


```

/*
 * PROCEDURE INTERRUPT TIMER0
 * Wordt aangeroepen door TIMER0 interrupt
 * Wijzig afhandeling van het programma.
 * INPUT: - - -
 * OUTPUT: - -
 * Veranderd de volgende variabelen
 * Elk 10 ms Timer1 = Stopwatch1
 * Timer2 = Stopwatch2
 * Als GO * waar is wordt er een copy in D time * bewaard.
 * Als stopwatch * groter als Par(3) wordt wordt stop
 * Als stopwatch * groter als Par(4) wordt ga in
 * start positie
 * Als stopwatch * grootter als Par(4) wordt ga in
 * disable;
 * ISR TO: PROCEDURE INTERRUPT TIMER0 using I!
 * ISR TO: ISR = 122;
 * do;
 * if Time_5m then
 * do;
 * if Time_1 then
 * Time_1(0)=dec(Time_1(1)+Time_1(0)); /* Time_1oms bewaart 10 ms in BCD vorm */
 * Time_1(1)=dec(Time_1(1)+Time_1(0)); /* Time_1oms bewaart 10 ms in BCD vorm */
 * CY=FALSE; /* Time1 ++ */
 * D_Time_1=Time1; /* copy time */
 */

```

```

init_timer_0: PROCEDURE !
    TMOD = TMOD AND 1111$0000B; /* RESET TIMER 0 FLAGS
    TMOD = TMOD OR 0000$0000B; /* SET COUNTER 0 IN MODE
    TH0 = 12; /* RELOAD WARRDE 112
    TR0 = 1; /* TIMER 0 STARTEN
    ET0 = 1; /* interrupt enable
    PT0 = 1; /* TIMER 0 EFFECT PRIORITYD */
    PWMP = OFF; /* PRESCALER PWM OUTPUT
    Time_5m=FALSE; /* RESET 5 ms TIMER */
END init_timer_0;

```

```
/* **** PROCEDURE init_timer_0 ****/
/* **** Wordt aangetrokken door INITIALISATIE ****/
PROCEDURE init_timer_0
{
    /* **** Rodept - aan ****/
    WORDT_aangetrokken_door_INITIALISATIE
    /* **** INITIAALISATIE ****/
    Roeppt - aan
    /* **** INPUT : - ****/
    /* **** OUTPUT: - ****/
    /* **** Initialiseert TO en prescaler PWM. ****/
}
```

```
    return DE;
    return CE;
    return DE;
    return EE;
    return EF;
end;
```



```

/*
PROCEDURE INTERRUPT CAPTURE_1
Wordt aangeroepen door CAPTURE_1 (TIME2) interrupt
ROept - aan
Start impuls afhandeling van baan 2
Veranderd de volgende variabelen
Als start verwacht wordt :
Wordt aangeroepen door CAPTURE_1 (TIME2) interrupt
ROept - aan
Stop impuls afhandeling van baan 1
END CT1;
end;
enable;
RESET CAPTURE_1;
do;
if (CTCON and 0000$0100b) > 0 then /* Als opganeide flank komt
Debounce =2= TRUE;
/* Schakel debounce in
Dender(1) = 0;
/* Reset dender timer
do;
disable;
if (CTCON and 0000$0100b) > 0 then /* Als opganeide flank komt
Debounce =2= TRUE;
/* Schakel debounce in
Dender(1) = 0;
/* Reset dender timer
do;
end;
enable;
RESET CAPTURE_1;
end;
END CT1;

```

```

/*
 *****
 * CTO: PROCEDURE INTERRUPT CAPTURE_0
 * Wordt aangeroepen door CAPTURE_0 (TIME2) interrupt
 * Roppt - aan
 * Start impuls afhandelend van baan 1
 * Veranderd de volgende variabele
 * Als start verwacht wordt :
 * Debounce 1 valg gehesen
 * Dender (0) voor baan 1 gereset
 * *****
 */
/* CTO: PROCEDURE INTERRUPT CAPTURE_0 using 3:
 *****
 * Disabale!
 * do;
 * if ((CTCON and 0000$0001b) > 0 then /* Als opgaande flank komt
 *      /* schakel debounce in
 *      /* Reset dender timer
 *      /* Reset interrupt valg
 *      /* enable;
 * end;
 * END CTO;

```

```

do! disabale;
do! if (CTCON and 0100$0000b) > 0 then /* Als opgaaande flank komt
/* Debounce_2 = TRUE;
/* Schakel debounce_in
/* Reset dender timer
Dender(1) = 0;
do! end;
end! enable;
RESET_CAPTURE_3;
/* Reset interrupt valg
if (CTCON and 0100$0000b) > 0 then /* Als opgaaande flank komt
/* Debounce_2 = TRUE;
/* Schakel debounce_in
/* Reset dender timer
Dender(1) = 0;
do! end;
end! enable;
RESET_CAPTURE_3;
/* Reset interrupt valg
END CTS;
/*
***** PROCEDURE Read Ad
/*
Wordt aangeroepen door Les$Interface
/*
Deze functie geeft de waarde van de AD converter af
/*
die een spanning omgezet heeft op input Chan
/*
Read Ad: procedure (Chan) word;
/*
declare Chan byte;
/*
dcl Temp1 byte;
/*
dcl Temp2 byte;
/*
dcl Result word;
/*
do! if Temp1 > 0 then
Temp1 = ADCON and Ad_Mask_If;
/*
Als bit hooog is resetten */
/*
Chan = Chan and 7;
/*
/* Beperek kanbaar keuze byte */
/*
do!

```

```
/*
/* PROCEDURE INTERRUPT CAPTURE_3
/* Wordt aangeroepen_door CAPTURE_3 (TIME2) interrupt
/* ROept - aan
/* Stop impuls afhandeling van baan 2
/* Veranderd de volgende variabeleën
/* Als start verwacht wordt :
/* Debounce 2 vlag gehesen
/* Dender (1) voor baan 2 gereset
/* */
/*
```

```

CT2: PROCEDURE INTERRUPT CAPTURE_2 Using 3;
      if (CTCON and 0001$0000b) > 0 then /* Als opgaaande flank komt
do; disable; /* debounce in Schakel debounce in
      Dender(0) = 0; /* Reset dender timer
end; /* end;
      RESET CAPTURE_2;
      /* Reset interrupt vlag
enable; /* end;
END CT2;

```

```

/*
***** PROCEDURE Lees$Interfase *****

Wordt aangeroepen door hoofd programma dus
/*
/* Wordt de stand van potmeter uit en geeft evenstueel de ingestelde
/* Waarden van de programma variabelen weer
/*
***** Leest de stand van potmeter uit en geeft evenstueel de ingestelde
/*
***** DCL Temp byte; procedure byte;
/*
/* Dumb = Read_Ad(7); /* Lees de stand van de 'potmeter' in
/*
/* Temp = not(P4); /* Lees de stand van de schakelaar in
/*
/* Als de schakelaar op 0 staat en druk-
/*
/* If Temp = 0 then
/*
/* Else
/*
/* Par(Temp and 0fh) = Bin2BCD(Dummy); /* Wordt alleen Par(0) inlezen
/*
/* Par(Temp and 0fh) = Bin2BCD(Dummy); /* Converteer BIN naar BCD
/*
/* If Temp >= 80h then /* Vaker het volgende uit als drukknop
/*
/* Else
/*
/* Par(Temp and 0fh) = Par(Temp and 0fh); /* Minimaal impuls lengte 1 *
/*
/* Par(Temp and 0fh)=Par(Temp and 0fh)/4; /* Minimaal impuls lengte 2 *
/*
/* Par(Temp and 0fh)=Par(Temp and 0fh)*1; /* Minimaal impuls lengte 0 *
/*
/* Par(Temp and 0fh)=Par(Temp and 0fh)*4; /* Maximaal impuls lengte 4 *
/*
/* Par(Temp and 0fh)=Par(Temp and 0fh)*1; /* Minimaal impuls lengte 3 *
/*
/* Par(Temp and 0fh)=Par(Temp and 0fh)/4; /* Maximaal impuls lengte 4 *
/*
/* Par(Temp and 0fh)=Par(Temp and 0fh)*1; /* Minimaal impuls lengte 5 *
/*
/* Par(Temp and 0fh)=Par(Temp and 0fh)/4; /* Maximaal impuls lengte 4 *
/*
/* Par(Temp and 0fh)=Par(Temp and 0fh)*1; /* Minimaal impuls lengte 3 *
/*
/* Par(Temp and 0fh)=Par(Temp and 0fh)/4; /* Maximaal impuls lengte 4 *
/*
/* Par(Temp and 0fh)=Par(Temp and 0fh)*1; /* Minimaal impuls lengte 2 *
/*
/* Par(Temp and 0fh)=Par(Temp and 0fh); /* Minimaal impuls lengte 1 *
/*
/* Par(Temp and 0fh)=Par(Temp and 0fh); /* Minimaal impuls lengte 0 *
/*
/* Par(Temp and 0fh) = Dumb;
/*
/* Do case temp and 0fh;
/*
/* End;
/*
***** SLA=SLA+0; /* R/W Schrijf EPPROM adres pointer. */
/*
/* MTD(0)=(Temp and 0fh)*2; /* Schrijf EPPROM adres pointer. */
/*
/* MTD(0)=HIGH (Par(Temp and 0fh)); /* en vervolgens het lange byte
/*
/* MTD(1)=LOW (Par(Temp and 0fh)); /* Verzend drie bytes
/*
/* Call START MASTER TX(3); /* Wacht tot dit verzoenden is
/*
/* Do while NumBytMst<>0;
/*
/* End;

```

```

ADCON = 0;
end;
ADCON = Chan or Ad_Start;
/* KIES KANAL EN START CONVERSE*/
Temp1 = 0;
do While (Temp1 = 0);
    /* Wacht tot converteerde klap is */
    Temp1 = ADCON and Ad_Mask_IF;
    /* Test bit
    end;
    /* Lees 8 bit resultaat */
Temp1 = ADCH;
/* ADCH!
Temp2 = ADCON;
/* ADCON;
Temp2 = shr(Temp2,6) and 3;
/* Les 2 bits */
Temp2 = Temp1 + Temp2;
/* Schuif 2 bits 6 naar rechts */
Result = Temp2;
/* Verlaat de subroutine met het resultaat
naar links en tel de */
return Result;
END read_Ad;

```



```

/*
***** INITIAALISATIE PROCEDURE *****
***** END VAN PROCEDURES *****
***** BEGIN HOOFD PROGRAMMA *****
/* Begijn hoofd programma */

/* Lamp1=ROOD;
Lamp2=ROOD;
/* Zet lampen op rood.
/* Eigen slave IIC adres is 12H
/* Init IIC(12H);
call Init_IIC(12H);

P1 = OFF;
/* Mak output van P1 hoog voor input */
/* Mak output van P1 hoog voor input */
P1 = OFF;

P4 = OFF;
/* Refresh;
/* Flag150ms = FALSE;
Display_1 = FALSE;
Display_2 = FALSE;
G0_1 = FALSE;
G0_2 = FALSE;
/* Reset glijvlag een
/* Reset glijvlag twee
/* Reset uitzwem-vlag een
/* Reset uitzwem-vlag twee
/* Schakel capture interrupt select
/* Schakel capture interrupt
IEN1 = OFF;
/* CAPTURE PRIORITY select
/* CAPTURE interrupt 0..3 in.
CTCON = 0000$0101b;
/* Disp$1=0;
Time1 = 0h;
Time2 = 0h;
DummY = 0!;
TIme1 = 0h;
TIme2 = 0h;
DummY = 0!;
MID(0)=1;
SLA=0A0H;
SLA=SLA+0;
/* Lees Par(info) uit EEPROM
/* Det te verrenden data
/* Laad het adres regiester
/* Set het schrijf bit
/* Start het de transmissie
/* Watch tot de byte weg zijn
do while NumBytMS<>0;
call START MASTER TX(1);
do while NumBytMS<>0;
call START MASTER RX(2);
do DummY = 0 to 6;
/* Set het lees bit
/* Lees de instellingen uit de
/* EEPROM in tweede stukken
/* Deze routinewerkt wel maar
/* ik kan het nog niet goed
end;
Par(DummY) = MID(1)*256;
/* Volgen waarom.....
Par(DummY) = MID(0)! + Par(DummY) + MRD(0)! /* Laten onderzoeken. ?
call TIME(50);
end;

```

```

/*
***** HOOFD PROGRAMMA LUS. *****
/* DO WHILE(1) ! /* Doe, tot de steekker getrokken wordt, het volgende
/* DummY=Lees$Interface; /* Roep de 'human interface' aan
/* DummY = 0 then
/* If Go_1 then
/* do!
/* Out$buffer1 (0) = BLANK; /* Als stopwatch loopt alleen
/* Out$buffer1 (1) = BLANK; /* Als digits op display
/* Out$buffer1 (2) = BLANK; /* MS digits op display
/* Out$buffer1 (3) = BLANK; /* Als MS digit is 0 BLANK,en,
/* else
/* if (shx(Time1 ,12) > 0 then
/* Out$buffer1 (2) = BCD2Seg( shx(Time1 ,8) and 0fh );
/* Out$buffer1 (1) = BCD2Seg( shx(Time1 ,4) and 0fh );
/* Out$buffer1 (0) = BCD2Seg( shx(D_Time_1 ,12) and 0fh );
/* else
/* if (shx(D_Time_1 ,12) and 0fh ) < 0 then
/* Out$buffer1 (2) = BCD2Seg( shx(D_Time_1 ,8) and 0fh );
/* Out$buffer1 (1) = BCD2Seg( shx(D_Time_1 ,4) and 0fh );
/* Out$buffer1 (0) = BCD2Seg( D_Time_1 and 0fh );
/* else
/* if (shx(D_Time_1 ,12) and 0fh ) > 0 then
/* Out$buffer1 (3) = BCD2Seg( shx(D_Time_1 ,12) and 0fh );
/* else
/* do;
/* if Display_1 = FALSE and Debounce_1 = FALSE then
/* Out$buffer1 (0) = BLANK; /* Initiëller standaard tekst*
/* Out$buffer1 (1) = Y; /* Alles in rust: baan is vrij */
/* Out$buffer1 (2) = EL;
/* Out$buffer1 (3) = GE;
/* do!
/* if DisPlay_1 = FALSE and Debounce_1 = FALSE then
/* do;
/* else
/* end;
/* do;
/* if DisPlay_1 = FALSE and Debounce_1 = FALSE then
/* do;
/* else
/* end;
/* else
/* do;
/* if DisPlay_1 = FALSE and Debounce_1 = FALSE then
/* do;
/* else
/* end;
/* end;

```

```

/*
***** BEGIN INTRO *****
*/
/* Verstuur intro data
out$buffer1 (3) = HA;
out$buffer1 (1) = ES;
out$buffer1 (0) = PE;
out$buffer1 (2) = BLANK;
out$buffer2 (3) = HA;
out$buffer2 (2) = ES;
out$buffer2 (1) = PE;
out$buffer2 (0) = BLANK;
out$buffer2 (1) = PE;
out$buffer2 (2) = BLANK;
out$buffer2 (3) = HA;
out$buffer2 (2) = ES;
out$buffer2 (1) = PE;
out$buffer2 (0) = BLANK;
out$buffer2 (1) = PE;
out$buffer2 (2) = BLANK;
out$buffer2 (3) = HA;
out$buffer2 (2) = ES;
out$buffer2 (1) = PE;
out$buffer2 (0) = BLANK;
do while(Teller < 16);
do whi1e(Teller < 16);
do Refresh();
    /* Als de ververs timer afgeopen is */
    /* If DISPLAYRefresh=TRUE then */
    /* If DISPLAYRefresh=FALSE do */
        /* Laad de watchdog timer */
        /* Sturen nieuwe informatie naar display */
        /* Call Refresh();
        /* Teller = Teller + 1;
        /* T3 = 0;
    end;
end;
/*
***** END INTRO *****
*/

```

```

        Out$Buffer1 (3) = BLANK; /* Als MS digit is 0 BLANK'en! */
        else
            Out$Buffer1 (3) = BLANK; /* Als MS digit is 0 BLANK'en! */

        if Go_2 then
            do!
                if Display_2 = FALSE and Debounce_2 = false then
                    do!
                        if Debounce_2 = FALSE then
                            do!
                                Out$Buffer2 (0) = BLANK;
                                Out$Buffer2 (1) = BLANK;
                                Out$Buffer2 (2) = BCD2Seg (shx(Time2,8) and 0fh);
                                Out$Buffer2 (3) = BCD2Seg (shx(Time2,12) and 0fh);
                                Out$Buffer2 (4) = BCD2Seg (shx(D_Time2,4) and 0fh);
                                Out$Buffer2 (5) = BCD2Seg (shx(D_Time2,8) and 0fh);
                                Out$Buffer2 (6) = BCD2Seg (shx(D_Time2,12) and 0fh);
                                Out$Buffer2 (7) = BCD2Seg (shx(D_Time2,16) and 0fh);
                                Out$Buffer2 (8) = BCD2Seg (shx(D_Time2,20) and 0fh);
                                Out$Buffer2 (9) = BCD2Seg (shx(D_Time2,24) and 0fh);
                                Out$Buffer2 (A) = BCD2Seg (shx(D_Time2,28) and 0fh);
                                Out$Buffer2 (B) = BCD2Seg (shx(D_Time2,32) and 0fh);
                                Out$Buffer2 (C) = BCD2Seg (shx(D_Time2,36) and 0fh);
                                Out$Buffer2 (D) = BCD2Seg (shx(D_Time2,40) and 0fh);
                                Out$Buffer2 (E) = BCD2Seg (shx(D_Time2,44) and 0fh);
                                Out$Buffer2 (F) = BCD2Seg (shx(D_Time2,48) and 0fh);

                                if Debounce_2 = TRUE then
                                    do!
                                        if Debounce_1 = TRUE then
                                            do!
                                                if Debounce_1 = TRUE then
                                                    if Debounce_1 = TRUE then
                                                        Lamp1=RED;
                                                        Lamp2=GREEN;
                                                    else
                                                        Lamp1=GREEN;
                                                        Lamp2=RED;
                                                else
                                                    if Debounce_1 = TRUE then
                                                        Lamp1=RED;
                                                        Lamp2=GREEN;
                                                    else
                                                        Lamp1=GREEN;
                                                        Lamp2=RED;
                                                end;
                                            end;
                                        end;
                                    end;
                                end;
                            end;
                        end;
                    end;
                end;
            end;
        end;
    end;
}

```

END Glijbaan

end;