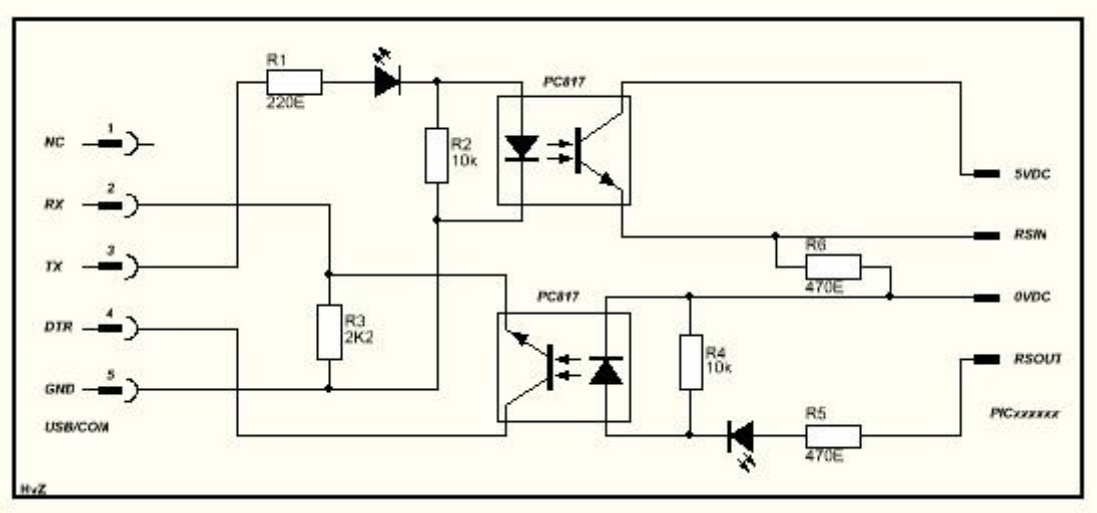
STEPPER MOTOR CONTROL WITH PROFILAB EXPERT 4.0

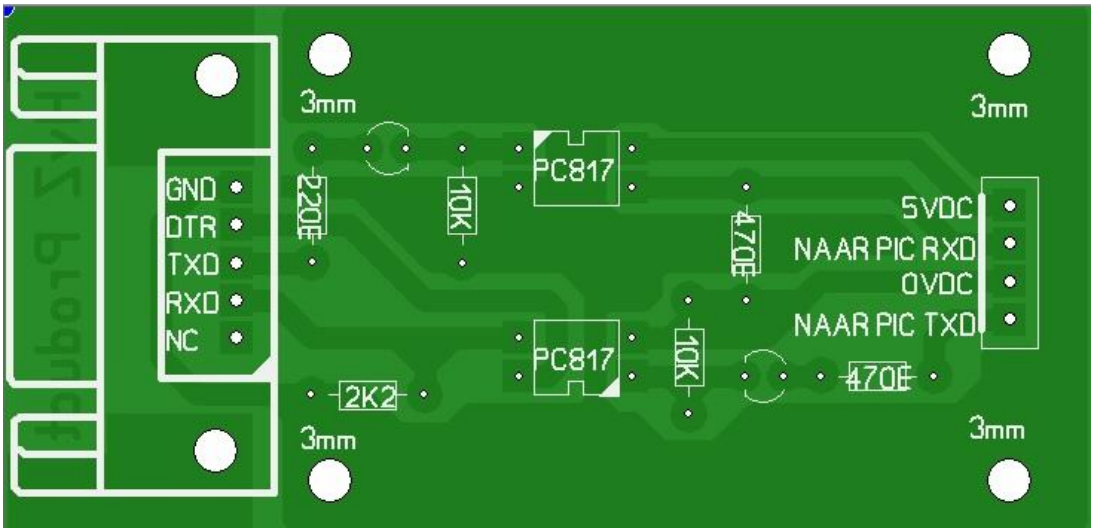
I would like to show you how to control a stepper motor via Profilab Expert 4.0 in combination with a microcontroller. The microcontroller used for this is in this case a 16F887, which is a controller from Microchip. But it is also possible with a different type of course. The control is via the USB port of the PC, with Profilab you can access the USB port. You can find this under the heading hardware in Profilab. To switch from the USB port to RS232 you need a converter, that is such a device, see below. It is of course also possible via the serial port of the PC, then you do not need the converter.

This converts the signal from the USB port to RS232. To protect the microcontroller I made a PCB with a few optocoplers on it, this ensures that the signals remain galvanically separated. Below the schedule.



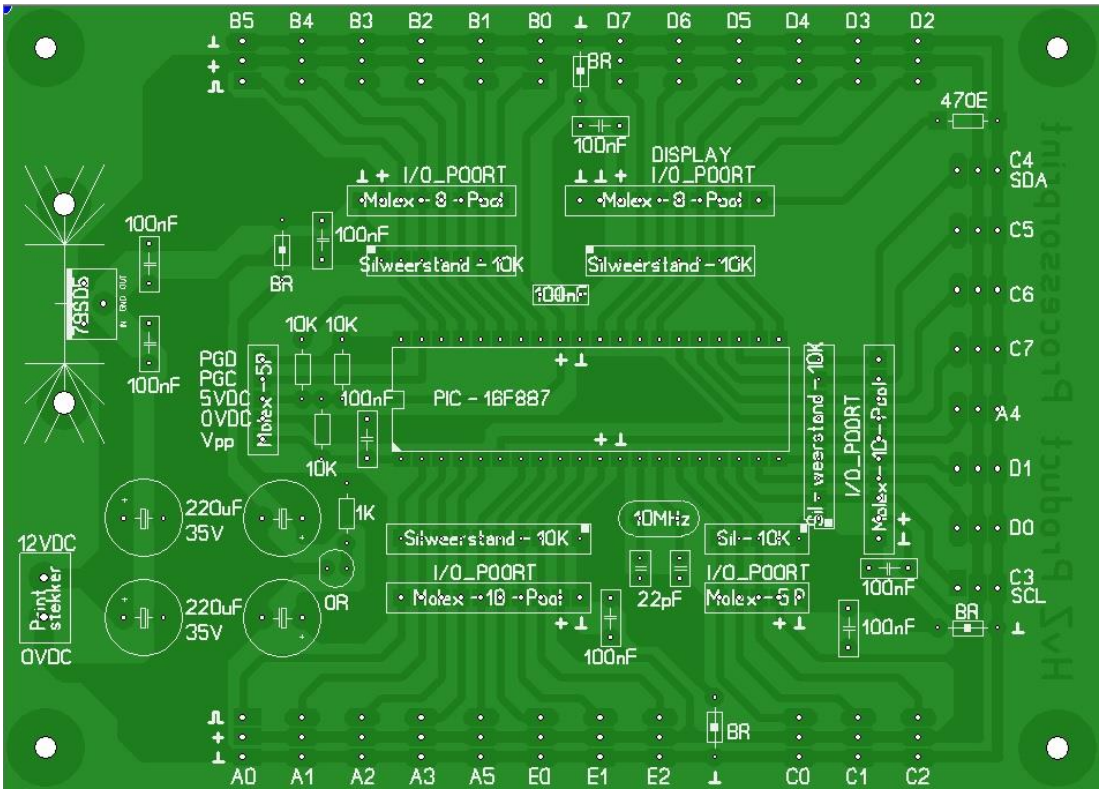
The power supply for the RX signal comes from the DTR pin, which becomes high when Profilab is active. If the RSOUT signal from the controller now goes high, then the RX pin will also go high. The TX pin in turn controls the RSIN pin on the controller. On the right side of the diagram you can see that the power is coming from the controller, in this case it is 5VDC.

Below you see the PCB for the interface.



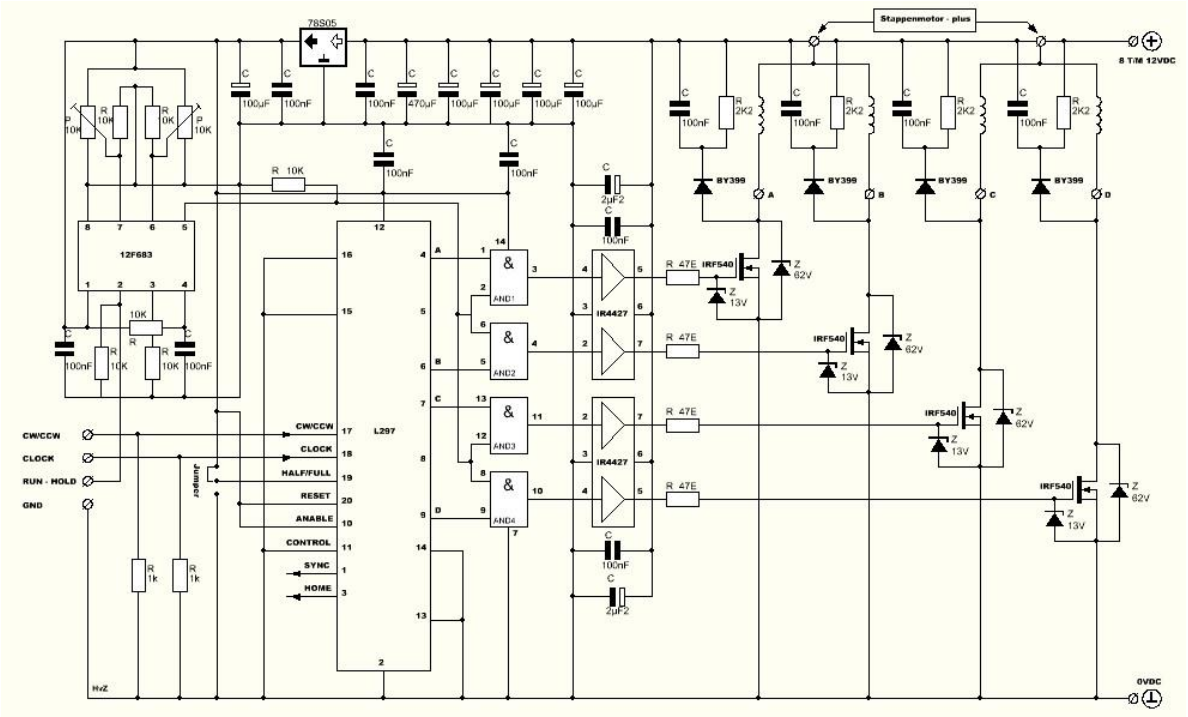
The PCB is made with Sprint-Layout 4.0, just like the rest of the PCBs that follow. On the left side of the PCB you see a 9-pin SUBD connector for the RS232 connection. On the right side of the PCB you can see the connection at the front of the controller, this is done with a four-pin connector from Molex.

This allows you to make communication between the PC and the controller. This way you can send data back and forth. Below you can see the PCB of the controller.



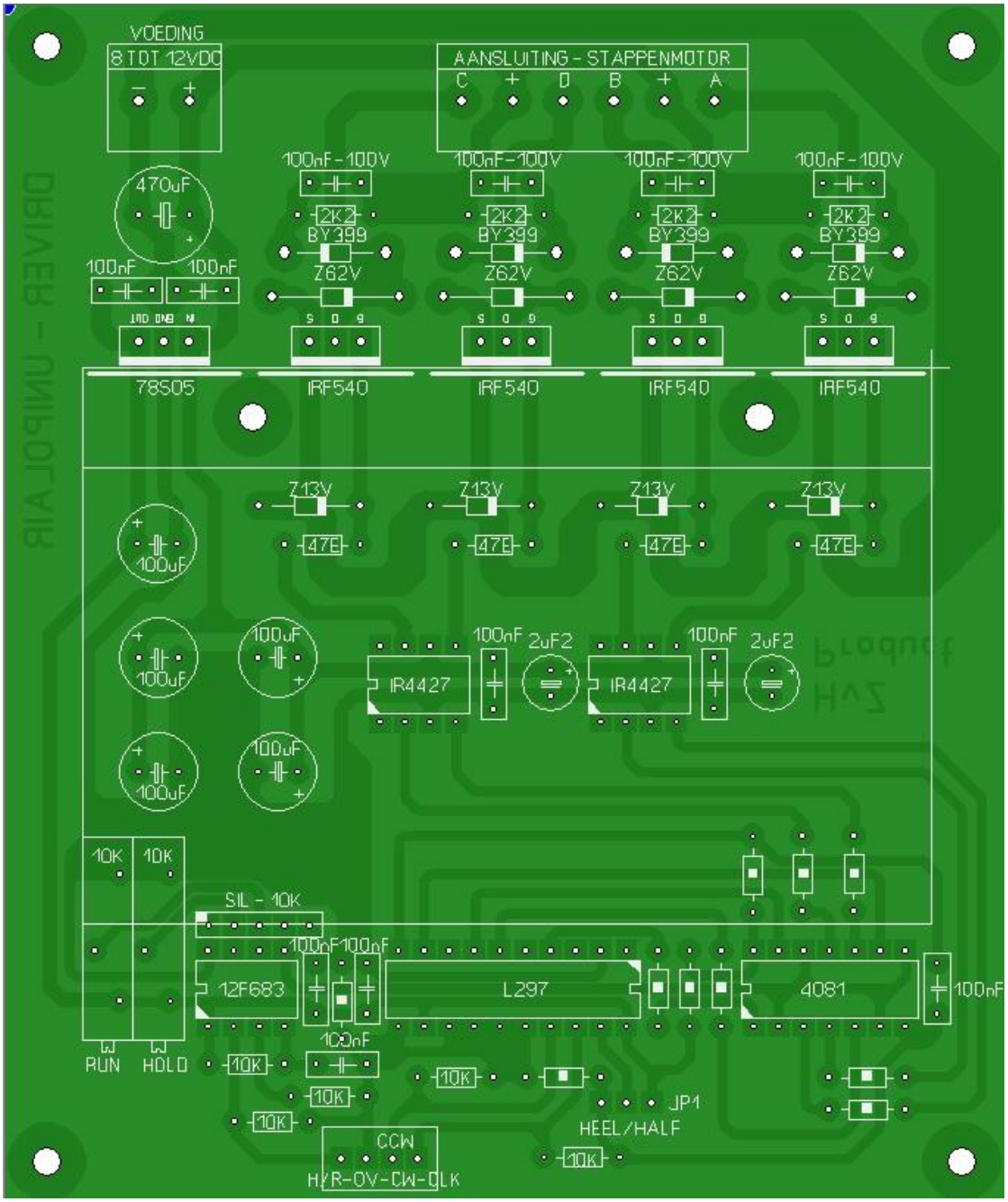
Here too molex connectors are used, each port pin is on a three-pole connector the port and a plus and minus connection. This way you can easily connect something to the ports. All ports are also connected to a multi-pin connector, port-A and port-E are together, port-C, port-D and port-B. There is also an ICSP connector to program the controller. The interface print is connected to port-B.0 and port-B.1 for communication.

So we are going to use this to create a stepper motor control together with Profilab. To properly control a stepper motor you need a driver, below you can see the diagram.



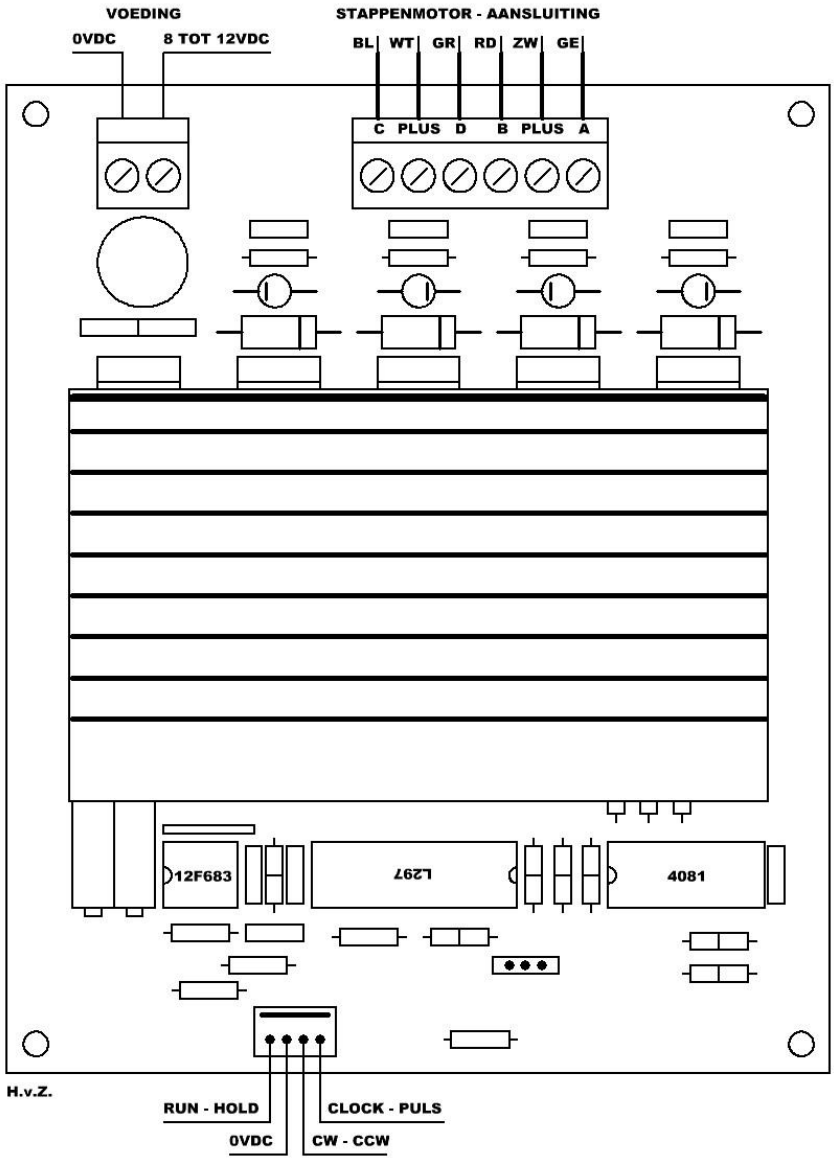
You can buy a driver for this or make your own, I chose to make it myself. This is a driver for a unipolar stepper motor, not many drivers can be found for that either. Usually bipolar motors are used and you have plenty of drivers for that. I used a L297 here, you can make full and half steps with it. This is often used in combination with an L298, but I opted for mosfets because they can provide much more power. I also used an extra microcontroller for the flow control, this was done with a 12F683. This regulates the current when the motor is stopped and running. At standstill, the current does not have to be that high (the hold function), but sufficient torque must be supplied while the motor is running. The control is via a four-pole connector, DIR, PULS, and RUN / HOLD and the GND. With the run and hold pin the current is determined during standstill or running. Dir is for the direction and with pulse the step frequency is determined for the motor, so you determine the speed of the motor. Around the mosfets you see a lot of protection and interference suppression in the form of resistors, capacitors, diodes and zener diodes.

Below you see the print for the stepper motor control.

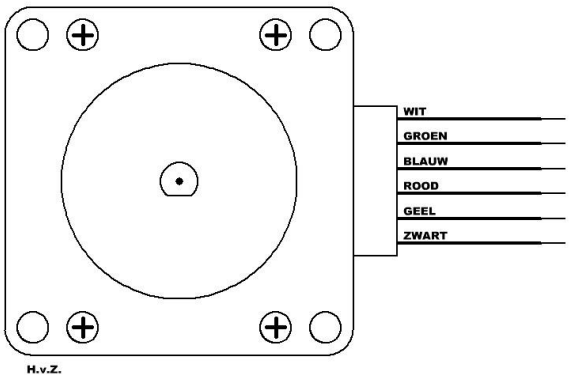


On the PCB you see the two potentiometers at the bottom left with which you can set the current. The mosfets are controlled by mosfet drivers of the type IR4427, in this way there is 12VDC on the gate so that the fets are properly opened. I kept the supply voltage here at 12VDC, but that should be up to 18 volts. The VGS for this type of mosfet is +/- 20 V. The mosfets and the voltage regulator (78S05) are together on a cooling element. The mounting of the mosfets and the voltage regulator is done here by means of springs, I personally think that works better than those plastic mounting rings with bolts.

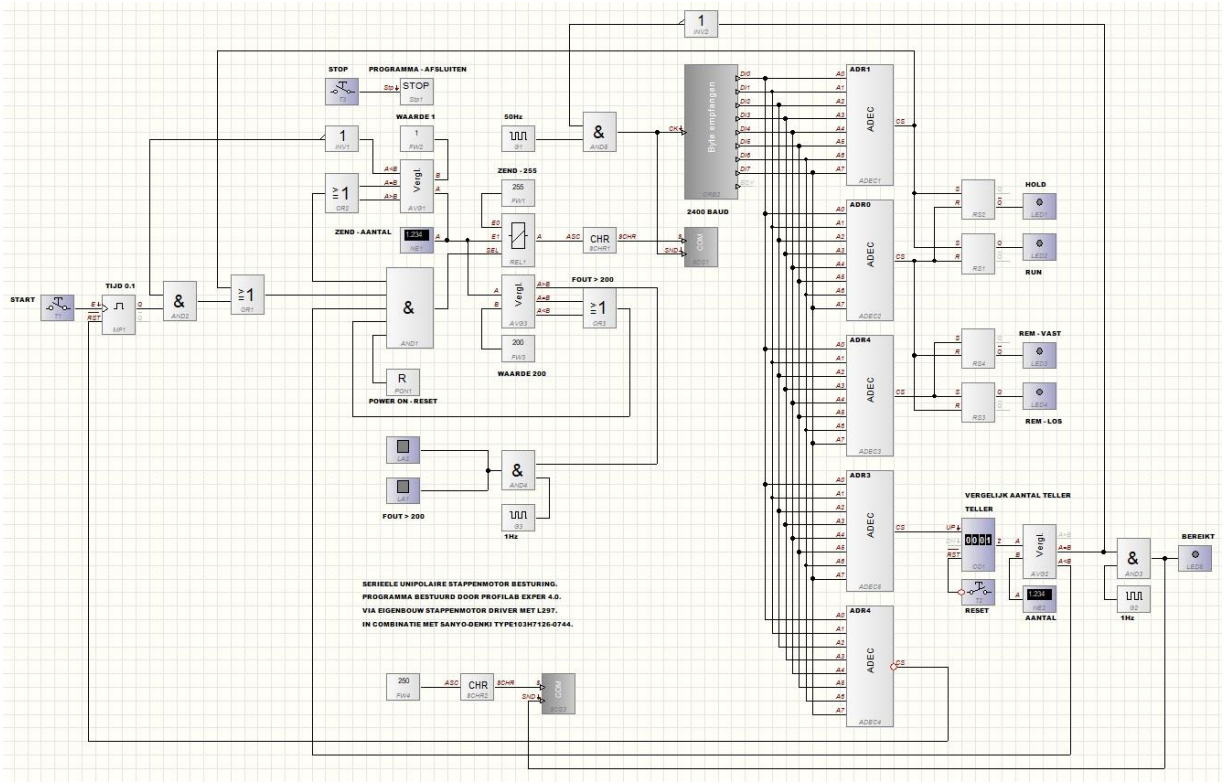
Below you see the connections for the stepper motor driver.



Below the stepper motor used.



We are now going to look at the program made in Prifilab.



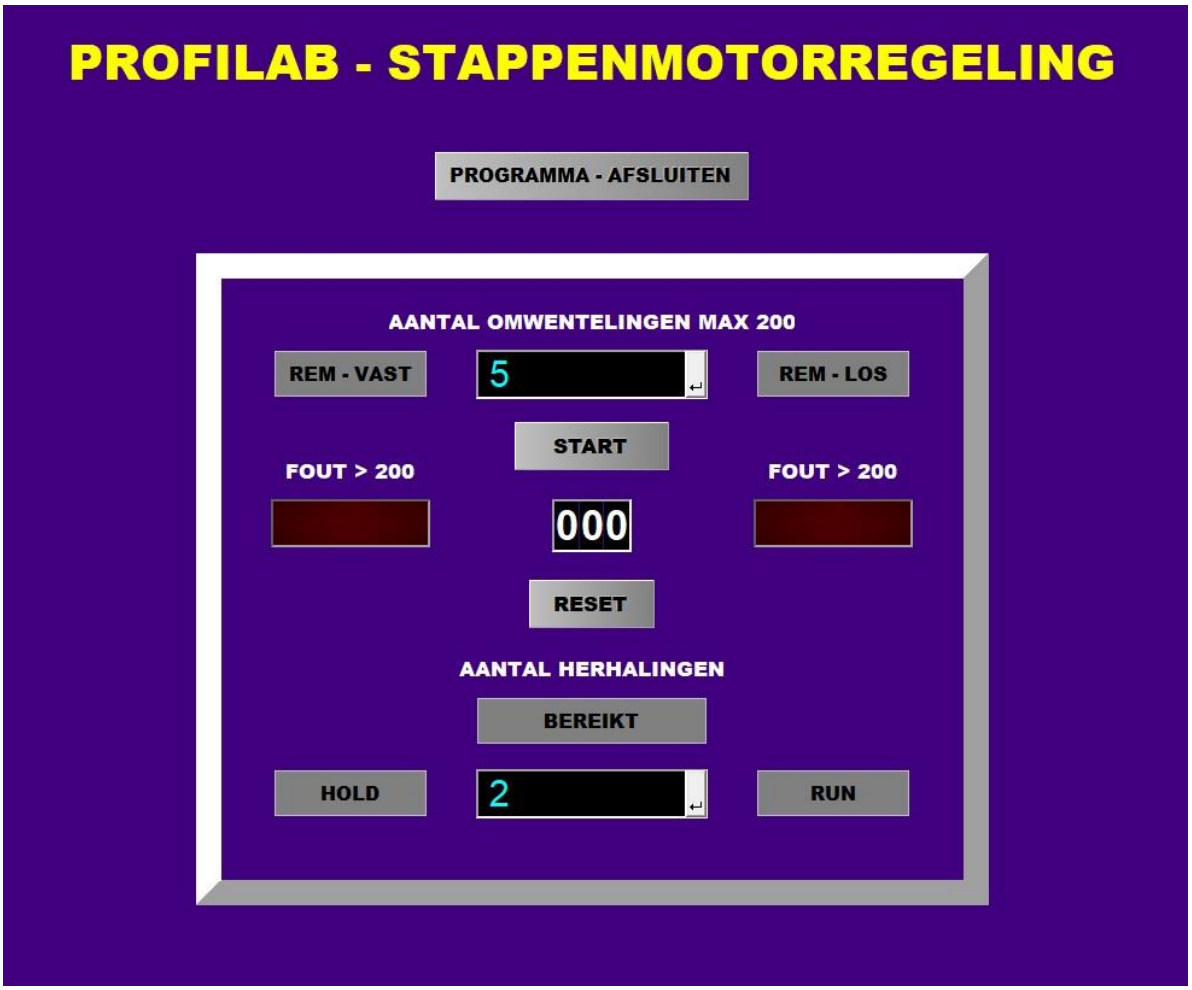
The data is sent at 50Hz, which is more than fast enough in this case and the baut rate is set at 2400. I do not do anything other than send a command, the controller continues to process it. This is also the best method if you use a PC, because in this way you have little or nothing to do with Windows interrups.

On the right you see a few address decoders with the addresses 0, 1, 3, and 4. These values are sent by the controller, via address 0 and 1 the leds RUN and HOLD are switched on and off on the front plate. The LEDs REM-VAST and REM-LOS are switched on and off on the front plate via addresses 0 and 4. Via address 3, the counter readings are compared with the number of set repetitions that must be made. When address 4 arrives, a value 250 is sent to the controller. What goes through the addresses are all incoming values that are sent by the controller.

The only thing that is sent by Profilab is the value of the number of revolutions to be made and the value 250 for when the number of revolutions has been reached. You can only send 8 bits this way, so you can send values between 0 and 255 with this example. As you can see you cannot make the stepper motor more than 200 rounds, if you enter a value that is greater than 200 then two LED bars will flash and everything is blocked.

You can make this as complicated as you want, of course, but this is an example of how you can communicate with a controller via profilab.

This also includes a front plate, of course, see below.



At the top you see the field where you can enter the number of revolutions, here you can enter a value from 0 to 200. At the bottom you see the field where you can enter the number of repetitions, every time an X number of revolutions have been sent. the controller value 3 and that is again compared in Profilab with the entered value for the number of repetitions. The number of repetitions is also kept in a counter, located below the start button. When the value is reached, everything is blocked and the counter must first be reset before continuing.

In the LED bars you see the text brake-fix, brake-release, hold and run, which light up when the motor is running or stopped, that data is sent by the controller. So there is continuous interaction between the controller and Profilab.

This was pretty much it as far as Profilab is concerned. We will now look at the program of the microcontroller. As said before, the stepper motor driver has a 12F683, which also contains a piece of software. See below.

```
'*  Name    : STEP EN HOLD VOOR STAPPENMOTORREGELING.
'*  Author  : H.van Zwieten.
'*  Notice  : Copyright (c) 2015 H.v.Z.
'*          : All Rights Reserved
'*  Date    : 22-9-2015
'*  Version : 1.0
'*  Notes   :

Device 12F683                           ; processor type

Xtal 4                                  ; Interne oscilator

Config WDT_OFF,_                        ; WatchDog Timer uit
       PWRTE_ON,_                       ; Power-up Timer Enable aan
       MCLRE_OFF,_                      ; Externe Master Reset Enable uit
       INTRC_OSC_NOCLKOUT               ; Interne oscilator aan

All_Digital true                        ; Alle poorten digitaal

Declare CCP1_Pin GPIO.2                 ; PWM uitgang

Declare Adin_Res    = 8                 ; Resolutie 8 bits
Declare Adin_Tad    = frc               ; Set rc osc
Declare Adin_Stime = 50                 ; Sample tijd 5

Symbol VRIJGAVE_RUN_HOLD = GPIO.5       ; Ingang vrijgave run_hold
Symbol FREQUENTIE = 20000               ; PWM frequentie 20KHz

Dim HOLD_FUNCTIE As Byte                ; Variabele analoog_in 0
Dim RUN_FUNCTIE As Byte                 ; Variabele analoog_in 1

Clear                                   ; Wis geheugen

         ;543210                        ; Hulpregel
GPIO    = %000000                       ; Maak poort laag
TRISIO = %110011                        ; Poort I/O

         ;76543210                      ; Hulpregel control register
ADCON0 = %00000001                      ; A/D control register

;-------------------------------------------------------------------------
; HOLD; STROOM INSTELLING DOOR DE SPOELEN - INSTELLEN VIA ANALOGE INGANG 0.
;-------------------------------------------------------------------------

HOLD:
    HOLD_FUNCTIE = ADIn 0
    HPWM 1,HOLD_FUNCTIE,FREQUENTIE

    If VRIJGAVE_RUN_HOLD = 1 Then
     GoTo RUN
    EndIf
GoTo HOLD
```

```
;-------------------------------------------------------------------------
; STEP; STROOM INSTELLING DOOR DE SPOELEN - INSTELLEN VIA ANALOGE INGANG 1.
;-------------------------------------------------------------------------

RUN:
    RUN_FUNCTIE = ADIn 1
    HPWM 1,RUN_FUNCTIE,FREQUENTIE

    If VRIJGAVE_RUN_HOLD = 0 Then
     GoTo HOLD
    EndIf
GoTo RUN

End
```

As you can see it is only a very short program. Three entries are used, they are HOLD_FUNCTION ADIN 0, RUN_FUNCTION ADIN 1 and RELEASE_RUN_HOLD. And then we have one more output and that is HPWM 1, which provides the pwm signal and thus determines the current through the stepper motor.

The frequency with which this happens is 20KHz, which frequency has been chosen so high to ensure that the background noises are less audible. Otherwise, you would hear an annoying beep if the frequency ware lower.

We will now look at the program for the communication between Profilab and the 16F887.

```
'*  Name    : SERIEELE UNIPOLAIRE STAPPENMOTOR BESTURING.
'*  Author  : H van Zwieten.
'*  Notice  : Copyright (c) 2016 H.v.Z.
'*          : All Rights Reserved
'*  Date    : 14-11-2016
'*  Version : 1.0
'*  Notes   : PROGRAMMA BESTUURD DOOR PROFILAB EXPER 4.0.
'*  Notes   : VIA EIGENBOUW STAPPENMOTOR DRIVER MET L297.
'*  Notes   : IN COMBINATIE MET SANYO-DENKI TYPE103H7126-0744.

Device 16F887                          ; Processor type

Xtal 10                                ; Cristal 10Mhz

Asm                                    ; Config settings
CONFIG_REQ
__CONFIG _CONFIG1, HS_OSC & WDT_OFF & DEBUG_OFF & FCMEN_OFF & LVP_OFF &
IESO_OFF & BOR_OFF & CPD_OFF & CP_OFF & MCLRE_OFF & PWRTE_ON
__CONFIG _CONFIG2, WRT_OFF & BOR40V
EndAsm

All_Digital true                       ; Alle poorten digitaal

Declare Serial_Baud 2400               ; Baudrate 2400

Declare Rsin_Pin  PORTB.0              ; Data in
Declare Rsout_Pin PORTB.1              ; Data uit
```

```
Declare LCD_RSPin PORTD.2           ; Reset display poort D.2
Declare LCD_ENPin PORTD.3           ; Enable display poort D.3
Declare LCD_DTPin PORTD.4           ; Data display poort D.4 t/m D.7

Symbol PULS = PORTA.2               ; Ingang puls
Symbol DIR  = PORTA.1               ; Ingang DIR 0 = rechts DIR 1 = links
Symbol STEP_HOLD = PORTA.0          ; Ingang houd stroom

Symbol STAPPEN = 400                ; Instelling halve stappen

Symbol S1 = PORTE.2                 ; Ingang vrijgave puls

Symbol MOTORREM = PORTC.0           ; Uitgang motorrem
Symbol LED = PORTC.2                ; Uitgang controle led

Symbol FREQUENTIE = 400             ; Instelling stap frequentie
Symbol WACHT = 250                  ; Instelling wacht links rechts

Dim DATA_IN As Byte                 ; Serieële darta in

Dim I As Dword                      ; Variabele FOR NEXT
Dim VAR1 As Dword                   ; Variabele data in X stappen

Clear                               ; Wis geheugen

Cls                                 ; Wis display

DelayMS 500                         ; Pauze 0.5 sec

         ;543210                    ; Hulpregel poort A
PORTA  = %000000                    ; Maak poort A laag
TRISA  = %000000                    ; Poort_A I/O

         ;543210                    ; Hulpregel poort B
PORTB  = %000000                    ; Maak poort B laag
TRISB  = %000001                    ; Poort_B I/O

         ;76543210                  ; Hulpregel poort C
PORTC  = %00000000                  ; Maak poort C laag
TRISC  = %00000010                  ; Poort_C I/O

         ;76543210                  ; Hulpregel poort D
PORTD  = %00000000                  ; Maak poort D laag
TRISD  = %00000000                  ; Poort_D I/O

         ;210                       ; Hulpregel poort E
PORTE  = %000                       ; Maak poort E laag
TRISE  = %111                       ; Poort_E I/O

         ;76543210                  ; Hulpregel analoog
ADCON0 = %00000001                  ; ADCON0 register analoog
         ;543210                    ; Hulpregel analoog poort_B
ANSELH = %000000                    ; ANSEL register analoog poort_B
```

```
; -----------------------------------------------------
; PROGRAMMA SERIEELE UNIPOLAIRE STAPPENMOTOR BESTURING.
; -----------------------------------------------------


DATA_IN = 0                         ; Zet data_in op nul
VAR1 = 0                            ; Zet Var1 op nul
I = 0                               ; Zet I op nul

START:                              ; Start

    MOTORREM = 1                    ; Rem is aan
    STEP_HOLD = 0                   ; Stroom stappenmotor hold functie

    PULS = 0                        ; Stappuls is nul

    LED = 0                         ; Controle led is uit

    Print At 1,1,"PULSEN"           ; Print data
    Print At 2,1,"OMWENTELING"      ; Print data

    DATA_IN = RSIn                  ; Omzetten serieel naar variabele
    DelayMS 1                       ; Wachttijd

    If DATA_IN < 201 Then           ; Als waar dan
     MOTORREM = 0                   ; Rem is los
      DelayMS 1                     ; Wachttijd
     GoTo RUN_STAPPENMOTOR          ; Ga naar run stappenmotor
    EndIf                           ; Einde als

    If DATA_IN = 250 Then           ; Als waar dan
     Cls                            ; Wis display
      GoSub WAARDE_BEREIKT          ; Ga naar waarde bereikt
    EndIf                           ; Einde als


    If S1 = 1 Then                  ; Als waar dan
     RSOut 1                        ; Zend adres 1 / set run
      DelayMS 1                     ; Wachttijd
    EndIf                           ; Einde als

    If S1 = 0 Then                  ; Als waar dan
     RSOut 0                        ; Zend ades 0 / Set hold / rem vast
      DelayMS 1                     ; Wachttijd
    EndIf                           ; einde als

GoTo START                          ; Ga naar start
```

```
; ------------------------------------------------
; UITGAVE AANTAL STAPPEN MET UNIPOLAIRE STAPPENMOTOR.
; ------------------------------------------------

RUN_STAPPENMOTOR:                   ; Run stappenmotor

    MOTORREM = 0                    ; Rem is uit
    DIR = 0                         ; Ingave richting stappenmotor
    STEP_HOLD = 1                   ; Stroom stappenmotor step functie
    LED = 1                         ; Controle led is aan

    RSOut 1                         ; Zend adres 1 / set run
    DelayMS 1                       ; Wachttijd
    RSOut 4                         ; Zend adres 4 / rem los
    DelayMS WACHT                   ; Wachttijd wacht

    VAR1 = DATA_IN * STAPPEN        ; Var1 is serieele data maal stappen

    Print At 1,8,Dec VAR1,          ; Print data
    Print At 2,13,Dec VAR1/STAPPEN  ; Print data

    If VAR1 = 0 Then                ; Als waar dan
     STEP_HOLD = 0                  ; Stroom stappenmotor hold functie
      GoTo START                    ; Ga naar start ontvangen en verzenden
data
    EndIf                           ; Einde als

    For I = VAR1 To 1 Step - 1      ; Herhaal waarde aantal stappen tot één
      PULS = 1                      ; Stappuls is één
      DelayUS FREQUENTIE            ; Stap frequentie
      PULS = 0                      ; Stappuls is nul
      DelayUS FREQUENTIE            ; Stap frequentie

      If I = 1 Then                 ; Als waar dan
       I = 0                        ; Zet I op nul
        VAR1 = 0                    ; Zet Var1 op nul
         RSOut 0                    ; Zend adres 0 / set hold / rem vast
          DelayMS 1                 ; Wachttijd
           RSOut 3                  ; Zend adres 3 hoog teller op
          DelayMS 1                 ; Wachttijd
         DelayMS WACHT              ; Wachttijd wacht
        Cls                         ; Wis display
       GoTo START                   ; Naar begin start
      EndIf                         ; Einde als
    Next I                          ; Herhaal waarde aantal STAPPEN tot één

GoTo RUN_STAPPENMOTOR               ; Ga naar run stappenmotor


WAARDE_BEREIKT:                     ; Waarde bereikt

    Print At 1,1,"WAARDE_BEREIKT"   ; Print data
    DelayMS 500                     ; Wacht
    Cls                             ; Wis display
    Return                          ; Terug naar start

GoTo WAARDE_BEREIKT                 ; Ga naar waarde bereikt

End                                 ; Einde programma
```

As you can see, this program is a bit more extensive, we start at the top. There you can see that we work with a baud rate of 2400 for communication. Below that we set the RSIN_PIN and the RSOUT_PIN, which will be on port B.0 and B.1. Below that we put the declaration for the display, we will be able to put different texts on it.

Then we will use a symbol to determine which gates the inputs and outputs will be used for, and we can also enter fixed values there, which cannot be changed. We put values that do change in a variable, such as DATA_IN, I and VAR1. And finally, we specify whether a particular gate should become an entrance or exit.

The program.

Before we enter the start loop, we first set all three variables to zero. In the start loop you can see that the brake is still on, step hold is zero, pulse is zero and the control led is zero. Then a print command is given with the text PULSEN and REVOLUTION. After that, the program does not continue because it is waiting for data from Profilab. If data comes in from Profilab with a value less than 200, the program will go to the run loop, if the value is greater than 200, the program will not jump to the run loop. The engine can also be started with S1, then a value is sent to Profilab and the engine will also start running.

In the run loop, the data from Profilab is put in the variable DATA_IN X STEPS in VAR1 and that is further processed in the program. You can see that at the symbol steps 400 has been entered, which 400 is the number of steps required for one revolution of the stepper motor. The stepper motor driver is therefore in half step operation. With this program you can make the motor with only one complete revolution each time, so if you want to steer the motor per degree, a different program is needed. But that is also possible, the possibilities are endless with the combination of a microcontroller together with Profilab.

The program in the run loop now enters the for next loop and the number of steps is executed there. When the number of entered steps have been made, the controller sends data to Profilab again, the LEDs are set correctly and the counter is increased. This process will repeat itself after the start command has been given in Profilab or by S1. When the number of repetitions is reached (for example 10 X) then the VALUE_RANGE loop is jumped and a print command is given again and the display is reset there.

This was about it, I hope it is a bit clear how you can communicate in a relatively simple way between a controller and Profilab Expert 4.0.

Author:

H. van Zwieten.