Motorsturing

Het is niet de bedoeling van onze programmeerwerkzaamheden om alleen maar een of meer lampjes te laten branden om ze daarna weer uit doen gaan. We willen een racerobot bouwen, een racerobot, die in staat is een uitgezet traject binnen de kortste keren af te leggen. De robot is voorzien van een PIC-processor.
In het vorige hebben we bereikt, dat we weten welke hulpmiddelen nodig zijn om de PIC-processor te programmeren en we leerden hoe we het programma schrijven en overdragen naar het geheugen van de processor. Die voert dan uit, wat we in ons programma hebben beschreven.
Het is nu tijd om de motorsturing ter hand te nemen. Daarna volgt nog de sensors te onderzoeken. Tenslotte wordt voor een aantal soorten robots met verschillende aandrijvingen door middel van flow-sheets het geheel geïntegreerd.
Drie soorten motors komen voor ons in aanmerking voor de racerobot. Elk heeft specifieke eigenschappen en vraagt eigen soorten programmatuur. Verder is het zo, dat de vorm van de aandrijving van de robot met het stuurmechanisme eigen programma's vereisen.
We beschrijven eerst de drie soorten motors:

- de servomotor
- de stappenmotor met vermogenstoevoer
- de gelijkstroommotor met H-brug.

De servomotor

Servomotors heb je in twee soorten, waarvan de ene eenvoudig in de andere kan worden omgebouwd. In de CCFZ-beschrijving van het programmeren van de BoE-bot staat beschreven hoe dat moet. De as van de ene soort kan ca. 360 ° draaien, bij de tweede soort kan de as doordraaien. De eerste soort wordt in de hobbywereld gebruikt voor stuurmechanismen en bij brandstofmotors voor de gastoevoer. De tweede soort kan worden gebruikt voor de voortbeweging, zoals bij de BoE-bot gebeurt. Deze soort zullen we hier aansturen met de PIC-processor.
Een servomotor bestaat in wezen uit een gelijkstroommotor met een elektronische schakeling, die de draairichting en de snelheid regelt.
Een servomotor heeft drie aansluitingen. Twee voor de energietoevoer (de plus en de min), en een voor de commandopulsen van de processor. Deze laatste (logische) aansluiting vraagt 0 of 5 volt voor de 0 en de 1. Deze beschrijving is getest op de servomotor van Parallax (continuous rotation, www.parallax.com).
De aansturing hiervan gebeurt op basis van pulsbreedte. Een pulsbreedte van 1,5 ms (milliseconde) doet de motor stilstaan. Wordt de pulsbreedte verminderd tot 1,3 ms, dan draait de motor de ene richting uit in toenemende snelheid, wordt de pulsbreedte vermeerderd tot 1,7 ms, dan neemt de draaisnelheid steeds toe in de andere richting. Bij 1,3 en 1,7 ms worden de maximale snelheden bereikt. Het aansturen van een servomotor wordt hiermee dus een studie met de stopwatch van MPLAB.

```
; ------------------------------------------------------------------
;
;            Parallax servomotor aansturen
;
;                  juni 2008
;
;                  Hans Dorst
;
; ------------------------------------------------------------------
            list              p=16F84
            errorlevel -302
            __config   3FFD

porta       equ               0x05
portb       equ               0x06
trisa       equ               0x85
trisb       equ               0x86

rp0         equ               5
status      equ               0x03
temp        equ               0x34
loop1       equ               0x30
loop2       equ               0x31

            bsf               status,rp0          ; poorta alles ingang
            movlw             0xff
            movwf             trisa
            bcf               status,rp0

            bsf               status,rp0
            movlw             0x00                ; portb alles uitgang
            movwf             trisb
            bcf               status,rp0

            bsf               portb,0x00
            movlw             portb

; ------------------------------------------------------------------
puls                                              ; routine, die puls van 650 ms stuurt naar rb0

            bsf               portb,0x01
            movlw             portb
            call              pulsduur
            bsf               portb,0x00
            movlw             portb
            call              wachten
            goto              puls

; ------------------------------------------------------------------
pulsduur                                          ; hier 1,5 ms instellen

            movlw             0x36
            movwf             loop2               ; 20h in geheugenplaats 31h invoeren

nestlus                                           ; te verwachten is dat met een lus tot 256 onvoldoende is
            movlw             0xff                ; geneste lus, ook tot 20h
            movwf             loop1
            goto              aftellen

aftellen
            decfsz            loop1,f             ; verminder waarde in loop1-geheugen met 1
            goto              aftellen            ; wordt overgeslagen bij loop1 = 0
            decfsz            loop2,f             ; verminder waarde in loop2-geheugen met 1
            goto              nestlus
            return

; ------------------------------------------------------------------
wachten
            movlw             0xff
            movwf             loop2               ; ffh in geheugenplaats 31h invoeren
```

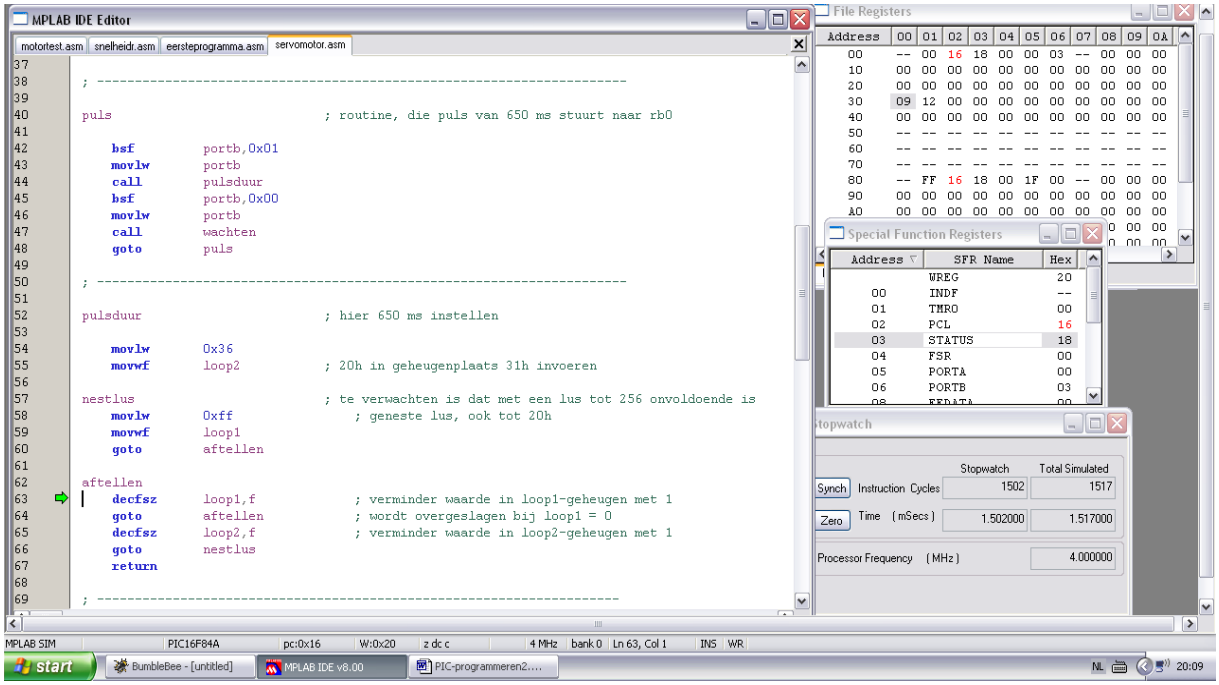
```
nestluswachten                                  ; te verwachten is dat met een lus tot 256 onvoldoende is
        movlw       0x36                        ; geneste lus, ook tot 72h
        movwf       loop1
        goto        aftellenwachten

aftellenwachten
        decfsz      loop1,f                     ; verminder waarde in loop1-geheugen met 1
        goto        aftellenwachten             ; wordt overgeslagen bij loop1 = 0
        decfsz      loop2,f                     ; verminder waarde in loop2-geheugen met 1
        goto        nestluswachten

        return

        end
```

Hierboven de listing met twee keer twee geneste lussen. Een keer voor de pulsbreedte, en een keer voor de pauze. Beide lussen maken gebruik van de variabelen “loop1” en “loop2”. Bij pulsduur heb ik voorlopig de waarden 36 voor loop2 en ff voor loop 1 ingevuld. Met de stopwatch ga ik nu na welke de standen zijn voor geheugenplaatsen 30 en 31 bij het bereiken van de 1,5 ms. Het onderzoek zal zich hierop richten.



Toets F6 indrukken om de stopwatch aan het begin van het programma te plaatsen. (zal telkens moeten gebeuren.) En nu herhaaldelijk klikken op F7 om de cursor, en dus het begin van het programma te plaatsen op “aftellen”, “decfsz”. Bij mij staat de stopwatch dan op 20 µs. Als je goed kijkt, dan zie je meteen, dat elke twee klikken op F7 de stopwatch 3 µs verder is gekomen. Controleer dit alles. Je kunt nooit weten of er bij jouw schakeling iets anders is dan bij mij.
Hieronder dus het deel van de listing, waar we nu naar kijken.

```
pulsduur                                    ; hier 1,3,  1,5  en 1,7 ms instellen
        movlw       0x01
        movwf       portb                   ; maak poort0: 1
        movlw       0x12                    ; voor 1,5 ms
        movwf       loop2                   ; waarde in geheugenplaats 31h invoeren

nestlus
        movlw       0x05                    ; geneste lus; ook voor 1,5 ms
        movwf       loop1

        goto        aftellen

aftellen
        decfsz      loop1,f                 ; verminder waarde in loop1-geheugen met 1
        goto        aftellen                ; wordt overgeslagen bij inhoud loop1 = 0
        decfsz      loop2,f                 ; verminder waarde in loop2-geheugen met 1
        goto        nestlus                 ; wordt overgeslagen bij inhoud loop2 = 0
        goto        wachten
```

Om het uitzoeken niet te veel te vertragen heb ik in deze listing de waarden voor loop1 en voor loop2 toch maar wat kleiner gemaakt. Klik weer op F6 en telkens op F7 en kijk wat er gebeurt in de schermen van de stopwatch, het “Special Function Register” en de “File registers”. De veranderingen in de waarden door de laatst genomen stap worden steeds rood gekleurd. Dat vergemakkelijkt het waarnemen. Onderscheid maak ik tussen de grote lus en de kleine lus. De kleine lus wordt telkens opnieuw doorlopen bij elke vermindering van de waarde voor de grote lus. Na zorgvuldig kijken en telkens opnieuw checken kom ik tot de volgende resultaten:

Ten eerste:
Elke afname voor loop1 met een, vraagt 2 stappen (2 x F7) en kost 3 µs.

Vervolgens:
Om opnieuw een volgende serie te gaan aftellen in de kleine lus zijn er 5 stappen (5 x F7) in de grote lus nodig, die 14 µs duren.

Tenslotte:
Als je bijvoorbeeld van 5 naar 0 laat aftellen, dan doet het mechanisme er 4 stappen over.

Dat laatste vraagt een toelichting. Heb je de aftelwaarde 5 ingevuld en je start, dan gaat de teller meteen naar vier. Kijk maar! Als je bij nul bent, dan wordt de lus meteen verlaten. Dat betekent, dat de kleine lus een keertje te weinig wordt doorlopen: 4 keer. Bij de berekende waarde moet 1 (een) worden bijgeteld.

We kiezen ervoor om de kleine lus op 100 µs in te stellen. De grote lus moet dan respectievelijk 13, 15 of 17 keer worden doorlopen om 1,3, 1,5 of 1,7 ms te bereiken. Het kan zijn, dat op deze manier niet precies de waarden kunnen worden bereikt. Met 5 % grens nauwkeurigheid nemen we genoegen.

Telkens als we van de kleine lus naar de grote gaan besteden we 14 µs. Dat wil zeggen, dat de kleine lus zelf 100 – 14 = 86 µs moet worden. Het aantal stappen (aantal keer F7) is hiermee 2/3 x 86 = 58. Dat komt overeen met 0x3a.

Wat hieraan niet goed is weet ik niet, maar het resultaat had 1f moeten zijn. Daarbij is de kleine lus met de 14 µs van de grote lus totaal 99 µs. Graag mailtje naar hando@planet.nl.

Deze kleine lus moet voor 1,3, 1,5 en 1,7 ms-puls 13, 15 en 17 keer doorlopen worden. In de grote lus moeten daarvoor respectievelijk 14, 16 en 18 worden gekozen (eentje meer dus, het middelste resultaat van de beschouwing.) Dat worden resp. 0x0d, 0x0f en 0x11. En zo werkt het ook.

Voor de pauze tussen de pulsen doen we hetzelfde. Kleine lus 110 µs. Met de grote lus 200 keer doorlopen. Daarom grote lus op 201 = 0xd9. Het programma wordt zodoende voor 1,3 ms pulsen en 20 ms pauze:

```
; ------------------------------------------------------------------
;
;               Parallax servomotor aansturen
;
;                       juni 2008
;                       Hans Dorst
;
; ------------------------------------------------------------------
            list                p=16F84
            errorlevel -302
            __config   3FFD

porta       equ                 0x05
portb       equ                 0x06
trisa       equ                 0x85
trisb       equ                 0x86

rp0         equ                 5
status      equ                 0x03
temp        equ                 0x34
loop1       equ                 0x30
loop2       equ                 0x31

            bsf                 status,rp0              ; poorta alles ingang
            movlw               0xff
            movwf               trisa
            bcf                 status,rp0

            bsf                 status,rp0
            movlw               0x00                    ; portb alles uitgang
            movwf               trisb
            bcf                 status,rp0

            bsf                 portb,0x00
            movlw               portb
; ------------------------------------------------------------------
puls                                                    ; routine, die puls van 650 ms stuurt naar rb0
            bsf                 portb,0x01
            movlw               portb
            call                pulsduur
            bsf                 portb,0x00
            movlw               portb
            call                wachten
            goto                puls
```

```
; -------------------------------------------------------------------
pulsduur                                        ; hier 1,5 ms instellen
        movlw       0x11
        movwf       loop2                       ; 20h in geheugenplaats 31h invoeren

nestlus                                         ; te verwachten is dat met een lus tot 256 onvoldoende is
        movlw       0x1f                        ; geneste lus, ook tot 20h
        movwf       loop1
        goto        aftellen

aftellen
        decfsz      loop1,f                     ; verminder waarde in loop1-geheugen met 1
        goto        aftellen                    ; wordt overgeslagen bij loop1 = 0
        decfsz      loop2,f                     ; verminder waarde in loop2-geheugen met 1
        goto        nestlus
        return
; -------------------------------------------------------------------
wachten
        movlw       0xd9
        movwf       loop2                       ; 20h in geheugenplaats 31h invoeren
nestluswachten                                  ; te verwachten is dat met een lus tot 256 onvoldoende is
        movlw       0x1f                        ; geneste lus, ook tot 20h
        movwf       loop1
        goto        aftellenwachten

aftellenwachten
        decfsz      loop1,f                     ; verminder waarde in loop1-geheugen met 1
        goto        aftellenwachten             ; wordt overgeslagen bij loop1 = 0
        decfsz      loop2,f                     ; verminder waarde in loop2-geheugen met 1
        goto        nestluswachten

        return

        end
```

Hiermee kunnen we de servomotortjes van Parallax met de maximale snelheid vooruit, stilstaan en maximale snelheid achteruit aansturen. Tussenliggende pulsbreedten resulteren in tussenliggende snelheiden. De pauze kan daarvoor hetzelfde blijven.

De stappenmotor

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!! In bewerking !!!!!!!!!!!!!!!!!!!!!!!!!!!!

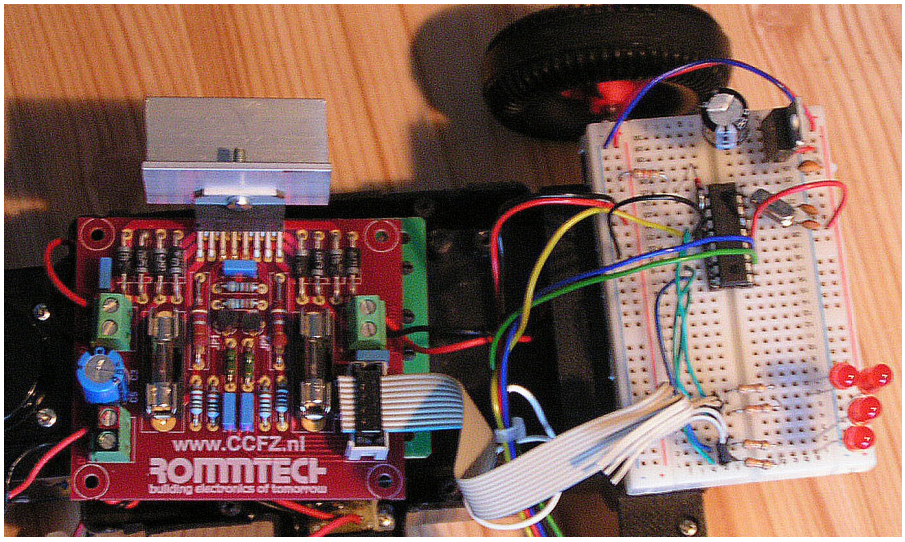
De gelijkstroommotor met H-brug

De H-brug is beschreven in een apart document, dat opgehaald kan worden van onze site www.ccfz.nl. Deze H-brug is geschikt om met de PIC samen te werken. Van ons Breadboard kan de 5 volt voedingsspanning worden gebruikt om de H-brug van 5 volt logicaspanning te voorzien. Op de punten 9 en 10 van de H-brug dient de aansluiting plaats te vinden. Op de punten 1 en 2 wordt de draairichting van motor 1 en van motor 2 bestuurd en op aansluitingen 3 en 4 wordt voor motor 1 en voor motor 2 de energietoevoer voor de motors geregeld door middel van pulsbreedtemodulatie. Dit is allemaal beschreven in het document CCFZ Motorsturing, dat over de H-brug gaat.

Van de PIC-processor gebruiken we de volgende aansluitingen:

- poort b0 voor motor 1, de draairichting
- poort b1 voor motor 1, de snelheidsregeling
- poort b2 voor motor 2, de draairichting
- poort b3 voor motor 2, de snelheidsregeling.

De LED-lampjes laten we aangesloten, zodat we kunnen zien wat er gebeurt. Naast deze LED-lampjes sluiten we motor 1 en motor 2 aan. Deze zijn van de robot, die in ontwikkeling is. De vermogenstoevoer is meteen ook aangesloten (linksonder rode en zwarte draden), zodat het effect van ons programma op de aandrijving kan worden vastgesteld. De verbinding naar de Whisp 648 hebben we natuurlijk ook nog nodig.
Het geheel ziet er dan als volgt uit:



Het gewenste programma is:

```
; ------------------------------------------------------------------------
;
;                       Eerste experiment
;
;                        Voorjaar 2008
;
;                        Hans Dorst
;
; ------------------------------------------------------------------------
;
; Hierboven, net als deze tekst, dus een inleidende tekst achter de ; (punt-komma),
; welke tekst geen deel uitmaakt van het programma, maar als toelichting is bedoeld.
;
; We gaan een programma schrijven, waarmee twee lampjes, aangesloten op RA1 en RB4
; om en om elke halve seconde aan en uitgaan.
;
            list                p=16F84
            errorlevel -302
            __config  3FFD

porta               equ     0x05
portb               equ     0x06
trisa               equ     0x85
trisb               equ     0x86

rp0                 equ     5
status              equ     0x03
```

```
                bsf     status,rp0      ; porta alles ingang
                movlw   0xff
                movwf   trisa
                bcf     status,rp0

                bsf     status,rp0      ; portb alles uitgang
                movlw   0x00
                movwf   trisb
                bcf     status,rp0


                goto    rijden


rijden          movlw   0x03            ; plaats de waarde in geheugen w; rechtdoor
                movwf   portb           ; kopieer w naar portb

                goto    rijden          ; opnieuw

                end
```

Het grootste deel van dit programma zal bekend moeten zijn uit de eerste hoofdstukken. Maar met de drie laatste regels voor : “end” geven we aan wat we willen: zowel op poort b0 als op poort b1 willen we een 1 (een) geven . Beide poorten hoog wil zeggen: H-brug laat stroom continue door en draairichting is kant “1” op. Dat is eenvoudig.
Bij het volgende programma proberen we de pulsbreedtemodulatie te laten functioneren om de motor minder energie te kunnen laten opnemen (telkens even de stroomtoevoer via de H-brug onderbreken) en daardoor langzamer te laten draaien. Dat bereiden we voor met onderstaand programma.

```
; ------------------------------------------------------------------------------
;
;                       Eerste experiment
;
;                        Voorjaar 2008
;
;                        Hans Dorst
;
; ------------------------------------------------------------------------------
;
; Hierboven, net als deze tekst, dus een inleidende tekst achter de ; (punt-komma),
; welke tekst geen deel uitmaakt van het programma, maar als toelichting is bedoeld.
;
; We gaan een programma schrijven, waarmee twee lampjes, aangesloten op RA1 en RB4
; om en om elke halve seconde aan en uitgaan.
;
                list            p=16F84
                errorlevel -302
                __config  3FFD

porta           equ             0x05
portb           equ             0x06
trisa           equ             0x85
trisb           equ             0x86

rp0             equ             5
status          equ             0x03
temp            equ             0x34

                bsf     status,rp0      ; porta alles ingang
                movlw   0xff
                movwf   trisa
                bcf     status,rp0
                bsf     status,rp0      ; portb alles uitgang
```

```
        movlw   0x00
        movwf   trisb
        bcf     status,rp0
        goto    rijden


; -----------------------------------------------------------------

rijden  movlw   0x03                    ; plaats de waarde in geheugen w; rechtdoor
        movwf   portb           ; kopieer w naar portb
        movlw   0x02            ; nu zetten we de poort even op 0
        movwf   portb           ; energietoevoer even stoppen

        goto    rijden

        end
```

Achter “rijden” zie je de aansturing van de poorten b0 en b1, zoals in het vorige programma. Laat de motor draaien, zodat je kunt controleren wat er gebeurt. Meet de spanning over de motor als die draait. Het programma maakt een voortdurende lus. De motors blijven derhalve draaien totdat de spanning wordt weggehaald. Ga dat even na. Bij mij is de spanning over de motor 6,95 volt bij een voedingsspanning van 9,0 volt.
We voegen nu toe de volgende twee regels, waarin poort b1 laag, nul of 0 wordt. Daarna gaan we weer naar aftellen om opnieuw te beginnen. Dit is het resultaat voor de eerste twee regels na “aftellen”:

```
rijden          movlw   0x03
                movwf   portb
                movlw   0x02
                movwf   portb
```

Probeer na te gaan of de motor inderdaad langzamer draait doordat de spanning voor de helft van de tijd onderbroken wordt. De gemiddelde spanning is dus de helft van wat die eerst was. Dat wil zeggen, dat de energie een kwart bedraagt. Ik meet 4.03 volt. Dat kan te weinig zijn, want de energie op de motor neemt kwadratisch af met de spanning.
Willen we die percentages veranderen, dan kunnen we de volgende truc toepassen:

```
rijden          movlw   0x03
                movwf   portb
                movwf   portb
                movlw   0x02
                movwf   portb
```

Hiermee wordt twee derde van de tijd de spanning hoog. Dat wil zeggen 66%. De energie wordt daarmee ongeveer 44 % van volt.
In het volgende geval:

```
rijden          movlw   0x03
                movwf   portb
                movlw   0x02
                movwf   portb
                movwf   portb
```

wordt de spanning in 33 % van de tijd toegevoerd. Dat wil zeggen, dat de energie naar de motor ongeveer 11 % van het maxiale wordt. Meet de spanning over de

motor. (2,38 volt meet ik) Voor nadere informatie over deze berekening: zie de beschrijving van de H-brug in Motorsturing.
Dit kan ook: tweederde van de tijd spnning over de motor, 4,45 volt.

```
rijden          movlw   0x03            ; plaats de waarde in geheugen w; rechtdoor
                movwf   portb           ; kopieer w naar portb
                movwf   portb

                movlw   0x02            ; nu zetten we de poort even op 0
                movwf   portb           ; energietoevoer even stoppen


                goto    rijden
```

Willen we hetzelfde effect bereiken met de tegengestelde draairichting, dan moeten we achtereenvolgens kiezen:

```
                movlw   0x01            ; ipv 0x03
```

en

```
                movlw   0x00            ; ipv 0x02
```

Ga dit allemaal na met hulp van de tabel op bladzijde 6 van deze handleiding (deel 1).

Gestuurd door onze PIC-processor kunnen we dus op deze manier een gelijkstroommotor aansturen.