INTELLIGENT HUMIDISTAT

Submitted by: Group-71

Group Members:

Shreyansh Joshi	2018A7PS0097G
S Sethuram	2018A7PS0101G
Wani Susmit Nitin	2018A7PS0116G
Varun	2018A8PS0877G
Siddharth Sharma	2018A7PS0199G

Prepared in partial fulfilment of the requirements of the course

"Microprocessors and Interfacing"

Course No. CS/ECE/EEE/INSTR F241

at

BIRLA INSTITUTE OF TECHNOLOGY & SCIENCE, PILANI
K K BIRLA GOA CAMPUS – 403726

Submission Date: 19th April 2020

Design Question No: 23
Group Number: 71

Submitted to: Prof. K.R. Anupama
Designation: Instructor-in-Charge (IC)

Project Areas: Microprocessors and Interfacing, Assembly Language Programming

Abstract: This project aims to make an Intelligent Humidistat device. As per the problem statement, we design and emulate the hardware and block diagram of this device in a software called “Proteus”. Using this hardware design and block diagram, we prepare a flowchart on how the system will work when programmed. Various external components are to be used like sensors, decoders, ROMs, RAMs, 8255, 8253, ADC, etc. Following that chart, we hence, write a program in Assembly Language for the device using MASM 611.

ACKNOWLEDGEMENT

The successful completion of this project and report required a lot of guidance and support. We extend our deepest gratitude to the IC – Prof. K.R. Anupama for giving us the opportunity to work on such an interesting assignment. The video lectures as well as in-class lectures and tutorials were really helpful in clearing our basics required in order to complete the project successfully.

Last but not the least, we would like to extend our gratitude to all lab instructors and the TAs who helped us get a command over Assembly Language.

TABLE OF CONTENTS

Contents **Page No.**

PROBLEM STATEMENT

P23: System to be designed: **An Intelligent Humidistat**

A humidistat is supposed to be reset according to the outside temperature – as the outside temperature falls, the humidity level inside the house should be set lower. The purpose of this project is to develop a humidistat which senses the outside temperature and adjusts the humidity accordingly. Two sensors are required: outside temperature and inside humidity. Output is provided via a simple relay with the humidifier (presumably on the furnace) being on or off. Also, readings from the humidity and temperature sensors must be displayed on an LCD display. The entire system can be turned on or off using a single switch.

ASSUMPTIONS

The following are the assumptions made regarding the system:

- The outside temperature is between -40°C and 60°C.
- Room is reasonably small. Hence, only 1 sensor is required for measuring humidity level of the room.

- Resolution of 1°C and 1% is required in temperature and humidity sensors respectively.
- The humidifier turns on when the LED glows, and consequently, the humidifier turns off when the LED stops glowing.

- **There is a linear relationship between temperature and humidity**, i.e., for an increase in temperature by 1°C, there is an increase in relative humidity by 1%, and at 0°C the relative humidity must be 40%. For example – if the temperature is 34°C, the corresponding RH should be 74%.
- We have used potentiometers in proteus for simulating the temperature and humidity sensors. In real-life implementation, 2048RH/T model by HydroLynx Systems could be used as temperature and humidity sensors respectively.

SYSTEM DESCRIPTION

The humidistat is supposed to change the humidity level inside a room according to the outside temperature. If the current humidity is less than the ideal humidity which is meant for the current outside temperature, the humidifier should be turned on. The role of humidifier is to increase the humidity inside the room. After attaining the ideal humidity, if the outside temperature increases, then also the humidifier is tuned on.

For this, two sensors are required: *Outside Temperature Sensor* and *Inside Humidity Sensor*

The temperature sensors are mounted outside the room and are open to the atmosphere. The humidity sensors are mounted inside the room. The humidity sensor measures the humidistat in % Relative Humidity. The sensors give **analog** output. These outputs are converted to digital form through A/D converters.

ADC is set such that the output received from the ADC ranges from 00h to 64h, where with increase in temperature by 1°C, or with increase in RH by 1%, results in increase in the output by 01h.

Digital Output (8-bit)	Corresponding Temperature	Corresponding RH
00h	-40°C	0%
01h	-39°C	1%
. . .	. . .	. . .
64h	60°C	100%

LIST OF COMPONENTS USED

Chip No.	Qty	Chip	Purpose
8086	1	Microprocessor	Central Processing Unit
2732	2	EPROM	Erasable Programmable Read Only Memory; in which the code resides
6116	2	SRAM	Used to store the temporary data (like temperature values, stack, etc.)
74138	2	3x8 Decoder	To select among the two PPIs (8255), 8253; and for memory interfacing
8255	2	Programmable Peripheral Interface	Provides I/O ports for the other devices
ADC0808	1	Analog to Digital Converter	Converts the analog voltage to its digital equivalent
74LS373	3	8-bit octal latches	Latching the address bus
74LS245	2	8-bit bidirectional buffer	Buffering Data Bus
8253	1	Programmable Interval Timer	To generate clock input for ADC
LM016L	1	16x2 alphanumeric LCD	Displays the current temperature and humidity

OTHER HARDWARE USED

1. **Logic Gates –** These are primarily used for building decoding logic for memory interfacing and I/O interfacing.

2. **Solid-state Relay –** It is used as a switch to power on high voltage devices.

3. **LED –** It is used to show the turning on/off of humidifier.

4. **Potentiometers –** They are used to simulate input from sensors.

5. **Switches –** They are used to power off/on the system and the LCD.

MEMORY ORGANIZATION

Memory is divided into odd and even banks for word and byte transfer (8086 has an 8-bit data line).

ROM (Each of size 4 KB, 2nos.)

EVEN:	00000h – 01FFEh
ODD:	00001h – 01FFFh

RAM (Each of size 2KB, 2nos.):

EVEN:	02000h – 02FFEh
ODD:	02001h – 02FFFh

The code resides in ROM and begins at address 0000h. The address loaded as soon as system starts is FFFF0h.

MEMORY AND ADDRESS MAP

CHIP	$A_{19}A_{18}A_{17}A_{16}$	$A_{15}A_{14}A_{13}A_{12}$	$A_{11}A_{10}A_{9}A_{8}$	$A_{7}A_{6}A_{5}A_{4}$	$A_{3}A_{2}A_{1}A_{0}$
EPROM 2732					
00000h	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0
01FFFh	0 0 0 0	0 0 0 1	1 1 1 1	1 1 1 1	1 1 1 1
SRAM 6116					
02000h	0 0 0 0	0 0 1 0	0 0 0 0	0 0 0 0	0 0 0 0
02FFFh	0 0 0 0	0 0 1 0	1 1 1 1	1 1 1 1	1 1 1 1

I/O INTERFACING

The input and output devices of the system are connected to the processor using 8255 Programmable Peripheral Interfacing controllers, and 8253 Programmable Interval Timer is used to provide clock signals to ADC.

Here we connect A_0 and A_1 of 8255s and 8253 to A_1 and A_2 of 8086's address bus. The CS of 8255s and 8253 are connected to the corresponding outputs of decoder 74138 for selecting the chips. Addresses for the chips are as follow (All have been given even address space as data lines $D_0 – D_7$ are used):

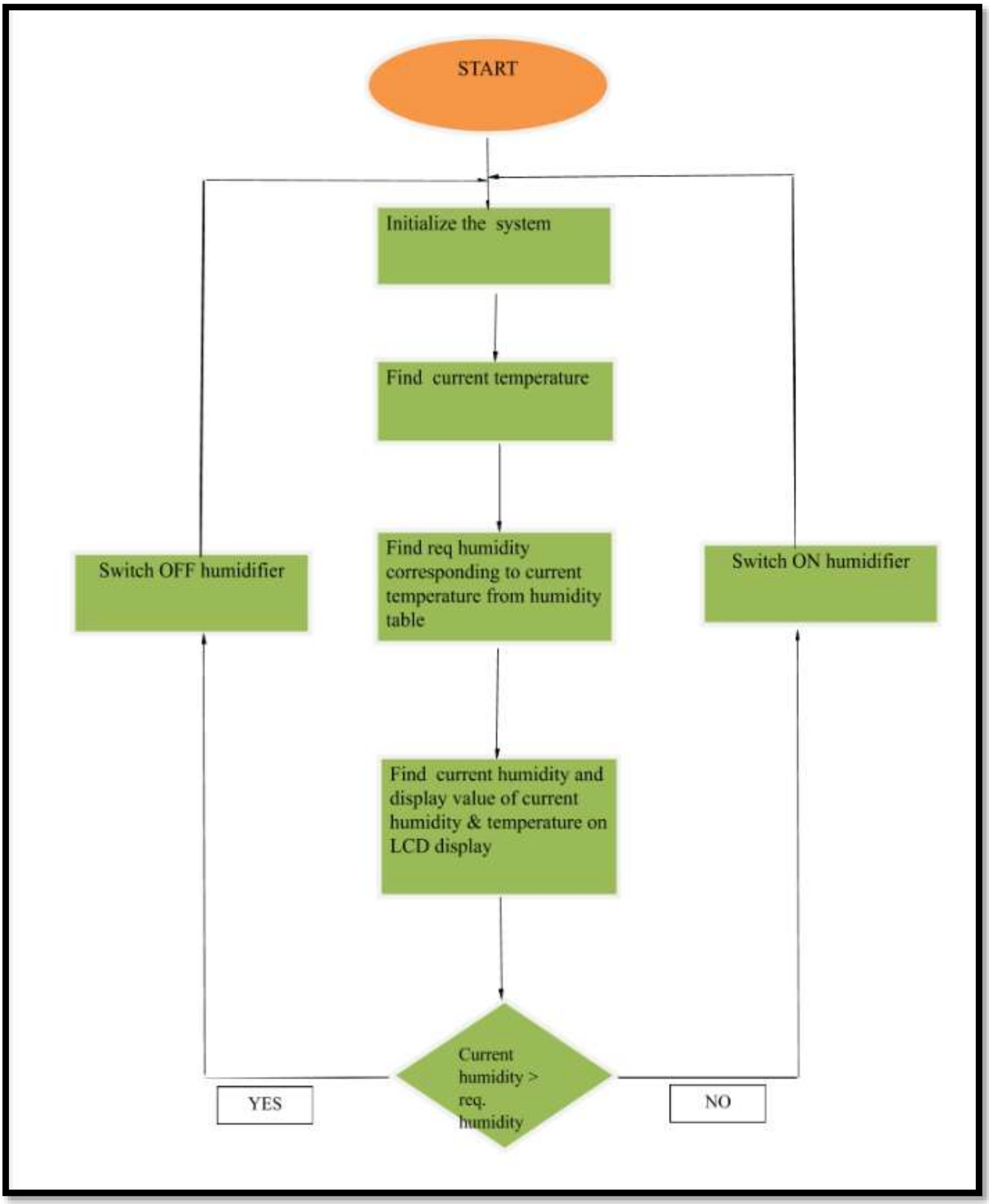
I/O MAPPING

Address of 8255_1 Port-A	: 00h
Address of 8255_1 Port-B	: 02h
Address of 8255_1 Port-C	: 04h
Address of 8255_1 Control Register	: 06h
Address of 8255_2 Port-A	: 08h
Address of 8255_2 Port-B	: 0Ah
Address of 8255_2 Port-C	: 0Ch
Address of 8255_2 Control Register	: 0Eh
Address of 8253 Counter-0	: 10h
Address of 8253 Counter-1	: 12h
Address of 8253 Counter-2	: 14h
Address of 8253 Counter Register	: 1Eh

PORT ADDRESS MAPS

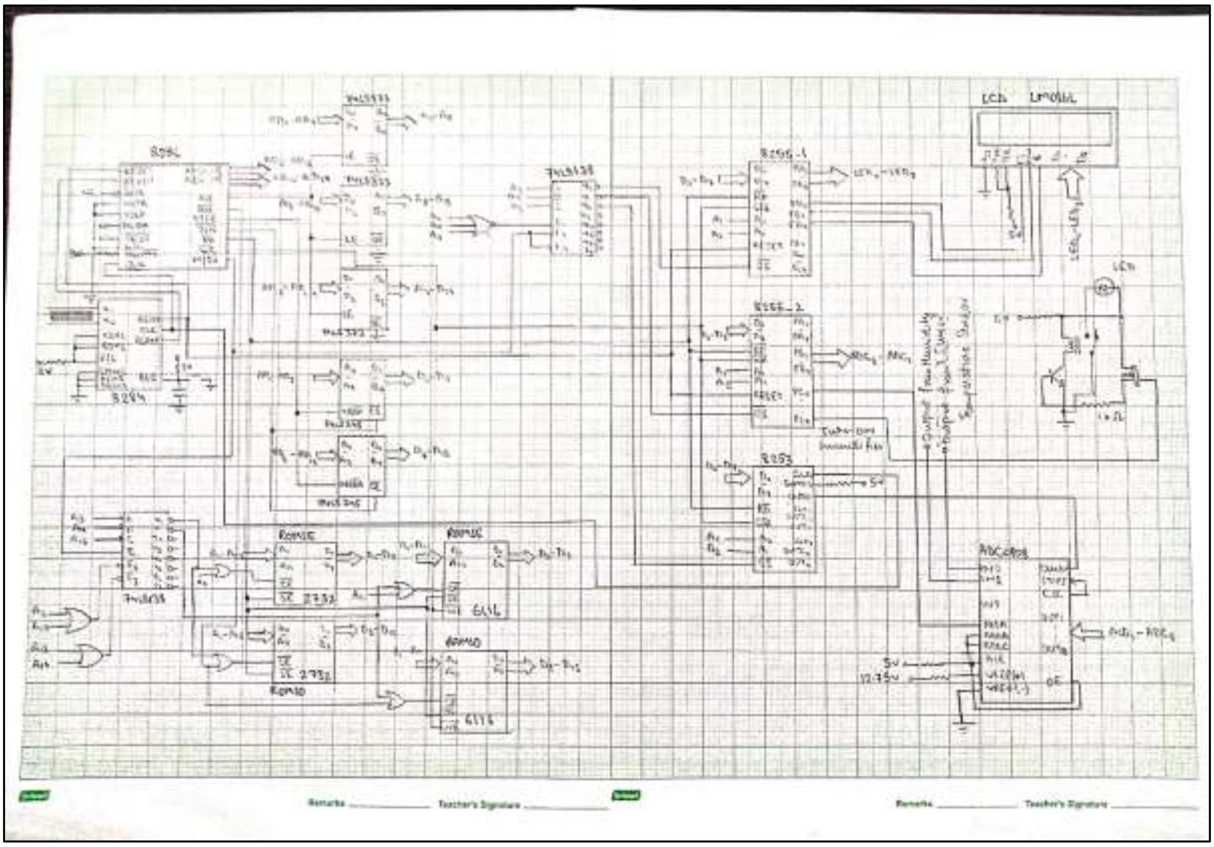
CHIP	A7	A6	A5	A4	A3	A2	A1	A0
8255_1								
00h	0	0	0	0	0	0	0	0
06h	0	0	0	0	0	1	1	0
8255_2								
08h	0	0	0	0	1	0	0	0
0Eh	0	0	0	0	1	1	1	0
8253								
10h	0	0	0	1	0	0	0	0
16h	0	0	0	1	0	1	1	0

SOFTWARE FLOWCHART


START
Initialize the system
Find current temperature
Find req humidity corresponding to current temperature from humidity table
Find current humidity and display value of current humidity & temperature on LCD display
Current humidity > req. humidity
YES
NO
Switch OFF humidifier
Switch ON humidifier

APPENDIX

COMPLETE DESIGN DRAWING



Separately attached (master.pdf)

ASM CODE

```
#make_bin#

#LOAD_SEGMENT=FFFFh#
#LOAD_OFFSET=0000h#

#CS=0000h#
#IP=0000h#

#DS=0000h#
#ES=0000h#

#SS=0000h#
#SP=FFFEh#

#AX=0000h#
#BX=0000h#
#CX=0000h#
#DX=0000h#
#SI=0000h#
#DI=0000h#
#BP=0000h#

jmp     start

            db      1024 dup(0)

curr_temp   db      ?                   ;current temperature
curr_humdt  db      ?                   ;current humidity
ideal_humdt db      ?                   ;ideal humidity
neg_flag    db      00h

print_temp  db      "Temperature"
print_humd  db      "Humidity   "

;8255_1 for LCD
port1A equu 00h     ;input to LCD
port1B equ  02h     ;controlling the LCD
port1C equ  04h
creg1  equ  06h     ;control register (8255_1)
```

```
;8255_2 for ADC
port2A equ     08h
port2B equ     0ah      ;taking output from ADC
port2C equ     0ch      ;PC0 - controlling whether to select temperature sensor or humidity
sensor
                        ;PC7 - turning on the humidifier
creg2   equu   0eh      ;control register (8255_2)


;8253 for generating clock signal to ADC
cnt0    equ    10h      ;counter 0
creg3   equ    16h      ;control register


start:  cli

        ;intialize ds, es, ss to the start of ROM
        mov    ax, 0000h
        mov    ds, ax
        mov    es, ax
        mov    ss, ax
        mov    sp, 0fffeh

        ;initializing 8255
        sti

        mov    al, 88h          ;control word for 8255_1 (for LCD)
        out    creg1, al        ;Port-A for 8-bit data to be sent to LCD
                                ;PB0 to control RS of LCD
                                ;PB1 to control R/W of LCD
                                ;PB2 to control E of LCD

        mov    al, 82h          ;control word for 8255_2 (for ADC)
        out    creg2, al        ;Port-B for output from ADC
                                ;PC0 to select from Temperature Sensor and Humidity Sensor
                                ;PC7 to control whether to switch on the humidifier or not

        ;initializing 8253
        mov    al, 16h          ;Counter 0 to work in mode-3
        out    creg3, al        ;control word for 8253

        mov    al, 5
        out    cnt0, al;count value of 5 given to Counter 0

        ;initializing LCD
        ;FUNCTION SET

        ;D7     D6      D5      D4      D3      D2      D1      D0
        ;0      0       1       DL      N       F       *       *

        ;DL = Data Length (1: 8-bit, 0: 4-bit)
        ;N  = Number of Display Lines (1: 2 lines, 0: 1 line)
        ;F  = Character Font (1: 5x10 dots, 0: 5x7 dots)

        ;Cannot display 2 lines with 5x10 dots character font

        mov     al, 38h         ;function set
        call    cmndwrt


        ;DISPLAY ON

        ;D7     D6      D5      D4      D3      D2      D1      D0
        ;0      0       0       0       1       D       C       B

        ;D = Display (1: On, 0: Off)
        ;C = Display Cursor (1: On, 0: Off)
        ;B = Blink (1: On, 0: Off)

        mov     al, 0ch         ;display on
        call    cmndwrt


        ;ENTRY MODE SET

        ;D7     D6      D5      D4      D3      D2      D1      D0
        ;0      0       0       0       0       1       I/D     S

        ;I/D = Sets the cursor move direction
        ;S   = Whether to shift the display after read/write operation

        mov     al, 06h
        call    cmndwrt



main:   call    getHumdt
        call    getTemp

        call    display_LCD
        mov     al, ideal_humdt
        mov     bl, curr_humdt
        cmp     bl, al
        jl      inc_hum
        jmp     rpt

inc_hum:        call    inc_humdt
                jmp     rpt

rpt:            call    delay_major
                jmp     main



delay_minor proc near                           ;0.1 ms delay

        push    cx

        mov     cl, 30
d1:     dec     cl
        jnz     d1

        pop     cx

        ret

delay_minor endp



delay_std proc near                     ;3 ms delay

        push    cx

        mov     cx, 900
d2:     dec     cx
        jnz     d2

        pop     cx

        ret

delay_std endp
```

```
delay_major proc near                   ;218 ms delay

        push    cx

        mov     cx, 0ffffh
d3:     dec     cx
        jnz     d3

        pop     cx

        ret

delay_major endp



getTemp proc  near                      ;get Temperature through ADC

        mov     al, 00h
        out     creg2, al               ;PC0 = 0 (Using BSR)
                                        ;to get the current temperature from the sensor via ADC
                                        ;(ADD A = 0 in ADC is connected to temperature
sensor)
        call    delay_major
        mov     al,82h
     out    creg2,al
        in      al, port2B              ;The ADC Output values shall be ranging from 00h -
64h (0 - 100 in decimals)

        mov     curr_temp, al
        mov     ideal_humdt, al

        sub     curr_temp, 40
        cmp     curr_temp, 0
        jge     pos

        mov     neg_flag, 01h
        mov     al, ideal_humdt
        mov     cl, 40
        sub     cl, al
        mov     al, cl
        jmp     con

pos:    mov     neg_flag, 00h
        mov     al, curr_temp
```

```
con:    call    convBCD

        ret

getTemp         endp


getHumdt proc           near                    ;get Temperature through ADC

        mov     al, 01h
        out     creg2, al               ;PC0 = 1 (Using BSR)
                                        ;to get the current humidity from the sensor via ADC
                                        ;(ADD A = 1 in ADC is connected to humidity sensor)
        call    delay_major

        mov     al,82h
      out     creg2,al
        in      al, port2B

        mov     curr_humdt, al

        call    convBCD
        mov     dx, bx

        ret

getHumdt endp

inc_humdt proc          near



        mov     al, 0fh
        out     creg2, al

        ret

inc_humdt endp



display_LCD proc        near            ;Displays temperature and humidity on LCD

        push    ax
        push    cx
        push    si

        lea     si, print_temp          ;Displays the word "Temperature"
        mov     cx, 11

pt1:    mov     al, [si]
        call    datawrt

        inc     si
        dec     cx
        jnz     pt1

        ;call   delay_minor

        cmp     neg_flag, 00h
        jz      p1

        mov     al, "-"                 ;Displays '-' if the temperature is negative
        jmp     n1
p1:     mov     al, " "                 ;Displays space if the temperature is positive
n1:     call    datawrt


        ;BX register stores the value of current temperature
        ;The contents in BH and BL registers are already converted to the corresponding
ASCII values

        mov     al, bh                  ;Displays the contents of BH register
        call    datawrt

        mov     al, bl                  ;Displays the contents of BL register
        call    datawrt

        mov     al, 0dfh                ;Displays "°C" (degree celsius)
        call    datawrt

        mov     al, "C"
        call    datawrt


        ;call   delay_std
        ;call   delay_std

        mov     al, 0c0h                ;Shift to next line of LCD Display
        call    cmndwrt
        lea     si, print_humd          ;Displays the word "Humidity"
        mov     cx, 11

ph1:    mov     al, [si]
        call    datawrt

        inc     si
        dec     cx
        jnz     ph1


        ;DX register stores the value of humidity
        ;The contents in DH and DL registers are already converted to the corresponding ASCII values
        cmp     curr_humdt, 100
        jne     x1
        mov     al, "1"                 ;Displays "1" only if curr_humdt = 100%,
        jmp     x2                      ;else displays " "

x1:     mov     al, " "
x2:     call    datawrt

        mov     al, dh                  ;Displays the contents of DH register
        call    datawrt

        mov     al, dl                  ;Displays the contents of DL register
        call    datawrt

        mov     al, "%"                 ;Displays "%" symbol
        call    datawrt


        mov     al, 00h
        out     port1B, al
        call    delay_minor
        mov     al, 80h
        call    cmndwrt


        pop si
        pop cx
        pop ax

        ret
```

```
display_LCD endp


cmndwrt proc  near                         ;Writing the commands to LCD Display

           out      port1A, al
           call     delay_minor
           mov      al, 04h                ;Giving high to low transition to Enable signal keeping
RS = 0 and R/W = 0
           out      port1B, al
           call     delay_minor
           mov      al, 00h
           out      port1B, al
           call     delay_minor

           ret

cmndwrt endp

datawrt proc    near                       ;Writing the data to display on LCD Display

           out      port1A, al
           call     delay_minor
           mov      al, 05h                ;Giving high to low transition to Enable signal keeping
RS = 1 and R/W = 0
           out      port1B, al
           call     delay_minor
           mov      al, 01h
           out      port1B, al
           call     delay_minor

           ret

datawrt endp



;To convert Binary numbers to BCD form
;The binary number to be converted is stored in AL register
;The BCD form is stored in BX register
;This procedure has already converted the BCD form to its corresponding ASCII values to
display them on LCD

convBCD proc            near               ;convert Binary to BCD

           mov      bh,0ffH
```

```
c1:     inc     bh
      sub       al, 0ah
        jnc     c1
        add     al, 0ah
        mov     bl, 30h
        add     bh, bl
        add     bl, al

        cmp     bh, 3ah                 ;if curr_humdt is equal to 100%, value of bh becomes
3ah
        jne     c2
        mov     bh, 30h                         ;bh = 30h and bl = 30h, to display "00" of "100"

c2:     ret

convBCD endp
```

VARIATIONS IN PROTEUS IMPLEMENTATION WITH JUSTIFICATION

1. Using 8253 instead of 8254 because of unavailability of 8254 on Proteus.

2. The minimum storage of ROMs available in Proteus is of 4KB (2732).

3. Temperature/Humidity sensor – replaced by a potentiometer in the Proteus design to provide voltage between 0 – 5 V, as all sensors are not available on Proteus.

LIST OF ATTACHMENTS

1. Component-wise connections – *drawings.pdf*

2. Complete Design diagram – *master.pdf*

3. Manuals
   - 2048RH/T (Temperature and Relative Humidity Sensors)
   - LCD LM016L

4. Proteus Design File – *design.DSN*

5. EMU8086 ASM File – *code.asm*

6. Binary File after assembly – *code.bin*