TECHNISCH TALENT KRIJGT TOEKOMST
mechanica/elektriciteit - hout - auto

VERY TECHNICAL INDEED

INDUSTRIEËLE WETENSCHAPPEN

De stappenmotor

Leerling(en) :
Jeroen Maton
Pieter Verelle

Mentor:
Verhaeghe Dirk

2004 - 2005

VTI TorhoutSint-Aloysius | Papebrugstraat 8a, 8820 Torhout
Telefoon: 050 23 15 15 | Fax: 050 23 15 25
E-mail: vti@sint-rembert.be | Site: http://vtiweb.sint-rembert.be/
Site GIP'S 6IW: www.gip6iw.be

Inhoudsopgave

Voorwoord

Een robot neemt een voorruit op en draait die in één sierlijke beweging tot enkele millimeters van het frame waar de ruit in past; het koetswerk van een auto wordt binnen enkele seconden perfect met puntlassen aan elkaar gehecht; even later pikt een andere mechanische arm het volledige koetswerk op en dompelt het onder in een verfbad, …
Een bekend Frans automerk geeft een nieuw model de naam van een beroemde Spaanse schilder die jarenlang in Parijs vertoefde. De robots gaan gezellig uit de bol en zetten hun eerste stappen in de artistieke wereld van het kubisme… Neen, dit laatste voorbeeld gaat overduidelijk te ver en geeft de robot net iets te veel krediet. Maar het zet je wel aan het denken: wat kan er nu en wat kan er niet? Zouden wij zelf in staat zijn om een robot aan het werk te zetten?
Aangezien wij beiden erg geïnteresseerd zijn in computers, robots en andere technologische snufjes, waren we op zoek naar een geïntegreerde proef die deze richting uitging. Na lang zoeken, kregen we het voorstel om een robot te besturen via de computer. Met veel (over)enthousiasme begonnen we aan het project, maar na een tijdje zagen we in dat het besturen van een robot allerminst een eenvoudige opdracht is, zelfs niet voor de experts. Nadat we ons dit realiseerden, zijn we bescheiden overgestapt naar het besturen van één enkele stappenmotor en leerden we met andere woorden eerst stevig wandelen om ons wellicht later aan een loopnummer te wagen.
Ons dossier is gemaakt met een blik op de toekomst. We wilden een uitdaging aangaan en probeerden zoveel mogelijk de opdracht zelfstandig tot een goed einde te brengen. Zoals bijvoorbeeld het schrijven van de programma's en de toepassing op de werking van een lift op schaalmodel. In het begin van dit schooljaar wisten we nog niet goed wat we van onze geïntegreerde proef moesten verwachten, maar de opdracht was duidelijk: zoveel mogelijk kennis opdoen.
Een stappenmotor zo correct mogelijk besturen via de computer werd ons doel. Dat probeerden we zo goed mogelijk uit te werken in dit dossier.

Uiteraard zou deze opdracht niet geslaagd zijn zonder de hulp van verschillende mensen zowel binnen de school, als daarbuiten. Daarom willen we een speciaal dankwoord aan hen richten. Allereerst onze ouders voor de morele en logistieke steun. Onze klassenleraar, de heer Verhaeghe, die onze begeleider was bij deze geïntegreerde proef, voor zijn steun doorheen het hele schooljaar en voor het in goede banen leiden van dit eindwerk. Ook de heer Vanhooren die ons geholpen heeft bij het aanleren en het gebruik van de besturing die we nodig hadden in dit project. De heren Boterberge en Werbrouck, voor de vele instructies en nuttige tips die ons menige valkuil deden vermijden. Mevrouw Loosveld voor het opzoeken en uitleggen van specifieke informatie. De heer Verhelle omdat hij ons van enkele noodzakelijke, elektrische componenten wilde voorzien. We bedanken ook de heer Tanghe omdat hij twee metalen stukken voor ons draaide, waardoor wij een praktische toepassing konden vervolledigen. De heer De Bruyne voor de houten constructie die we bij een deel van de praktische uitvoering nodig hadden. Mevrouw De Laere bedanken we ook voor het geven van vele interessante tips. Ook dank aan de heer Deschepper voor het ter beschikking stellen van een computer op school en de hulp bij de kennismaking met de werking van computerpoorten. Verder nog een dankwoord voor de heren Bart Vanwalleghem en Frederik D'hulster van het PIH te Kortrijk voor de boeiende samenwerking.
Tot slot danken we allen die van ver of van nabij bij dit eindwerk betrokken waren en elk hun steentje bijdroegen om het tot een goed einde te brengen.

1 Inleiding

Stappenmotoren en robots zijn het onderwerp van deze GIP. Als onderwerp wilden we allebei iets met de computer realiseren. Een haalbaar onderwerp leek dan toch eerder moeilijk te vinden, omdat er hierbij ook een deel praktische uitwerking verwacht wordt. Een programma schrijven, zou dus niet voldoende zijn.
De helpende hand kwam echter van de school, die ons voorstelde om een robotarm (die op school aanwezig was) te gebruiken én deze te sturen met een computer.
De mens maakt al vanaf zijn ontstaan gebruik van werktuigen om zijn werk te verlichten en te vereenvoudigen. Als hij dus een werktuig kon ontwikkelen dat een deel van zijn werk (bijvoorbeeld het gevaarlijke of intensieve werk) van hem overneemt, dan is dit een grote stap vooruit.
Terug naar de robotarm. Dit onderwerp zou zowel een onderdeel programmeren als een onderdeel elektronica bevatten. Dan zou onze GIP ook in samenwerking met het PIH kunnen gebeuren, dat sinds twee jaar GIP's begeleidt en voor de nodige bijstand en bedrijfsbezoeken zorgt. Het PIH is vooral gericht op de theoretische kant van robotica, maar het leek ons toch ook interessant om de meer praktische kant van de robot te bestuderen. Van een robotarm is bitterweinig bekend. Het enige dat we met zekerheid kennen, zijn wat basiseigenschappen van de stappenmotoren die op de robot aanwezig zijn. De rest blijft tot op heden een eerder vaag domein.
Wat stond ons nu te wachten? Om te beginnen bestudeerden we de werking van stappenmotoren. Vorige schooljaar hebben we bijna alle lessen elektriciteit aan motoren gewijd, maar stappenmotoren kwamen hierbij niet aan bod. Dit was een extra motivatie om deze dan ook te bespreken. Na het onderzoek van de werking van de stappenmotor moest hierbij dan ook nog een gepaste sturing gezocht worden. Eerst dachten we aan de SAA1027, maar die bleek verouderd en financieel niet haalbaar. Daarna vonden we de L297: dit is een IC die speciaal ontworpen is voor het sturen van stappenmotoren. Aan de L297 kan men eenvoudig de L298 schakelen, die dan dient om het signaal van de L297 te versterken, zodat die de motor van voldoende spanning en stroom kan voorzien. Een ander onderdeel waar we veel aandacht aan besteed hebben, is het gebruik van de computer om de L297 te sturen.
Door middel van de parallelle poort kunnen we alle signalen die nodig zijn voor het sturen van de L297 nu genereren.

1 Robotica

1.1 Wat is een robot?

Robot; Het woordenboek van Dale omschrijft het als 'kunstmens, mechanisme dat min of meer de gedaante van een mens heeft, en bewegingen, verrichtingen of arbeid kan uitvoeren'.
Een robot kunnen we het best omschrijven als een programmeerbaar, zelfcontrolerend apparaat dat bestaat uit elektronische, elektrische en mechanische onderdelen. De robot is ontworpen om gevaarlijk of intensief werk van de mens over te nemen, hoewel hij in de 21ste eeuw al voor veel meer toepassingen gebruikt wordt. Het grote voordeel van robots is dat ze nooit moe worden, dat ze zonder protest oncomfortabel of gevaarlijk werk kunnen doen (vb. het verven van auto's, het verplaatsen van 'zware' lasten, etc). Een robot heeft ook nooit last van verveling en wordt ook niet van zijn werk afgeleid.
Het concept van de robot is heel oud, maar het woord *robot* wordt pas vanaf de 20ste eeuw gebruikt. Het woord is afgeleid van het Tsjechische woord robota of robotnik, wat zoveel wil zeggen als 'slaaf' of 'bediende'.
De eerste industriële robots werden gebruikt bij het bewerken van radioactief materiaal in labo's. Toen werden ze nog met behulp van schakelaars en 'joysticks' bediend. Tegenwoordig worden robots bediend door meer gesofisticeerde apparaten, zoals computers, PLC's, etc. Vele robots hebben ook sensoren, zodat ze zelf kunnen 'inspelen en reageren' op de omgeving. Dit kan men een zekere vorm al Artificial Intelligence (AI) noemen, omdat de robots dan zelf al kunnen ingrijpen bij een abnormale situatie (al was het maar onmiddellijk tot stilstand komen).

1.2 De 3 wetten van de robotica

1ste wet: A robot may not harm a human being or, through inaction, allow a human being to come to harm.
Een robot mag nooit een mens verwonden, of door niets laten toestaan dat een mens zich verwondt. Die eerste wet is zeker de belangrijkste: hij zorgt ervoor dat robots die een zekere AI bezitten, zich niet tegen de mens zullen verzetten, maar de mens zullen helpen in noodsituaties.

2de wet: A robot must obey the orders given to it by human beings, except where such orders would conflict with the First Law.
Een robot moet altijd de gekregen orders van een mens opvolgen, zolang deze niet in strijd zijn met wet 1. Deze wet zal ervoor zorgen dat de robot steeds een slaaf blijft en niet een eigen leven gaat leiden waarbij de mens opzij geschoven wordt. Ook verzekert men met deze wet dat de robot niet gebruikt kan/mag worden in bijvoorbeeld oorlogen.

3de wet: A robot must protect its own existence, as long as such protection does not conflict with the First or Second Law.
Een robot moet zijn eigen bestaan beschermen, zolang deze niet in strijd is met wet 1 en wet 2. Die derde wet zorgt ervoor dat robots onder elkaar niet

in ruzie kunnen komen. Het heeft geen zin om een oorlog tussen robots en mensen uit te sluiten, als de robots onderling oorlog kunnen voeren.

Het spreekt voor zich dat deze 3 wetten van de robotica pas echt gelden als men een robot met AI bestudeerd. Bij alle andere gevallen zal de mens zelf moeten zorgen dat hij veiligheidsmaatregelen neemt.

1.3 De componenten van een robot

1.3.1 De sturing

Figuur 1: De robotsturing

Als eerste groot onderdeel van de robot hebben we de sturing. Het spreekt voor zich dat een hoop aaneengeschroefd ijzer met wat motoren niet uit zichzelf zal beginnen werken, maar dat een andere (externe) component zal zorgen voor het aansturen van de motoren, werktuigen, etc.
Natuurlijk kan ook die sturing de programma's nog niet zelf schrijven. Daarvoor zijn er twee oplossingen.
Een eerst oplossing is het programma zelf schrijven, maar dit brengt opnieuw wat extra problemen met zich mee. Een eerste probleem zou kunnen zijn dat men om een bepaalde positie te bereiken met de robot al enkele keren te ver gegaan is, m.a.w. dat men het werktuig op de robotarm, of het werkstuk al enkele keren beschadigd kan hebben. Dat is natuurlijk niet de bedoeling. Een tweede probleem is dat het intypen van de programmaregels heel wat tijd in beslag neemt. Twee problemen dus die men in de industrie zoveel mogelijk wil vermijden.

Een tweede oplossing is het gebruik van een 'controlepaneel'. Hiermee kan men op een relatief eenvoudige manier de robot laten bewegen. Die bewegingen worden dan automatisch door de PC onthouden en in het programma gebracht. Hiernaast zien we zo'n controlepaneel van een KUKA-robot. Dit is hetzelfde type van besturingspaneel dat we gebruikt hebben tijdens onze tweede bezoek aan het PIH. (Zie verslag bedrijfsbezoek PIH/KUKA)

1.3.2 De robotarm

Hiernaast zien we de foto van een industriële robotarm. Hij ziet er heel wat minder mooi uit als de robots die in sciencefictionfilms getoond worden. Logisch, want het uiterlijk komt hier natuurlijk op de tweede plaats. Op de foto hebben we de motor met een kadertje aangeduid. Dit is het onderdeel waar wij het meest in geïnteresseerd zijn. De meeste van de robots bevatten ofwel een servomotor ofwel een stappenmotor.
Wij hebben gekozen om de stappenmotor van naderbij te bestuderen. De basis in verband met magnetisme is dezelfde als die van alle motoren. Daarom eerst nog een korte toelichting over elektrische motoren.

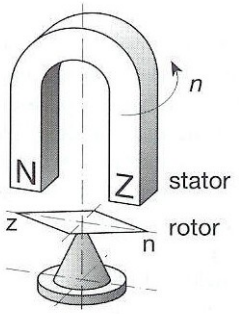
2 Elektrische motoren

2.1 Inleiding

Het doel van een motor is om elektrische energie om te zetten in mechanische energie, m.a.w. in een beweging. Dit kan op twee manieren. Ofwel met een synchrone motor, ofwel met een asynchrone motor. Omdat de SM (stappenmotor) een synchrone motor is, zullen we enkel de basis van de synchrone motor bespreken.

2.1.1 Principewerking

Het principe van een synchrone motor kunnen we het best verduidelijken aan de hand van de volgende figuur.



Figuur 2: Principe van een synchrone motor.

We zien een hoefijzermagneet, met daaronder een draaibare kompasnaald. Als we de magneet dicht genoeg bij de kompasnaald brengen, zal de kompasnaald zich draaien zodat de ongelijknamige polen elkaar aantrekken en vasthouden. Men zegt dat de magneten nu magnetisch gekoppeld zijn. Als we nu de hoefijzermagneet laten draaien, zien we dat de kompasnaald meedraait. Draaien we sneller, dan zal de kompasnaald ook sneller draaien. Draaien we trager, dan draait de kompasnaald ook trager. We kunnen nu zeggen dan de kompasnaald synchroon (met dezelfde snelheid) draait met de hoefijzermagneet.

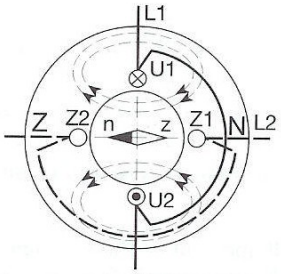
2.1.2 Het draaiveld

Natuurlijk is in bovenstaand principe nog geen gebruik gemaakt van elektriciteit. Daarom vervangen we nu de hoefijzermagneet door een spoel. Die spoel bevestigen we dan in een stator. De magneetnaald vervangen we door een permanente magneet of door een elektromagneet.
Dit hangt af van de grootte en het doel van de motor. De magneetnaald wordt nu de rotor.
Wat we nu bekomen, is een rotor en een stator met één fasewikkeling.
Zie onderstaande figuur:

Figuur 3 : Rotor + stator met 1 fasewikkeling

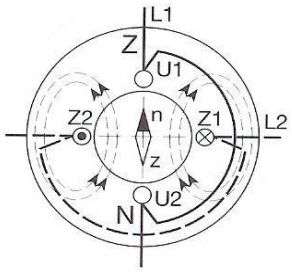
We laten nu een wisselstroom door de fase stromen. We merken dat het magnetisch veld steeds wisselt, en dit met dezelfde snelheid als de frequentie van de stroom. Als we nu de rotor van dichterbij bekijken, stellen we vast dat deze niet draait (men spreekt hier ook niet van een draaiveld, maar van een wisselveld). Dit is logisch, want de rotor weet simpelweg niet naar welke kant hij moet draaien. De ene keer draait hij naar links weg, de andere keer naar rechts. Dit is niet wat we van een motor verwachten, dus moet er een oplossing gevonden worden.

We kijken naar de onderstaande figuur:



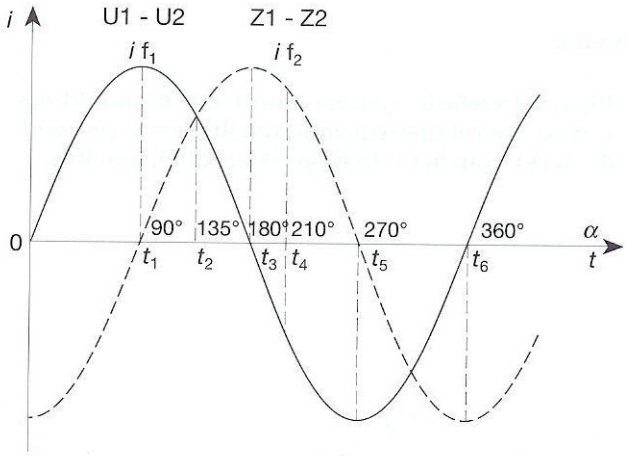
Figuur 4 : Rotor + tweefasige statorwikkeling

We zien dat de stator nu voorzien is van twee wikkelingen. Die worden elk door een verschillende geleider van stroom voorzien. Als we nu de twee stromen zo regelen, zodat de ene stroom 90° naijlt op de eerste, dan zien we dat de volgende stap, dit wordt:



Figuur 5 : Rotor + tweefasige statorwikkeling

Als de eerste stroom zijn nuldoorgang bereikt, bereikt de andere stroom zijn maximum. De twee fasen zullen dus afwisselend een maximum flux opwekken. De stroom ziet er dan als volgt uit:



Figuur 6 : Stroomdiagram bij twee statorwikkelingen

Op die manier bekomen we een draaiveld.
Dit principe kunnen we ook toepassen op 3-fasige motoren. Het faseverschil zal daar 60° zijn, maar de basis blijft dezelfde.
In de praktijk komen tweefasige motoren zelden voor. Dit om de simpele reden dat nergens een tweefasige spanning voor handen is (tenzij die zelf gegenereerd wordt). Driefasige spanning is wel eenvoudig voor handen en wordt om die reden dan ook veel gebruikt.

2.2 De stappenmotor

2.2.1 Inleiding

Wat is een stappenmotor?
Een SM is een motor die in staat is om een digitaal elektrisch signaal om te vormen in een zekere hoekverplaatsing. Daar deze motor zo precies kan draaien (telkens met een vaste hoekverandering), wordt die heel vaak gebruikt bij positionering (komen we later nog op terug). Bij de stappenmotor is het dus belangrijk dat je op elk moment kan vaststellen onder welke hoek de rotor zich bevindt.

Hieronder enkele voordelen van de SM.

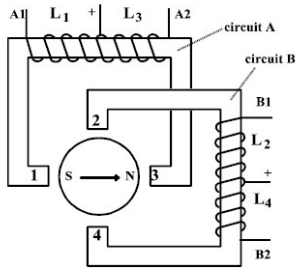
Voordelen:

- Heel snel innemen van een nieuwe hoekpositie
- Heel nauwkeurig in open keten (dus zonder terugkoppeling)
- Direct bruikbaar in geval van digitale controle-instructies
- Groot snelheidsbereik
- Constante snelheid bij stabiele controlefrequentie
- Korte aanloop- en afremtijden
- Betrouwbare werking
- Geen onderhoud
- Grote levensduur
- Groot houdkoppel bij stilstaande rotor

Door al deze voordelen wordt de SM vaak toegepast in robotica, coördinatentafels, plotters, printers, discdrives, etc.

2.2.2 Principiële werking

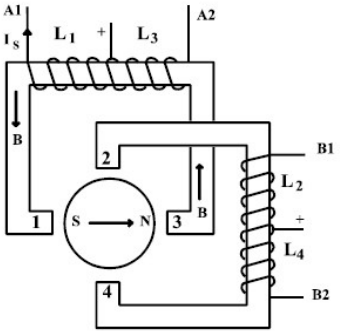
Om de werking van een stappenmotor te begrijpen, bekijken we eerst de onderstaande figuur:



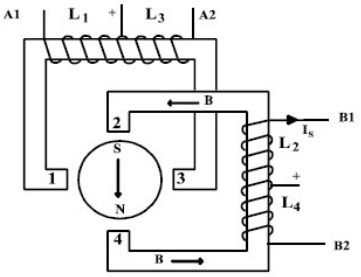
Figuur 7 : Schematische opstelling van een SM

Op bovenstaande figuur zien we tweepolige PM-stappenmotoren (de unipolaire motor wordt later uitgelegd) . Elk circuit bestaat uit twee lichamelijke polen en twee deelwikkelingen. Voor Circuit A zijn dat de polen 1 en 2, en de deelwikkelingen L1 en L3. Voor Circuit B zijn dat de polen 2 en 4, en de deelwikkelingen L2 en L4. Als men nu één van de fasen bekrachtigt, zal de rotor een voorkeurspositie innemen (de positie met de

minste magnetische weerstand). Daar er vier fasen zijn, zijn er vier voorkeursposities. We kunnen zeggen dat de motor vier stappen heeft per omwenteling.

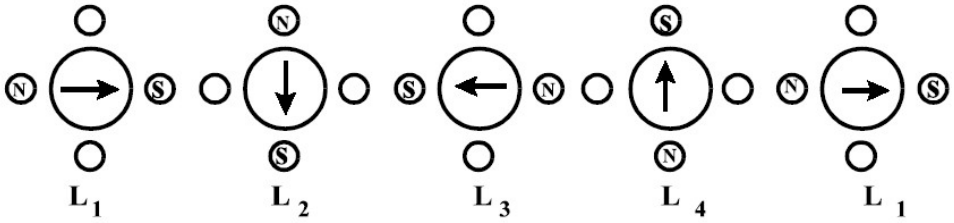


Figuur 8 : Fase L2 bekrachtigd



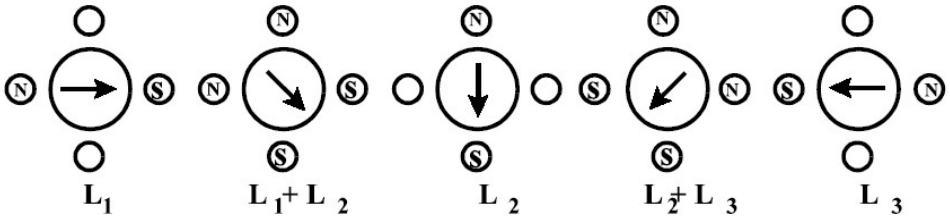
Figuur 9 : Fase L1 bekrachtigd

Stel, we bekrachtigen fase L1 (fig. 7). Met de rechterhandregel kunnen we nu de richting van het magnetisch veld bepalen en zo de stand van de rotor. We laten vervolgens de spanning in fase L1 wegvallen, en zetten fase L2 onder spanning. De rotor zal nu een nieuwe positie innemen, zodat de veldlijnen weer de weg van de minste magnetische weerstand kunnen volgen. Het spreekt voor zich, dat als we nu fase L3 bekrachtigen, de rotor zich opnieuw 90° zal draaien. Voor fase L4 geldt net hetzelfde.



Figuur 10 : Schematische voorstelling volstapsbedrijf (Full step)

Wat we daarnet gezien hebben, is dat de rotor telkens 90° zal draaien als we een andere fase bekrachtigen. Stel dat we nu tussen twee stappen twee fasen bekrachtigen. Dit wil dan zeggen dat de voorkeurspositie tussen twee fasen ligt (in dit geval op 45°). Dit is een eenvoudige manier op het aantal stappen te verdubbelen. Schematisch wordt het dan zo voorgesteld:



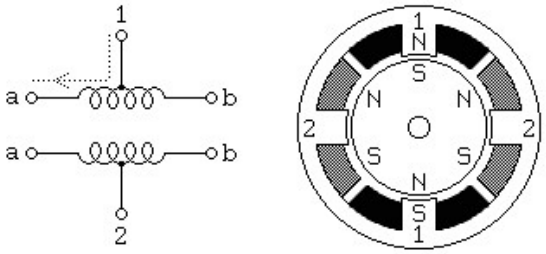
Figuur 11: Half stap bedrijf (Half Step)

Dit zijn de twee meest voor de hand liggende stapmethoden. Later volgt nog een stuk waarin andere stapmethoden uitgelegd worden.

2.2.3 Soorten stappenmotoren

De stappenmotoren kunnen we indelen in 2 grote groepen. Motoren met variabele reluctantie (VR), en motoren met een permanent magnetisme (PM). De motoren kan men onderscheiden door simpelweg aan de rotor te draaien. Als je dan een weerstand ondervindt (weliswaar heel beperkt), heb je te doen met een PM-motor. Natuurlijk zal je bij een VR-motor ook enige weerstand ondervinden, maar die is afkomstig van het remanent magnetisme. Het onderscheid kun je ook waarnemen als je met de ohm-meter over de klemmen van de motor meet. Heb je 3 of 4 spoelen die dezelfde massa delen, dan heb je een VR-motor. Als je twee gescheiden spoelen hebt, dan spreekt men van een PM-motor. Hebben die twee spoelen dan nog eens een middenaftakking, dan noemt men die motoren unipolaire stappenmotoren. In het andere geval spreekt men van bipolaire motoren.

2.2.3.1 Unipolaire motoren



Figuur 12 : De schematische voorstelling van een unipolaire motor

Op bovenstaande figuur zien we de schematische voorstelling van een unipolaire motor. We merken duidelijk dat de fasewikkelingen uit twee afzonderlijke wikkelingen bestaan en dat die wikkelingen elk een middenaftakking hebben. Deze middenaftakking wordt doorgaans aan de positieve pool gelegd. Om de fase om te polen, hoeft men nu enkel de andere klemmen afwisselend met de massa te verbinden.
We zien ook dat de rotor 3 polenparen heeft (3 keer noord en 3 keer zuid). Dit is enkel om de staphoek te verkleinen. Stel, fase 1 is bekrachtigd. De motor staat nu stil. Willen we de motor doen draaien, dan bekrachtigen we nu fase 2. Als we goed naar de rotor kijken, zullen we vaststellen, dat die 30° zal draaien voor hij weer in zijn voorkeurspositie komt.

Nu moeten we de motor wel nog laten ronddraaien. Hiervoor kunnen we de volgende sequentie gebruiken (4 omwentelingen).

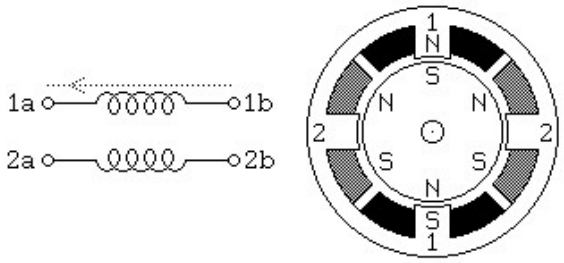
```
Winding 1a 10001000100010001000010001
Winding 1b 00100010001000100010001000
Winding 2a 01000100010001000100010001000
Winding 2b 00010001000100010001000100010
          Tijd =>
```

We merken dat telkens één van de spoelen bekrachtigd is. De andere spoelen doen dus niets. We kunnen dit ook anders doen, zoals hieronder:

Winding 1a 11001100110011001100110011001
Winding 1b 00110011001100110011001100110
Winding 2a 01100110011001100110011001100
Winding 2b 10011001100110011001100110011
Tijd =>

We stellen vast dat hier telkens twee fasen bekrachtigd zijn. Dit heeft natuurlijk een groot voordeel. Hoe meer veldlijnen er inwerken op de rotor, hoe groter het koppel dat de motor kan leveren. In deze tweede situatie kan de motor een koppel ontwikkelen dat 1.4 keer groter is dan de eerste.

2.2.3.2 Bipolaire motoren



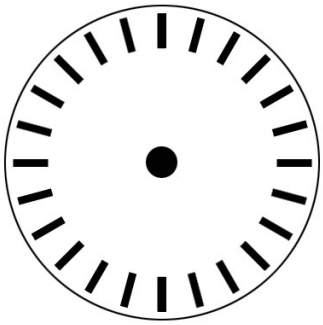
De bouw van de bipolaire motoren is heel gelijklopend met die van de unipolaire motoren. Alleen zal men bij de bipolaire motoren geen middenaftakking vinden. De fase heeft dus maar twee aansluitklemmen. Dit wil ook zeggen dat de sturing die erachter zit, helemaal anders is, want om de fase om te polen moet men beide klemmen ompolen.

2.2.4 Wat kunnen we aanvangen met een stappenmotor?

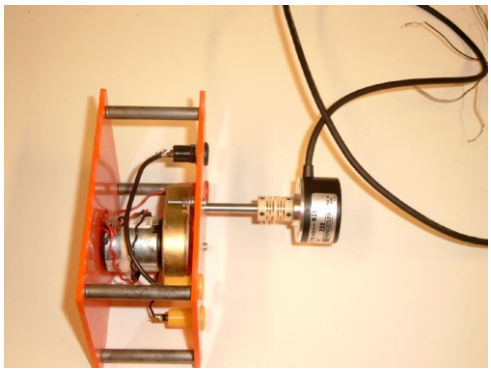
Nu we de verschillende soorten stappenmotoren en hun werking hebben besproken, verduidelijken we even waarom een stappenmotor gebruikt wordt.
Ten eerste kunnen we de stappenmotor als gewone motor gebruiken. Met dit voordeel dat de rotatiesnelheid heel gemakkelijk te regelen is. Natuurlijk is het zinloos een stappenmotor te maken die in stappen van 1,8° stapt als die enkel continue rond moet draaien. Dus, er moet nog een andere toepassing zijn. En die is er ook onder de vorm van 'positioneren'.
Positioneren kunnen we het best beschrijven als het in de juiste positie brengen van een werkstuk/werktuig/rotor. Als we dit toepassen op stappenmotoren wil dit dus zeggen dat we op ieder moment van het proces exact weten waar de rotor zich bevindt.
Bij positioneren hebben we twee basisprincipes: positioneren met terugkoppeling én positioneren zonder terugkoppeling.

2.2.4.1 Positioneren met terugschakeling

Bij positioneren met terugschakeling maakt men meestal gebruik van een gewone DC-motor, met op de motor nog een encoder bevestigd. Een encoder is een toestel dat door middel van licht de hoekverandering kan opmeten.



Figuur 13 : Voorstelling encoderschijf



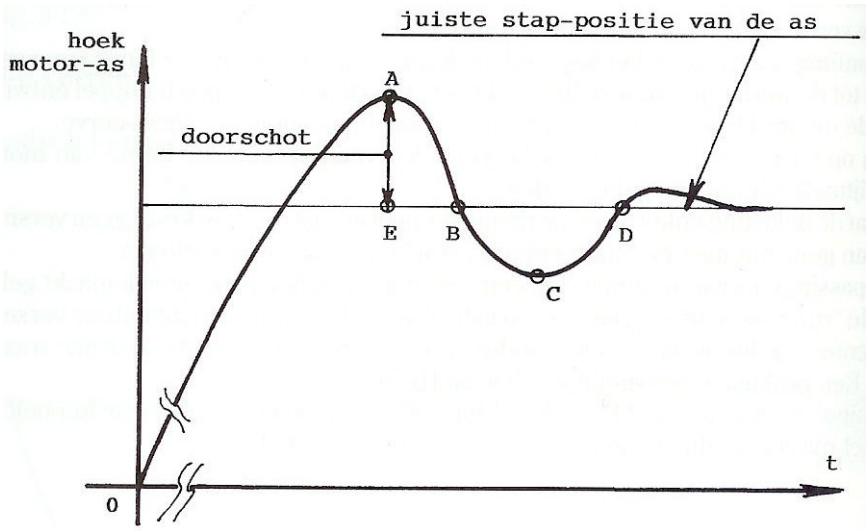
Figuur 14Encoder op een DC motor

Op bovenstaande figuur zien we de voorstelling van een encoderschijf. De encoder wordt zo geplaatst dat hij met dezelfde rotatiesnelheid draait als de motor (of de as, zolang men maar ergens een vaste referentie heeft). Aan de ene kant van de schijf zit een lichtbron, aan de andere kant een lichtgevoelige cel. Het spreekt voor zich dat bij deze opstelling extra elektronica of software nodig is om te bepalen hoeveel omwentelingen de as gemaakt heeft. Het nadeel van deze methode is dat er resonantie kan optreden (door overshoot zal de sturing steeds willen bijsturen, en hierdoor in resonantie komen).

2.2.4.2 Positioneren zonder terugkoppeling

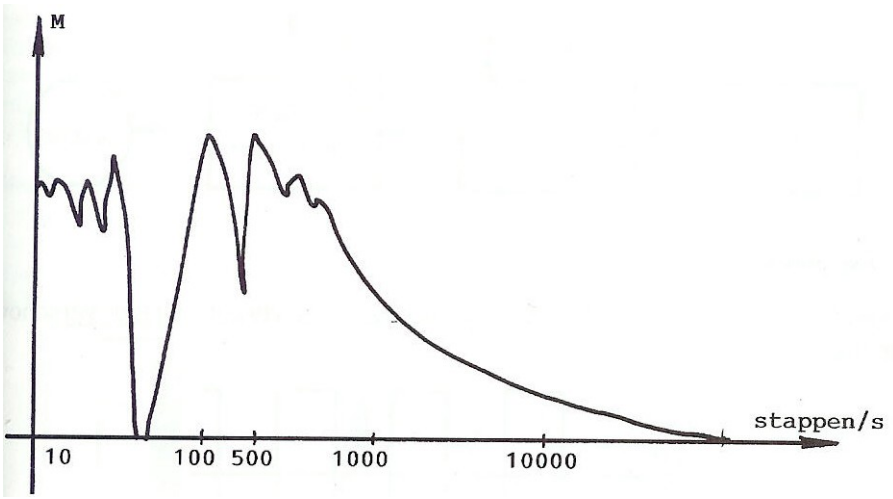
Bij positioneren zonder terugkoppeling heeft de motor geen apparaat dat de sturing vertelt hoeveel omwentelingen de rotor gemaakt heeft. Er moet dus iets anders zijn waardoor men weet hoeveel omwentelingen de rotor al gemaakt heeft. Hiervoor zal men nu gebruik maken van een stappenmotor. De stappenmotor wordt altijd zo gestuurd dat men weet hoeveel stappen de motor gemaakt heeft. Aan de hand van de hoekverandering per stap en het aantal stappen kan men zo dus het aantal omwentelingen berekenen. Dit systeem heeft natuurlijk ook een groot nadeel. En dat is dat je nooit zeker weet als de rotor op zijn plaats aangekomen is. Het is heel goed mogelijk dat de rotor door overbelasting enkele stappen overslaat. Als de motor, die eventueel een plotter aandrijft, dan terug naar de nulpositie gaat, kan hij heel wat schade aanrichten.

2.2.5 Doorschot (overshoot)



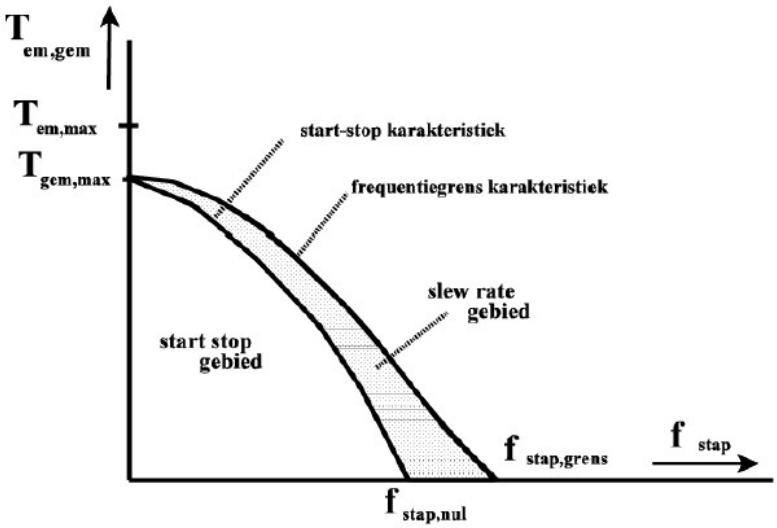
'Doorschot' kunnen we het best uitleggen als het te ver draaien van de rotor. Dit komt door de massatraagheid van de rotor zelf, of van de mechanische onderdelen die de motor aanstuurt. Het is logisch dat we die doorschot zoveel mogelijk willen beperken. Het blijft de bedoeling om een bepaalde afgelegde weg af te leggen, en niet om te veel weg af te leggen om dan op je passen te moeten terugkomen.
Op de figuur zien we tussen A en E het maximum doorschot. Het koppel dat hier ontwikkeld is, is niet groot genoeg om naar de volgende voorkeurspositie te stappen en de rotor zal dus een stuk terugdraaien. Tijdens die beweging is het bijna onmogelijk de motor opnieuw te starten. De rotor zal nu blijven draaien tot ze in punt C komt, om zo weer naar de andere kant te schieten. Natuurlijk is dit een gedempte trilling, die van heel korte duur is. Maar als de motor snel na elkaar gestart en gestopt moet worden, kan dit voor problemen zorgen.

2.2.6 Resonantie



Zoals elk mechanisch geheel heeft ook de stappenmotor een resonantie-frequentie. Vooral RM-motoren kunnen hier hinder van ondervinden. Bij een stappenimpulsfrequentie die in de omgeving komt of samenvalt met de resonantiefrequentie kan, bij onvoldoende demping, de mechanische resonantie opgewekt en versterkt worden. In dit gebied zal de stappenmotor heel onstabiel werken. Het is zelfs mogelijk dat de motor op z'n stappen terugkeert in plaats van vooruit te draaien.
PM-motoren hebben een koppelverlies rond 1000Hz als gevolg van de resonantieverschijnselen. De resonantie kan minimaal gemaakt worden door het traagheidsmoment van de belasting te vergroten. Dit gaat natuurlijk ten kosten van de snelle reactie en veroorzaakt meer doorschot.

2.2.7 De toerental-moment karakteristiek



Figuur 14 : Koppel-toerental karakteristiek

Als het toerental van de motor stijgt, zal de flux dalen. Dit heeft natuurlijk als gevolg dat het koppel ook zal dalen. Dit zal men altijd weergeven in twee grafieken. Een 'running' grafiek ook wel slew-curve en een 'start-without-error' grafiek. Op de grafiek komt 'running' overeen met start-stop karakteristiek, en 'start without error' overeen met de frequentiegrens karakteristiek.
Op de running grafiek zien we welk koppel gepaard gaat met welk toerental als de motor draait. Dit toerental moet behaald worden door de motor langzaam te doen versnellen.
Naast de running grafiek heeft men ook nog een 'start-without-error' grafiek. Deze grafiek toont het toerental waarmee de motor mag starten en versnellen. De 'start-without-error' grafiek is lager, omdat men hier natuurlijk rekening moet houden met de massatraagheid van de rotor en het mechanisme.

Op de grafiek zien we ook nog een paar andere aanduidingen staan.

Holding torque is het maximum koppel dat aan de motor gelegd mag worden zonder dat die begint te draaien. (De fasen zijn bekrachtigd)

Detent torque is het koppel dat aan de motor gelegd mag worden als die niet belast is (afkomstig van het remanent magnetisme).

Pull-out torque is het maximum koppel dat aan de motor gelegd mag worden bij een bepaald toerentaal. Als men dit koppel overschrijdt, zal de motor stappen beginnen overslaan.

Pull-in torque is het maximum koppel waarmee de motor mag starten bij een bepaald toerental. Idem als pull-out torque: als men dit koppel overschrijdt, zal de motor stappen overslaan.

Pull-out rate is het maximum toerental waarmee de rotor mag draaien bij een bepaald koppel. Als dit toerental overschreden wordt, zal de motor niet meer synchroon draaien en dus stappen verliezen.

Pull-in rate is het maximum toerental waarmee de rotor mag starten. Net zoals bij pull-out rate, mag dit toerental niet overschreden worden of de motor zal stappen overslaan.

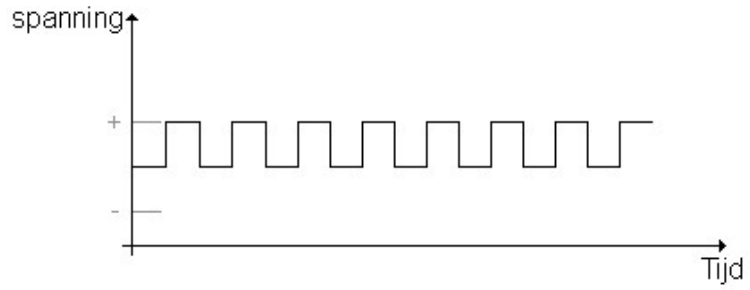
Maximum slew rate is het maximum aantal stappen per seconde dat de motor mag stappen. Zoals we op de grafiek kunnen zien, kan de motor bij dit toerental geen koppel meer leveren.

3 Het stuursignaal

3.1 Vooraf

In het vorige hoofdstuk hebben we gezien dat de stappenmotor ons in staat stelt om een digitaal signaal rechtstreeks (met de sturing weliswaar) om te zetten in een hoekverandering. Eerst en vooral kan men een gewone pulsgenerator gebruiken (Bv. Een L555). Dit brengt natuurlijk heel wat nadelen met zich mee. Bijvoorbeeld: het is moeilijker om de frequentie precies te regelen, of om een programma te schrijven dat de motor moet doorlopen. Als alternatief kan men een µ-processor of zelfs een computer gebruiken om de puls te genereren. Het grote voordeel echter is dat de frequentie precies te regelen is (bijvoorbeeld door het kloksignaal van de computer als 'basissignaal' te gaan gebruiken), en ook relatief eenvoudig te wijzigen is qua frequentie (hiervoor zorgt de gepaste software dan).

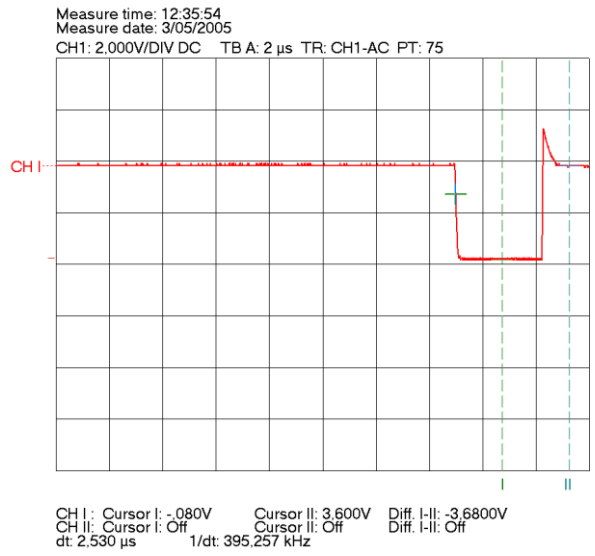
Wie de puls genereert, is eender. Hoe het signaal eruit moet zien niet. Als de IC reageert op een stijgende flank, dan is het nodig dat die puls een stijgende flank bezit die steil genoeg is zodat de IC hem als een stijgende flank waarneemt. De meest gewenste puls is natuurlijk een perfecte blokgolf, zoals getekend op volgende figuur.



In onze bespreking zullen we twee types van stuursignalen gebruiken. Een eerste type zal het sync-signaal van de L297 zijn, dat we aan de hand van de 4024 zullen opsplitsen in andere frequenties. De tweede signaalvorm zal een signaal uit de PC zijn.

3.2 Het sync-signaal van de L297, samen met de 4024 als stuursignaal

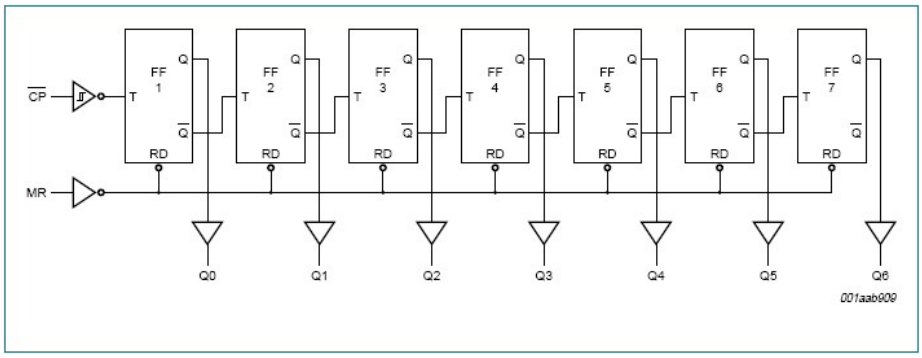
Op onze schakeling wordt gebruik gemaakt van het sync-signaal dat aan pin 11 op de L297 te vinden is. Dit signaal is een constante puls. Maar aangezien we de motor op verschillende snelheden willen laten draaien, wordt een 4024 gebruikt.



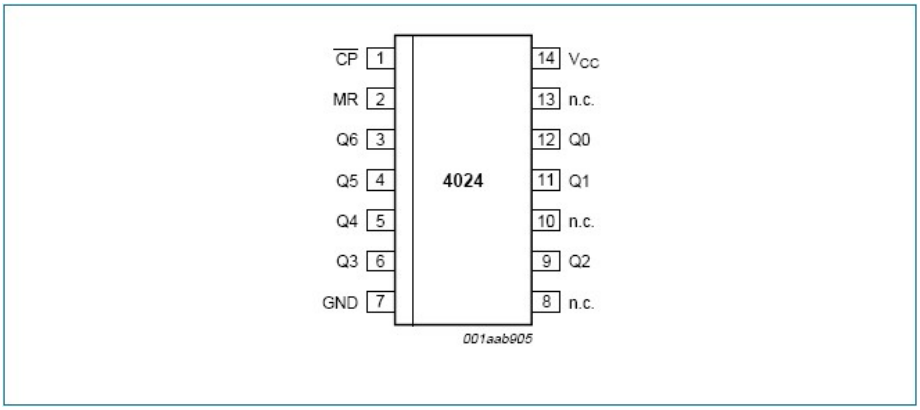
Figuur 15 : Sync-singaal van de L297

De HCT4024 7-stage binary ripple counter is, zoals de naam het al verraadt, een teller. Deze teller heeft zeven bits, wat ons in staat stelt om tot 128 te tellen. Maar deze functie van de 4024 wordt echter niet gebruikt. We zullen hem wel gebruiken om de puls die de L297 geeft, te delen. Met andere woorden : we kunnen de frequentie enkel verlagen, maar niet verhogen.

Laat ons eerst het logische schema van de 4024 bekijken:



Figuur 16 : Logisch diagram



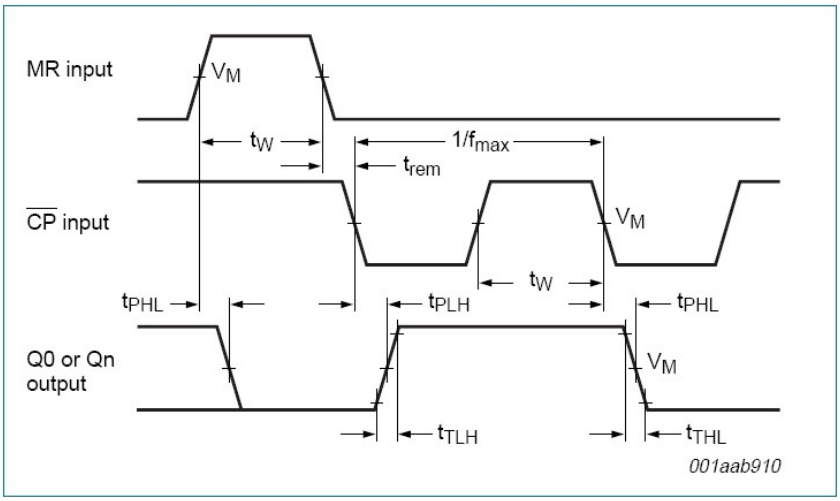
Figuur 17 : Pin configuratie

In volgend rooster vinden we de pinconfiguratie van de 4024:

Symbool	**Pin**	**Beschrijving**
CP	1	Klok ingang, getriggert op de negatieve flank
MR	2	Master reset input. (actief = hoog)
Q6	3	Parallelle uitgang 6
Q5	4	Parallelle uitgang 5
Q4	5	Parallelle uitgang 4
Q3	6	Parallelle uitgang 3
GND	7	Massa (Ground)
n.c.	8	Niet aangesloten (not connected)
Q2	9	Parallelle uitgang 2
n.c.	10	Niet aangesloten (not connected)
Q1	11	Parallelle uitgang 1
Q0	12	Parallelle uitgang 0
n.c.	13	Niet aangesloten (not connected)
Vcc	14	Voeding

Wat uitleg bij de functie van elke pin:

Uitgang op 4024	Deling
Q0	deling door 2
Q1	deling door 4
Q2	deling door 8
Q3	deling door 16
Q4	deling door 32
Q5	deling door 64
Q6	deling door 128



Figuur 18 : Functie van MR en CP

(Reactietijden zijn terug te vinden in de datasheets in de bijlage)
Op bovenstaande figuur wordt de functie van MR (Master Reset) en CP (Clock Input) verduidelijkt. Als MR hoog wordt, zien we dat alle uitgangssignalen van de teller (Q_0 → Q_n) laag worden. Bij het CP-signaal wordt bij een negatieve flank geteld. Dus, bij de eerste negatieve flank zal de 1ste bit van de teller van 0 naar 1 veranderen. Bij de tweede negatieve flank zal de 1ste bit weer veranderen van 1 naar 0. Dus, als we gebruik maken van de 4024 wordt de frequentie van de stuurpuls altijd minstens door twee gedeeld. Hiervoor moeten we gebruik maken van uitgang Q_0. Maar, de IC heeft 7 uitgangen, d.w.z. dat we onze frequentie kunnen delen door 2 (Q_0), door 4, 8, 16, 32, 64 en 128.

De sequenties die de teller dan zal doorlopen zijn de volgende:

```
Q0: 0101010101010101010101010101010101010101010101010101010101010101010101010101010101010101
Q1: 0011001100110011001100110011001100110011001100110011001100110011001100110011001100110011
Q2: 0000111100001111000011110000111100001111000011110000111100001111000011110000111100001111
Q3: 0000000011111111000000001111111100000000111111110000000011111111000000001111111100000000
Q4: 0000000000000000111111111111111100000000000000001111111111111111000000000000000011111111
Q5: 0000000000000000000000000000000011111111111111111111111111111111000000000000000000000000
Q6: 0000000000000000000000000000000000000000000000000000000000000000111111111111111111111111
```

Om de duidelijkheid te bewaren, zullen we de tekst niet nog kleiner zetten om ook de laatste volledige signaalvorm voor te stellen. Wat je ziet, zou al een duidelijk beeld moeten geven van wat de 4024 met ons signaal doet.

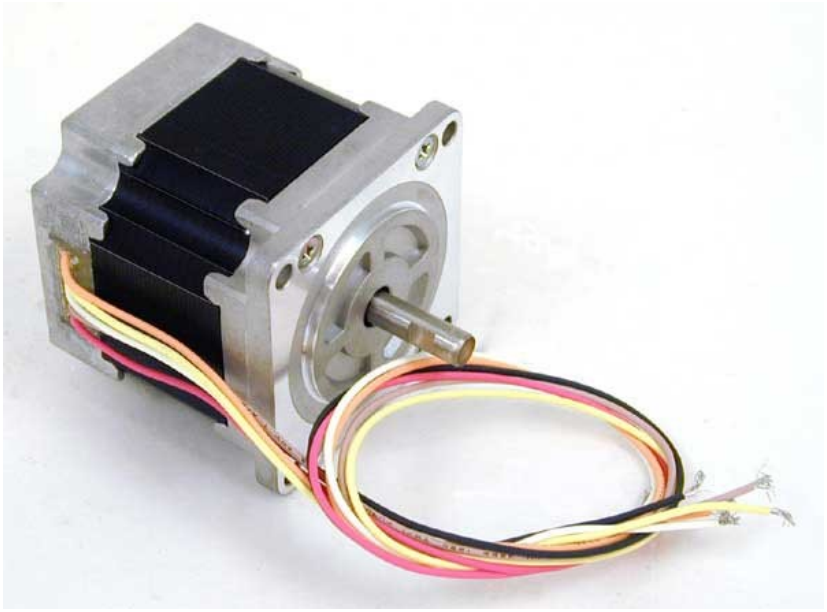
3.3 Sturen met de PC

Wordt in hoofdstuk 6 uitgebreid besproken.

4 De L297

4.1 Vooraf

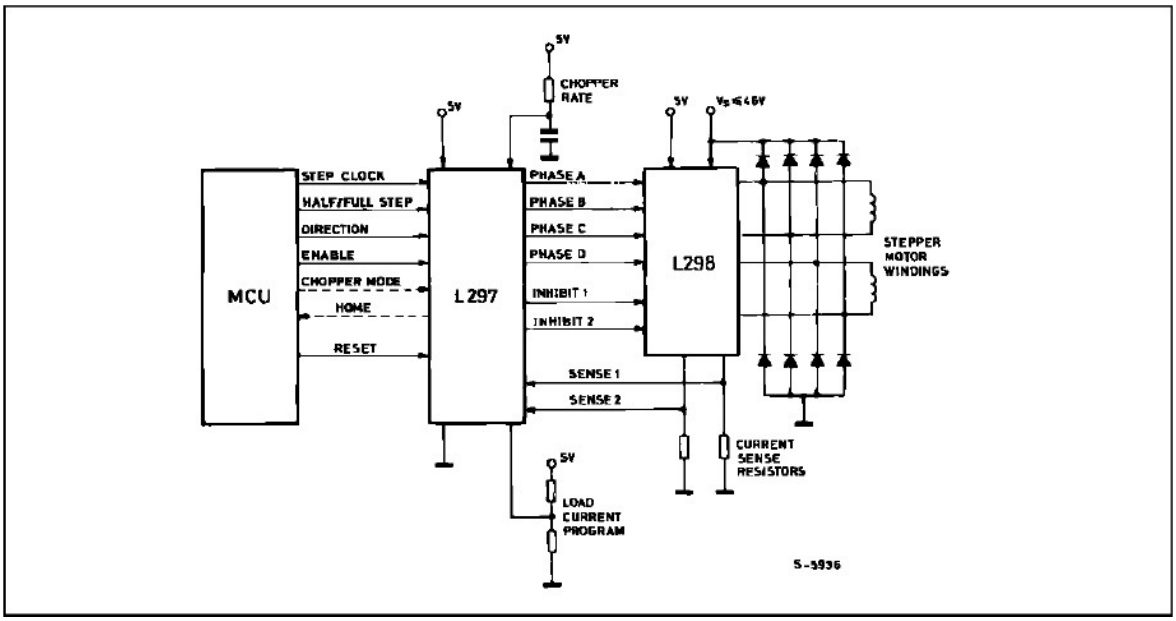
Zoals we al bij de stappenmotor hebben uitgelegd, is het van het grootste belang dat we weten waar de rotor zich bevindt. En dit kunnen we enkel doordat we weten dan de rotor zich zal positioneren volgens de magnetische positie van de stator. (ZN-NZ). Het is dus nodig dat we zelf kunnen beslissen op welk moment welke fase bekrachtigd moet worden, en daarvoor gebruikt men een L297.



Figuur 19 : De stappenmotor

4.2 Algemene info

De L297 wordt hoofdzakelijk gebruikt in samenwerking met een L298 of een L293 als stappenmotor controller. De L297 wordt meestal door een ander externe component gestuurd, meestal een microprocessor, computer of VCO. Wij zullen gebruik maken van het sync-signaal van de L297, een besturing via de PC (zoals besproken in vorig hoofdstuk). Deze microprocessor zorgt meestal voor alle nodige stuursignalen. Met de juiste vermogencomponenten kan men zonder probleem een tweefasige bipolaire PM SM, of een vier fasige VR SM sturen. De L297 is ook in staat om in normal drive, wave drive en half step drive the stappen. Van de L297 zijn tevens ook twee versies te verkrijgen. De L297 en de L297A. De L297A is uitgerust met een stap puls verdubbelaar, en is speciaal gemaakt voor floppydiskkop/CD positionering.



Figuur 20 : De L297 geschakeld met de L298

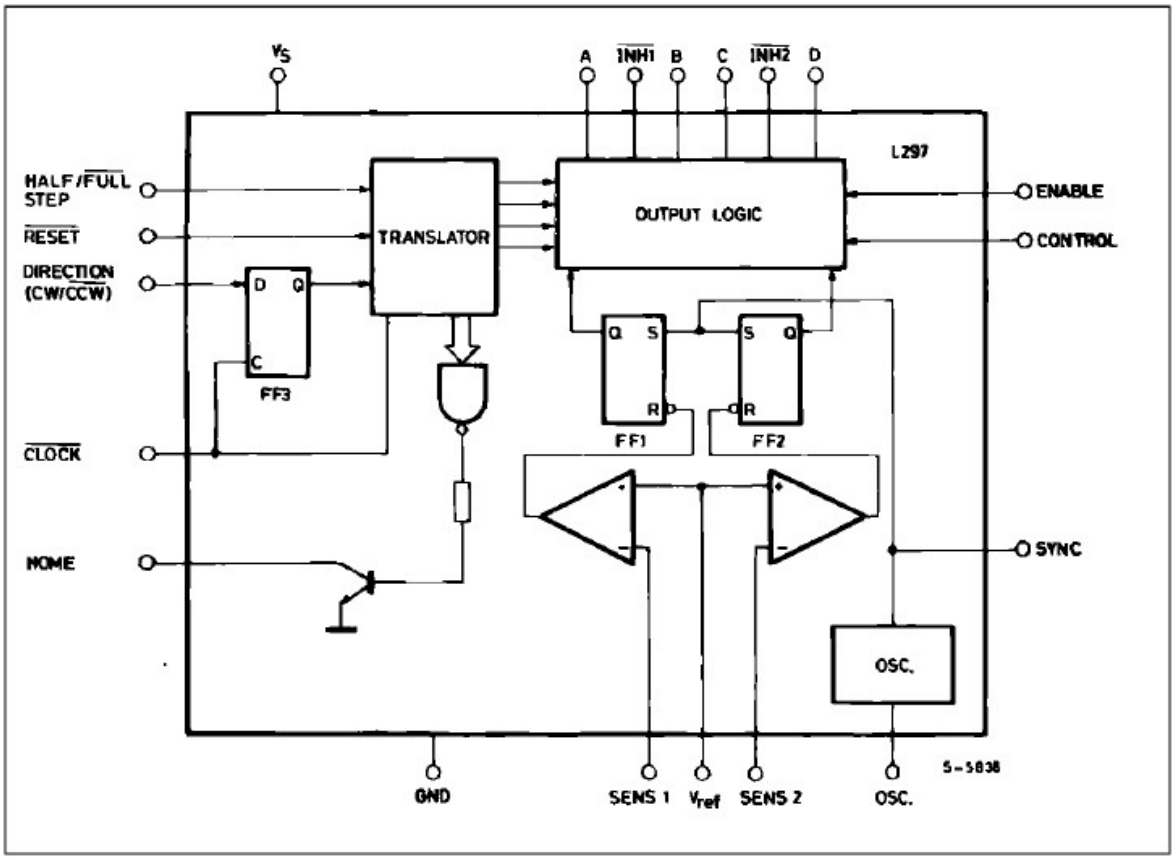
4.3 Voordelen

De L297 heeft vele voordelen. Een eerste voordeel is de lage kostprijs. Het feit dat je voor een schakeling met de L297 weinig componenten nodig hebt (grotere betrouwbaarheid, lagere kostprijs en minder plaats nodig), en dat de belasting om de microprocessor verminderd wordt. Breidt men de L297 uit met een L298, dan kan men bipolaire motoren tot 2 A sturen. Wil men nog grotere stromen sturen, dan moet men kiezen voor vermogentransitoren of 'Darlingtons' (bv de ULN2075B)

Een tweede voordeel van de L297 is de zeer frequent gebruikte IC. Hieronder vind je een lijst met toepassingen:
Printers (cartrige positionering, papieraanvoer), elektronische typemachines, plotters, allerhande CNC-toepassingen, robotten, floppy disk-drivers, CD-stations, elektronische schakelmechanismen, elektronische naaimachines, elektronische carburatoren, telex machines, fototoestellen, elektronische kleppen, en nog veel meer.

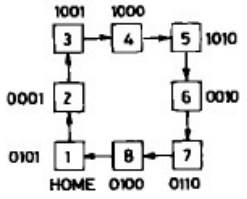
4.4 Het genereren van de fasesequenties

Zoals we al eerder zagen bij de stappenmotoren is de basis van een stappenmotorsturing een component dat elke fase van de SM op zijn beurt kan bekrachtigen. De L297 is zo'n component. Hieronder vind je het inwendige schema van de L297



Figuur 21: Inwendige schema van de L297

Dit is het hart van de L297. Deze bovenstaande 'blok' wordt gestuurd door drie invoersignalen (er zijn meer stuursignalen mogelijk, maar dit is het absolute minimum). Twee signalen voor de manier van stappen namelijk een richtingssignaal (eng: direction) dat verantwoordelijk is voor de draairichting van de motor (cw of ccw) en een halfstaps/volstaps-signaal (half/full step). Dat is verantwoordelijk voor het ofwel halfstaps stappen, of volstaps stappen. Het derde signaal is een kloksignaal dat de schakeling zegt hoe snel ze moet schakelen (dit is een blokgolf).
Het binnenkomende kloksignaal zal door de vertaler (Eng: translator), die bestaat uit een 3bit counter en een 'logic', omgezet worden in een 8-stapssequentie. Deze sequentie noemt men de mastersequentie en bevat alle mogelijke stapssequenties die de L297 kan genereren (3 in dit geval). De mastersequentie ziet er als volgt uit:

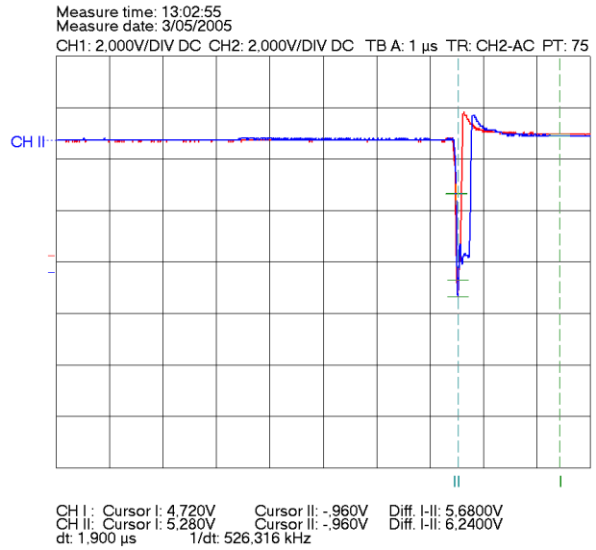


Figuur 22 :
Schematische voorstelling van de mastersequentie

De mastersequentie is de sequentie die gebruikt zal worden voor halfstaps bedrijf. Het volstapsbedrijf kan men verkrijgen door telkens het signaal, waar twee spoelen op hetzelfde moment bekrachtigd worden, over te slaan. Hierdoor zal telkens maar 1 spoel per stap bekrachtigd worden.
(Halfstepping en fullstepping wordt verder in het dossier nog uitgelegd).

4.5 Andere in- en uitgangen van de L297

De in- en uitgangsignalen die we hierboven besproken hebben (Half/Full Step, Direction, Clock, A, B, C en D) zijn veruit de belangrijkste. Maar de L297 heeft natuurlijk nog meer in- en uitgangsignalen die we hier nu zullen bespreken.
Reset: dit signaal is een invoersignaal, dat 'actief' wordt als men op de ingang een hoog signaal aanlegt. Dit signaal zal de L297 terug in zijn beginpositie plaatsen. Het speelt geen rol welke fase bekrachtigd is: als de reset-input hoog is, zal de volgende stap van de motor altijd ABCD=0101 zijn. Deze toestand wordt ook state 1 genoemd.
Home: Dit is een output-signaal dat zal aangeven wanneer de sequentie zich een state 1 bevindt. Als de sequentie zich in state 1 bevindt, zal de transistor open staan en kan er dus stroom vloeien. Met dit signaal kan men eventueel het aantal stappen tellen, indien de L297 met een VCO of andere constante pulsgenerator verbonden is, en de pulsen niet geteld kunnen worden.
INH1 en INH2: Dit zijn twee signalen die men rechtstreeks met de L298 zal verbinden. Deze signalen zullen ervoor zorgen dat de spoelen in de motor sneller ontladen, waardoor men ze sneller kan ompolen. Dit is nodig om het toerental op te drijven. Deze signalen worden rechtstreeks met pin 5 en 11 van de L298 verbonden.



Figuur 23 : Signaalvorm van INH1 en INH2

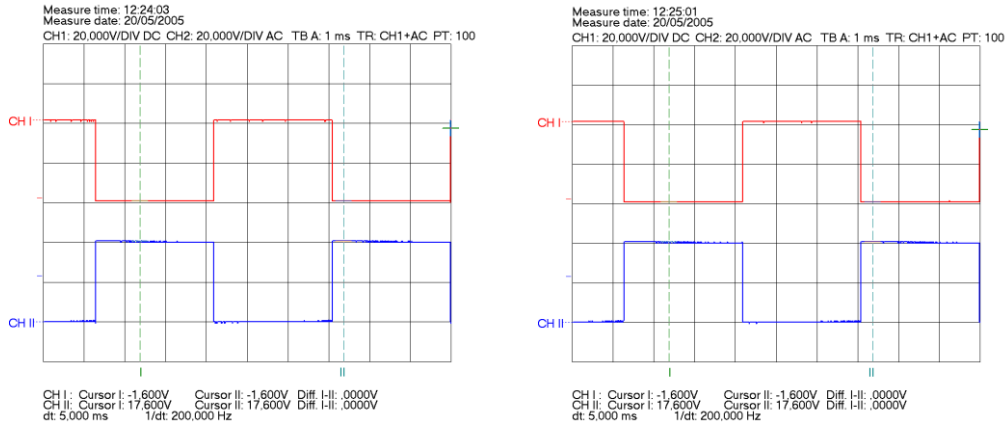
Enable: Dit is een input-signaal dat men kan gebruiken om de L297 aan en af te schakelen. Als men de input laag brengt, zal de output van A, B, C, D, INH1 en INH2 laag gebracht worden. Men kan deze functie als de aan-uitschakelaar van de IC beschouwen.

4.6 Stuurmodes

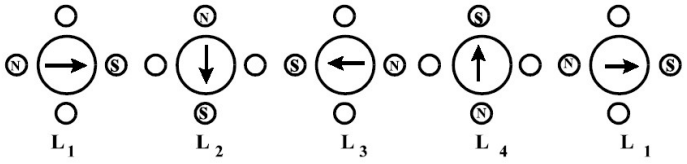
Onder stuurmodes verstaan we de sequenties die de uitgangen A, B, C en D van de L297 doorlopen. Deze kunnen in alle opzichten van elkaar verschillen, met als gevolg dat de motor ook anders zal reageren op belastingen.

4.6.1 Full step drive

Zoals eerder al besproke, zal bij fullstepdrive elke spoel afzonderlijk bekrachtigd worden. Bij onze tweefasige unipolaire motor wil dit dus zeggen dat de motor per sequentie 4 stappen zal zetten.

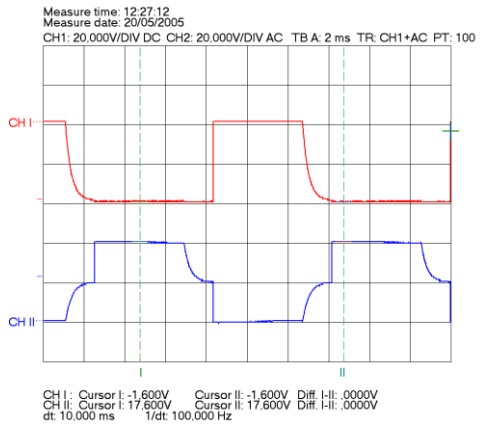


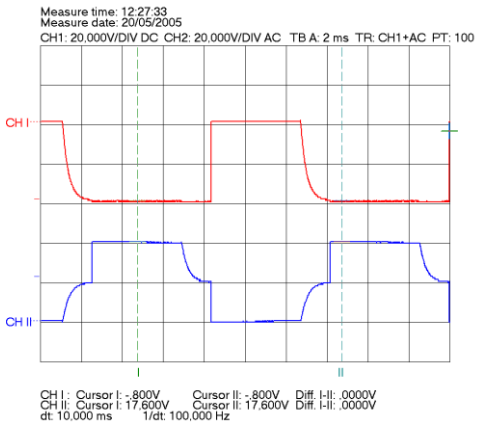
Op de eerste karakteristiek zien we de signaalvorm aan uitgang A en B van de L297, op de tweede karakteristiek zien we de signaalvorm aan uitgang C en D van de L297. We zien duidelijk dat het ene signaal pas opkomt als het andere ophoudt. Hierdoor zal de rotor van de ene spoel rechtstreeks naar de andere 'stappen'.
Herinner je deze figuur uit het begin van het dossier:

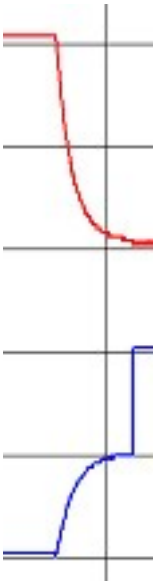


4.6.2 Half step drive

Half step drive kan men vergelijken met full step drive, met dit verschil dat men bij half step drive twee spoelen op hetzelfde moment zal aansturen. Dit houdt in dat men tussen elke stap die men bij full step drive neemt nog een extra stap tussenstapt bij half step drive. Hieronder volgen de oscillogrammen van half step drive

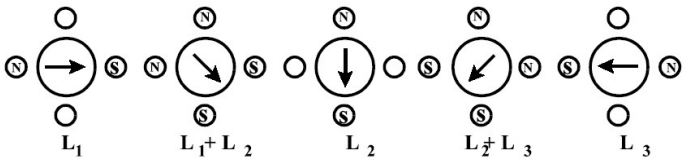






Op de figuur links staat een stuk van hetzelfde oscillogram uitvergroot. We kunnen duidelijk zien dat er een bepaalde periode is waar zowel de ene als de andere spoel bekrachtigd is. (Uitgang A zal als een hyperbool dalen van maximum tot 0, uitgang B zal hyperbolisch stijgen tot ½ van de uitgangsspanning, om dan direct te stijgen naar de maximum spanning). In deze tussenperiode zal de rotor zich dus positioneren tussen beide spoelen.

Ook hier denken we terug aan de figuur die we in het begin van het dossier gebruikt hebben.



5 De L298

5.1 Vooraf

We hebben nu al de VCO en de L297 besproken. De VCO diende als Input voor de L297, die de pulstrein omzet naar een sequentie die aan de uitgang van de L297 waargenomen kan worden. Zoals bij de meeste stuur-IC's is de IC niet zelf in staat om het nodige vermogen te ontwikkelen om de toepassing (in dit geval de fasen van de motor) te sturen. De stroom en/of spanning zijn daarvoor niet voldoende. Daarom schakelt men de uitgang van de L297 rechtstreeks aan een versterkend element (de L298). Op die manier kan de vermogen-IC L298 een veel groter vermogen over de spoelen van de motor plaatsen waardoor het koppel groter wordt.

5.2 Wat is de L298

De L298 is een monolithische chip met een ingebouwd circuit dat gemaakt is om een signaal te versterken. De L298 is gemaakt om onder andere inductieve belastingen te sturen zoals de spoelen van motoren, in ons geval een stappenmotor.

5.3 Algemene info

Zoals al eerder vermeld werd, is de L298 een element dat een signaal dat reeds bestaat, moet versterken. Daarom wordt de L298 in gebruikte circuits in combinatie met andere stuur-IC 's geplaatst. De L297 zorgt in het gebruikte circuit voor de gepaste pulsen op het gepaste moment en de L298 zet die signalen om naar een hoger voltage en een circuit waar meer vermogen kan verplaatst worden. Vandaar de naam vermogen-IC. Hieruit volgt het logische feit dat de L298 niet alleen kan gebruikt worden. Dat is ook niet het geval in onze schakeling.

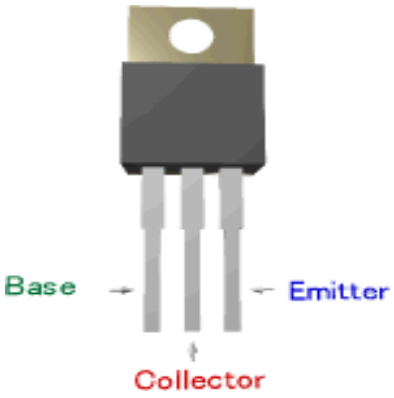
5.4 Voordelen

De L298 heeft verschillende voordelen. Eén ervan is dat de voeding-spanning van de IC mag oplopen tot 46 volt. Dat biedt veel voordelen bij het voeden van een zware motor. De motor in onze sturing heeft een spanning van 40 volt nodig om gestuurd te worden. Met deze IC is dit dus mogelijk. Een ander voordeel is de toegelaten DC stroom. Die mag oplopen tot 4 ampère. Al is deze stroom zeker niet klein, de stroom kan nog verhoogd worden door de 4 uitgangen aan de IC, twee aan twee in parallel te schakelen. Vanzelfsprekend kan dit niet toegepast worden op een stappenmotor omdat bij een toepassing met een stappenmotor minimum vier uitgangen nodig zijn. Nog een voordeel aan de L298 is dat er een beveiliging in zit tegen te hoge temperaturen. Niet alleen de beveiliging is voorzien. Op sommige praktische uitvoeringen is ook een boring voorzien om eventueel een koellichaam aan op te hangen. Handig is ook dat de L298 in drie verschillende praktische uitvoeringen te verkrijgen is:

de PowerSO 20 (L298P) uitvoering, een platte IC zonder boring om een koellichaam aan te bevestigen. De Multiwatt15 horizontaal (L298HN) met een boring en de Multiwatt15 verticaal (L298N), ook met een boring. In ons circuit wordt de L298N gebruikt omdat op deze het gemakkelijkst een koellichaam kan aangebracht worden.

5.5 Versterkers

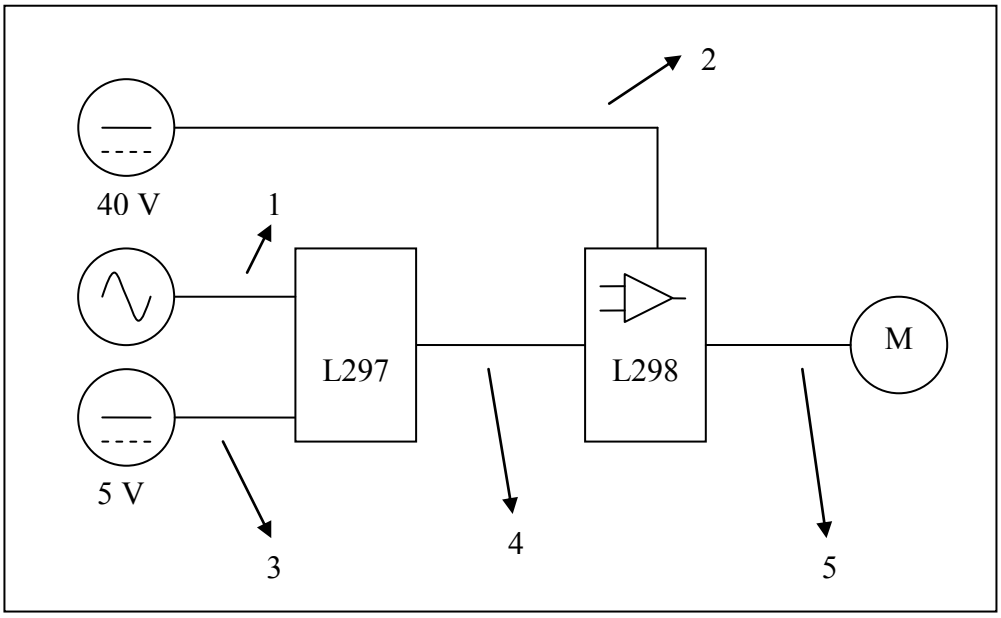
De L298 is in zekere zin een versterker. Daarom nog even een korte uitleg over wat een versterker eigenlijk is en wat een versterker precies doet.
Versterkers zijn noodzakelijke componenten die gebruikt worden om te zwakke signalen met een te kleine stroom en een te kleine spanning te vergroten tot een bruikbaar signaal. Het symbool van een versterker ziet er als volgt uit  . Om een versterker te kunnen gebruiken, hebben we altijd vier basisdelen nodig, namelijk: een voeding, een signaalbron, een versterker en een belasting. Eenmaal alle basisdelen aanwezig, kunnen we beginnen met de versterking.
In vele versterkers is de basis een transistor. Iedere gewone transistor heeft drie aansluitklemmen: de collector, de emitor en de basis.



Bij de L298 is dit ook het geval. Er zijn 8 transistoren geplaatst in het IC, twee ervan horen bij elke uitgang van het IC. Op iedere twee transitoren die bij een uitgang horen, zijn er poorten geplaatst die aangeven wanneer de transistor mag inschakelen of niet.
Omdat de transitoren niet altijd mogen ingeschakeld zijn, komt er nog een EN-poort voor te staan om er voor te zorgen dat het signaal enkel wordt versterkt wanneer er een bepaalde combinatie van signalen op de EN-poorten toekomt.

5.6 De L298 als versterker

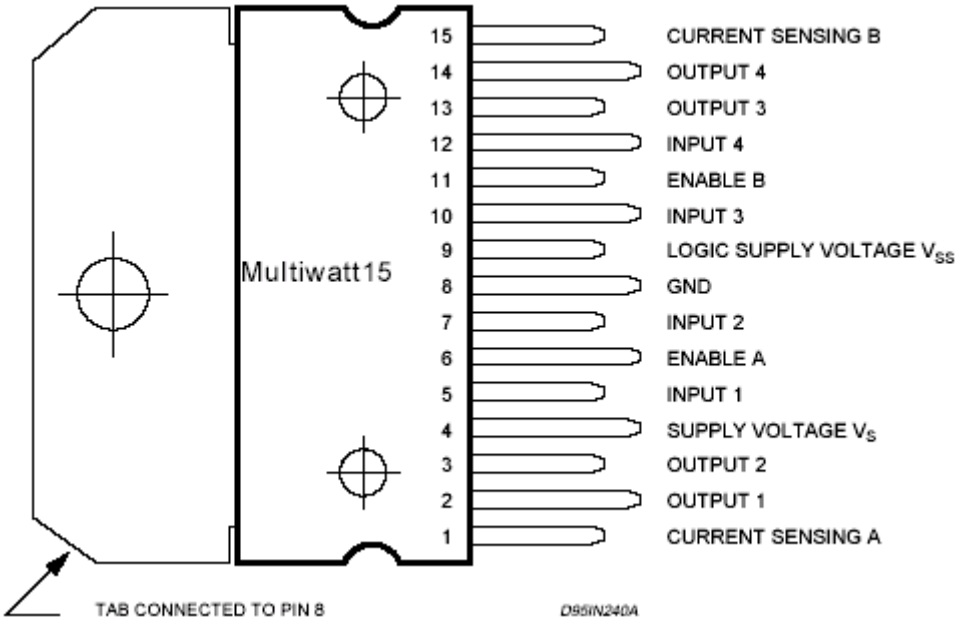
In onze schakeling is de versterker de L298. De signaalbron bestaat uit de signalen die uit de L297 komen. De belasting is de motor, en de voeding stelt de 40 volt voor, afkomstig van een bron. Als we alles samen in een schema plaatsen, bekomen we het volgende:



In dit schema kunnen we 4 als de signaalbron zien, M als belasting, 2 als voeding voor de versterker en L298 als de versterker.
Signaal één is het signaal, afkomstig van de functie generator, de pc of een andere pulsgenerator. Op twee hebben we de voedingspanning op de versterker. Drie is de vaste 5 volt die wordt omgezet naar signaal vier door de L297 die op zijn buurt de signaalbron voorstelt.

5.7 De klemmen van de L298

Zoals bij alles IC 's het geval is, staat iedere klem van een IC voor een totaal andere functie. De besprekingen van de klemmen van de L298 worden nu uitgelegd.



5 – 7 – 10 – 12 : Dit zijn de ingangsklemmen van het IC. De signalen die afkomstig zijn vanaf de L297 met de verschillende pulsen erop, worden op deze pinnen aangesloten. Inwendig worden de signalen versterkt en doorgestuurd naar de uitgangspinnen van de L298.

2 – 3 – 13 – 14 : Deze klemmen zijn de uitgangen van het IC die rechtstreeks met een stappenmotor worden aangesloten. Ze worden inwendig uit het IC afgetakt vanaf de plaats waar het signaal versterkt is door de twee transistors die bij die uitgangen passen. Deze signalen zijn logisch "1" als de inputsignalen die erbij passen logisch "1" zijn.

9 : De V_{ss} klem is aangebracht om het IC zelf te voeden zodat het IC zijn bewerkingen kan uitvoeren en het gegeven signaal kan versterken. Bij normale omstandigheden is dit signaal 5V maar het mag oplopen tot 7 V. Zonder dit voedingssignaal kunnen de EN-poorten niet functioneren zoals het hoort en kan het IC de signalen niet versterken.

4 : De aansluiting V_s is de voedingsaansluiting voor de IC-uitgangen die naar de motor gaan. Dit signaal wordt gebruikt om de 5 volt te versterken en die dan door te sturen naar de stappenmotor die ze omzet in een magnetisch veld en er vervolgens een beweging van maakt. De vaste (in ons geval) 40 V wordt aangesloten op de collector van de bovenste van de twee transistoren per uitgang. Op die manier worden de 5V-signalen vertrekt met dit 40 V signaal.

6 – 11 : Deze twee klemmen zorgen ervoor dat een spoel sneller kan ontladen bij de demagnetisering. Op beide poorten komt een signaal toe dat ervoor zorgt dat de spoelen van de stappenmotor geen stroom meer krijgen en dus veel vlugger ontladen waardoor ze veel vlugger kunnen omgekeerd worden gepolariseerd. Op die manier kan de stappenmotor een hoger toerental bereiken en een beter rendement. De signalen die op deze poorten aangesloten zijn, komen rechtstreeks van de L297.

1 – 15 : aan deze uitgangen zijn de signalen Sense A en Sense B aangesloten, maar er zijn ook twee weerstanden op aangesloten. De weerstanden zorgen ervoor dat de warmte op deze weerstanden wordt overgebracht. Als dit niet gebeurt, kunnen de transistoren in de L298 opwarmen, waardoor het werkpunt van de versterker zich verplaatst. Als dit gebeurt, zullen de transistoren zich niet meer op de juiste manier gedragen en zal het signaal verkeerd versterkt worden.

8 : De laatste pin is er om aangesloten te worden op een aarding. De 8 pin is inwendig rechtstreeks verbonden met het koelvlak om eventuele spanningen die er aanwezig zouden zijn, te kunnen verwerken.

Bijkomend: Aan de V_{ss} en de V_s klemmen moet er een niet inductieve capaciteit aangebracht worden en die klemmen moeten dan met de bron verbonden zijn. De capaciteit is meestal 100 nF.

6 Sturen met de computer

6.1 Vooraf

Op vandaag staat de computer in het middelpunt van de moderne wereld. Het is voor sommigen zelfs zover gekomen dat ze letterlijk niet meer kunnen overleven zonder de hulp van een computer. Op zich is dit niet erg, maar wij spelen op dit gebruik van de pc in en gebruiken de pc in onze geïntegreerde proef om een liftsysteem te besturen.

6.2 Poorten

Poorten op een computer zijn eigenlijk contactpluggen die meestal langs achter op de computer zitten om te communiceren met een bepaald apparaat of met een andere computer. Iedere poort heeft een aantal signaallijnen om gegevens uit te zenden. Iedere lijn kan een hoog "1" of een laag "0" signaal uitzenden of ontvangen.

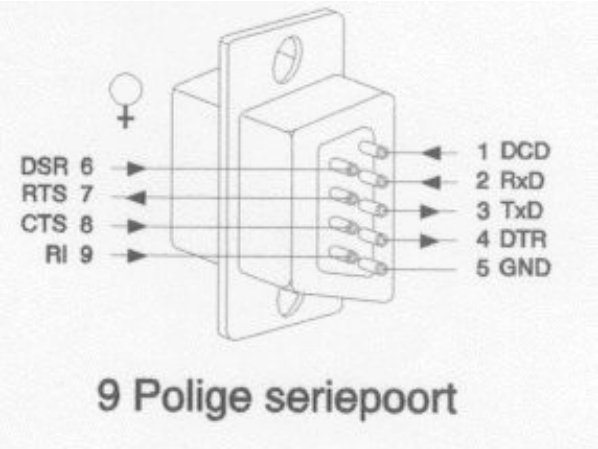
6.3 De seriële poort

6.3.1 Werking

De seriële poort heeft negen lijnen. De poort kan één bit per keer verzenden of ontvangen waardoor de snelheid relatief traag is ten opzichte van bijvoorbeeld de parallelle poort waar verschillende signalen per keer kunnen verzonden of ontvangen worden.

6.3.2 Gebruik

De seriële poort heeft acht lijnen, waarvan drie ingangen en vijf uitgangen. De poort heeft naast deze acht lijnen ook nog één nullijn. In de figuur zie je de "mannelijke" kant die zich in de computer bevindt en waar bijvoorbeeld een oude muis of een modem kan op aangesloten worden.



9 Polige seriepoort

6.3.3 Besluit

Deze poort is dus niet geschikt om gebruikt te worden in onze toepassing omdat de poort relatief traag is en omdat er niet genoeg in- en uitgangen zijn. Dit laatste kan verholpen worden door een schuifregister toe te passen op de poort, maar er moest een andere, meer simpele oplossing gevonden worden.

6.4 De parallelle poort

6.4.1 Werking

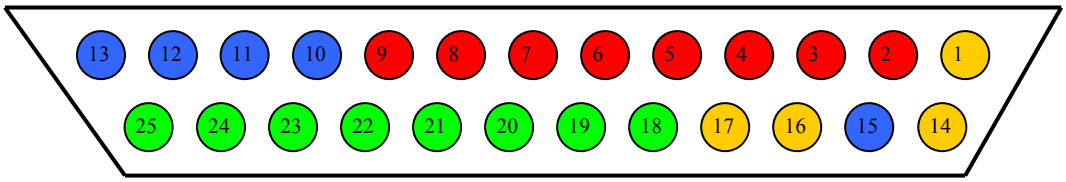
De parallelle poort is in feite een seriële poort waar al een schuifregister aan toegevoegd is. Dit maakt de poort veel makkelijker te besturen dan een seriële poort met een zelf toegevoegd schuifregister. De parallelle poort heeft vijfentwintig lijnen. Ze kan verschillende bits tegelijkertijd verzenden en ontvangen.

6.4.2 Gebruik

De printerpoort werd oorspronkelijk alleen voor de printer gebruikt, vandaar de naam "printerpoort", maar de dag van vandaag wordt hij voor scanners, fototoestellen en zelfs huissystemen gebruikt.
Natuurlijk is deze poort niet de nieuwste, maar daarom niet de minste. De parallelle poort heeft verschillende soorten lijnen: er zijn datalijnen, status-lijnen, controlelijnen en nullijnen. Iedere soort heeft zijn eigen commando om aangedreven te worden. Bij de datalijnen is dat bijvoorbeeld bij een standaard instelling "888". Let wel dit kan verschillen van computer tot computer.
De waarde hangt af van hoe de poort geïnstalleerd is op de pc. Als de poort geïnstalleerd is op "0x378", dan is het "888". Het nummer waarop de poort staat, is geschreven in het hexadecimale talstelsel en hoeft enkel omgezet te worden in het decimaal stelsel om tot het besturingscommando te komen.
De hexadecimale waarde kan men terugvinden in het hardware profiel van de computer.

6.4.3 De verschillende lijnen

Zoals reeds vermeld, zijn er op de printerpoort verschillende soorten lijnen aanwezig. Deze lijnen hebben elk hun eigen doel. Vele lijnen zijn specifiek ontworpen voor het gebruik van een printer omdat men niet dacht dat deze poort ooit voor iets anders zou moeten dienen.



Datalijnen: 

Er zijn in het totaal acht "data"-lijnen. Die worden gebruikt om uitgangen aan te sturen. Aangezien er acht van deze lijnen zijn en iedere lijn een hoog"1" of een laag"0" signaal kan uitsturen, zijn er "2^8 = 256" verschillende mogelijke combinaties te maken met deze lijnen.
Iedere lijn geeft een laag signaal "0" bij een "uit" toestand en een hoog signaal "1" bij een "aan" toestand.
Deze lijnen zijn standaard geïnstalleerd op "0x378".

Pin	Bit	Soort	PC-signaal	Pinsignaal
2	D0	uitgang	0/1	0/1
3	D1	uitgang	0/1	0/1
4	D2	uitgang	0/1	0/1
5	D3	uitgang	0/1	0/1
6	D4	uitgang	0/1	0/1
7	D5	uitgang	0/1	0/1
8	D6	uitgang	0/1	0/1
9	D7	uitgang	0/1	0/1

Statuslijnen: 

Deze lijnen zijn specifiek voor de printer ontworpen. Ze geven de status van de printer aan, bijvoorbeeld "Papier op" of "Bezig".
S0, S1 en S2 zijn niet zichtbaar in de connector.
Deze lijnen zijn standaard geïnstalleerd op "0x379".

Pin	Bit	Soort	PC-signaal	Pinsignaal
10	S6	Status	Data uitvoer 1/0	0/1
11	S7	Status	Bezig 1/0	0/1
12	S5	Status	Papier 1/0	0/1
13	S4	Status	Data invoer 1/0	1/0
15	S3	Status	Printer error 1/0	0/1
	S2	Status	Geen	Geen
	S1	Status	Geen	Geen
	S0	Status	EPP error 1/0	1/0

Controlelijnen:

Deze lijnen zijn gemaakt om controles uit te voeren.
C4, C5, C6 en C7 zijn niet zichtbaar op de connector.
De lijnen zijn standaard geïnstalleerd op "Ox37A".
De controlelijnen kunnen ook gebruikt worden om uitgangen te sturen, let wel één van de lijnen staat negatief opgesteld ten opzichte van de andere.
Als we telkens een ander commandosignaal ingeven naar de controlelijnen, merk je al vlug dat lijn C3 omgekeerd staat ten opzicht van alle drie andere lijnen.
Als we C3 telkens omdraaien, zien we dat de codes dan wel kloppen.

Commando signaal	C4	C3	C2	C1	PC signaal	C4	C3	C2	C1
0	1	0	1	1	0	1	1	1	1
1	1	0	1	0	1	1	1	1	0
2	1	0	0	1	2	1	1	0	1
3	1	0	0	0	3	1	1	0	0
4	1	1	1	1	4	1	0	1	1
5	1	1	1	0	5	1	0	1	0
6	1	1	0	1	6	1	0	0	1
7	1	1	0	0	7	1	0	0	0
8	0	0	1	1	8	0	1	1	1
9	0	0	1	0	9	0	1	1	0
10	0	0	0	1	10	0	1	0	1
11	0	0	0	0	11	0	1	0	0
12	0	1	1	1	12	0	0	1	1
13	0	1	1	0	13	0	0	1	0
14	0	1	0	1	14	0	0	0	1
15	0	1	0	0	15	0	0	0	0

Pin	Bit	Soort	Gebruik
1	C0	Contole	Controleert de data lijnen
14	C1	Contole	Voert het volgende printercommando aan
16	C2	Contole	Reset de buffer van de printer
17	C3	Contole	Bestuurt de data lijnen
	C4	Contole	Stuurt het "cut" commando van de printer
	C5	Contole	Bestuurd de richting van enkele poorten
	C6	Contole	Geen
	C7	Contole	Soms als C5 gebruikt

Nullijnen:

Deze lijnen zijn de nullijnen van de parallelle poort en worden gebruikt om het circuit met bijvoorbeeld de datapinnen aan te sluiten. Deze lijnen kunnen niet gestuurd worden.

6.4.4 De datalijnen

De datalijnen zijn in feite de enige lijnen die gemaakt zijn om andere apparaten of computers te besturen. Daarom zijn deze lijnen geschikt voor ons om ze te gebruiken. Dit gebeurt met het commando "888" dat afkomstig is van het hexadecimale getal "0x358".
Wanneer een hoog signaal "1" uit de pin van de printerpoort verwacht wordt, moet die pin ook aangedreven worden met een hoog signaal "1". Hetzelfde geldt voor een laag signaal "0".

Pin	Bit	Soort	PC-signaal	Pinsignaal
2	D0	uitgang	0/1	0/1
3	D1	uitgang	0/1	0/1
4	D2	uitgang	0/1	0/1
5	D3	uitgang	0/1	0/1
6	D4	uitgang	0/1	0/1
7	D5	uitgang	0/1	0/1
8	D6	uitgang	0/1	0/1
9	D7	uitgang	0/1	0/1

Aangezien er acht hoge of lage signalen zijn om aan te sturen, kun je deze lijnen aansturen met een binaire code van acht tekens lang. Dit betekent dat als je een signaal uit de printerpoort wilt krijgen dat gelijk staat aan "2" decimaal, je de lijnen moet aansturen met een "00000010" binaire code.
Er zijn 256 mogelijke combinaties maar aangezien het signaal "0" hier ook deel van uitmaakt, is de hoogst mogelijke decimale combinatie 255 en niet 256. Dit valt natuurlijk te omzeilen door bijvoorbeeld in een programma bij de decimale code een "1" af te trekken of op te tellen.

6.5 Toegang tot een poort

6.5.1 Toegang verschaffen

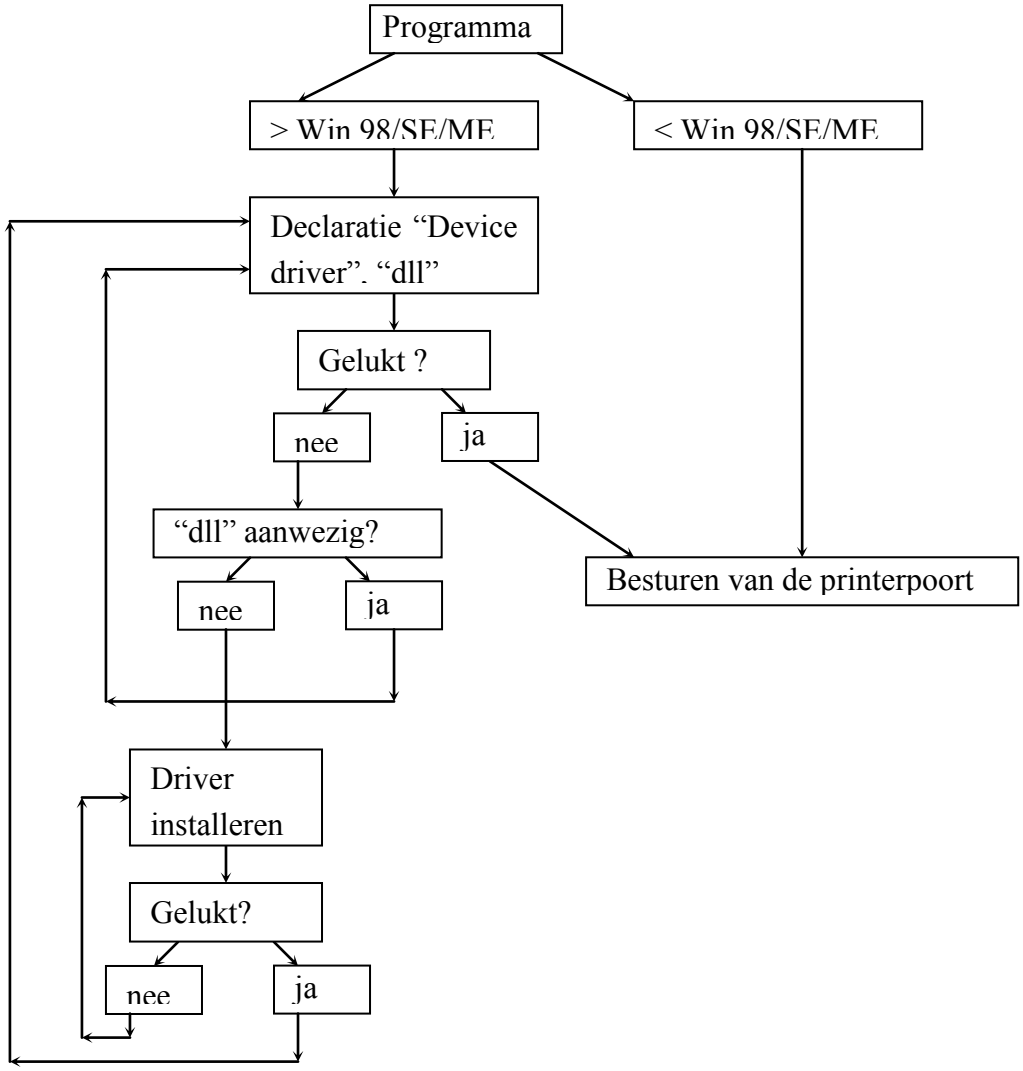
De toegang tot een poort van om het even welke computer is niet alledaags.
Om een poort te besturen:

- De commando's van de poort en eventueel van de onderverdeling in de poort moeten gekend zijn om deze te kunnen besturen. Om een auto te besturen heb je tenslotte ook de sleutel nodig.
- Het kan zijn dat de poorten beveiligd zijn. Dit hangt af van het besturingssysteem. Wanneer het OS (besturingssysteem) Windows is, dan moet de versie nieuwer zijn dan de Windows 98/SE/ME om geen last te hebben van beveiliging. Die beveiliging is door Microsoft ingevoerd in de nieuwere versies. In feite zit de beveiliging niet in de OS maar in de "386 processor" die in beveiligde modus staat.
- In geval van beveiliging moet deze omzeild worden. Dit kan door een "device driver". Er zijn verschillende soorten "device drivers", maar de meest eenvoudige en voor de hand liggende is een "dll", wat staat voor "Dynamic Link Library". Een "dll" is eigenlijk een bibliotheek met verschillende besturingen voor een programma.

6.5.2 Een "dll"

De "dll"-bestanden worden in twee verschillende soorten onderverdeeld. Er zijn deze die in "basic" gemaakt zijn (de zogenaamde "activeX") en de overige die in "C++", "delphi", enzovoort gemaakt zijn.
Om een "dll"-bestand te kunnen gebruiken, moet je als het ware eerst de gegevens oproepen voor je iets doet. Dit gebeurt door middel van declaratie. Iedere "dll" moet je weer anders declarenen. Ook iedere programmeercode heeft zijn eigen declaratiemethode. In het programma "Code" wordt het "dll" bestand "IO.dll" gebruikt. De programmeertaal van het programma is "basic" en het is gemaakt in het programma "Visual Basic". Maar het "dll"- bestand is echter gemaakt in "C++".

Natuurlijk kan er iets mislopen tijdens de hele procedure.
Daarvoor is er het volgende schema.



6.5.3 IO.dll in Visual Basic

Het "dll"-bestand "IO.dll" dat gebruikt is voor het programma "Code", moet net als alle andere "dll"-bestanden voor het gebruik gedeclareerd worden. Aangezien deze "dll" gemaakt is door iemand anders, bestaan de declaraties reeds en zijn deze terug te vinden. De declaratie is nodig om de "dll" als het ware te activeren voor het gebruik. Als dit niet gebeurt, blijft de "dll" inactief en kunnen we niets doen met de poorten.

De declaraties zijn:

```
Private Declare Sub PortOut Lib "IO.DLL" (ByVal Port As Integer, ByVal Data As Byte)
Private Declare Sub PortWordOut Lib "IO.DLL" (ByVal Port As Integer, ByVal Data As Integer)
Private Declare Sub PortDWordOut Lib "IO.DLL" (ByVal Port As Integer, ByVal Data As Long)
Private Declare Function PortIn Lib "IO.DLL" (ByVal Port As Integer) As Byte
Private Declare Function PortWordIn Lib "IO.DLL" (ByVal Port As Integer) As Integer
Private Declare Function PortDWordIn Lib "IO.DLL" (ByVal Port As Integer) As Long
Private Declare Sub SetPortBit Lib "IO.DLL" (ByVal Port As Integer, ByVal Bit As Byte)
Private Declare Sub ClrPortBit Lib "IO.DLL" (ByVal Port As Integer, ByVal Bit As Byte)
Private Declare Sub NotPortBit Lib "IO.DLL" (ByVal Port As Integer, ByVal Bit As Byte)
Private Declare Function GetPortBit Lib "IO.DLL" (ByVal Port As Integer, ByVal Bit As Byte) As Boolean
Private Declare Function RightPortShift Lib "IO.DLL" (ByVal Port As Integer, ByVal Val As Boolean) As Boolean
Private Declare Function LeftPortShift Lib "IO.DLL" (ByVal Port As Integer, ByVal Val As Boolean) As Boolean
Private Declare Function IsDriverInstalled Lib "IO.DLL" As Boolean
```

Zoals te zien is, zijn dit er heel wat, maar niet alles wordt gebruikt. Wij hebben alleen de eerste declaratie nodig: Private Declare Sub PortOut Lib "IO.DLL" (ByVal Port As Integer, ByVal Data As Byte).

De "PortOut" functie drijft de uitgangen van de poorten aan. Dat gebeurt met de gepaste code. Voor de acht uitgangen van de printerpoort is dat "888". De juiste waarde zorgt ervoor dat de juiste pinnen het gewenste signaal verschaffen.

Het gebruik van deze functie is:

```
PortOut 888, 0
```

"Portout" betekent dat er een uitgang bestuurd wordt van de computer. "888" geeft aan welke uitgangen het zijn: in dit geval, de uitgangen van de printerpoort. "0" staat voor het gewenste signaal. Dat signaal dient in decimaal ingevuld te worden. Wanneer bijvoorbeeld een binaire uitgang "11111111" nodig is, moet er "255" ingevuld worden, want zoals hoger vermeld, wordt er gestart bij nul en zijn er 256 combinaties mogelijk. De code wordt dan:

```
PortOut 888, 255
```

6.6 Een programma schrijven

6.6.1 Verschillende programmeertalen

Er zijn verschillende programmeertalen die gebruikt kunnen worden om een programma te schrijven. Voor de programma's die geschreven werden voor onze geïntegreerde proef is deze taal "basic" omdat dit een vrij krachtige programmeertaal is die relatief eenvoudig aan te leren is. Het programma dat gebruikt werd om de taal "basic" te programmeren, is "Visual Basic". Het programma geeft zoals de naam het zegt een soort "WYSIWYG" versie van uw programma weer op het display wat het zeer efficiënt en gemakkelijk in gebruik maakt.(1)
Dit is de code van het programma "code" . De regels waar een " ' " teken voor staat zijn info-regels. Overal waar een "Private Sub" of een "Sub" staat, begint een nieuw onderdeel van het programma. Elk onderdeel stop met "End Sub".

6.6.2 Het programma "Code"

Het programma 'code' is gemaakt om de datalijnen van de printerpoort te besturen. Dit gebeurt door middel van aanklikvakjes waarmee je iedere uitgang apart kunt besturen. Als je dit gedaan hebt, kun je de decimale code, de binaire code en de hexadecimale code van het signaal - dat naar de datalijnen gestuurd wordt - aflezen en eventueel gebruiken in een ander programma. Er is ook nog een deel om zelf een decimale code in te geven voor het geval je het resultaat ervan wilt weten.

6.6.3 De programmalijnen van "Code"

Code van het programma zelf:

```
'aanleggen van dll file en declareren van variabelen
Private Declare Sub PortOut Lib "IO.DLL" (ByVal Port As Integer, ByVal Data As Byte)
Private Declare Sub PortWordOut Lib "IO.DLL" (ByVal Port As Integer, ByVal Data As Integer)
Private Declare Sub PortDWordOut Lib "IO.DLL" (ByVal Port As Integer, ByVal Data As Long)
Private Declare Function PortIn Lib "IO.DLL" (ByVal Port As Integer) As Byte
Private Declare Function PortWordIn Lib "IO.DLL" (ByVal Port As Integer) As Integer
Private Declare Function PortDWordIn Lib "IO.DLL" (ByVal Port As Integer) As Long
Private Declare Sub SetPortBit Lib "IO.DLL" (ByVal Port As Integer, ByVal Bit As Byte)
Private Declare Sub ClrPortBit Lib "IO.DLL" (ByVal Port As Integer, ByVal Bit As Byte)
Private Declare Sub NotPortBit Lib "IO.DLL" (ByVal Port As Integer, ByVal Bit As Byte)
Private Declare Function GetPortBit Lib "IO.DLL" (ByVal Port As Integer, ByVal Bit As Byte) As Boolean
Private Declare Function RightPortShift Lib "IO.DLL" (ByVal Port As Integer, ByVal Val As Boolean) As Boolean
Private Declare Function LeftPortShift Lib "IO.DLL" (ByVal Port As Integer, ByVal Val As Boolean) As Boolean
Private Declare Function IsDriverInstalled Lib "IO.DLL" () As Boolean

Private Sub Check0_Click()
   'Controleren van de waarde, de gevonden waarde doorgeven aan een variabele
   'de juiste waarde bij de decimale code tellen en aftrekken
   'de decimale code omzetten naar hexadecimale code en die doorsturen naar het gepaste label
   'ook de cirkelvorm een andere kleur geven om aan te geven of de LED opgelicht is of niet
   'code geven aan LPT poort
   If Check0.Value = Checked Then
      lblBin0.Caption = 1
```

```
        lblDec.Caption = lblDec + 1
        lblHex.Caption = Hex(lblDec)
        shpLED0.FillColor = &HC000&
        PortOut 888, lblDec
    ElseIf Check0.Value = False Then
        lblBin0.Caption = 0
        lblDec.Caption = lblDec - 1
        lblHex.Caption = Hex(lblDec)
        shpLED0.FillColor = &HC0FFC0
        PortOut 888, lblDec
    End If
End Sub

Private Sub Check1_Click()
    'Controleren van de waarde, de gevonden waarde doorgeven aan een variabele
    'de juiste waarde bij de decimale code tellen en aftrekken
    'de decimale code omzetten naar hexadecimale code en die doorsturen naar het gepaste label
    'ook de cirkelvorm een andere kleur geven om aan te geven of de LED opgelicht is of niet
    'code geven aan LPT poort
    If Check1.Value = Checked Then
        lblBin1.Caption = 1
        lblDec.Caption = lblDec + 2
        lblHex.Caption = Hex(lblDec)
        shpLED1.FillColor = &HC000&
        PortOut 888, lblDec
    ElseIf Check1.Value = False Then
        lblBin1.Caption = 0
        lblDec.Caption = lblDec - 2
        lblHex.Caption = Hex(lblDec)
        shpLED1.FillColor = &HC0FFC0
        PortOut 888, lblDec
    End If
End Sub

Private Sub Check2_Click()
    'Controleren van de waarde, de gevonden waarde doorgeven aan een variabele
    'de juiste waarde bij de decimale code tellen en aftrekken
    'de decimale code omzetten naar hexadecimale code en die doorsturen naar het gepaste label
    'ook de cirkelvorm een andere kleur geven om aan te geven of de LED opgelicht is of niet
    'code geven aan LPT poort
    If Check2.Value = Checked Then
        lblBin2.Caption = 1
        lblDec.Caption = lblDec + 4
        lblHex.Caption = Hex(lblDec)
        shpLED2.FillColor = &HC000&
        PortOut 888, lblDec
    ElseIf Check2.Value = False Then
        lblBin2.Caption = 0
        lblDec.Caption = lblDec - 4
        lblHex.Caption = Hex(lblDec)
        shpLED2.FillColor = &HC0FFC0
        PortOut 888, lblDec
    End If
End Sub

Private Sub Check3_Click()
    'Controleren van de waarde, de gevonden waarde doorgeven aan een variabele
    'de juiste waarde bij de decimale code tellen en aftrekken
    'de decimale code omzetten naar hexadecimale code en die doorsturen naar het gepaste label
    'ook de cirkelvorm een andere kleur geven om aan te geven of de LED opgelicht is of niet
    'code geven aan LPT poort
```

```
    If Check3.Value = Checked Then
        lblBin3.Caption = 1
        lblDec.Caption = lblDec + 8
        lblHex.Caption = Hex(lblDec)
        shpLED3.FillColor = &HC000&
        PortOut 888, lblDec
    ElseIf Check3.Value = False Then
        lblBin3.Caption = 0
        lblDec.Caption = lblDec - 8
        lblHex.Caption = Hex(lblDec)
        shpLED3.FillColor = &HC0FFC0
        PortOut 888, lblDec
    End If
End Sub

Private Sub Check4_Click()
    'Controleren van de waarde, de gevonden waarde doorgeven aan een variabele
    'de juiste waarde bij de decimale code tellen en aftrekken
    'de decimale code omzetten naar hexadecimale code en die doorsturen naar het gepaste label
    'ook de cirkelvorm een andere kleur geven om aan te geven of de LED opgelicht is of niet
    'code geven aan LPT poort
        If Check4.Value = Checked Then
        lblBin4.Caption = 1
        lblDec.Caption = lblDec + 16
        lblHex.Caption = Hex(lblDec)
        shpLED4.FillColor = &HC000&
        PortOut 888, lblDec
    ElseIf Check4.Value = False Then
        lblBin4.Caption = 0
        lblDec.Caption = lblDec - 16
        lblHex.Caption = Hex(lblDec)
        shpLED4.FillColor = &HC0FFC0
        PortOut 888, lblDec
    End If
End Sub

Private Sub Check5_Click()
    'Controleren van de waarde, de gevonden waarde doorgeven aan een variabele
    'de juiste waarde bij de decimale code tellen en aftrekken
    'de decimale code omzetten naar hexadecimale code en die doorsturen naar het gepaste label
    'ook de cirkelvorm een andere kleur geven om aan te geven of de LED opgelicht is of niet
    'code geven aan LPT poort
    If Check5.Value = Checked Then
        lblBin5.Caption = 1
        lblDec.Caption = lblDec + 32
        lblHex.Caption = Hex(lblDec)
        shpLED5.FillColor = &HC000&
        PortOut 888, lblDec
    ElseIf Check5.Value = False Then
        lblBin5.Caption = 0
        lblDec.Caption = lblDec - 32
        lblHex.Caption = Hex(lblDec)
        shpLED5.FillColor = &HC0FFC0
        PortOut 888, lblDec
    End If
End Sub

Private Sub Check6_Click()
    'Controleren van de waarde, de gevonden waarde doorgeven aan een variabele
    'de juiste waarde bij de decimale code tellen en aftrekken
    'de decimale code omzetten naar hexadecimale code en die doorsturen naar het gepaste label
```

```
    'ook de cirkelvorm een andere kleur geven om aan te geven of de LED opgelicht is of niet
    'code geven aan LPT poort
    If Check6.Value = Checked Then
        lblBin6.Caption = 1
        lblDec.Caption = lblDec + 64
        lblHex.Caption = Hex(lblDec)
        shpLED6.FillColor = &HC000&
        PortOut 888, lblDec
    ElseIf Check6.Value = False Then
        lblBin6.Caption = 0
        lblDec.Caption = lblDec - 64
        lblHex.Caption = Hex(lblDec)
        shpLED6.FillColor = &HC0FFC0
        PortOut 888, lblDec
    End If
End Sub

Private Sub Check7_Click()
    'Controleren van de waarde, de gevonden waarde doorgeven aan een variabele
    'de juiste waarde bij de decimale code tellen en aftrekken
    'de decimale code omzetten naar hexadecimale code en die doorsturen naar het gepaste label
    'ook de cirkelvorm een andere kleur geven om aan te geven of de LED opgelicht is of niet
    'code geven aan LPT poort
    If Check7.Value = Checked Then
        lblBin7.Caption = 1
        lblDec.Caption = lblDec + 128
        lblHex.Caption = Hex(lblDec)
        shpLED7.FillColor = &HC000&
        PortOut 888, lblDec
    ElseIf Check7.Value = False Then
        lblBin7.Caption = 0
        lblDec.Caption = lblDec - 128
        lblHex.Caption = Hex(lblDec)
        shpLED7.FillColor = &HC0FFC0
        PortOut 888, lblDec
    End If
End Sub

Private Sub cmdTest_Click()
    'tweede testframe bovenhalen
    Code2.Show
    Code.Hide
End Sub

Private Sub Form_Activate()
    PortOut 888, lblDec
End Sub

Private Sub Form_Load()
    'code geven aan LPTpoort
    PortOut 888, lblDec
End Sub

Private Sub Form_Unload(Cancel As Integer)
    'code geven aan LPTpoort
    PortOut 888, 0
End Sub

Private Sub mnuAuteurs_Click()
    'een msgbox met de Auteur er op doorgeven
    MsgBox "Dit programma is gemaakt door Pieter Verhelle." + vbCrLf + "Copyright 2004 ©", vbOKOnly, "Scripter"
```

```
End Sub

Private Sub mnuHoe_Click()
   MsgBox "Voeg Het bijgevoegde 'IO.dll' bestand aan de 'C:\WINDOWS\system32' map toe om het programma volledig te
doen werken.", vbOKOnly, "Hoe"
End Sub

Private Sub mnuInfo_Click()
   'een msgbox met info doorgeven
   MsgBox "Dit programma is gemaakt voor de G.I.P. 2004 van het VTI van Torhout.", vbOKOnly, "Info"
End Sub

Private Sub mnuIO_Click()
   MsgBox "Meer info over IO.dll vind je op deze site http://www.geekhideout.com/iodll.shtml", vbOKOnly, "IO.dll"
End Sub

Private Sub mnuUit_Click()
   'alles op de nulwaarden zetten
   'code geven aan LPTpoort
   shpLED0.FillColor = &HC0FFC0
   shpLED1.FillColor = &HC0FFC0
   shpLED2.FillColor = &HC0FFC0
   shpLED3.FillColor = &HC0FFC0
   shpLED4.FillColor = &HC0FFC0
   shpLED5.FillColor = &HC0FFC0
   shpLED6.FillColor = &HC0FFC0
   shpLED7.FillColor = &HC0FFC0
   lblBin0.Caption = 0
   lblBin1.Caption = 0
   lblBin2.Caption = 0
   lblBin3.Caption = 0
   lblBin4.Caption = 0
   lblBin5.Caption = 0
   lblBin6.Caption = 0
   lblBin7.Caption = 0
   Check0.Value = Unchecked
   Check1.Value = Unchecked
   Check2.Value = Unchecked
   Check3.Value = Unchecked
   Check4.Value = Unchecked
   Check5.Value = Unchecked
   Check6.Value = Unchecked
   Check7.Value = Unchecked
   lblDec.Caption = 0
   lblHex.Caption = 0
   PortOut 888, lblDec
End Sub

Private Sub mnuAan_Click()
   'alles op de nulwaarden zetten
   'code geven aan LPTpoort
   shpLED0.FillColor = &HC000&
   shpLED1.FillColor = &HC000&
   shpLED2.FillColor = &HC000&
   shpLED3.FillColor = &HC000&
   shpLED4.FillColor = &HC000&
   shpLED5.FillColor = &HC000&
   shpLED6.FillColor = &HC000&
   shpLED7.FillColor = &HC000&
   lblBin0.Caption = 1
   lblBin1.Caption = 1
```

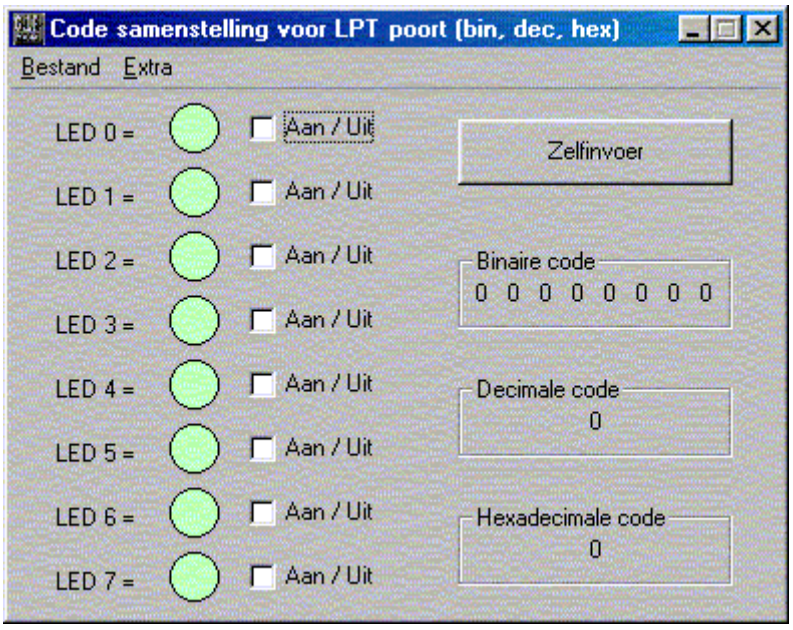
```
    lblBin2.Caption = 1
    lblBin3.Caption = 1
    lblBin4.Caption = 1
    lblBin5.Caption = 1
    lblBin6.Caption = 1
    lblBin7.Caption = 1
    Check0.Value = Checked
    Check1.Value = Checked
    Check2.Value = Checked
    Check3.Value = Checked
    Check4.Value = Checked
    Check5.Value = Checked
    Check6.Value = Checked
    Check7.Value = Checked
    lblDec.Caption = 255
    lblHex.Caption = Hex(lblDec)
    PortOut 888, lblDec
End Sub

Private Sub mnuSluiten_Click()
    'sluiten
    'code geven aan LPT poort
    PortOut 888, 0
    End
End Sub
```

Code van het tweede dialoogvenster voor de zelfinvoer:

```
Private Sub cmdTest_Click()
    'cijfer uit tekstvak halen en toepassen op de poort
    Dim codedec As Variant
    codedec = Val(txtDec2.Text)
    If codedec < 256 Then
        If codedec >= -1 Then
            PortOut 888, codedec
        Else
            MsgBox "de ingegeven waarde is te klein, geef een waarde in van 0 tot en met 255.", vbOKOnly + vbCritical, "Error!"
        End If
    Else
        MsgBox "de ingegeven waarde is te groot, geef een waarde in van 0 tot en met 255.", vbOKOnly + vbCritical, "Error!"
    End If
End Sub

Private Sub Form_Activate()
    'poort op 0 zetten
    codedec = Val(txtDec2.Text)
    If codedec < 256 Then
        If codedec >= -1 Then
            PortOut 888, codedec
        Else
            MsgBox "de ingegeven waarde is te klein, geef een waarde in van 0 tot en met 255.", vbOKOnly + vbCritical, "Error!"
        End If
    Else
        MsgBox "de ingegeven waarde is te groot, geef een waarde in van 0 tot en met 255.", vbOKOnly + vbCritical, "Error!"
    End If
End Sub

Private Sub Form_Load()
    'poort op 0 zetten
    PortOut 888, 0
```

```
End Sub
Private Sub txtDec2_Change()
End Sub
```

6.6.4 "Code"



6.7 Het programma "C lijnen"

Het programma "C lijnen" is bijna hetzelfde als het programma "code" met dat verschil dat er in "C lijnen" een register is toegevoegd omdat C3 negatief staat ten opzichte van de andere C lijnen. Telkens we een signaal willen doorsturen, kijkt het programma wat het commando signaal is en zoekt vervolgens in een register op wat de echte code moet zijn om dit signaal te verkrijgen op de printerpoort. Op deze manier hoeft de gebruiker van het programma geen rekening meer te houden met lijn C3 die negatief is opgesteld.

6.8 De programma lijnen van "C lijnen"

```
'Declareren van variabelen
Dim Vraag As Byte
Dim lblCode As Byte
Private Declare Sub PortOut Lib "IO.DLL" (ByVal Port As Integer, ByVal Data As Byte)
Private Declare Sub PortWordOut Lib "IO.DLL" (ByVal Port As Integer, ByVal Data As Integer)
Private Declare Sub PortDWordOut Lib "IO.DLL" (ByVal Port As Integer, ByVal Data As Long)
Private Declare Function PortIn Lib "IO.DLL" (ByVal Port As Integer) As Byte
Private Declare Function PortWordIn Lib "IO.DLL" (ByVal Port As Integer) As Integer
Private Declare Function PortDWordIn Lib "IO.DLL" (ByVal Port As Integer) As Long
Private Declare Sub SetPortBit Lib "IO.DLL" (ByVal Port As Integer, ByVal Bit As Byte)
Private Declare Sub ClrPortBit Lib "IO.DLL" (ByVal Port As Integer, ByVal Bit As Byte)
Private Declare Sub NotPortBit Lib "IO.DLL" (ByVal Port As Integer, ByVal Bit As Byte)
Private Declare Function GetPortBit Lib "IO.DLL" (ByVal Port As Integer, ByVal Bit As Byte) As Boolean
Private Declare Function RightPortShift Lib "IO.DLL" (ByVal Port As Integer, ByVal Val As Boolean) As Boolean
Private Declare Function LeftPortShift Lib "IO.DLL" (ByVal Port As Integer, ByVal Val As Boolean) As Boolean
Private Declare Function IsDriverInstalled Lib "IO.DLL" () As Boolean

Private Sub Check1_Click()
   If Check1.Value = Checked Then
      shp1.FillColor = &HC000&
      lblBin1.Caption = 1
      lblDec.Caption = lblDec + 1
      lblHex.Caption = Hex(lblDec)
      If lblDec = 0 Then
         lblCode = 11
      ElseIf lblDec = 1 Then
         lblCode = 10
      ElseIf lblDec = 2 Then
         lblCode = 9
      ElseIf lblDec = 3 Then
         lblCode = 8
      ElseIf lblDec = 4 Then
         lblCode = 15
      ElseIf lblDec = 5 Then
         lblCode = 14
      ElseIf lblDec = 6 Then
         lblCode = 13
      ElseIf lblDec = 7 Then
         lblCode = 12
      ElseIf lblDec = 8 Then
         lblCode = 3
      ElseIf lblDec = 9 Then
         lblCode = 2
      ElseIf lblDec = 10 Then
         lblCode = 1
      ElseIf lblDec = 11 Then
         lblCode = 0
      ElseIf lblDec = 12 Then
         lblCode = 7
      ElseIf lblDec = 13 Then
         lblCode = 6
      ElseIf lblDec = 14 Then
         lblCode = 5
      ElseIf lblDec = 15 Then
         lblCode = 4
      End If
      PortOut 890, lblCode
   ElseIf Check1.Value = Unchecked Then
```

```
      shp1.FillColor = &HC0FFC0
      lblBin1.Caption = 0
      lblDec.Caption = lblDec - 1
      lblHex.Caption = Hex(lblDec)
      If lblDec = 0 Then
         lblCode = 11
      ElseIf lblDec = 1 Then
         lblCode = 10
      ElseIf lblDec = 2 Then
         lblCode = 9
      ElseIf lblDec = 3 Then
         lblCode = 8
      ElseIf lblDec = 4 Then
         lblCode = 15
      ElseIf lblDec = 5 Then
         lblCode = 14
      ElseIf lblDec = 6 Then
         lblCode = 13
      ElseIf lblDec = 7 Then
         lblCode = 12
      ElseIf lblDec = 8 Then
         lblCode = 3
      ElseIf lblDec = 9 Then
         lblCode = 2
      ElseIf lblDec = 10 Then
         lblCode = 1
      ElseIf lblDec = 11 Then
         lblCode = 0
      ElseIf lblDec = 12 Then
         lblCode = 7
      ElseIf lblDec = 13 Then
         lblCode = 6
      ElseIf lblDec = 14 Then
         lblCode = 5
      ElseIf lblDec = 15 Then
         lblCode = 4
      End If
      PortOut 890, lblCode
   End If
End Sub
Private Sub Check2_Click()
   If Check2.Value = Checked Then
      shp2.FillColor = &HC000&
      lblBin2.Caption = 1
      lblDec.Caption = lblDec + 2
      lblHex.Caption = Hex(lblDec)
      If lblDec = 0 Then
         lblCode = 11
      ElseIf lblDec = 1 Then
         lblCode = 10
      ElseIf lblDec = 2 Then
         lblCode = 9
      ElseIf lblDec = 3 Then
         lblCode = 8
      ElseIf lblDec = 4 Then
         lblCode = 15
      ElseIf lblDec = 5 Then
         lblCode = 14
      ElseIf lblDec = 6 Then
         lblCode = 13
      ElseIf lblDec = 7 Then
```

```
            lblCode = 12
        ElseIf lblDec = 8 Then
            lblCode = 3
        ElseIf lblDec = 9 Then
            lblCode = 2
        ElseIf lblDec = 10 Then
            lblCode = 1
        ElseIf lblDec = 11 Then
            lblCode = 0
        ElseIf lblDec = 12 Then
            lblCode = 7
        ElseIf lblDec = 13 Then
            lblCode = 6
        ElseIf lblDec = 14 Then
            lblCode = 5
        ElseIf lblDec = 15 Then
            lblCode = 4
        End If
        PortOut 890, lblCode
    ElseIf Check2.Value = Unchecked Then
        shp2.FillColor = &HC0FFC0
        lblBin2.Caption = 0
        lblDec.Caption = lblDec - 2
        lblHex.Caption = Hex(lblDec)
        If lblDec = 0 Then
            lblCode = 11
        ElseIf lblDec = 1 Then
            lblCode = 10
        ElseIf lblDec = 2 Then
            lblCode = 9
        ElseIf lblDec = 3 Then
            lblCode = 8
        ElseIf lblDec = 4 Then
            lblCode = 15
        ElseIf lblDec = 5 Then
            lblCode = 14
        ElseIf lblDec = 6 Then
            lblCode = 13
        ElseIf lblDec = 7 Then
            lblCode = 12
        ElseIf lblDec = 8 Then
            lblCode = 3
        ElseIf lblDec = 9 Then
            lblCode = 2
        ElseIf lblDec = 10 Then
            lblCode = 1
        ElseIf lblDec = 11 Then
            lblCode = 0
        ElseIf lblDec = 12 Then
            lblCode = 7
        ElseIf lblDec = 13 Then
            lblCode = 6
        ElseIf lblDec = 14 Then
            lblCode = 5
        ElseIf lblDec = 15 Then
            lblCode = 4
        End If
        PortOut 890, lblCode
    End If
End Sub
Private Sub Check3_Click()
```

```
If Check3.Value = Checked Then
    shp3.FillColor = &HC000&
    lblBin3.Caption = 1
    lblDec.Caption = lblDec + 4
    lblHex.Caption = Hex(lblDec)
    If lblDec = 0 Then
        lblCode = 11
    ElseIf lblDec = 1 Then
        lblCode = 10
    ElseIf lblDec = 2 Then
        lblCode = 9
    ElseIf lblDec = 3 Then
        lblCode = 8
    ElseIf lblDec = 4 Then
        lblCode = 15
    ElseIf lblDec = 5 Then
        lblCode = 14
    ElseIf lblDec = 6 Then
        lblCode = 13
    ElseIf lblDec = 7 Then
        lblCode = 12
    ElseIf lblDec = 8 Then
        lblCode = 3
    ElseIf lblDec = 9 Then
        lblCode = 2
    ElseIf lblDec = 10 Then
        lblCode = 1
    ElseIf lblDec = 11 Then
        lblCode = 0
    ElseIf lblDec = 12 Then
        lblCode = 7
    ElseIf lblDec = 13 Then
        lblCode = 6
    ElseIf lblDec = 14 Then
        lblCode = 5
    ElseIf lblDec = 15 Then
        lblCode = 4
    End If
    PortOut 890, lblCode
ElseIf Check3.Value = Unchecked Then
    shp3.FillColor = &HC0FFC0
    lblBin3.Caption = 0
    lblDec.Caption = lblDec - 4
    lblHex.Caption = Hex(lblDec)
    If lblDec = 0 Then
        lblCode = 11
    ElseIf lblDec = 1 Then
        lblCode = 10
    ElseIf lblDec = 2 Then
        lblCode = 9
    ElseIf lblDec = 3 Then
        lblCode = 8
    ElseIf lblDec = 4 Then
        lblCode = 15
    ElseIf lblDec = 5 Then
        lblCode = 14
    ElseIf lblDec = 6 Then
        lblCode = 13
    ElseIf lblDec = 7 Then
        lblCode = 12
    ElseIf lblDec = 8 Then
```

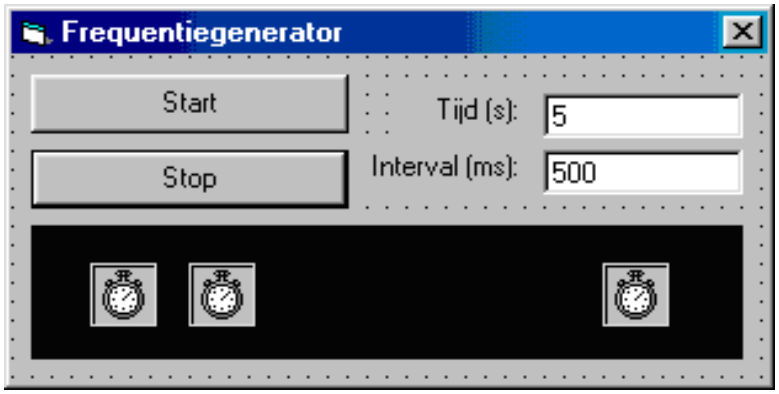
```
        lblCode = 3
      ElseIf lblDec = 9 Then
        lblCode = 2
      ElseIf lblDec = 10 Then
        lblCode = 1
      ElseIf lblDec = 11 Then
        lblCode = 0
      ElseIf lblDec = 12 Then
        lblCode = 7
      ElseIf lblDec = 13 Then
        lblCode = 6
      ElseIf lblDec = 14 Then
        lblCode = 5
      ElseIf lblDec = 15 Then
        lblCode = 4
      End If
      PortOut 890, lblCode
  End If
End Sub
Private Sub Check4_Click()
  If Check4.Value = Checked Then
    shp4.FillColor = &HC000&
    lblBin4.Caption = 1
    lblDec.Caption = lblDec + 8
    lblHex.Caption = Hex(lblDec)
    If lblDec = 0 Then
      lblCode = 11
    ElseIf lblDec = 1 Then
      lblCode = 10
    ElseIf lblDec = 2 Then
      lblCode = 9
    ElseIf lblDec = 3 Then
      lblCode = 8
    ElseIf lblDec = 4 Then
      lblCode = 15
    ElseIf lblDec = 5 Then
      lblCode = 14
    ElseIf lblDec = 6 Then
      lblCode = 13
    ElseIf lblDec = 7 Then
      lblCode = 12
    ElseIf lblDec = 8 Then
      lblCode = 3
    ElseIf lblDec = 9 Then
      lblCode = 2
    ElseIf lblDec = 10 Then
      lblCode = 1
    ElseIf lblDec = 11 Then
      lblCode = 0
    ElseIf lblDec = 12 Then
      lblCode = 7
    ElseIf lblDec = 13 Then
      lblCode = 6
    ElseIf lblDec = 14 Then
      lblCode = 5
    ElseIf lblDec = 15 Then
      lblCode = 4
    End If
    PortOut 890, lblCode
  ElseIf Check4.Value = Unchecked Then
    shp4.FillColor = &HC0FFC0
```

```
        lblBin4.Caption = 0
        lblDec.Caption = lblDec - 8
        lblHex.Caption = Hex(lblDec)
        If lblDec = 0 Then
            lblCode = 11
        ElseIf lblDec = 1 Then
            lblCode = 10
        ElseIf lblDec = 2 Then
            lblCode = 9
        ElseIf lblDec = 3 Then
            lblCode = 8
        ElseIf lblDec = 4 Then
            lblCode = 15
        ElseIf lblDec = 5 Then
            lblCode = 14
        ElseIf lblDec = 6 Then
            lblCode = 13
        ElseIf lblDec = 7 Then
            lblCode = 12
        ElseIf lblDec = 8 Then
            lblCode = 3
        ElseIf lblDec = 9 Then
            lblCode = 2
        ElseIf lblDec = 10 Then
            lblCode = 1
        ElseIf lblDec = 11 Then
            lblCode = 0
        ElseIf lblDec = 12 Then
            lblCode = 7
        ElseIf lblDec = 13 Then
            lblCode = 6
        ElseIf lblDec = 14 Then
            lblCode = 5
        ElseIf lblDec = 15 Then
            lblCode = 4
        End If
        PortOut 890, lblCode
    End If
End Sub

Private Sub cmdStop_Click()
    Vraag = MsgBox("Weet u zeker dat u wilt stoppen?", vbYesNo, "Stopen?!")
    If Vraag = vbYes Then
        PortOut 890, 11
        End
    Else
    End If
End Sub

Private Sub cmdZelf_Click()
    'wisselen van kader
    Clijnen.Hide
    ClijnenZ.Show
End Sub

Private Sub Form_Activate()
    'poortsignaal juist zetten
        If lblDec = 0 Then
            lblCode = 11
        ElseIf lblDec = 1 Then
            lblCode = 10
```

```
    ElseIf lblDec = 2 Then
       lblCode = 9
    ElseIf lblDec = 3 Then
       lblCode = 8
    ElseIf lblDec = 4 Then
       lblCode = 15
    ElseIf lblDec = 5 Then
       lblCode = 14
    ElseIf lblDec = 6 Then
       lblCode = 13
    ElseIf lblDec = 7 Then
       lblCode = 12
    ElseIf lblDec = 8 Then
       lblCode = 3
    ElseIf lblDec = 9 Then
       lblCode = 2
    ElseIf lblDec = 10 Then
       lblCode = 1
    ElseIf lblDec = 11 Then
       lblCode = 0
    ElseIf lblDec = 12 Then
       lblCode = 7
    ElseIf lblDec = 13 Then
       lblCode = 6
    ElseIf lblDec = 14 Then
       lblCode = 5
    ElseIf lblDec = 15 Then
       lblCode = 4
    End If
    PortOut 890, lblCode
End Sub

Private Sub Form_Load()
   'poortsignaal effectief op nul zetten
   PortOut 890, 11
End Sub
```

6.9 Het programma "FG"

FG staat voor frequentie generator. Het logische gevolg is dus dat "FG" gemaakt is om een frequentie uit de printerpoort te laten komen. Deze frequentie is nodig om aangesloten te worden om de CLK-ingang van onze besturing. Het programma heeft onder andere drie timers. Twee timers die afwisselend werken en de frequentie maken, en één timer die de duur van de frequentie maakt.



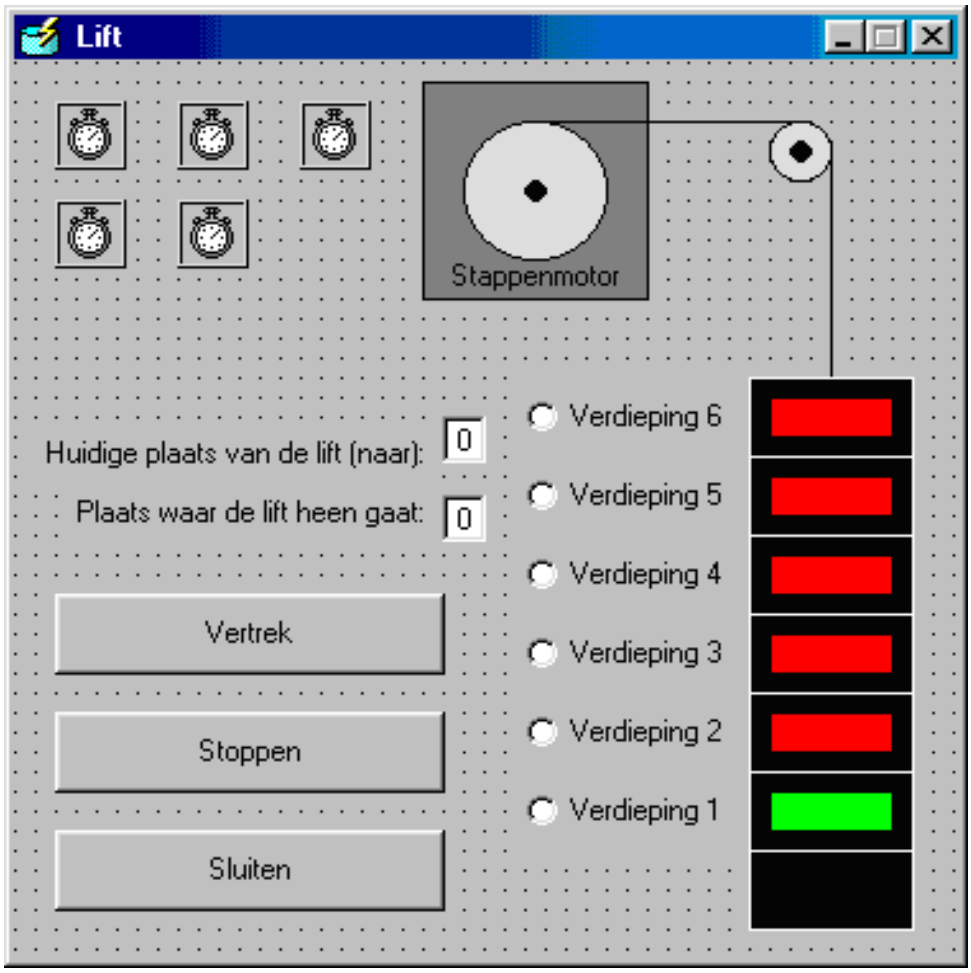
6.10 Programma lijnen van "FG"

```
Private Declare Sub PortOut Lib "IO.DLL" (ByVal Port As Integer, ByVal Data As Byte)
Private Declare Sub PortWordOut Lib "IO.DLL" (ByVal Port As Integer, ByVal Data As Integer)
Private Declare Sub PortDWordOut Lib "IO.DLL" (ByVal Port As Integer, ByVal Data As Long)
Private Declare Function PortIn Lib "IO.DLL" (ByVal Port As Integer) As Byte
Private Declare Function PortWordIn Lib "IO.DLL" (ByVal Port As Integer) As Integer
Private Declare Function PortDWordIn Lib "IO.DLL" (ByVal Port As Integer) As Long
Private Declare Sub SetPortBit Lib "IO.DLL" (ByVal Port As Integer, ByVal Bit As Byte)
Private Declare Sub ClrPortBit Lib "IO.DLL" (ByVal Port As Integer, ByVal Bit As Byte)
Private Declare Sub NotPortBit Lib "IO.DLL" (ByVal Port As Integer, ByVal Bit As Byte)
Private Declare Function GetPortBit Lib "IO.DLL" (ByVal Port As Integer, ByVal Bit As Byte) As Boolean
Private Declare Function RightPortShift Lib "IO.DLL" (ByVal Port As Integer, ByVal Val As Boolean) As Boolean
Private Declare Function LeftPortShift Lib "IO.DLL" (ByVal Port As Integer, ByVal Val As Boolean) As Boolean
Private Declare Function IsDriverInstalled Lib "IO.DLL" () As Boolean

Private Sub cmdStart_Click()
   Dim interval As Integer
   Dim sec
   Dim Aantal As Integer
   sec = Val(txtSec.Text)
   'Aantal = sec / interval
   interval = Val(txtInterval.Text)
   sec = sec * 1000
   If sec = 0 Then
      MsgBox "Voer een tijd in", vbOKOnly & vbError, "Tijd!?"
   ElseIf interval = 0 Then
      MsgBox "Voer een interval in", vbOKOnly & vbError, "Tijd!?"
   Else
      TmrTest.interval = interval
      TmrTest2.interval = interval
      tmrDuur.interval = sec
      'Debug.Print "Einde cmd toets"
      'Debug.Print TmrTest.interval
      'Debug.Print TmrTest2.interval
      tmrDuur.Enabled = True
      TmrTest.Enabled = True
      'Debug.Print tmrDuur.interval
   End If
End Sub

Private Sub cmdStop_Click()
   TmrTest.Enabled = False
   TmrTest2.Enabled = False
   PortOut 888, 0
   ShpTest.FillColor = &H0&
End Sub

Private Sub Form_Load()
   PortOut 888, 0
End Sub

Private Sub tmrDuur_Timer()
   tmrDuur.Enabled = False
End Sub

Private Sub TmrTest_Timer()
   If tmrDuur = False Then
      TmrTest.Enabled = False
      TmrTest2.Enabled = False
```

```
        PortOut 888, 0
        ShpTest.FillColor = &H0&
    Else
        PortOut 888, 255
        ShpTest.FillColor = &HFF&
        TmrTest.Enabled = False
        TmrTest2.Enabled = True
    End If
End Sub

Private Sub TmrTest2_Timer()
If tmrDuur = False Then
        TmrTest.Enabled = False
        TmrTest2.Enabled = False
        PortOut 888, 0
        ShpTest.FillColor = &H0&
    Else
        PortOut 888, 240
        ShpTest.FillColor = &H0&
        TmrTest2.Enabled = False
        TmrTest.Enabled = True
    End If
End Sub
```

6.11 Het programma "Lift"

Zoals de naam het al zegt, is dit programma gemaakt om ons liftsysteem te besturen. Het programma is een samensmelting van alle andere reeds gemaakt programma's waardoor het programma op dezelfde manier is opgebouwd. Er zijn vijf timers aanwezig, twee om naar boven te bewegen, twee om naar beneden te gaan en één om de duur te bepalen.



6.12 Programma lijnen van "Lift"

```
Private Declare Sub PortOut Lib "IO.DLL" (ByVal Port As Integer, ByVal Data As Byte)
Private Declare Sub PortWordOut Lib "IO.DLL" (ByVal Port As Integer, ByVal Data As Integer)
Private Declare Sub PortDWordOut Lib "IO.DLL" (ByVal Port As Integer, ByVal Data As Long)
Private Declare Function PortIn Lib "IO.DLL" (ByVal Port As Integer) As Byte
Private Declare Function PortWordIn Lib "IO.DLL" (ByVal Port As Integer) As Integer
Private Declare Function PortDWordIn Lib "IO.DLL" (ByVal Port As Integer) As Long
Private Declare Sub SetPortBit Lib "IO.DLL" (ByVal Port As Integer, ByVal Bit As Byte)
Private Declare Sub ClrPortBit Lib "IO.DLL" (ByVal Port As Integer, ByVal Bit As Byte)
Private Declare Sub NotPortBit Lib "IO.DLL" (ByVal Port As Integer, ByVal Bit As Byte)
Private Declare Function GetPortBit Lib "IO.DLL" (ByVal Port As Integer, ByVal Bit As Byte) As Boolean
Private Declare Function RightPortShift Lib "IO.DLL" (ByVal Port As Integer, ByVal Val As Boolean) As Boolean
Private Declare Function LeftPortShift Lib "IO.DLL" (ByVal Port As Integer, ByVal Val As Boolean) As Boolean
```

```
Private Declare Function IsDriverInstalled Lib "IO.DLL" () As Boolean
Private Sub cmdEnd_Click()
   PortOut 888, 0
   End
End Sub

Private Sub cmdGo_Click()
Dim Aan As Single
Dim Uit As Single
'De waarde van de plaats van de lift ophalen
If shp1.FillColor = &HFF00& Then
   txtPlaats.Text = "1"
ElseIf shp2.FillColor = &HFF00& Then
   txtPlaats.Text = "2"
ElseIf shp3.FillColor = &HFF00& Then
   txtPlaats.Text = "3"
ElseIf shp4.FillColor = &HFF00& Then
   txtPlaats.Text = "4"
ElseIf shp5.FillColor = &HFF00& Then
   txtPlaats.Text = "5"
ElseIf shp6.FillColor = &HFF00& Then
   txtPlaats.Text = "6"
End If
'de waarde van de plaats van de lift en de aangegeven plaats ophalen
Dim Plaats As Single
Dim Heen As Single
Heen = txtHeen.Text
Plaats = txtPlaats.Text
Dim Verhouding As Single
Verhouding = Plaats - Heen
'de richting van de beweging wordt gecontroleerd
If Verhouding = 0 Then
   MsgBox "Er is geen verplaatsing ingegeven," & vbCrLf & "probeer het opnieuw op met een andere verdieping",
vbInformation + vbOKOnly, "Verplaatsingsfout"
ElseIf Verhouding > 0 Then
   sec = 5000 * Verhouding
   tmrDuur.Interval = sec
   tmrDuur.Enabled = True
   tmrTest.Enabled = True
ElseIf Verhouding < 0 Then
   Verhouding = Verhouding * (-1)
   sec = 5000 * Verhouding
   tmrDuur.Interval = sec
   tmrDuur.Enabled = True
   tmrTest3.Enabled = True
End If
If Heen = 1 Then
   shp1.FillColor = &HFF00&
   shp2.FillColor = &HFF&
   shp3.FillColor = &HFF&
   shp4.FillColor = &HFF&
   shp5.FillColor = &HFF&
   shp6.FillColor = &HFF&
ElseIf Heen = 2 Then
   shp2.FillColor = &HFF00&
   shp1.FillColor = &HFF&
   shp3.FillColor = &HFF&
   shp4.FillColor = &HFF&
   shp5.FillColor = &HFF&
   shp6.FillColor = &HFF&
ElseIf Heen = 3 Then
```

```
    shp3.FillColor = &HFF00&
    shp2.FillColor = &HFF&
    shp1.FillColor = &HFF&
    shp4.FillColor = &HFF&
    shp5.FillColor = &HFF&
    shp6.FillColor = &HFF&
ElseIf Heen = 4 Then
    shp4.FillColor = &HFF00&
    shp2.FillColor = &HFF&
    shp3.FillColor = &HFF&
    shp1.FillColor = &HFF&
    shp5.FillColor = &HFF&
    shp6.FillColor = &HFF&
ElseIf Heen = 5 Then
    shp5.FillColor = &HFF00&
    shp2.FillColor = &HFF&
    shp3.FillColor = &HFF&
    shp1.FillColor = &HFF&
    shp4.FillColor = &HFF&
    shp6.FillColor = &HFF&
ElseIf Heen = 6 Then
    shp6.FillColor = &HFF00&
    shp2.FillColor = &HFF&
    shp3.FillColor = &HFF&
    shp1.FillColor = &HFF&
    shp5.FillColor = &HFF&
    shp4.FillColor = &HFF&
End If
If shp1.FillColor = &HFF00& Then
    txtPlaats.Text = "1"
ElseIf shp2.FillColor = &HFF00& Then
    txtPlaats.Text = "2"
ElseIf shp3.FillColor = &HFF00& Then
    txtPlaats.Text = "3"
ElseIf shp4.FillColor = &HFF00& Then
    txtPlaats.Text = "4"
ElseIf shp5.FillColor = &HFF00& Then
    txtPlaats.Text = "5"
ElseIf shp6.FillColor = &HFF00& Then
    txtPlaats.Text = "6"
End If
cmdGo.Enabled = False
End Sub

Private Sub cmdHalt_Click()
    tmrTest.Enabled = False
    tmrTest2.Enabled = False
    tmrTest3.Enabled = False
    tmrTest4.Enabled = False
    PortOut 888, 0
End Sub

Private Sub Form_Load()
    tmrTest.Enabled = False
    tmrTest2.Enabled = False
    tmrTest3.Enabled = False
    tmrTest4.Enabled = False
    PortOut 888, 0
End Sub
```

```
Private Sub opt1_Click()
'De juiste waarde doorgeven aan het tekstveld
   txtHeen.Text = 1
End Sub

Private Sub opt2_Click()
'De juiste waarde doorgeven aan het tekstveld
   txtHeen.Text = 2
End Sub

Private Sub opt3_Click()
'De juiste waarde doorgeven aan het tekstveld
   txtHeen.Text = 3
End Sub

Private Sub opt4_Click()
'De juiste waarde doorgeven aan het tekstveld
   txtHeen.Text = 4
End Sub

Private Sub opt5_Click()
'De juiste waarde doorgeven aan het tekstveld
   txtHeen.Text = 5
End Sub

Private Sub opt6_Click()
'De juiste waarde doorgeven aan het tekstveld
   txtHeen.Text = 6
End Sub

Private Sub tmrDuur_Timer()
   tmrDuur.Enabled = False
   cmdGo.Enabled = True
End Sub

Private Sub tmrTest_Timer()
   If tmrDuur = False Then
      PortOut 888, 0
      tmrTest.Enabled = False
      tmrTest2.Enabled = False
   Else
      PortOut 888, 0
      tmrTest.Enabled = False
      tmrTest2.Enabled = True
   End If
End Sub

Private Sub tmrTest2_Timer()
   If tmrDuur = False Then
      PortOut 888, 0
      tmrTest.Enabled = False
      tmrTest2.Enabled = False
   Else
      PortOut 888, 1
      tmrTest2.Enabled = False
      tmrTest.Enabled = True
   End If
End Sub

Private Sub tmrTest3_Timer()
```

```
    If tmrDuur = False Then
        PortOut 888, 0
        tmrTest3.Enabled = False
        tmrTest4.Enabled = False
    Else
        PortOut 888, 2
        tmrTest3.Enabled = False
        tmrTest4.Enabled = True
    End If
End Sub

Private Sub tmrTest4_Timer()
    If tmrDuur = False Then
        PortOut 888, 0
        tmrTest3.Enabled = False
        tmrTest4.Enabled = False
    Else
        PortOut 888, 3
        tmrTest4.Enabled = False
        tmrTest3.Enabled = True
    End If
End Sub
```

6.13 Besluit

Er zijn verschillende poorten op een computer die op verschillende manieren kunnen aangedreven worden. Dat kan ook in verschillende programmeertalen gebeuren. Al deze opties maken het mogelijk om flexibel te werk te gaan. Het is alleen een kwestie van een manier te kiezen die goed toepasbaar is en zo efficiënt mogelijk werkt.

De gebruiksvriendelijkheid van deze taal is mede de reden dat basic uiteindelijk meer succes heeft gehad dan andere, complexere talen die aanvankelijk meer aanzien genoten bij de eerste professionele programmeurs. Denken we maar aan Pascal en Fortram.

7 Bedrijfsbezoeken

7.1 PIH en KUKA (15 september 2004)

On Wednesday , September the 15th, some senior students of the VTI Torhout, visited the PIH in Kortrijk for the first time in this school year. Several technical schools working on different robot projects were invited by the PIH of Kortrijk to compare their work at the end of the school year. As we arrived at the PIH, we were given a friendly welcome by the coordinator of the PIH project that brought all the schools together. For this first presentation about robots in the PIH, the Kuka company was invited to give us some information about the matter. Kuka is one of the world leaders in the production of robots for the industrial world. They make large scale robots for the car industry, the glass industry and so on. Kuka's PR man gave us a very entertaining and quite interesting introduction about what robots are and what they do in the world as we know it today. We were shown several short movies and a couple of photos of Japanese toy robots and also a few of the ones they developed themselves. After the show was over, some people working in the PIH showed us several classes and robots they had in stock. This ended our most interesting visit.

7.2 PIH Robot en Positioneren (maart 2005)

The second in the line of four excursions was at the PIH itself. At the previous visit we could look at robots, and see how they moved and worked, but today we could make our own robot-program. We first needed some extra info about positioning. They explained the basics of positioning, like explained in the file. Also the principle of feedback (Principle that is used often with DC-motors, to have an exact location) was explained. A big disadvantage of this system with feedback is that there can be a sort of resonance. Imagine overshoot (the rotor is making an change of anglethat is to big). The feedback will now make the rotor turn back to the right position. But, if you again have overshoot the rotor will turn back, and even go to far. Again the feedback will adjust the rotor's position, and overshoot will be the result. We've seen this phenomenon at the PIH. A second type of motor that can be used is a stepper motor. This is the motor that is mainly discussed in our GIP, so when we get the info at the PIH we logically already knew quite a bit of how they worked. Still, this was the excellent opportunity to see how these

motor's were used in actual machines. (So far we've only seen our stepper motor with our test circuit). So far was the first part of our excursion. The second part had something to do with the KUKA-robot, at the PIH. With the help of a control-panel we were able to make the robot move. This was fairly easy, so we could do that ourselves. In the past people had to write hours and hours on a single program, but nowadays this could all be don in les that one hour. We took only 15 minutes to write our own simple program. Out of all the excursions we did I thing this one was the most interesting.

7.3 Picanol Diksmuide (20 oktober 2004)

Wednesday afternoon, 13h15. With the four of us standing on the parking space of the VTI, Ready to start our trip to Ieper for a company visit to the well-known company Picanol. Like most of you will probably know Picanol designs and fabricates weaving machines, and exports them to all the countries in the world. This makes Picanol on absolute market leader in the weaving industry.

The company Picanol was found in 1986, by a Belgian INDUSTIREEL Charles Steverlynck. (It then carried the name 'Weefautomaten Picanol NV'). But, it was Jaimé Picanol that designed a whole new weaving machine that became a big success. From that day on Picanol only climbed up. In 1966 Picanol was listed on the stock market of Belgium. In 1994 Picanol expanded to China, where they opened a new vestal, good for 60% of the company. During our visit we were absolutely surprised by all the things we saw. Maybe that's because we don't do that much excursions with the VTI itself…

Like every big company this one had a big hall with mill machines , a big hall with lathes. But Picanol also had it's own foundry, and a magazine that would make mister Verhelle jealous. They also had a KUKA industrial robot (the main reason for our visit). At that time we'd already saw quite some KUKA robots, but never one that was actually working in a factory. The job for that robot at Picanol was to place heavy pieces of work into a machine (a machine to HARDEN those pieces of work). If you would want this to do with regular employees, you'd need two of them to safely carry the piece of work. It would take them much longer that the few seconds the robot needed to replace the piece of work. This together make's it really interesting to invest in industrial robots.

7.4 Stäubli (20 april 2005)

On Wednesday , April the 20th , some senior students of the VTI Torhout, visited Stäubli. The Stäubli-company is specialised in developing robots and specific connectors for industrial use. During this schoolyear several technical schools worked on different robot projects. They were invited by the PIH of Kortrijk to do so and make a presentation of their individual projects at the end of the year. The PIH arranged a visit to Stäubli. The Stäubli company is one of the leading firms on the matter and that is why a visit was very interesting. Stäubli is specialized in finding complete solutions adapted to the specific needs of the industry. The Stäubli PR-man showed the various robot types that Stäubli developed for such cause, such as the cleanroom-robot used in factories where dust must be omitted at all costs. These robots are completely air-tight so that they don't interfere with the production of monolith chips. The so-called plastic robots are adapted to the plastic industry, and other trades as farming industry, health industry and food industry also find the kind of robots they need at Stäubli's. At the end of the presentation at Stäubli's in Bissegem we moved on to Boucherie-company in Izegem. That company that was founded at the end of the 50's at the time that Izegem still was the Mekka of shoes and brushes industry. Boucherie specialized in developing machinery for the brushes industry, a highly specialized narrow part of the industrial market. The visit to Boucherie nv was very enlightening as well, but unfortunately it was very short in duration. It didn't make it less worthwhile though. After this visit we went home to continue our work for school.

8 Slot

We zijn nu gekomen aan het einde van onze geïntegreerde proef.
De stappenmotoren hebben heel wat van hun geheimen prijs gegeven.
We hebben ook de sturing met de L297 en de L298 grondig bestudeerd.
We moesten helaas afzien van ons oorspronkelijke idee om de robotarm met de computer te sturen. Dit omdat het sturen van zes stappenmotoren met een parallelle poort niet rechtstreeks mogelijk is. Er zijn onvoldoende uitgangen aan de poort om alle motoren apart te sturen zoals het hoort. Het zou hebben geleid tot een uiterst complex sturingssysteem.
De precieze uitwerking daarvan was echt wel te hoog gegrepen. Daarom hebben we ons toegelegd op een sturing waarin de L297 en de L298 werden gebruikt. Het heeft ons ondermeer geleerd dat het zeker geen lachertje is om een bestaande schakeling te hergebruiken of te herstellen.
Ook zijn we tot de conclusie gekomen dat één stappenmotor sturen met een computer al een vrij moeilijke opdracht is. Nadat we ons geruime tijd verdiepten in de programmeertaal Basic en na het schrijven van verscheidene testprogramma's, slaagden we erin een programma uit te werken dat complementair functioneert met de hoger genoemde sturing die gebruik maakt van de L297 en L298. Dit programma zorgt voor de nodige pulsen en signalen die dan weer omgezet worden door de sturing om zodoende de motor aan te drijven. Uiteraard lukte dit niet allemaal op één dag en moesten we verscheidene poorten uittesten vooraleer we met zekerheid konden kiezen voor de parallelle poort als zijnde de beste oplossing voor dit specifieke probleem.

9 Bronvermelding

HEISERMAN, D.L., *How to design an build your own custom robot*, 3de druk, TAB BOOKS .Inc, Blue Ridge Summit, Pa., 17214, 1940, 462 pagina's.

PHILIPS, *Stepping motors and associated electronics (C17)*, 1984, 136 pagina's.

PHILIPS, *Synchronous motors and gearboxes (Part 6)* , 1984, 95 pagina's.

PHILIPS, *Components and materials (Part 6) Electric motors and accessoires*, mei 1981, 203 pagina's.

PHILIPS,*Direct current motors (part 18)*, 1984, 110 pagina's.

POLLEFLIET, J., *Elektronische vermogencontrole*, 4de druk, Uitgeverij Nevelland, Nevele, 1995, 776 pagina's.

KATZENMEIER, H., *Robots zelf construeren*, 1ste druk, Uitgeverij Segment B.V., 2004, 232 pagina's.

PAUWELS, G., *Ontwikkelen van een automatische boormachine voor PCB's*, 2001-2002, 75 pagina's, (Thesis PIH ivm stappenmotoren en positioneren).

CLAERHOUT, L., DEKELVER, V., DE SCHEPPER, F., LIBBRECHT, J., MAESEN, I., *Serie elektrotechniek Elektriciteit*, 2de druk, Wolters Plantyn, Deurne, 2001, 226 pagina's.

Departement industriële wetenschappen en technologie, *cursus visual basic*, C. Daniels, 122 pagina's

Three Laws Of Robotics, internet, Wikipedia, (http://en.wikipedia.org/wiki/Three_Laws_of_Robotics).

Bellis, M., *Robots*, internet, About, (http://inventors.about.com/library/inventors/blrobots.htm).

Jones, D., *Control of stepping motors*, internet, uiowa.edu, 2004-09-2, (http://www.cs.uiowa.edu/~jones/step/).

Circuit Online, *algemene informatie,* internet, (http://www.circuitsonline.net/forum).

STMicroelectronics, *L298 datasheet,* internet, ST
(http://www.st.com/stonline/books/pdf/docs/1773.pdf).

STMicroelectronics, *L297 datasheet,* internet, ST
(http://www.alltronics.com/download/1734.pdf).

Nidhy, *uitleg bij de uitgangen van de L298*, 2004, Internet, Indian institute of technologie Kanpur,(http://www.iitk.ac.in/eclub/Circuit/l298.htm).

IBM , *informatie over de uitgangen op de parallelle poort*, Internet, Zhahai Stewart
(http://www.timgoldstein.com/CNC/ParallelPortPrimer.htm).

Geek hideout, *IO.dll,* internet, Fred,
(http://www.geekhideout.com/whoami.shtml).

10 Bijlagen