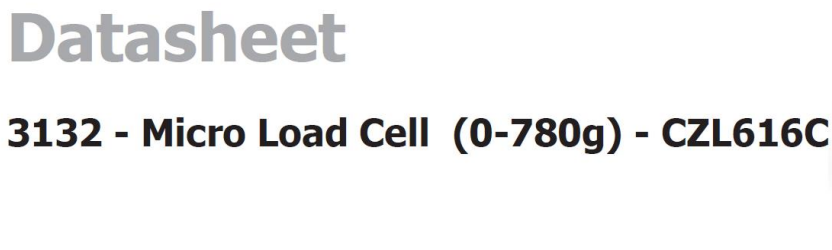


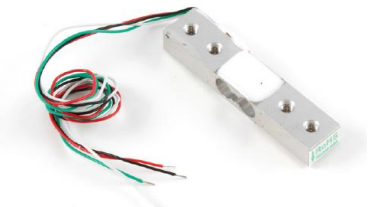


Electret microphone – Rapid: 35-0192

Dit schema kan gebruikt worden om het zwakke signaal van een goedkope 'electret microphone' om te zetten naar een mooi analoog signaal voor de microcontroller. Dit is in feite een 1 traps versterker. De versterking kan worden ingesteld met de regelbare 100K weerstand. Dit schema kan ook worden gebruikt op een digitale ingang om de microcontroller te laten reageren op klap-geluiden.

Kracht/Gewicht



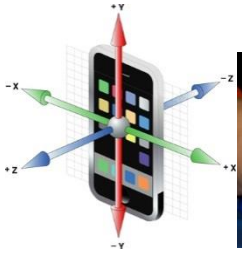
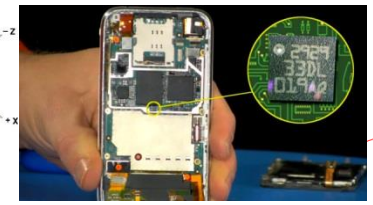
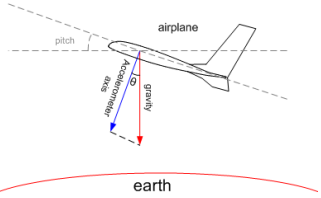


Op de verschillende robotica-sensoren websites kan je een heel gamma terugvinden van deze zeer betaalbare Micro load cells. Elk met hun eigen meetbereik – sommige tot wel 100kg en allemaal verkrijgbaar voor ongeveer 10€. Deze sensoren maken gebruik van rekstrookjes, wat wel inhoud dat het signaal nog wat moet worden bewerkt alvorens het kan worden ingelezen door een microcontroller, maar daarvoor dient de goede datasheet die er bij geleverd wordt.

Accelerometer - Gyroscoop





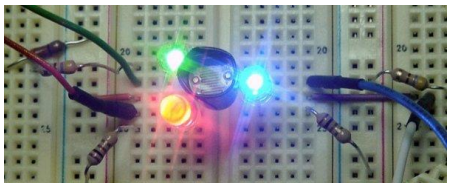
Accelerometers en gyroscopen worden dikwijls samen gebruikt om te weten te komen hoe een bepaald voorwerp – zij het nu een 'nunchuck' van een Wii console, een quadrocopter, een balancerende robot of een segway – gedraaid is. Ze worden bij tablets en smartphones ook gebruikt om te meten wanneer je scherm moet gedraaid worden en in vliegtuigen worden ze veelvuldig toegepast om horizontaal te blijven. Meestal wordt er gemeten in 3 assen t.o.v. de aarde.

De MPU-6050 is een combinatie van een Gyro en Accelero sensor

Als je hier iets mee wilt doen, dan zal je eerst wat tijd moeten investeren in het uitzoeken wat de juiste werking is van zowel de accelerometer als de gyro-sensor. Je vindt 10-tallen accelero en gyro sensors op het net – dikwijls voor slechts enkele euro's. Kort samengevat meet de accelerometer de versnelling en de gyro meet de verdraaiing rond een as. Accelerometers zijn traag en geven pas een correcte waarde na een bepaalde tijd en gyro's geven wel correcte waarden op korte termijn, maar zijn op lange termijn dan weer minder nauwkeurig. Ze vullen elkaar dus perfect aan en zijn daarom dikwijls samen gebruikt.

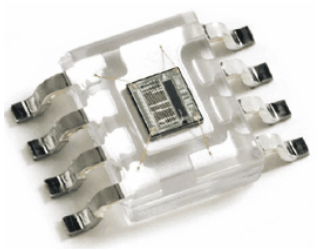
Kleursensor

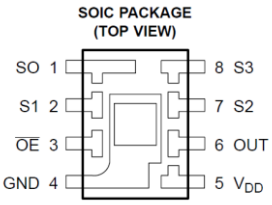
ZELFBOUW MET RGB LED EN LDR

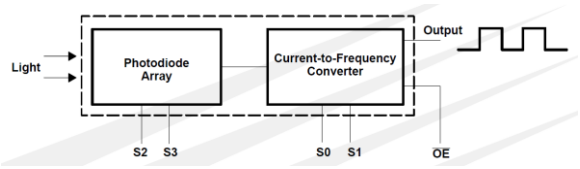


Je zou zelf een kleursensor kunnen maken met een LDR – in combinatie met een blauwe, een groene en een rode led. Elk van de 3 leds licht je afzonderlijk op een bepaald voorwerp – afhankelijk van de kleur van dat voorwerp weerkaatst er meer of minder licht op de LDR. De 3 meetwaardes van de LDR, die je met de analoge ingang van de uC meet, geven je een indicatie voor de kleur. In een labo-omgeving werkt dit vrij goed, maar voor robotica toepassingen wordt dit type van sensor te veel beïnvloed door externe factoren zoals zonlicht ed.

TAOS TCS230 SENSOR





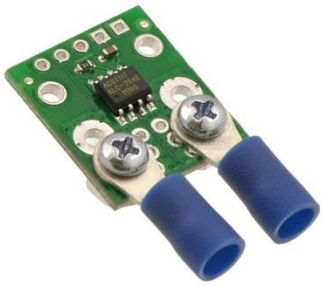


De TCS230 sensor is speciaal ontworpen voor kleurmeting. Deze bevat 64 fotodiodes – 16 met een rode filter, 16 met een blauwe, 16 met een groene en 16 zonder filter. Met de S2 en S3 pinnen selecteer je een bepaalde filter. De frequentie die je dan op de output meet is een maat voor de gemeten kleur met deze bepaalde filter actief. Deze meting herhaal je voor de 4 filters om een totaalbeeld te krijgen. Deze sensor meet héél nauwkeurig, maar blijft gevoelig voor omgevingsinvloeden.

TAOS TCS230 DB (DAUGHTER BOARD)

Dit product – eveneens van TAOS – bevat de TCS230 colour sensor maar door hier gebruik te maken van een lens en door het product gecontroleerd te belichten met witte leds worden de storende omgevingsinvloeden volledig weg gefilterd.

Stroom



Al voor enkele euro's kan je op het internet stroom-sensoren vinden. De goedkopere werken met een hall sensor die het magnetisch veld rond een stroombaan meten – Hall sensoren worden voornamelijk gebruikt voor dc stromen, maar bijvoorbeeld de MLX91209CA van Melexis kan zowel AC als DC meten. De duurdere stroommeters – vooral voor AC stromen dan - werken met split core current transformatoren. Elke sensor heeft zo ook zijn eigen stroom bereik.

Gas: Co2, Alcohol, Luchtkwaliteit, …


Op robotica sites zijn er sets van GAS sensoren beschikbaar – elk voor hun specifieke gassen. Zo zijn een CO2 sensoren, LPG sensoren, alcohol sensoren, luchtkwaliteit sensoren en nog veel meer. Dikwijls reageren de goedkopere modellen niet even snel als de duurdere tegenhangers – maar er is stof genoeg om u te laten inspireren.

De GP2Y1010AU0F van Sharp is een sensor die speciaal ontworpen is om luchtkwaliteit en fijn stof te meten.

Uitdagingen Analoge Sensoren

- Bereken in de schakeling van de LDR hoeveel spanning er op de ingangspin van de uC staat als de LDR 6K meet (maximaal licht) en als de LDR 200K meet (maximaal donker).
- Selecteer één van deze 'analoge' sensoren – sluit deze correct aan op je microcontroller en gebruik je inspiratie om er iets zinvols mee te doen.