```
$Regfile = "m8def.dat"                  'M8-Deklarationen
$Crystal = 1000000                      'interner RC-Oszillator reicht

CONFIG LCDPIN = PIN , DB4 = PORTC.2 , DB5 = PORTC.3 , DB6 = PORTC.4 , DB7 = PORTC.5 , E = PORTB.1 , RS =
PORTB.2
Config LCD = 16 * 2

dim retvalue as byte
dim zahlstr as string * 5

dim frequ as Word : frequ = 961         'Programm-Default-Werte
dim stereo as Byte : stereo = 1         'alles Ein
dim driver as Byte : driver = 1
dim licht as Byte : licht = 1

dim dummy_ee as ERAM Long               'Werte fürs EEPROM
dim frequ_ee as ERAM Word
dim stereo_ee as ERAM Byte
dim driver_ee as ERAM Byte
dim licht_ee as ERAM Byte

dim menu1(8) as string * 8              'Menüwerte
menu1(1) = "Frequenz"
menu1(2) = "Stereo  "
menu1(3) = "Treiber "
menu1(4) = "Licht   "
menu1(5) = "Sichern "
menu1(6) = "AbstimmU"
menu1(7) = "AntenneU"
menu1(8) = "zurueck "

dim menu2(2) as string * 3
menu2(1) = "Ein"
menu2(2) = "Aus"


ddrd.2 = 0                              'drehimpuls 1 = INT0
portd.2 = 1                             'Pullup
ddrb.0 = 0                              'drehimpuls 2
portb.0 = 1                             'Pullup
ddrd.3 = 0                              'Set-Button   = INT1
portd.3 = 1                             'Pull-Up
setbutton alias pind.3                  'Menübestätigungs-Knopf

dim drehgeber as word                   'wird von ISR (INT0) verändert
on INT0 OnInt0
mcucr.isc00 = 1                         'INT0 bei FALLING + RISING
mcucr.isc01 = 0
gicr.INT0 = 1


config INT1 = LOW LEVEL                 'nur zum Aufwachen aus Sleep-Modus
on INT1 OnInt1


mcucr.sm1 = 1
mcucr.se = 1                            'Sleep-Modus=Power-Down


ddrd.4 = 1                              'PD4= "Licht"

ddrd.5 = 1                              'PD5= Ausgang zum BH1415F (Data)
datas alias portd.5
ddrd.6 = 1                              'PD6= Clock
clock alias portd.6
ddrd.7 = 1                              'PD7= Chip Enable
chipena alias portd.7


admux.refs0 = 1                         'ADC-Config:
admux.refs1 = 1                         'interne Referenz
adcsr.adps0 = 1
adcsr.adps1 = 1
adcsr.adps2 = 1                         'Prescaler=128
adcsr.aden = 1                          'ADC einschalten
ddrc.0 = 0                              'als Eingänge schalten (ADC0-1)
ddrc.1 = 0


sreg.7 = 1                              'alle Interrupts freigeben


declare Function menu(menuitems() as string , byval maxentry as byte , byval default as Byte ) as Byte
declare Function frequenz(byval default as Word) as Word
declare Sub toBH1415F
```

```
declare Sub showADC(byval anz_mod as Byte)


if frequ_ee <> &HFFFF then              'abgespeicherte Default-Werte
   frequ = frequ_ee
   stereo = stereo_ee
   driver = driver_ee
   licht = licht_ee
end IF

portd.4 = not licht.0                   'Display-Licht; LOW-Aktiv
toBH1415F                               ' schiebt frequ, stereo u. driver i.d. Chip

cls
locate 2 , 1
lcd " (c)  DiLi-Soft"

do
   cursor off
   locate 1 , 1 : lcd menu1(1)
   locate 1 , 10 : zahlstr = str(frequ) : zahlstr = format(zahlstr , " 00.0") : lcd zahlstr ; spc(2)

   enable INT1
   !sleep                               'nur Druck (LOW LEVEL-Int) weckt auf
   disable INT1

   waitms 500

   locate 1 , 10 : lcd spc(6)
   cursor on blink

   do
      locate 1 , 1
      retvalue = menu(menu1(1) , 8 , 1)
      locate 1 , 10
      select case retvalue
         case 1 : frequ = frequenz(frequ)
                  exit DO

         case 2 : stereo = menu(menu2(1) , 2 , stereo)
                  exit DO

         case 3 : driver = menu(menu2(1) , 2 , driver)
                  exit DO

         case 4 : licht = menu(menu2(1) , 2 , licht)
                  portd.4 = not licht.0 'Low-Aktiv
                  goto loopend

         case 5 : sreg.7 = 0            'zum Abspeichern INTs abschalten
                  frequ_ee = frequ
                  stereo_ee = stereo
                  driver_ee = driver
                  licht_ee = licht
                  sreg.7 = 1
                  goto loopend

         case 6 : admux.0 = 0           'ADC-Kanal 0
                  call showADC(0)
                  goto loopend2

         case 7 : admux.0 = 1           'ADC-Kanal 1
                  call showADC(1)
                  goto loopend2

         case 8 : goto loopend2
      end select
   loop

   call toBH1415F
   loopend:
   locate 1 , 10 : lcd "ok" ; spc(3)
   wait 1
   loopend2:
loop


end



Function menu(menuitems() as string , byval maxentry as byte , byval default as Byte ) as Byte
   local curent as byte
   local mlen as byte
   local drehgeber_alt as word
```

```
   local lauf as Byte

   drehgeber = &H8000
   drehgeber_alt = &H8000
   mlen = len(menuitems(1))
   lcd menuitems(default);
   curent = default

   do
      if setbutton = 0 then
         menu = curent
         waitms 500
         exit FUNCTION
      end IF

      if drehgeber <> drehgeber_alt then
         if drehgeber > drehgeber_alt then
            incr curent
            if curent > maxentry then
               curent = 1
            end IF
         end if

         if drehgeber < drehgeber_alt then
            decr curent
            if curent = 0 then
               curent = maxentry
            end IF
         end if

         for lauf = 1 to mlen
            shiftcursor left
         next lauf
         lcd menuitems(curent);
         drehgeber_alt = drehgeber
      end IF
   loop
end FUNCTION



Function frequenz(byval default as word) as Word
   local s as String * 5
   local drehgeber_alt as word
   local lauf as Byte

   drehgeber = default
   drehgeber_alt = drehgeber
   s = str(drehgeber)
   s = format(s , " 00.0" )
   lcd s
   do
      if drehgeber <> drehgeber_alt then

         if drehgeber > 1080 then
            drehgeber = 875
         end IF

         if drehgeber < 875 then
            drehgeber = 1080
         end IF

         drehgeber_alt = drehgeber

         for lauf = 1 to 5
            shiftcursor left
         next lauf
         s = str(drehgeber_alt)
         s = format(s , " 00.0" )
         lcd s

      end IF

      if setbutton = 0 then
         frequenz = drehgeber_alt
         waitms 500
         exit FUNCTION
      end IF
   loop
end FUNCTION



Sub toBH1415F
   local seriellword as Word
```

```
   local lauf as Byte
   seriellword = frequ
   seriellword.11 = stereo.0
   seriellword.14 = driver.0
   'locate 2 , 1
   'lcd bin(seriellword )
   chipena = 1                          'Chip selektieren
   for lauf = 1 to 16
      datas = seriellword.0             'Datenbit 0 anlegen
      clock = 1                         'und übergeben
      waitus 250
      clock = 0
      shift seriellword , RIGHT         'nächstes Bit in Position 0 schieben
   next lauf                            'Chip deselektieren
   chipena = 0
end SUB



Sub showADC(byval anz_mod as Byte)
   local adc_h as word
   local adc_val as word                '10 bit
   adcsr.adfr = 1                       'free running
   adcsr.adsc = 1                       'starten
   cursor noblink
   cursor off
   do
      adc_val = ADCL
      adc_h = ADCH
      shift adc_h , LEFT , 8
      adc_val = adc_val OR adc_h

      locate 1 , 10
      select case anz_mod
         case 0 : zahlstr = str(adc_val)       'Abstimmspannung (V)
                  zahlstr = format(zahlstr , " 0.00")
                  lcd zahlstr ; " V"
         case 1 : zahlstr = str(adc_val)       'Antennenspannung (mV)
                  zahlstr = format(zahlstr , "   0")
                  lcd zahlstr ; "   "
      end select

      waitms 100
   loop until setbutton = 0
   adcsr.adsc = 0
   adcsr.adfr = 0                       'free running aus
   locate 1 , 10
   waitms 500
end SUB



drehimpuls1 alias pind.2
drehimpuls2 alias pinb.0
OnINT0:
   if drehimpuls1 = 1 then              'steigende Flanke
      if drehimpuls2 = 1 then
         decr drehgeber
      else
         incr drehgeber
      end IF
   else                                 'fallende Flanke
      if drehimpuls2 = 0 then
         decr drehgeber
      else
         incr drehgeber
      end IF
   end IF
   waitms 2
   GIFR.INTF0 = 1                       'evtl. Preller entfernen
return

OnINT1:                                 'nur zum Aufwachen!
return
```