A REPORT
ON

INTELLIGENT HUMIDISTAT

BY

Meet Kanani	**2017A7PS0128P**
Aditya Ramaswamy	**2017A7PS0130P**
Mukkamala Venkat Sai Ram	**2017A7PS0133P**
S Hariharan	**2017A7PS0134P**

Prepared in partial fulfilment of the requirements of the course
“Microprocessors and Interfacing” - Course Number: CS F241

AT

BIRLA INSTITUTE OF TECHNOLOGY & SCIENCE, PILANI

BIRLA INSTITUTE OF TECHNOLOGY & SCIENCE, PILANI
PILANI – 333031 (RAJASTHAN)

Submission Date and Place: 25th April 2019, Pilani

Title of the Project: Intelligent Humidistat

Project Number: 23
Group Number: 91

Name of Experts: Dr. Nitin Chaturvedi
Designation: Instructor-in-Charge (ICs)

Key Words: Microprocessors, Interfacing, Peripherals like ADC, RAM, Octal Buffers, Latches, LCD Display, Decoders, etc.

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". Various external components are to be used like sensors, potentiometers, RAMs, ROMs, decoders, etc. Using this hardware design and block diagram, we prepare a flowchart on how the system will work when programmed. Following that chart, we thereby write a program in Assembly Language for the device using MASM 611.

TABLE OF CONTENTS

PROBLEM STATEMENT

P23: System to be designed: An Intelligent Humidistat

System Description: 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 -50° C and 49° C.
- 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 humidity is 50%. For example – if the temperature is 17° C then the corresponding humidity should be 67% RH.

SYSTEM DESCRIPTION

The humidistat is supposed to change the humidity level inside a room according to the outside temperature. When the temperature goes low from the reference temperature (previously measured temperature), the humidifier is turned on through a solid-state relay. Then the humidity level is lowered by the humidifier and the humidity level is monitored until the humidity reaches a value corresponding to the current temperature. The humidifier is then turned off. And then the current temperature is made the reference value and the whole process is repeated again. The 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 humidity in % Relative Humidity. The sensors give analog output. These outputs are converted to digital form through A/D converters. In the memory, there is a look up table which stores the % RH values obtained from the linear relationship for the corresponding temperatures in the temperature range.

LIST OF COMPONENTS USED

Chip No.	Qty.	Chip	Purpose
8086	1	Microprocessor	Central Processing Unit
6116	2	SRAM	Used to store the temporary data (like temperature values, stack, etc.)
2732	2	EPROM	Erasable Programmable Read Only Memory; in which the code resides
74138	2	3:8 Decoder	To Select PPI (8255), 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
74LS245	2	8-bit bidirectional buffer	Buffering Data bus
LM016L	1	16x2 alphanumeric LCD	Displays the current temperature and humidity
74LS373	3	8-bit octal latches	Latching the address bus

OTHER HARDWARE USED

1. Logic Gates – These are primarily used for building decoding logic for memory interfacing and I/O interfacing.
2. Tri-state buffer – It is used to generate interrupt vector numbers.
3. Solid-State Relay – It is used as a switch to power on high voltage devices.
4. LED – It is used to signal the activity of the humidifier.
5. Potentiometers – They are used to simulate input from sensors.
6. 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.

ROM:

ROM1 EVEN: 00000h – 01FFEh
ROM1 ODD: 00001h - 01FFFh

RAM:

RAM1 EVEN: 02000h – 02FFEh
RAM1 ODD: 02001h -- 02FFFh

The data segment starts at the address of 00000h
Code Segment begins at 00000h.

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_1A_0**
EPROM 2732					
From 00000h	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0
To 01FFFh	0 0 0 0	0 0 0 1	1 1 1 1	1 1 1 1	1 1 1 1
SRAM 6116					
From 02000H	0 0 0 0	0 0 1 0	0 0 0 0	0 0 0 0	0 0 0 0
To 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. Here we connect the A_0 and A_1 of the 8255 to the A_1 and A_2 of the 8086's address bus respectively and also we connected the $\overline{CS}$ of 8255 to the output of decoder 74139 for selecting the chip. Addresses for the chips are as follow (All have been given even address space):

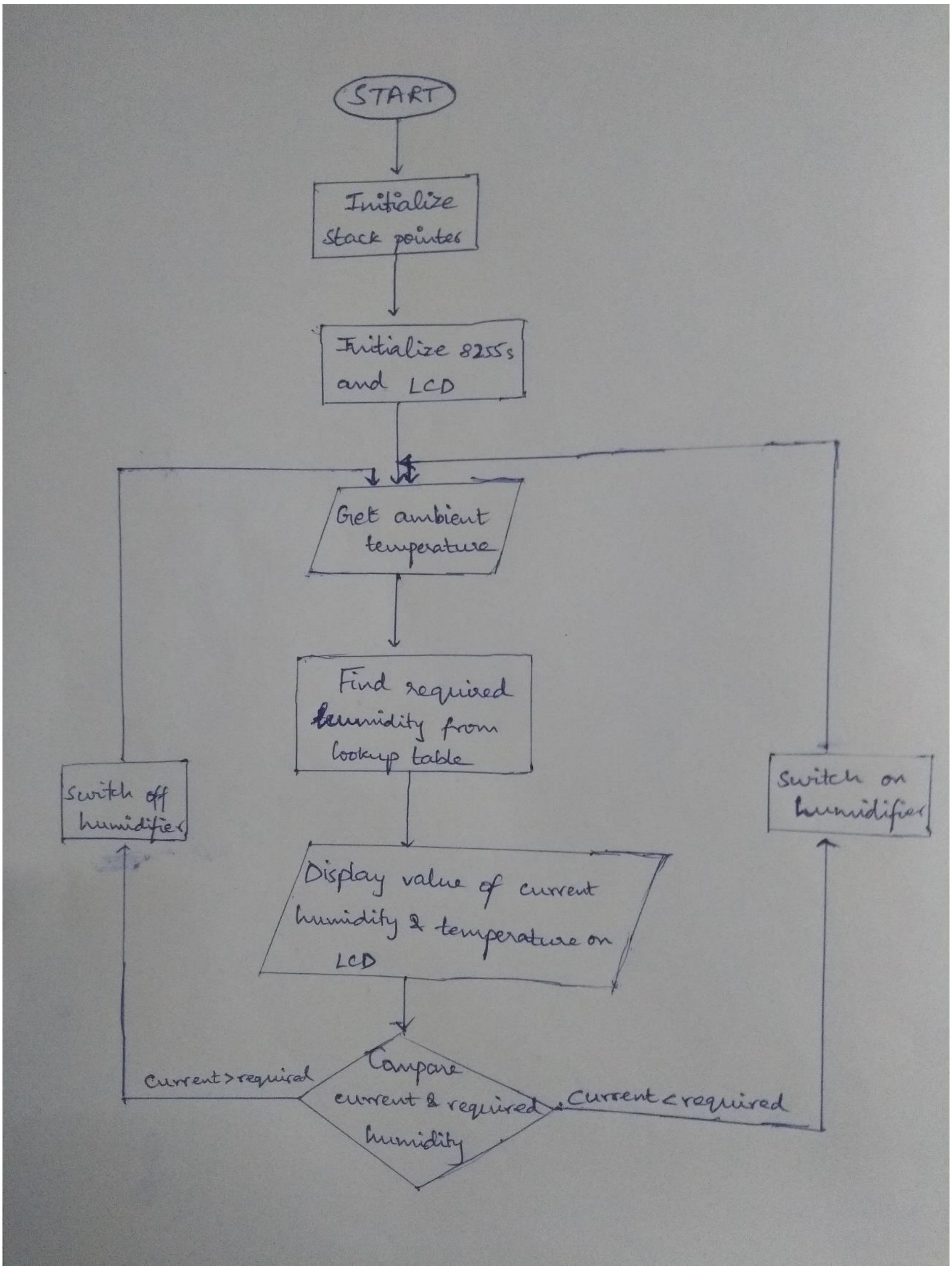
I/O Mapping

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

PORT ADDRESS MAPS

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} $A_{1}A_{0}$
8255 - 1					
From 00000h	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0
To 00006h	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	0 1 1 0
8255 - 2					
From 00008H	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	1 0 0 0
To 0000Eh	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	1 1 1 0

SOFTWARE FLOWCHART



APPENDIX

a. ASSEMBLY 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#

temp_ref     db
0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,21,23,24,25,26,27,28,29,30,31,32,33,
34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51,52,53,54,55,56,57,58,59,60,61,62,63,
64,65,66,67,68,69,70,71,72,73,74,75,76,77,78,79,80,81,82,83,84,85,86,87,87,89,90,91,92,93,
94,95,96,97,98,99,100

ADC_out      db
00h,02h,05h,08h,0ah,0dh,0fh,12h,14h,17h,1ah,1ch,1fh,21h,24h,26h,29h,2bh,2eh,30h,33h,36
h,38h,3bh,3dh,40h,42h,45h,47h,4ah,4dh,4fh,52h,54h,57h,59h,5ch,5eh,61h,63h,66h,69h,6bh,
6eh,70h,73h,75h,78h,7ah,7dh,7fh,82h,85h,87h,8ah,8ch,8fh,91h,94h,96h,99h,9ch,9eh,0a1h,0a
3h,0a6h,0a8h,0abh,0adh,0b0h,0b2h,0b5h,0b8h,0bah,0bdh,0bfh,0c2h,0c4h,0c7h,0c9h,0cch,0c
fh,0d1h,0d4h,0d6h,0d9h,0dbh,0deh,0e0h,0e3h,0e5h,0e8h,0ebh,0edh,0f0h,0f2h,0f5h,0f7h,0fa
h,0fch,0ffh

temp_t      db  ?                       ;current temperature
temp_h       db  ?                       ;current humidity
cmp_t       db  ?                       ;voltage for current temperature
neg_flag    db  00h                       ;check sign of temperature
origin      db  50                      ;voltage for 0 degree temperature

jmp     strt
        db     1024 dup(0)
```

```
    strt:     cli

    ;intialize ds, es,ss to start of RAM
   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(for LCD)
    out 06h,al

    mov al,89h  ; control word for 8255(for ADC)
    out 0Eh,al


    mov al,00h      ;default low output for PC0
    out 0ch,al



     ;initializing LCD

    call dly_minor
    mov al,04h
    out 02h,al
    call dly_minor

    mov al,04h      ; to make rs=0 and r/w=0
    out 02h,al

    mov al,38h      ;function set
    out 00h,al

    mov al,04h
    out 02h,al
    call dly_minor
    mov al,00h      ;to make rs=0 and r/w=0
    out 02h,al
    call dly_minor
    mov al,0Ch      ; display on
    out 00h,al
    mov al,04h
    out 02h,al
    call dly_minor
```

```
            mov al,00h
            out 02h,al

            mov al,06h      ; set entry mode
            out 00h,al
            call dly_minor
            mov al,04h
            out 02h,al
            call dly_minor
            mov al,00h
            out 02h,al
            mov al,4ch
            out 00h,al
            call dly_minor



start:    call    idle
        call    clear_LCD
                call    hello_world
                call    dly_std


seq:  call    getHmd
        call    getTemp
        call    clear_LCD
        call    dly_std
        call    display_lcd
        mov     al,cmp_t
        mov     bl,temp_h
        cmp     al,bl
        ja      inc_hum
        jb      dec_hum
        call    idle
        jmp     repeat

inc_hum:    call    inc_hmd
            jmp     repeat

dec_hum:    call    dec_hmd
            jmp     repeat

repeat:    call    dly_major
            jmp     seq
```

```
dly_minor proc          near

            mov         cl,30
        aa:
                dec     cl
                jnz     aa
        ret
dly_minor endp


dly_major proc          near

        mov     cx,0ffffh
        bb:
                dec     cx
                jnz     bb
        ret
dly_major endp



dly_std proc    near

        mov     cx,5555h
        st:
                dec     cx
                jnz     st
        ret
dly_std endp




getTemp    PROC    NEAR                 ;get temperature through ADC


     mov     al,00h
     out     0eh,al    ; PC0=0

     call    dly_major

     mov     al,82h
     out     0eh,al
     in      AL,0AH
     lea     si,ADC_out
     lea     di,temp_ref
     dec     si

cc:    inc    si
```

```
        cmp     al,[si]
        jnz     cc
        sub     si,offset ADC_out
        add     di,si

        mov     al,[di]
        mov     cmp_t,al

        cmp     [di],50
        jge     pos
        mov     neg_flag,01h       ;for negative temperature
        mov     al,[di]
        mov     origin,50
        sub     origin,al
        mov     al,origin
        mov     temp_t,al
        jmp     con

pos:    mov     neg_flag,00h       ;for positive temperature
        mov     al,[di]
        sub     al,50
        mov     temp_t,al


con:    call    CONVBCD
        ret

getTemp    ENDP


getHmd     PROC    NEAR                    ;get humidity through ADC

        mov     al,01h
        out     0eh,al    ; PC0=1

        call    dly_major

        mov     al,82h
        out     0eh,al
        in      al,0aH

        lea     si,ADC_out
        lea     di,temp_ref
        dec     si

dd:     inc     si

        mov     bl,[si]
        cmp     al,bl
        jnz     dd
        sub     si,offset ADC_out
        add     di,si
        mov     al,[di]
        mov     temp_h,al

        call    CONVBCD
        mov     dx,bx
        ret

getHmd     ENDP

;increase humidity
inc_hmd   proc    near

                mov     al,0eh
                out     0eh,al    ;reset decrease humidity signal

                mov     al,0dh
                out     0eh,al    ;set increase humidity signal

                ret
inc_hmd   endp

;decrease humidity
dec_hmd   proc    near

                mov     al,0ch
                out     0eh,al    ;reset increase humidity signal

                mov     al,0fh
                out     0eh,al    ;set decrease humidity signal

                ret
dec_hmd   endp

;idle humidifier when temperature and humidity are equal
idle       proc    near

                mov     al,0eh
                out     0eh,al

                mov     al,0ch
                out     0eh,al

                ret
idle       endp


clear_LCD proc          near
```

```
        mov al,00h
        out 02h,al
        call dly_minor
        mov al,01h                      ;Clear LCD display
        out 00h,al
        call dly_minor
        mov al,04h
        out 02h,al
        call dly_minor
        mov al,00h
        out 02h,al
RET
clear_LCD endp


hello_world proc        near

        mov al,0A0h
        out 00h,al
        call dly_minor
        mov al,05h
        out 02h,al
        call dly_minor
        mov al,01h
        out 02h,al  ;prints Space

        mov al,0A0h
        out 00h,al
        call dly_minor
        mov al,05h
        out 02h,al
        call dly_minor
        mov al,01h
        out 02h,al  ;prints Space

        mov al,0A0h
        out 00h,al
        call dly_minor
        mov al,05h
        out 02h,al
        call dly_minor
        mov al,01h
        out 02h,al  ;prints Space

        mov al,48h
        out 00h,al
        call dly_minor
        mov al,05h
        out 02h,al
        call dly_minor
```

```
mov al,01h
out 02h,al  ;prints H

mov al,65h
out 00h,al
call dly_minor
mov al,05h
out 02h,al
call dly_minor
mov al,01h
out 02h,al  ;prints e

mov al,6ch
out 00h,al
call dly_minor
mov al,05h
out 02h,al
call dly_minor
mov al,01h
out 02h,al  ;prints l

mov al,6ch
out 00h,al
call dly_minor
mov al,05h
out 02h,al
call dly_minor
mov al,01h
out 02h,al  ;prints l

mov al,6fh
out 00h,al
call dly_minor
mov al,05h
out 02h,al
call dly_minor
mov al,01h
out 02h,al  ;prints o

mov al,0A0h
out 00h,al
call dly_minor
mov al,05h
out 02h,al
call dly_minor
mov al,01h
out 02h,al  ;prints Space

mov al,57h
out 00h,al
```

```
        call dly_minor
        mov al,05h
        out 02h,al
        call dly_minor
        mov al,01h
        out 02h,al  ;prints W

        mov al,6fh
        out 00h,al
        call dly_minor
        mov al,05h
        out 02h,al
        call dly_minor
        mov al,01h
        out 02h,al  ;prints o

        mov al,72h
        out 00h,al
        call dly_minor
        mov al,05h
        out 02h,al
        call dly_minor
        mov al,01h
        out 02h,al  ;prints r

        mov al,6ch
        out 00h,al
        call dly_minor
        mov al,05h
        out 02h,al
        call dly_minor
        mov al,01h
        out 02h,al  ;prints l

        mov al,64h
        out 00h,al
        call dly_minor
        mov al,05h
        out 02h,al
        call dly_minor
        mov al,01h
        out 02h,al  ;prints d

ret
hello_world endp


display_lcd   PROC    NEAR   ;Display temperature and humidity on LCD
```

```
mov al,54h
out 00h,al
call dly_minor
mov al,05h
out 02h,al
call dly_minor
mov al,01h
out 02h,al  ;prints 'T'

mov al,65h
out 00h,al
call dly_minor
mov al,05h
out 02h,al
call dly_minor
mov al,01h
out 02h,al  ;prints 'e'

mov al,6dh
out 00h,al
call dly_minor
mov al,05h
out 02h,al
call dly_minor
mov al,01h
out 02h,al  ;prints 'm'

mov al,70h
out 00h,al
call dly_minor
mov al,05h
out 02h,al
call dly_minor
mov al,01h
out 02h,al  ;prints 'p'

mov al,65h
out 00h,al
call dly_minor
mov al,05h
out 02h,al
call dly_minor
mov al,01h
out 02h,al  ;prints 'e'

mov al,72h
out 00h,al
call dly_minor
```

```
mov al,05h
out 02h,al
call dly_minor
mov al,01h
out 02h,al  ;prints 'r'

mov al,61h
out 00h,al
call dly_minor
mov al,05h
out 02h,al
call dly_minor
mov al,01h
out 02h,al  ;prints 'a'

mov al,74h
out 00h,al
call dly_minor
mov al,05h
out 02h,al
call dly_minor
mov al,01h
out 02h,al  ;prints 't'


mov al,75h
out 00h,al
call dly_minor
mov al,05h
out 02h,al
call dly_minor
mov al,01h
out 02h,al  ;prints 'u'

mov al,72h
out 00h,al
call dly_minor
mov al,05h
out 02h,al
call dly_minor
mov al,01h
out 02h,al  ;prints 'r'

mov al,65h
out 00h,al
call dly_minor
mov al,05h
out 02h,al
call dly_minor
mov al,01h
```

```
        out 02h,al  ;prints 'e'


        mov al,0A0h
        out 00h,al
        call dly_minor
        mov al,05h
        out 02h,al
        call dly_minor
        mov al,01h
        out 02h,al  ;prints Space


        cmp neg_flag,00h
        jz  hh

        mov al,2dh
        out 00h,al
        call dly_minor
        mov al,05h
        out 02h,al
        call dly_minor
        mov al,01h
        out 02h,al  ;prints '-'
        jmp nn


hh:    mov al,0A0h
        out 00h,al
        call dly_minor
        mov al,05h
        out 02h,al
        call dly_minor
        mov al,01h
        out 02h,al  ;prints Space


nn:     mov al,bh
        out 00h,al
        call dly_minor
        mov al,05h
        out 02h,al
        call dly_minor
        mov al,01h
        out 02h,al  ;prints number stored in bh

        mov al,bl
        out 00h,al
        call dly_minor
        mov al,05h
```

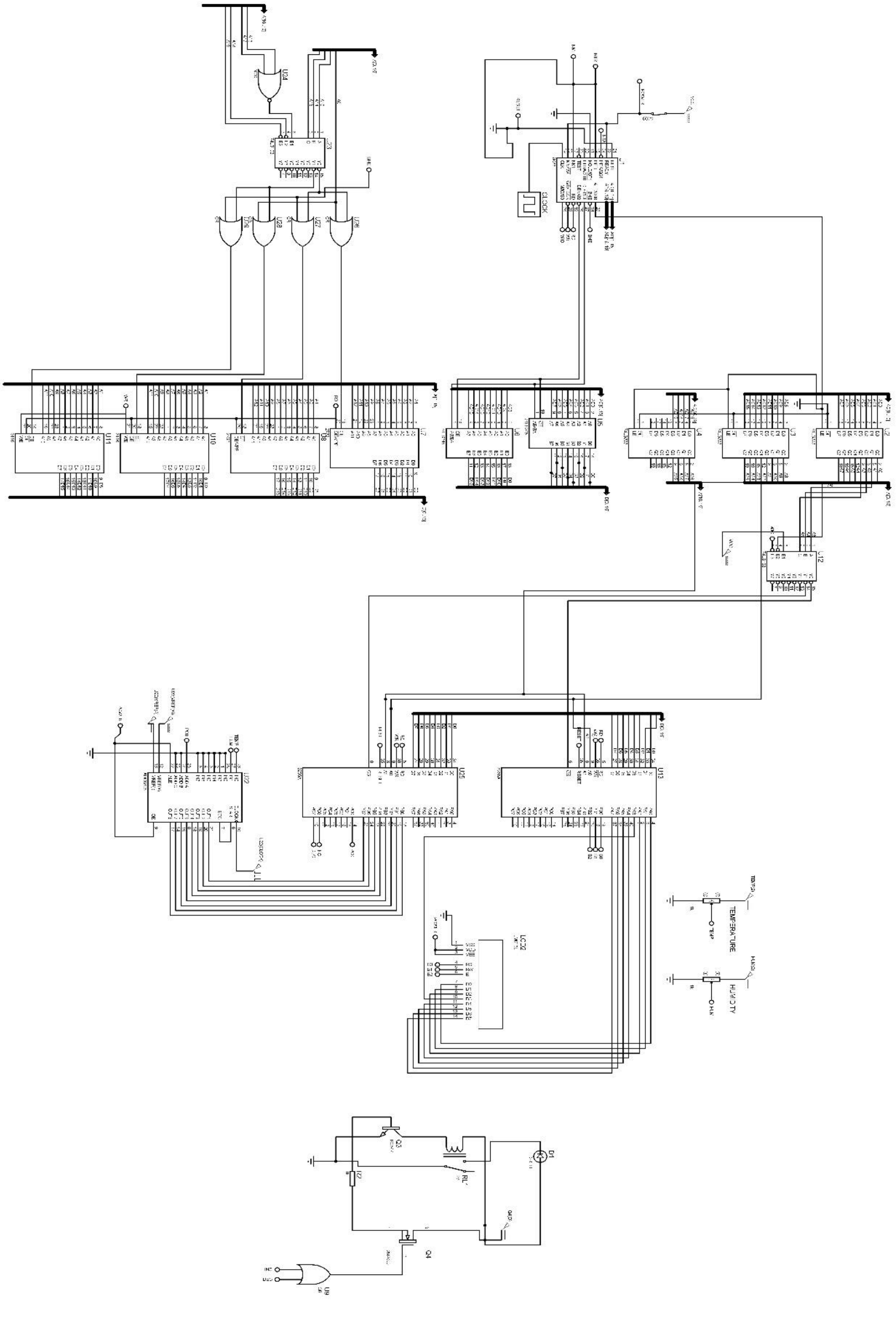
```
    out 02h,al
    call dly_minor
    mov al,01h
    out 02h,al  ;prints number store in bl

    mov al,0DFh
    out 00h,al
    call dly_minor
    mov al,05h
    out 02h,al
    call dly_minor
    mov al,01h
    out 02h,al  ;prints 'DEG'

    mov al,43h
    out 00h,al
    call dly_minor
    mov al,05h
    out 02h,al
    call dly_minor
    mov al,01h
    out 02h,al  ;prints 'C'

    mov al,0A0h
    out 00h,al
    call dly_minor
    mov al,05h
    out 02h,al
    call dly_minor
    mov al,01h
    out 02h,al  ;prints Space

mov al,0c0h
    out 00h,al
    mov al,04h
    out 02h,al
    call dly_minor
    mov al,00h
    out 02h,al
        call dly_minor


    mov al,48h
    out 00h,al
    call dly_minor
    mov al,05h
    out 02h,al
    call dly_minor
    mov al,01h
    out 02h,al  ;prints 'H'
```

```
mov al,75h
out 00h,al
call dly_minor
mov al,05h
out 02h,al
call dly_minor
mov al,01h
out 02h,al  ;prints 'u'

mov al,6dh
out 00h,al
call dly_minor
mov al,05h
out 02h,al
call dly_minor
mov al,01h
out 02h,al  ;prints 'm'

mov al,69h
out 00h,al
call dly_minor
mov al,05h
out 02h,al
call dly_minor
mov al,01h
out 02h,al  ;prints 'i'

mov al,64h
out 00h,al
call dly_minor
mov al,05h
out 02h,al
call dly_minor
mov al,01h
out 02h,al  ;prints 'd'

mov al,69h
out 00h,al
call dly_minor
mov al,05h
out 02h,al
call dly_minor
mov al,01h
out 02h,al  ;prints 'i'

mov al,74h
out 00h,al
call dly_minor
mov al,05h
```

```
out 02h,al
call dly_minor
mov al,01h
out 02h,al  ;prints 't'

mov al,79h
out 00h,al
call dly_minor
mov al,05h
out 02h,al
call dly_minor
mov al,01h
out 02h,al  ;prints 'y'


mov al,0A0h
out 00h,al
call dly_minor
mov al,05h
out 02h,al
call dly_minor
mov al,01h
out 02h,al  ;prints Space

mov al,0A0h
out 00h,al
call dly_minor
mov al,05h
out 02h,al
call dly_minor
mov al,01h
out 02h,al  ;prints Space

mov al,0A0h
out 00h,al
call dly_minor
mov al,05h
out 02h,al
call dly_minor
mov al,01h
out 02h,al  ;prints Space


mov al,0A0h
out 00h,al
call dly_minor
mov al,05h
out 02h,al
call dly_minor
mov al,01h
```

```
            out 02h,al  ;prints Space


            mov al,0A0h
            out 00h,al
            call dly_minor
            mov al,05h
            out 02h,al
            call dly_minor
            mov al,01h
            out 02h,al  ;prints Space

            mov al,dh
            out 00h,al
            call dly_minor
            mov al,05h
            out 02h,al
            call dly_minor
            mov al,01h
            out 02h,al  ;prints character in dh

            mov al,dl
            out 00h,al
            call dly_minor
            mov al,05h
            out 02h,al
            call dly_minor
            mov al,01h
            out 02h,al  ;prints character in dl

            mov al,25h
            out 00h,al
            call dly_minor
            mov al,05h
            out 02h,al
            call dly_minor
            mov al,01h
            out 02h,al  ;prints '%'
            ret
display_lcd    endp

CONVBCD PROC     NEAR                    ;convert binary to bcd
        mov     bh,0ffH

BACK1:  INC     BH
        SUB     AL,0AH
        JNC     BACK1
        ADD     AL,0AH
        MOV     BL,30H
        ADD     BH,BL
        ADD     BL,AL
        RET
CONVBCD ENDP
```

b. CIRCUIT DIAGRAM



c. REFERENCES

A 16x2 LCD was used. The following were used to understand its working.

1. https://panda-bg.com/datasheet/2134-091834-LCD-module-TC1602D-02WA0-16x2-STN.pdf
2. https://www.csus.edu/indiv/p/pangj/class/lcd/instruct.html