

'Pic Basic Code from Joel Murphy, physical computing, Parsons 2008

```
include "modedefs.bas"
```

```
OSCCON = %01100110 '4MHz INTERNAL OSCILLATOR
```

```
ANSEL = %00000000 'MAKE PortA PINS ALL DIGITAL
```

```
TRISA = %00000100 'INPUT = 1, OUTPUT = 0
```

```
TRISB = %00000000 '
```

```
'VARIABLES
```

```
RawTemp VAR WORD 'Input from SHT
```

```
DegreesC var word 'Temperature in Celcius
```

```
DegreesF var word 'Temperature in Farenheit
```

```
RawHumid var word 'Input from SHT
```

```
RH_Lin var word 'Humidity Linearalized
```

```
RH_Comp var word 'Humidity Compensated for Temperature
```

```
Dummy var word 'used with DIV32 to do math on LARGE numbers [MAX  
2,147,483,647]
```

```
Y var word
```

```
X var word 'used to help with math
```

```
Z var word
```

```
IObyte var byte 'Used to shift Data in and out of SHT
```

```
ACKbit var bit
```

```
WaitBit var bit
```

```
Time var word
```

```
'CONSTANTS
```

```
SHT_Temp con %00000011
```

```
SHT_Humid con %00000101
```

```
ACK CON 0
```

```
NoACK con 1
```

```
'PINS
```

```
SHT_Data var PortA.3
```

```
Clock var PortA.2
```

```
Serial var PortB.6
```

```
Blink var PortB.3
```

```
high serial
```

```
high sht_data
```

```
Pause 500
```

```
serout2 serial,84,[249] 'GET READY TO DEFINE CHARACTER 1
serout2 serial,84,[0] '
serout2 serial,84,[4] '
serout2 serial,84,[14] '      DEFINE YOUR OWN DEGREES SYMBOL HERE
serout2 serial,84,[31] '      SEND A [1] TO PRINT IT IN SEROUT2
serout2 serial,84,[14] '
serout2 serial,84,[4] '
serout2 serial,84,[0] '
serout2 serial,84,[1] '

```

```
serout2 serial,84,[12]      'LCD clear
pause 10
serout2 serial,84,[17]      'LCD backlight on

```

Main:

```
  gosub readtemp
'  serout2 serial,84,[dec rawtemp,1,13]
  Serout2 serial,84,[dec degreesc,1,"C",dec degreesf,1,"F",13]
  pause 100
  gosub readhumidity
'  serout2 serial,84,[dec rawhumid,"RH"]
  serout2 serial,84,["L= ",dec rh_lin]      ', "T= ",dec rh_comp]
  high blink
  pause 500      'used for debugging
  low blink
  serout2 serial,84,[13]

```

goto Main

ReadTemp:

```
  gosub sht_start
  iobyte = sht_temp
  gosub sht_write
  gosub sht_wait
  ackbit = ack
  gosub sht_read
  rawtemp.byte1 = iobyte
  ackbit = noack
  gosub sht_read
  rawtemp.byte0 = iobyte

```

```
' >>> DETERMINE DEGREES CELCIUS <<<
  degreesc = (rawtemp/100) - 40
' >>> DETERMINE DEGREES FARENHEIT <<<

```

```

dummy = rawtemp*10
degreesf = div32 555      'THE DIV COMMAND MAKES IT POSSIBLE TO
degreesf = degreesf-40   'DO MATH WITH BIG NUMBERS. LOOK IN
COMPILER MANUAL
Return                    'FOR DETAILS ON THIS

```

ReadHumidity:

```

GOSUB sht_start
iobyte = sht_humid
gosub sht_write
gosub sht_wait
ackbit = ack
gosub sht_read
rawhumid.byte1 = iobyte
ackbit = noack
gosub sht_read
rawhumid.byte0 = iobyte

```

```
' >>> RETURN LINEAR HUMIDITY <<<
```

```

dummy = rawhumid*10
x = div32 247
y = rawhumid/500
z = rawhumid/714
rh_lin = x-(y*z)-4

```

```
' >>> RETURN RH COMPENSATED FOR TEMPERATURE <<<
```

```

' rh_comp = degreesc-25
' rh_comp = rh_comp*((rawhumid/126)+1)+rh_lin
' *****still workin' on it !!
return

```

```
' >>>> TELLS SHT TO GET READY FOR TRANSMISSION <<<<
```

SHT_START:

```

input sht_data    'send data pin high with pullup    low clock
high clock        ' GENERATES SHT_START
low sht_data      '
low clock         'DATA  _____|_____
high clock        '
input sht_data    'CLOCK  ___|___|___
low clock

```

```
return
```

```
' >>>> SENDS COMMAND BYTE TO SHT <<<<
```

```
SHT_WRITE:
```

```
  shiftout sht_data,clock,1,[iobyte]    'CHECK COMPILER FOR MODE
```

```
DEFINITIONS
```

```
  shiftin sht_data,clock,0,[ackbit\1]   'MSBPRE/LSBPRE, ETC...
```

```
  return
```

```
' >>>> WAITS FOR SHT TO FINISH TAKING READING <<<<
```

```
SHT_WAIT:
```

```
  input sht_data
```

```
  for time = 0 to 300
```

```
    if sht_data = 0 then return
```

```
    pause 1
```

```
  next time
```

```
  return
```

```
' >>>> RECEIVES BYTE[s] FROM SHT <<<<
```

```
SHT_READ:
```

```
  shiftin sht_data,clock,0,[iobyte]
```

```
  shiftout sht_data,clock,0,[ackbit\1]
```

```
  input sht_data
```

```
  return
```