```
//////////////////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////////////////////////
//
//  Title:
//  Revision:
//  Rev. date:
//  Description:styrenhet ....
//  Language:       CCS-C
//  Author:         Matte Sundberg
//  Date:           2002-08-20
//
//////////////////////////////////////////////////////////////////////////
/*
SEP25b

1.07 fungerar med nyare compilator igen, ändrade nivåer för
temperaturkontrollen av displayvärme, går ned på "halvljus"
vid ca. +35, slocknar vid ca. +38
1.08 har aktiverat möjligheten att sätta lcd-utgången (plint 8)

SEP25b-TME

1.00 första versionen av slangmätning med enbart displaymodul

SEP25c

Grafisk display
1.00 (bygger på SEP25b 1.08)Används som teckendisplay
1.01 backlight kan släckas och gå ner på "halvljus" med knapp 1 & 4
25c.1.1.02 fix av rev.display vid kallstart

SEP25d

Grafisk display
1.02 (bygger på SEP25d 1.02)Används som teckendisplay
1.03 --- 2008-09-02 flyttar opto1 ett steg till vänster i byte[0]
1.04 --- 2009-01-31 EE init

*/
//////////////////////////////////////////////////////////////////////////

// 1.00 bygger på sep25b 1.08
/*
1.01 --- 2008-01-31
ökar på delay före displysinitiering
från 10 till 500 mS.
sänker LCD_REF_VOLT_REG ett snäpp till 13

1.02 --- 2008-02-20
pillar lite med lcdinit i drivarn, slipper reverserad displys
vid kallstart, trorja iaf. Tar bort piezo-läten. Pillar lite delayer i init().

1.03 --- 2008-09-02
```

```
flyttar opto1 ett steg till vänster i byte[0]

1.04 --- 2009-01-31
EE init

1.05 --- 2011-02-19
nytt mönsterkort me 1259 bussmöjligheter, div io:s omflyttad
slutar löda 6002:an och ändrar A0 och A1 till utgångar

1.06 --- 2011-05-04
ändrar RB6 å 7 till utgångar

1.07 --- 2011-05-11
fixar opto2 som används till varvtal

1.08 --- 2011-11-17
släck bakgrundsbelysningen om en knapp är intryckt vid strömsättning

*/
//
//
//
//*****************************************************************************

#include <18f2620.h>

//#use  delay(clock=3686400)
#use        delay(clock=29491200)           // Xtal
//#fuses    HS,WDT,NOPROTECT,BROWNOUT,NOLVP,PUT//,DEBUG
#fuses      H4,WDT,PROTECT,BROWNOUT, PUT,NOLVP  // Fuses
#use    rs232(baud=19200, xmit=PIN_C6, rcv=PIN_C7, ERRORS)

#use fast_io(A)
#use fast_io(B)
#use fast_io(C)
/************************************/

#include "font.h"
#include "S6B0724_9v.c"


// Konstanter --
#define ROW_1   5
#define ROW_2 19
#define ROW_3 33
#define ROW_4 47
///:~

//
#byte       PORTA   = 0x0F80
#byte       PORTB   = 0x0F81
#byte       PORTC   = 0x0F82
#byte       PORTE   = 0x0F84
#byte       TIMER2  = 0xFCA
```

```
#byte       TXSTA   = 0xFAC
#byte       RCSTA   = 0xFAB

#bit        TMR2ON  = TIMER2.2
#bit        TRMT    = TXSTA.1
#bit        CREN    = RCSTA.4

#bit        Dir485  = PORTC.4

#bit        Button1 = PORTA.4
#bit        Button2 = PORTC.0
#bit        Button3 = PORTB.1
#bit        Button4 = PORTB.0

#bit        Backlight = PORTA.2
//#bit      Piezo = PORTB.4
#bit        Output1         =   PORTA.3

#bit        Opto1           =   PORTB.5
#bit        Opto2           =   PORTB.4
///:~
#ROM 0xF00000 = {
                0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
                0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
                0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
                0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
                }

// Varabler --
int1 displayupdate, SampReport;

int8 sendbyte[8], comstate, lcdbufptr, SampInterval;

int16   rpm, rpmCount;

char lcdbuffer[18];
///:~

// Funktions deklarationer --
void Init();
void main();
void UpdateDisplay(int8 line);
void GoForSplit(int16 value, int8 index);

// interupt funktioner -
void SerialIn();
void SerialOut();
void ComDelay();
void tick();
void PulsCounter();
///:~


void Init()
```

```
{
//  set_tris_a(0x13);   // 00010011
    set_tris_a(0x10);   // 00010000
    set_tris_b(0x33);   // 00110011
    set_tris_c(0x81);   // 10000001

    output_a(0);
    output_b(0);
    output_c(0);
    delay_ms(40);

//  Piezo = 0;
//  Output1 = 0;

    Backlight = 1;
    delay_ms(200);
    cog_Init();             // sÃ¤tt igÃ¥ng displysen
    delay_ms(50);   // vila lite

    lcdbufptr = 0;
    displayupdate = false;
    Dir485 = 0;             // Ã¤ndrar kommunikations riktningen inÃ¥t
    CREN = 1;                   // trycker pÃ¥ mottagarn

    // sÃ¤tt igÃ¥ng en massa interupt
//  setup_spi(SPI_MASTER | SPI_L_TO_H | SPI_CLK_DIV_16);
    setup_timer_1(T1_INTERNAL|T1_DIV_BY_8);
    setup_timer_2(T2_DIV_BY_4, 0xc0, 16);
    enable_interrupts(INT_TIMER1);
    enable_interrupts(INT_TIMER2);
    enable_interrupts(INT_RB);
    enable_interrupts(GLOBAL);


    delay_ms(100);  // vila lite
    if(!Button1 || !Button2 || !Button3 || !Button4)
        Backlight = 0;
}

#zero_ram
void main()
{
    int8 i;

    Init();


    enable_interrupts(INT_RDA);

    while(TRUE)
    {
        if(displayupdate)
        {
            displayupdate = false;
```

```
            sendbyte[0]=(((128 + Opto1*64 + Button1*32 + Button2*16 + Button3*8 +
Button4*4)^255)>>2);
            sendbyte[7]=0;
            for(i = 0; i <= 6; i++)
            {
                sendbyte[7] += sendbyte[i];
            }
            UpdateDisplay(((lcdbuffer[1]-3)/16)+1);
        if (SampReport)
            {
            SampReport  = FALSE;
            GoForSplit(rpm,1);
            }
        }


    }
}


/*
    Updaterar en rad pÃ¥ displysen
*/
void UpdateDisplay(int8 line)
{
  int8 i;
  int8 y;
  int1 color=1;
    char text[17];  // texten att skriva ut

    switch(line)
    {
    case 1:
        y = ROW_1;
    break;

    case 2:
        y = ROW_2;
    break;

    case 3:
        y = ROW_3;
//      cog_Text(0,y,FONT_DEFAULT,text2);
//      return;
    break;

    case 4:
        y = ROW_4;
//      cog_Text(0,y,FONT_DEFAULT,text3);
//      return;
    break;

    default:
```

```
//      y = ROW_1;
//      cog_Text(0,y,FONT_DEFAULT,text4);
//      return;
    break;
    }

    // kopierar de 16 sista tecknen i lcdbuffer
  for(i = 0; i < 16; i++)
    {
    text[i] = lcdbuffer[i+2];
    if(y == ROW_4)
        {
            if(text[i] == 126 || text[i] == 127 || text[i] == 6)
        color=0;
        }
    else
        color=1;
    }
  text[16] = 0x00;  // null
  if(color == 0)
  {
      text[0] = '{'; // special för att dela upp fjärde raden
      text[3] = '}'; // till 4 st "knappar"
      text[4] = '{';
      text[7] = '}';
      text[8] = '{';
      text[11] = '}';
      text[12] = '{';
      text[15] = '}';
  }

  // skriver ut pÃ¥ displayen
    cog_Text(0,y,text,1,color);
    cog_Update(y, y+12);
}



/************************************/
/*  Interupts --                                            */
//*************************************************************************
//  Split Word to bytes
//*************************************************************************

//#separate
void GoForSplit(int16 value, int8 index)
{
    SendByte[index] = make8(value,1);
    SendByte[index+1] = make8(value,0);
}
//*************************************************************************

#INT_TIMER1
void tick()
```

```
{
    set_timer1(-9216);                  // Sets timer to interrupt at 10 mS
        interval
    restart_wdt();          // Ã¥ssÃ¥ startar vi om wdtÃ¤n

    if (++SampInterval == 100)
        {
        SampInterval = 0;
        SampReport = TRUE;
        rpm = rpmCount;
        rpmCount = 0;
        }
}


/*
    Utklockning av tecken delay
*/
#INT_TIMER2
void ComDelay() // 1 mS.
{
    TMR2ON=0;           // stÃ¤ng av timer 2

    if(CREN)            // Ã¤r mottagarn pÃ¥?
    {
        CREN = 0;                                       // stÃ¤ng av mottagarn
        Dir485 = 1;                                   // Ã¤ndrar kommunikations
            riktningen utÃ¥t
        putc(sendbyte[0]);                  // skicka fÃ¶rsta tecknet
        comstate=1;
        enable_interrupts(INT_TBE); // tryck pÃ¥ skicka interuptet
    }

    else                    // njee den va av
    {
        Dir485 = 0;     // Ã¤ndrar kommunikations riktningen inÃ¥t
        CREN = 1;           // tryck pÃ¥Ã¥Ã¥Ã¥Ã¥Ã¥Ã¥Ã¥Ã¥Ã¥Ã¥Ã¥Ã¥Ã¥Ã¥Ã¥Ã¥Ã¥ mottagarn
    }
}



/*
    Serial in interupt
*/
#INT_RDA
void SerialIn()
{
//  static int8 lcdbufptr;  // tecken rÃ¤knare (ok tydligen funkar inte static nÃ¥
    bra i denna kompilator, anvÃ¤nder en global variabel istï¿½llet)
    char inchar;                            // tecknet som ligger pÃ¥ porten

    inchar = getc();                    // hÃ¤mta tecknet
    switch(inchar)
```

```
    {
    case 1:                     // start tecken
        lcdbufptr = 0;  // nollstÃ¤ll tecken rÃ¤knarn
    break;

    case 2:                     // stop tecken
        if (lcdbufptr == 18 && (lcdbuffer[0] & 31) == 4)
            TMR2ON=1;           // pÃ¥ mÃ¤ timer 2
    break;

    default:                    // vanligt tecken
        if(lcdbufptr < 18)
            lcdbuffer[lcdbufptr++] = inchar;    // lÃ¤gg in tecknet i en buffer
    break;
    }
}



/*
    Serial ut interupt
*/
#INT_TBE
void SerialOut()
{
    switch(comstate)
    {
    case 1:
        putc(sendbyte[1]);
        comstate++;
    break;

    case 2:
        putc(sendbyte[2]);
        comstate++;
    break;

    case 3:
        putc(sendbyte[3]);
        comstate++;
    break;

    case 4:
        putc(sendbyte[4]);
        comstate++;
    break;

    case 5:
        putc(sendbyte[5]);
        comstate++;
    break;

    case 6:
        putc(sendbyte[6]);
        comstate++;
```

```
    break;

    case 7:
        disable_interrupts(INT_TBE);    // stÃ¤ng av skicka interuptet
        putc(sendbyte[7]);              // skicka sista tecket
        displayupdate=true;             // updatera displysen
        TMR2ON=1;                       // pÃ¥ mÃ¤ timer 2
    break;
    }
}
/***********************************/
#INT_RB
void PulsCounter()
    {
        if(!Opto2)
        rpmCount++;
    }
```