```
/*
  Phasenkomperator
  (C)2005 Hardy Lau

  Microchip dsPIC30F4013
  Interner RC-Oszillator mit 8 MHz
*/

#include <p30f4013.h>
#include <math.h>

#define E  _RF6
#define RW _RF5
#define RS _RF4

signed char phasen_vorzeichen = 0; //Vorzeichen der Phase -1,+1,0=Unbestimmt
double vorzeichen_at_phase = 0; //Betrag der Phase bei der das Vorzeichen
bestimmt wurde
double a, ph; //Amplitude in dB, Phase in Grad

//------------------------------------------------------------------------------
-

void delay( unsigned int ms) {
  unsigned int x, y;
  for( x = ms; x; x--) {
    for( y = 200; y; y--) E = 0;
  }
}

//------------------------------------------------------------------------------
-

unsigned int read_lcd( unsigned char rs) {
  unsigned int temp;
  E = 0;
  if( rs == 0) {
    RS = 0;
    RS = 0;
  } else {
    RS = 1;
    RS = 1;
  }
  TRISB != 0b0001111000000000; //RB9..RB12 4Bit LCD-Datenleitung (lesen)
  TRISD != 0b0000000000001111; //RD0..RD3 4Bit LCD-Datenleitung (lesen)
  RW = 1; //Read
  RW = 1;
  E = 0;
  E = 0;
  E = 1; //Enable fuer mindestens 450ns
  E = 1;
  E = 1;
  temp = ((PORTB & 0b0001111000000000) >> 5) | (PORTD & 0b0000000000001111);
  E = 0;  //Nicht enable fuer mindestens 450ns
  E = 0;
  TRISB &= 0b1110000111111111; //RB9..RB12 Ausgang
  TRISD &= 0b1111111111110000; //RD0..RD3 Ausgang
  E = 0;
  return( temp);
}

//------------------------------------------------------------------------------
-

void write_lcd( unsigned int data, unsigned char rs) {
  read_lcd( 0); // LCD-BUSY Abfragen
  E = 0;
  if( rs == 0) {
    RS = 0;
    RS = 0;
  } else {
    RS = 1;
    RS = 1;
  }
  RW = 0; //Write
  RW = 0;
  PORTB = (PORTB & 0b1110000111111111) | (data & 0b0000000011110000) << 5;
  PORTD = (PORTD & 0b1111111111110000) | (data & 0b0000000000001111);
  E = 0;
  E = 0;
  E = 1; //Enable fuer mindestens 450ns
  E = 1;
  E = 1;
  E = 0;
  E = 0;
  E = 0;
}

//------------------------------------------------------------------------------
-

void init_lcd() {
  unsigned int j;
  E = 0;
  TRISB &= 0b1110000111111111; //RB9..RB12 4Bit LCD-Datenleitung
  TRISD &= 0b1111111111110000; //RD0..RD3 4Bit LCD-Datenleitung
  TRISF &= 0b1111111110001111; //RF4 = RS, RF5 = R/W, RF6 = Enable
  write_lcd( 0b0000000000111000, 0); //Function set, 8 Bit, 2 Zeilen
  write_lcd( 0b0000000000001111, 0); //Display on
  write_lcd( 0b0000000000000001, 0); //Clear Display
  delay(5);
  write_lcd( 0b0000000000000110, 0); //Coursor Autoincrement
}

//------------------------------------------------------------------------------
-

void init_lcd_neue_zeichen() {
  //Delta chr(3)
  write_lcd( 0b01011000, 0);    //CG-RAM Address Set 24
  write_lcd( 0b00000000, 1);
  write_lcd( 0b01011001, 0);    //CG-RAM Address Set 25
  write_lcd( 0b00000000, 1);
  write_lcd( 0b01011010, 0);    //CG-RAM Address Set 26
  write_lcd( 0b00000000, 1);
  write_lcd( 0b01011011, 0);    //CG-RAM Address Set 27
  write_lcd( 0b00000100, 1);
  write_lcd( 0b01011100, 0);    //CG-RAM Address Set 28
  write_lcd( 0b00001010, 1);
  write_lcd( 0b01011101, 0);    //CG-RAM Address Set 29
  write_lcd( 0b00010001, 1);
  write_lcd( 0b01011110, 0);    //CG-RAM Address Set 30
  write_lcd( 0b00011111, 1);
  write_lcd( 0b01011111, 0);    //CG-RAM Address Set 31
  write_lcd( 0b00000000, 1);
  //Phi chr(2)
  write_lcd( 0b01010000, 0);    //CG-RAM Address Set 16
  write_lcd( 0b00010010, 1);
  write_lcd( 0b01010001, 0);    //CG-RAM Address Set 17
  write_lcd( 0b00010101, 1);
  write_lcd( 0b01010010, 0);    //CG-RAM Address Set 18
  write_lcd( 0b00010101, 1);
  write_lcd( 0b01010011, 0);    //CG-RAM Address Set 19
  write_lcd( 0b00010101, 1);
  write_lcd( 0b01010100, 0);    //CG-RAM Address Set 20
  write_lcd( 0b00001110, 1);
  write_lcd( 0b01010101, 0);    //CG-RAM Address Set 21
  write_lcd( 0b00000100, 1);
  write_lcd( 0b01010110, 0);    //CG-RAM Address Set 22
  write_lcd( 0b00000100, 1);
  write_lcd( 0b01010111, 0);    //CG-RAM Address Set 23
  write_lcd( 0b00000000, 1);
  //Plus-Minus chr(1)
  write_lcd( 0b01001000, 0);    //CG-RAM Address Set 8
  write_lcd( 0b00000100, 1);
  write_lcd( 0b01001001, 0);    //CG-RAM Address Set 9
  write_lcd( 0b00000100, 1);
  write_lcd( 0b01001010, 0);    //CG-RAM Address Set 10
  write_lcd( 0b00011111, 1);
  write_lcd( 0b01001011, 0);    //CG-RAM Address Set 11
  write_lcd( 0b00000100, 1);
  write_lcd( 0b01001100, 0);    //CG-RAM Address Set 12
  write_lcd( 0b00000100, 1);
  write_lcd( 0b01001101, 0);    //CG-RAM Address Set 13
  write_lcd( 0b00000000, 1);
  write_lcd( 0b01001110, 0);    //CG-RAM Address Set 14
  write_lcd( 0b00011111, 1);
  write_lcd( 0b01001111, 0);    //CG-RAM Address Set 15
  write_lcd( 0b00000000, 1);

  write_lcd( 0b10000000, 0);    //DD-RAM Adress Set 0
}

//------------------------------------------------------------------------------
-

void write_lcd_string( char *string) {
  while( *string) {
    write_lcd( *string++, 1);
  }
}

//------------------------------------------------------------------------------
-

void copyright() {
  write_lcd_string( "(c)2005 DL1GLH");
  write_lcd( 0b11000000, 0); //Gehe Anfang Zweite Zeile
  write_lcd_string( "Phasenkomperator");
  delay(2000);
  write_lcd( 0b11000000, 0); //Gehe Anfang Zweite Zeile
  write_lcd_string( "V0.97 22.07.2005");
  delay(2000);
}

//------------------------------------------------------------------------------
-

void init_12bit_ad() {
  TRISB !=  0b0000000000001111; //AN0..AN3 (RB0..RB3) als Input
  ADPCFG = 0b1111111111110000; //AN0..AN3 A/D-Port
}

//------------------------------------------------------------------------------
-

unsigned int ad_read( unsigned int channel) {
  unsigned long wert = 0;
  unsigned int i;
  ADCON1 = 0b0000000011100000;
  ADCON2 = 0b0000000000000000;
  ADCON3 = 0b0000001100000111;
  ADCHS =  channel;
  ADCSSL = 0;
  ADCON1bits.ADON = 1; // A/D-Einschalten
  for( i = 4096; i; i--) {
    ADCON1bits.SAMP = 1; // Start sampling
    while(!ADCON1bits.DONE);
    wert += ADCBUF0;
  }
  ADCON1bits.ADON = 0; // A/D-Ausschalten
  return( wert >> 12);
}

//------------------------------------------------------------------------------
-

void a_ph_read(unsigned char mit_korrektur) {
  double spref, spampl, sppha;
  double vh;
  double x[15] = { 0.0, 0.2, 1.6, 4.5, 5.6, 6.6, 8.7, 13.5, 17.4, 27.0, 37.0,
48.0, 90.0, 172.7, 180.0 };
  double y[15] = { 1.0, 7.0, 3.62, 1.95, 1.66, 1.51, 1.487, 1.27, 1.21, 1.126,
1.08, 1.044, 1.001, 0.985, 1.016 };
  unsigned int  i;

  spref = ad_read(1);
  sppha = ad_read(2);
  spampl = ad_read(0);

  //Amplitudenverhaeltnis
  vh = spampl - (spref * 0.5);
  vh = (vh * 60.0) / spref;
  a = vh;

  //Korrektur Amplitude
  if( mit_korrektur) {
    a = a - 0.06;
  }

  //Phase
  vh = 180.0 - ((180.0 * sppha) / spref);
  ph = vh;

  //Korrektur der Phase
  for( i = 13; i >= 0; i--) {
    if( ph >= x[i]) {
      vh = (y[i+1] - y[i]) / (x[i+1] - x[i]) * (ph - x[i]) + y[i];
      break;
    }
  }

  if( mit_korrektur) {
    ph = ph * vh;
  }

}

//------------------------------------------------------------------------------
-

void vers_spannung() {
  double spannung;

  write_lcd( 0b0000000000000001, 0); //Clear Display
  delay(10);
  write_lcd_string( "Batterie:");
  write_lcd( 0b11000000, 0); //Gehe Anfang Zweite Zeile

  spannung = (double)ad_read(3) * 0.005718;
  if( spannung > 10) {
    write_lcd( ((unsigned int)spannung/10.0) + '0', 1);
  }
  write_lcd( (unsigned int)spannung%10 + '0', 1);
  write_lcd( ',', 1);
  write_lcd( (unsigned int)(spannung*10.0)%10 + '0', 1);
  write_lcd( (unsigned int)(spannung*100.0)%10 + '0', 1);
  write_lcd_string( " Volt      ");
  delay( 3000);
}
```

```
//------------------------------------------------------------------------------
-

check_ad8302_ref() {
  double spannung;

  write_lcd( 0b0000000000000001, 0); //Clear Display
  delay(10);
  write_lcd_string( "AD8302-Referenz");
  write_lcd( 0b11000000, 0); //Gehe Anfang Zweite Zeile

  spannung = (double)ad_read(1) * 0.001;
  if( (spannung > 1.95) || (spannung < 1.75)) {
    write_lcd_string( "ERROR: ");
    write_lcd( (unsigned int)spannung%10 + '0', 1);
    write_lcd( ',', 1);
    write_lcd( (unsigned int)(spannung*10.0)%10 + '0', 1);
    write_lcd( (unsigned int)(spannung*100.0)%10 + '0', 1);
    write_lcd( (unsigned int)(spannung*1000.0)%10 + '0', 1);
    write_lcd_string( " V");
    while(1);
  }
  else {
    write_lcd_string( "OK");
    delay(1500);
  }
}

//------------------------------------------------------------------------------
-

void init_tasten() {
  TRISB !=  0b0000000000110000; //RB4,RB5 als Input
}

//------------------------------------------------------------------------------
-

unsigned char lese_tasten() {
  return( ~((PORTB & 0b0000000000110000) >> 4) & 3);
}

//------------------------------------------------------------------------------
-

init_relais() {
  TRISC &= 0b1101111111111111; //RC13 Ausgang
  PORTCbits.RC13 = 0;  //Relais = 0;
}

//------------------------------------------------------------------------------
-

void relais( unsigned char zustand) {
  if( zustand == 1)
    PORTCbits.RC13 = 1;
  else
    PORTCbits.RC13 = 0;
}

//------------------------------------------------------------------------------
-

void messe_phase_amplitude() {
  //Amplitudenverhaeltnis
  a_ph_read(1);
  write_lcd( 0b10000000, 0); //Gehe Anfang Erste Zeile
  write_lcd_string( "\003A= ");
  if( a < 0) {
    write_lcd_string( "-");
  } else {
    write_lcd_string( "+");
  }
  if((unsigned int)fabs(a) >= 10) {
    write_lcd( (unsigned int)fabs(a) / 10 + '0', 1);
  }
  write_lcd( (unsigned int)(fabs(a))%10 + '0', 1);
  write_lcd( ',', 1);
  write_lcd( (unsigned int)(fabs(a)*10.0)%10 + '0', 1);
  write_lcd( (unsigned int)(fabs(a)*100.0)%10 + '0', 1);
  //write_lcd( (unsigned int)(fabs(a)*1000.0)%10 + '0', 1);
  write_lcd_string( " dB          ");

  //Phase
  if( fabs( vorzeichen_at_phase - ph) > 3) phasen_vorzeichen = 0;

  write_lcd( 0b11000000, 0); //Gehe Anfang Zweite Zeile
  write_lcd_string( "\003\002= ");
  switch( phasen_vorzeichen) {
    case 0:  write_lcd_string( "\001"); //Unbekannt also Plusminus
             break;
    case -1: write_lcd( '-', 1); //Bekannt Minus
             break;
    case 1:  write_lcd( '+', 1); //Bekannt Plus
             break;
  }

  if((unsigned int)ph >= 100) {
    write_lcd( ((unsigned int)ph / 100)%10 + '0', 1);
  }
  if((unsigned int)ph >= 10) {
    write_lcd( ((unsigned int)ph / 10)%10 + '0', 1);
  }

  write_lcd( (unsigned int)ph%10 + '0', 1);
  write_lcd( ',', 1);
  write_lcd( (unsigned int)(ph*10.0)%10 + '0', 1);
  //write_lcd( (unsigned int)(ph*100.0)%10 + '0', 1);
  write_lcd( 0xdf, 1);
  write_lcd_string( "       ");
}

//------------------------------------------------------------------------------
-

void bestimme_vorzeichen() {
  double a_alt, ph_alt;

  a_ph_read(1);

  //Phase
  vorzeichen_at_phase = ph;
  ph_alt = ph;

  write_lcd( 0b10000000, 0); //Gehe Anfang Erste Zeile

  write_lcd_string( "Phasenvorzeichen");
  write_lcd( 0b11000000, 0); //Gehe Anfang Zweite Zeile

  //Amplitudenverhaeltnis
  if( fabs( a) > 20.0) { //Amplitudendifferenz zu gross (>20dB)
    write_lcd_string( "nicht bestimmbar");
    phasen_vorzeichen = 0;
    delay(1000);
    return;
  }

  write_lcd_string( "wird bestimmt...");
  //Phase mit Verlaengerung der Referenzleitung um 92cm
  relais(1); //Umwegeleitung einschalten
  delay(100);
  a_ph_read(1);
  relais(0);

  if( ph < ph_alt) {
    phasen_vorzeichen = -1;
  } else {
    phasen_vorzeichen = 1;
  }
}

//------------------------------------------------------------------------------
-

void bestimme_frequenz() {
  double ph_alt, freq;

  a_ph_read(0);

  write_lcd( 0b10000000, 0); //Gehe Anfang Erste Zeile
  write_lcd_string( "Frequenz        ");
  write_lcd( 0b11000000, 0); //Gehe Anfang Zweite Zeile

  //Amplitudenverhaeltnis
  if( fabs( a) > 20.0) { //Amplitudendifferenz zu gross (>20dB)
    write_lcd_string( "nicht bestimmbar");
    delay(1000);
    return;
  }

  //Phase ohne verlaengerung
  ph_alt = ph;

  //Phase mit Verlaengerung der Referenzleitung um 92cm
  relais(1); //Umwegeleitung einschalten
  delay(100);
  a_ph_read(0);

  freq = fabs(ph_alt - ph) * 562536.0;

  if( freq >= 10000000.0) {
    write_lcd( (unsigned int)(freq / 10000000)%10 + '0', 1);
  }
  write_lcd( (unsigned int)(freq / 1000000)%10 + '0', 1);
  write_lcd_string( ",");
  write_lcd( (unsigned int)(freq / 100000)%10 + '0', 1);
  write_lcd( (unsigned int)(freq / 10000)%10 + '0', 1);
  write_lcd_string( " MHz        ");
  relais(0);
  delay(1000);
}

//------------------------------------------------------------------------------
-

void funktion_3() {
  write_lcd( 0b10000000, 0); //Gehe Anfang Erste Zeile

  write_lcd_string( "Funktion 3      ");
  write_lcd( 0b11000000, 0); //Gehe Anfang Zweite Zeile
  write_lcd_string( "noch unbekannt  ");
  delay(1000);
}

//------------------------------------------------------------------------------
-
```

```
main()
{
 init_lcd();
 init_lcd_neue_zeichen();
 copyright();
 init_12bit_ad();
 vers_spannung();
 check_ad8302_ref();
 init_tasten();
 init_relais();

 while(1) {
   switch( lese_tasten()) {
     case 0: messe_phase_amplitude();
             break;
     case 1: bestimme_vorzeichen();
             break;
     case 2: bestimme_frequenz();
             break;
     case 3: vers_spannung();
             break;
   }
   delay(100);
 }
}

//--------------------------------------------------------------------------------
-
```