CONTROLLING DC FAN USING MICROPROCESSOR

Hardware Design and Assembly Level Programme

Report is completed in partial fulfilment of the course

Microprocessor Programming and Interfacing (CS F241)

Team Members:

1) Abhishek V Joshi – 2015A7PS116P
2) Ananyashree Garg – 2015A7PS117P
3) Anand Chordia – 2015A7PS118P
4) Karri Venkat Reddy – 2015A7PS119P

Problem Statement:

Description: This system senses the speed at which the fan is rotating and adjusts the speed, based on the user input. The user can select three different speeds of the fan. The current speed should be sensed and the control mechanism should gradually increase the speed to the desired speed.

User Interface:

1. Fan starts when user presses 'Start' button.
2. User can then set the required speed by using a keypad interface. This speed value should be displayed on the display.
3. After setting speed initially, user should be able to change the fan speed setting by an up and down switch. Each press on this arrow button increases/ decreases the speed by 1 unit. Min speed value is 1, whereas maximum speed value is 5 Units. Pressing 'UP' button after reaching to value.

5, should not change the display value or setting of fan speed. Same is true for lower bound.

4. Fan can be stopped by pressing 'Stop' button.
5. User can also set the mode of fan as 'Auto' mode besides a 'Regular mode' setting. In Auto mode, user should be able to enter the value of time in terms of hours after which the Fan has to be switched off automatically. (For example, if value entered is 2, then the Fan should switch off after 2 hours from the time this setting is applied

Assumptions:

1) CLK pin of the micro-processor is connected to a reliable clock generator to produce waveform of suitable frequency.
2) User can start the fan only by pressing the start button. No other key will function until the fan has been started.
3) The fan starts at speed 1 on press of the start button.
4) Pressing the start button after the fan has started results in no change of the system state.

5) User can then set the speed of the fan by pressing any of the push buttons numbered from 1 to 5.
6) Pressing button number 0 essentially serves the same purpose as that of the stop button.
7) Auto mode runs only at one speed. (Speed 3)
8) Hours in the Auto Mode have been scaled down to Seconds for the sake of simplicity of demonstration.
9) Display doesn't change when working in the auto mode.
10) Auto mode allows users to enter numbers from 0 to 10.

List of Hardware Used:

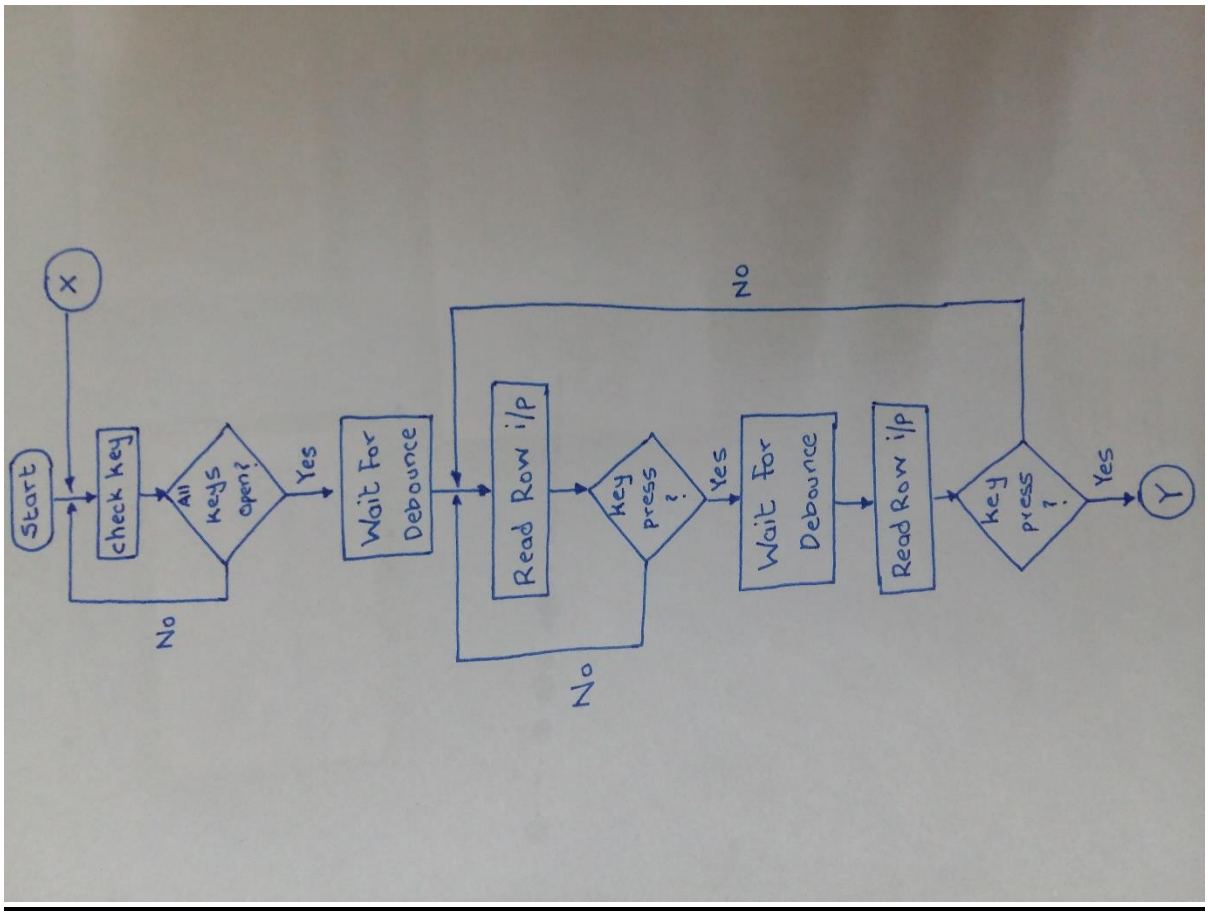
1) 8086 Micro-Processor = 1
2) 74LS373 Octal Latches = 3
3) 74LS138 3:8 Decoder = 1
4) 8255A Programmable Peripheral Interfacing Device = 1
5) 7SEG-COM-CAT-BLUE Display = 1
6) DAC_8 Digital to Analog Converter = 1
7) DC Fan = 1
8) Push Button Switches = 16
9) 74LS245 Bi-Directional Buffer = 2
10) 2732 SROM Chips (2KB each) = 2
11) 6116 RAM Chips (2KB each) = 2
12) OR Gates = 6
13) NOT Gates = 3
14) SPDT Switch = 1
15) Ground Terminal = as required
16) DC Voltage Sources = as required

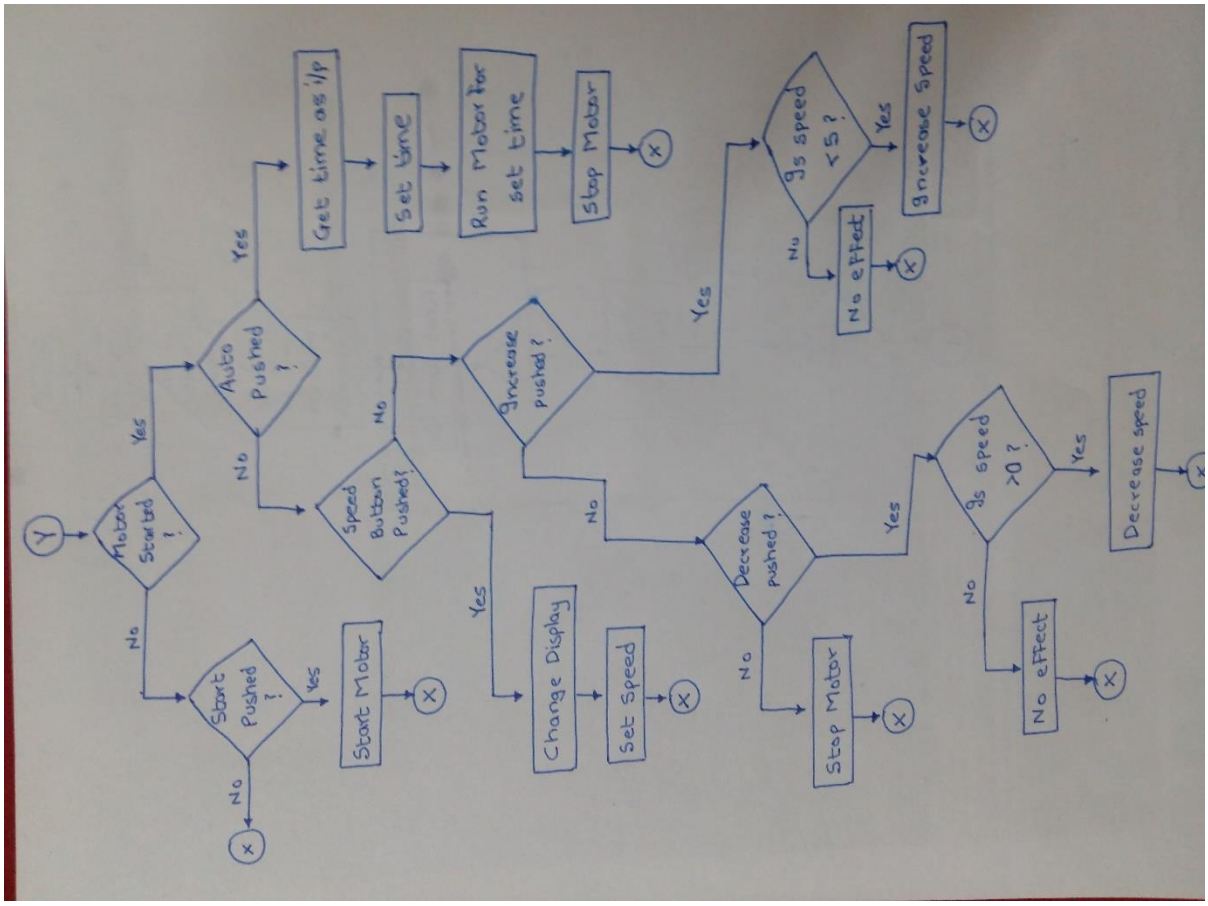
Memory Addressing:

Item	Address	Details

RAM (Memory)	02000H – 02FFFFH	2 * 2 KB Chips
ROM (Memory)	00000H – 1FFFFH	2 * 2 KB Chips
Port A (8255A)	00H	Output (DC Fan)
Port B (8255A)	02H	Output (7 Segment Display)
Port C (8255A)	04H	Input/Output (Keypad)
Control Register	06H	For loading the Control Word

Flow Chart:





CODE:

```
.model tiny

.data


table_kbrd  dw  0eeh, 0edh, 0ebh, 0e7h, 0deh, 0ddh, 0dbh, 0d7h, 0beh, 0bdh, 0bbh, 0b7h, 7eh, 7dh, 7bh, 77h

table_dis   db  3fh, 06h, 5bh, 4fh, 66h, 6dh;, 007dh, 0027h, 007fh, 006fh, 0077h, 007ch, 0039h,  005eh, 0079h, 0071h

speed       dw  00h

started     dw  00h

auto_speed  dw  03h

time        dw  00h

auto_flag   dw  00h
```

```
.code
.startup

   porta   equ     00h ;setting the 8253a ports
   portb   equ     02h
   portc   equ     04h
   creg    equ     06h

   ;initializing the ports of 8253a
   mov     al, 88h
   out     creg, al
x0: mov     al, 00h
   out     portc, al
x1: in      al, portc
   and     al, 0f0h
   cmp     al, 0f0h    ;check for key release
   jnz     x1
   call    delay_20ms ;debounce

   mov     al, 00h
   out     portc, al
x2: in      al, portc
   and     al, 0f0h
   cmp     al, 0f0h
   jz      x2
   call    delay_20ms ;debounce

   ;checking the validity of key press
```

```
mov     al, 00h
out     portc, al
in      al, portc
and     al, 0f0h
cmp     al, 0f0h
jz      x2

;check for key press column 1
mov     al, 0eh
mov     bl, al
out     portc, al
in      al, portc
and     al, 0f0h
cmp     al, 0f0h
jnz     x3

;check for key press column 2
mov     al, 0dh
mov     bl, al
out     portc, al
in      al, portc
and     al, 0f0h
cmp     al, 0f0h
jnz     x3

;check for key press column 3
mov     al, 0bh
mov     bl, al
out     portc, al
```

```
    in      al, portc
    and     al, 0f0h
    cmp     al, 0f0h
    jnz     x3

    ;check for key press column 4
    mov     al, 07h
    mov     bl, al
    out     portc, al
    in      al, portc
    and     al, 0f0h
    cmp     al, 0f0h
    jz      x2

    ;decode key
x3: or      al, bl
    mov     cx, 0fh
    mov     di, 00h
    lea     di, ds:table_kbrd
x4: cmp     al, [di]
    jz      x5
    inc     di
    loop    x4

    ;checks if motor is started
x5: cmp     started, 01h
    jz      auto
    cmp     al, 0b7h ;button encoding for start
    jz      start
```

```
    jmp     x0



start: call start_fan
    jmp x0

auto:   cmp al, 0b7h
    jz  x0
    cmp     auto_flag, 01h
    jz      time_set
    cmp     al, 77h
    jnz     speed_check
    mov     auto_flag, 01h
    jmp     x0

time_set:   cmp     al, 0eeh; checks which speed
    call    stop
    jmp     x0
    cmp     al, 0edh
    jz      set_time_1
    cmp     al, 0ebh
    jz      set_time_2
    cmp     al, 0e7h
    jz      set_time_3
    cmp     al, 0deh
    jz      set_time_4
    cmp     al, 0ddh
    jz      set_time_5
    cmp     al, 0dbh
```

```
    jz     set_time_6
    cmp    al, 0d7h
    jz     set_time_7
    cmp    al, 0beh
    jz     set_time_8
    cmp    al, 0bdh
    jz     set_time_9
    cmp    al, 0bbh
    jz     set_time_10
    jmp    x0

set_time_1:    mov    time, 01h
    call   set_time
    call   stop
    jmp    x0

set_time_2:    mov    time, 02h
    call   set_time
    call   stop
    jmp    x0

set_time_3:    mov    time, 03h
    call   set_time
    call   stop
    jmp    x0

set_time_4:    mov    time, 04h
    call   set_time
    call   stop
```

```
    jmp    x0

set_time_5:    mov    time, 05h
    call   set_time
    call   stop
    jmp    x0

set_time_6:    mov    time, 06h
    call   set_time
    call   stop
    jmp    x0

set_time_7:    mov    time, 07h
    call   set_time
    call   stop
    jmp    x0

set_time_8:    mov    time, 08h
    call   set_time
    call   stop
    jmp    x0

set_time_9:    mov    time, 09h
    call   set_time
    call   stop
    jmp    x0

set_time_10:    mov    time, 0ah
    call   set_time
```

```
   call    stop
   jmp     x0

;checks which speed to set and sets it
speed_check:    lea     bx, DS:table_dis
   cmp  al, 0eeh
   jz      set_speed_0
   cmp     al, 0edh
   jz      set_speed_1
   cmp     al, 0ebh
   jz      set_speed_2
   cmp     al, 0e7h
   jz      set_speed_3
   cmp     al, 0deh
   jz      set_speed_4
   cmp             al, 0ddh
   jz      set_speed_5
   jmp     increase

set_speed_0:  mov               speed, 00h
          call      stop
          mov               al, 3fh
          not               al
          out               portb, al
          jmp               x0

set_speed_1:    mov     speed, 01h
   call    set_speed
   ;mov            al, 06h
```

```
    mov     al, [bx + 01h]
            not             al
    out     portb, al
    jmp     x0

set_speed_2:    mov     speed, 02h
    call    set_speed
    ;mov            al, 5bh
    mov     al, byteptr[bx + 02h]
            not             al
    out     portb, al
    jmp     x0

set_speed_3:    mov     speed, 03h
    call    set_speed
    ;mov            al, 4fh
    mov     al, byteptr[bx + 03h]
            not             al
    out     portb, al
    jmp     x0

set_speed_4:    mov     speed, 04h
    call    set_speed
    ;mov            al, 66h
    mov     al, byteptr[bx + 04h]
            not             al
    out     portb, al
    jmp     x0
```

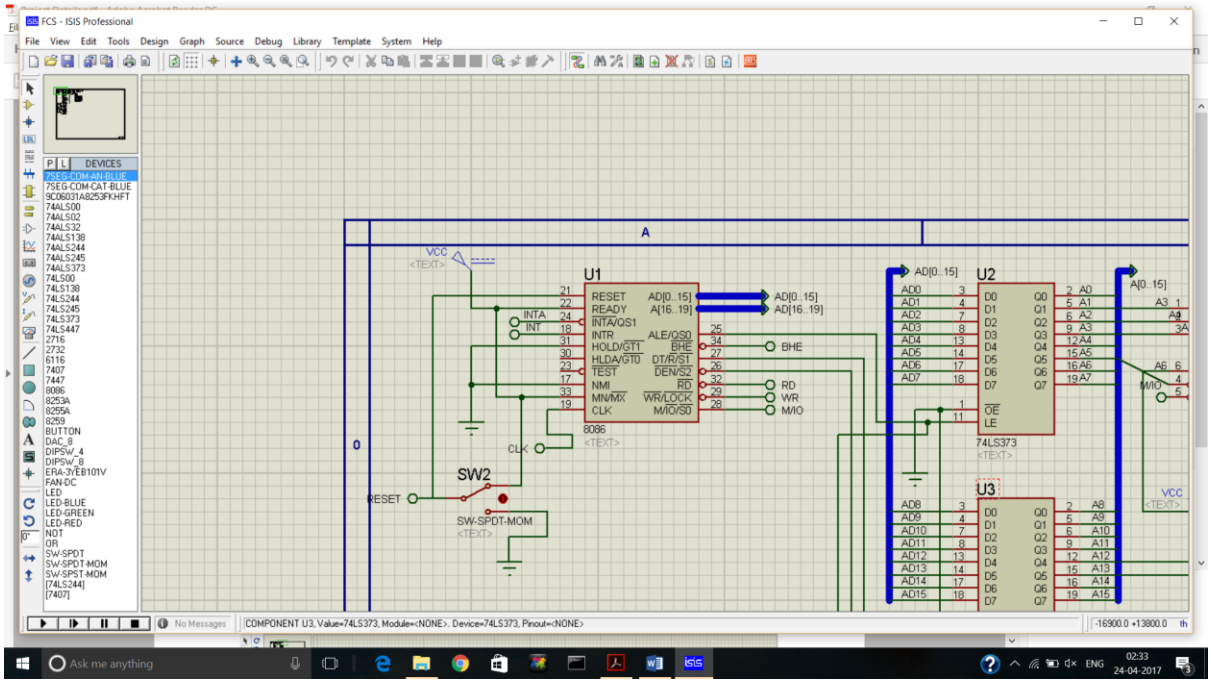
```
set_speed_5:    mov     speed, 05h
   call    set_speed
   ;mov            al, 6dh
   mov     al, byteptr[bx + 05h]
           not             al
   out     portb, al
   jmp     x0

increase:   lea     bx, DS:table_dis
           cmp     al, 7dh
   jnz     decrease
   call    incr
   call    set_speed
   jmp     x0

decrease:   cmp     al, 7bh
   jnz     stop_fan
   call    decr
   call    set_speed
   jmp     x0


stop_fan:  cmp      al, 7eh
   jnz     x0
   call   stop
   mov            al, 3fh
           not             al
           out             portb, al
   jmp     x0
```

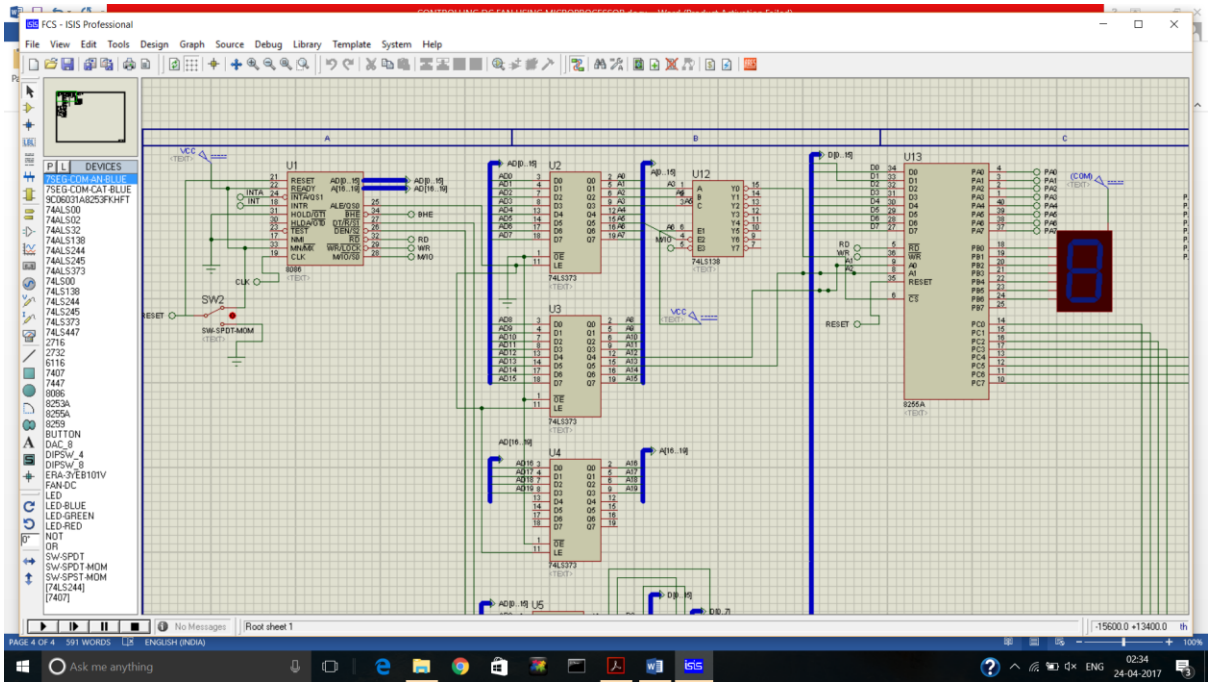
```
.exit

;Delay of 20ms
delay_20ms proc near
   mov     cx, 2220
x9: loop    x9
   ret
delay_20ms endp

;starts the fan on speed 1 on press of start button
start_fan proc near
   mov     speed, 01h
   mov     al, 33h
   out     porta, al
   mov     started, 01h
   ret
start_fan endp

;stops the fan on press of stop button
stop proc near
   mov     speed, 00h
   mov     al, 00h
   out     porta, al
   mov     started, 00h
   mov     auto_flag, 00h
   ret
stop endp
```

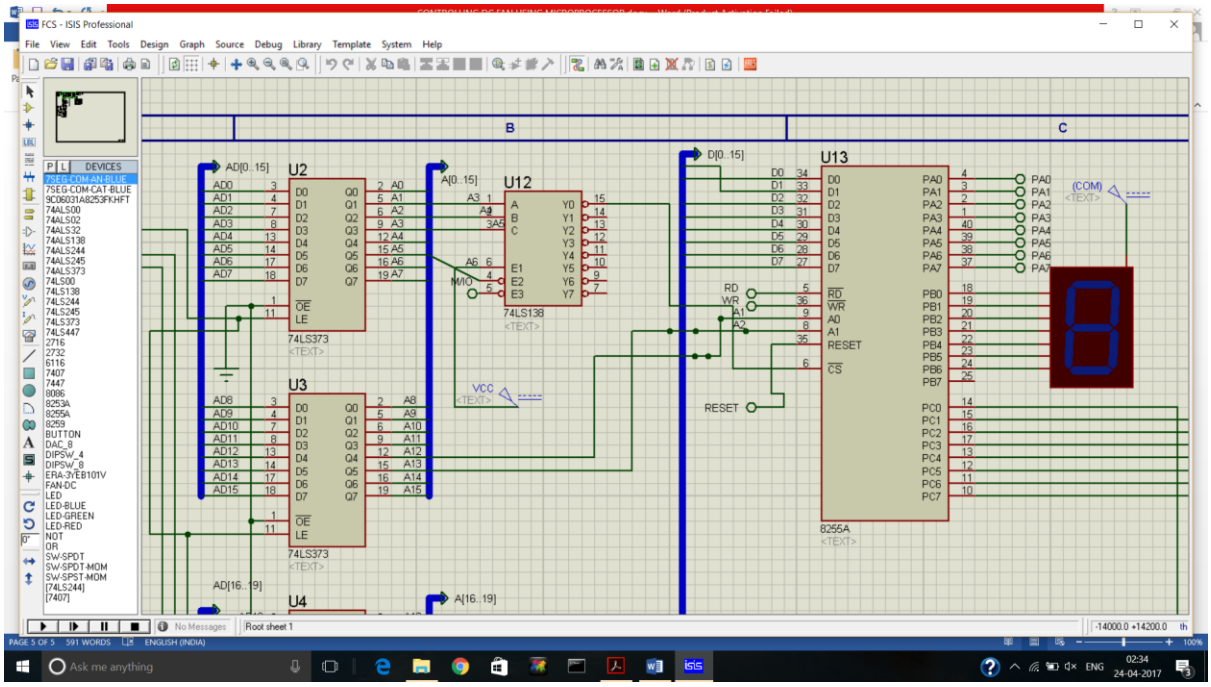
```
;sets the required speed
set_speed proc near
   mov     cx, speed
   mov     al, 00h
x:  add     al, 33h
   loop    x
   out     porta, al
   ret
set_speed endp

set_time proc near
   mov     cx, auto_speed
   mov     al, 00h
x8: add     al, 33h
   loop    x8
   out     porta, al

   mov     bx, time
loop3:  mov     dx, 50
loop2:  mov     cx, 2220
loop1:  nop
   dec     cx
   jnz     loop1
   dec     dx
   jnz     loop2
   dec     bx
   jnz     loop3

   ret
```

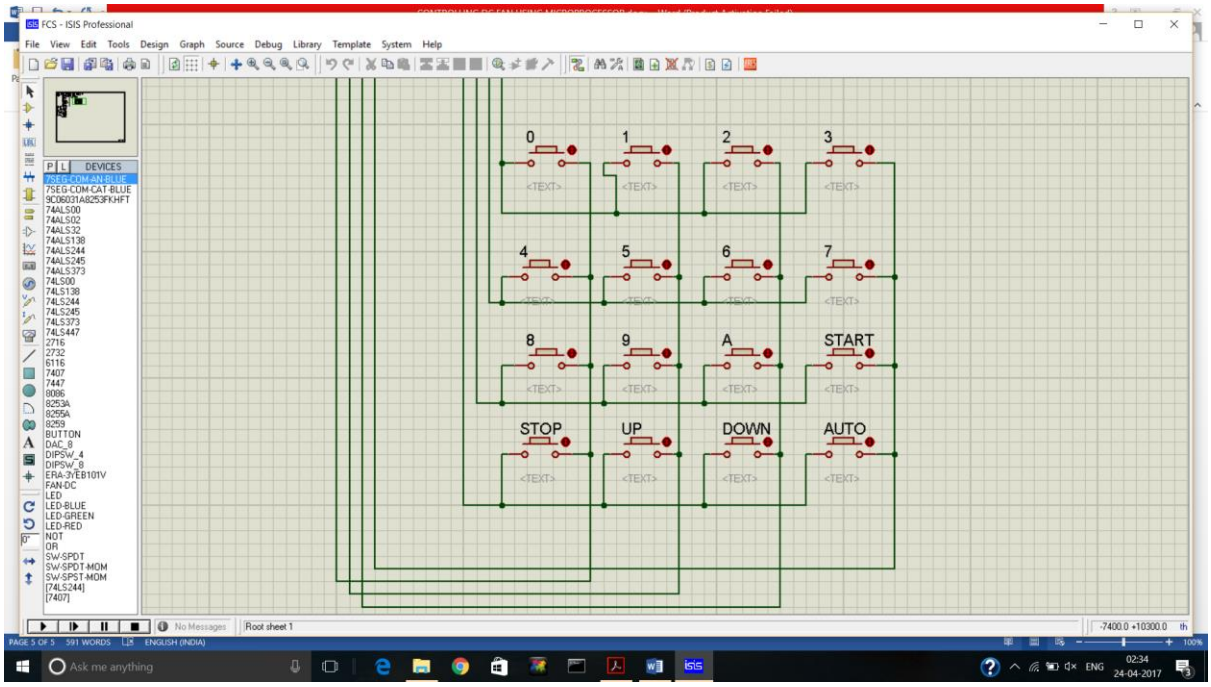
```
set_time endp

incr proc near
   cmp     speed, 05h
   jge     x6
   inc     speed
   mov     cx, speed
   add     bx, cx
   mov     al, [bx]
   not               al
   out     portb, al
x6: ret
incr endp

decr proc near
   cmp     speed, 00h
   jbe     x7
   dec     speed
   mov     cx, speed
   add     bx, cx
   mov     al, [bx]
   not               al
   out     portb, al
x7: ret
decr endp

end
```

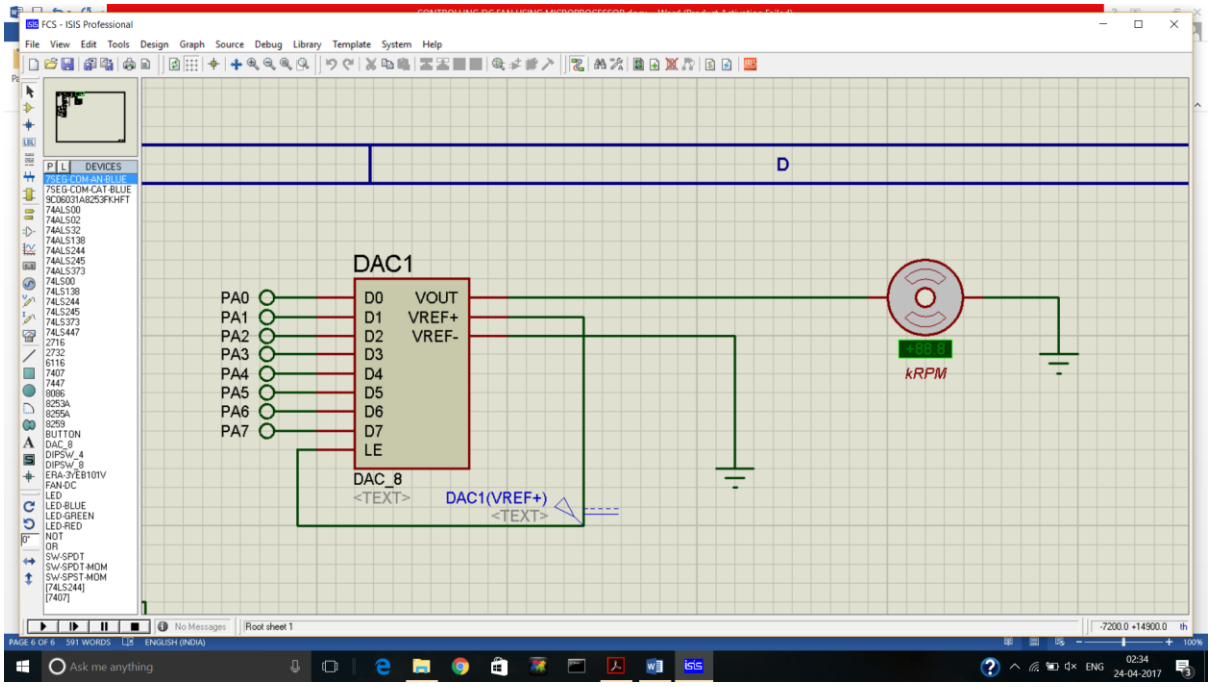
Screenshots of the Design:

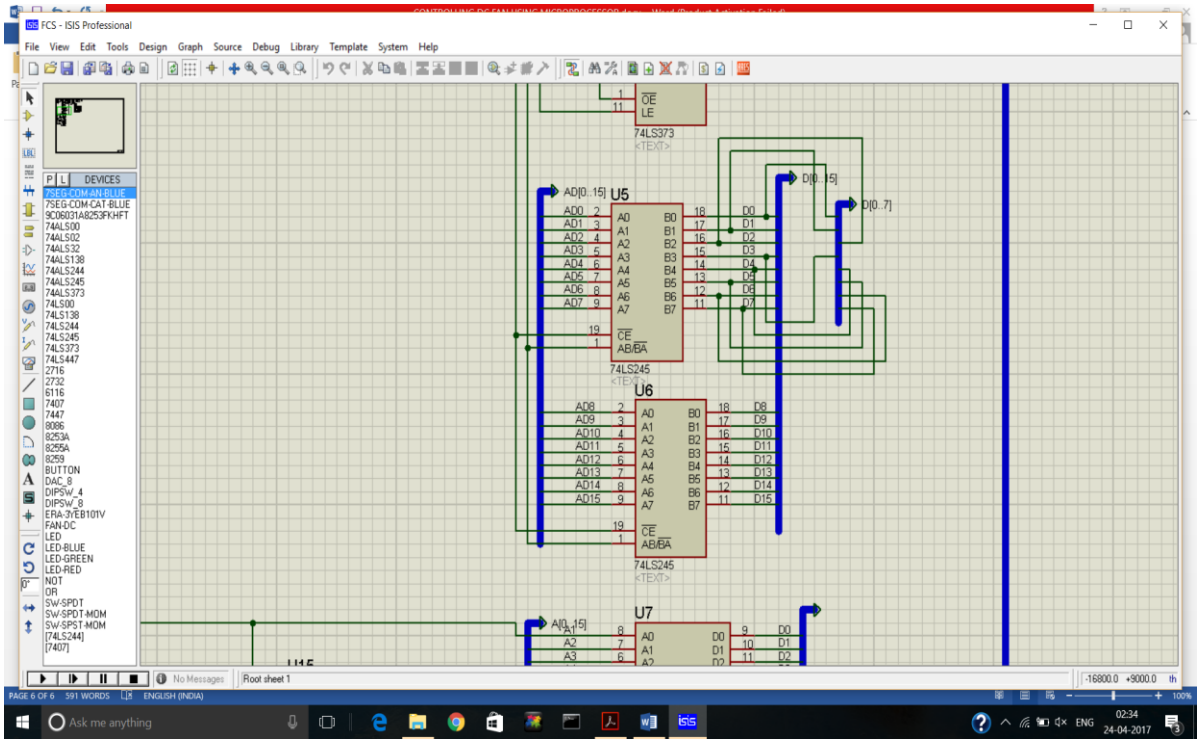


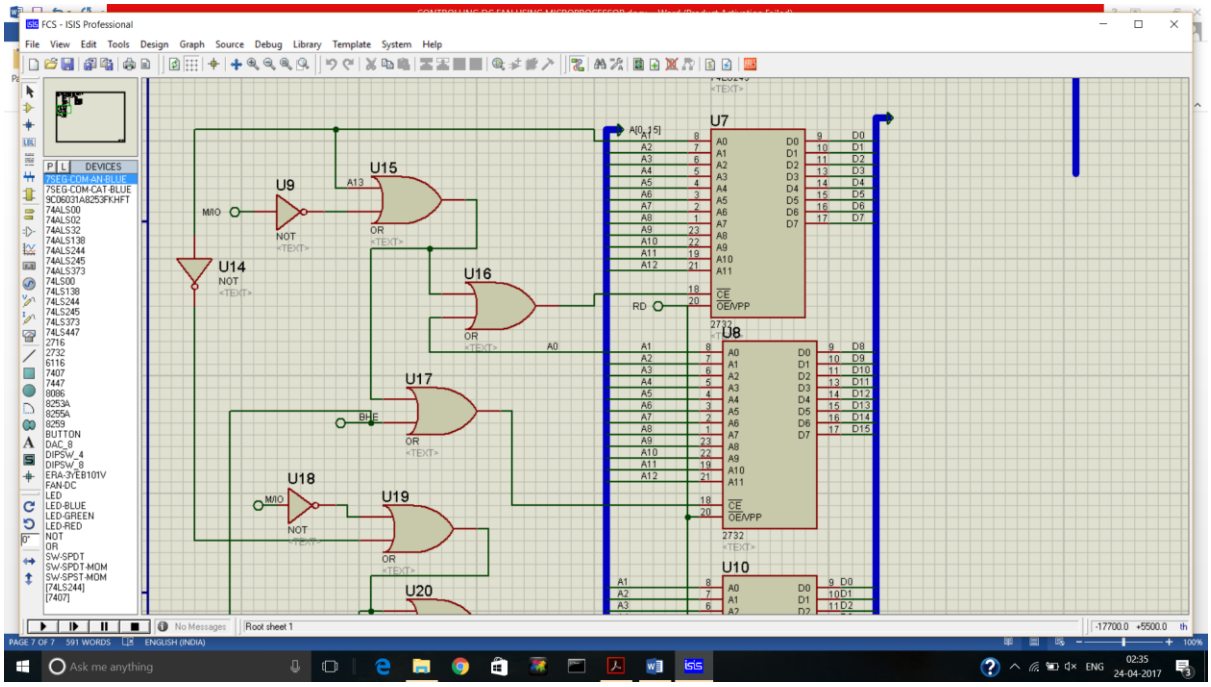


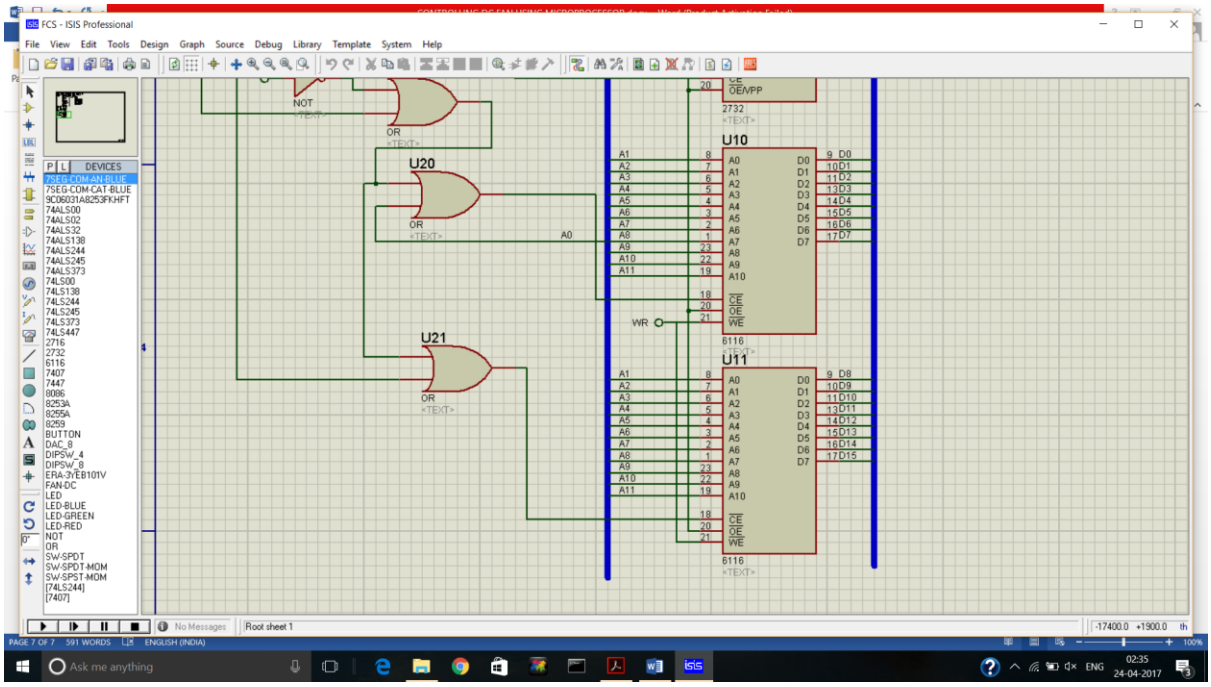

FCS - ISIS Professional
File View Edit Tools Design Graph Source Debug Library Template System Help
B
C
U2
U3
U4
U12
U13
74LS373
74LS138
8255A
AD[0..15]
A[0..15]
D[0..15]
AD[16..19]
A[16..19]
RD
WR
RESET
VCC
(COM)
DEVICES
Root sheet 1
No Messages
Ask me anything
ENG
02:34
24-04-2017


FCS - ISIS Professional
File View Edit Tools Design Graph Source Debug Library Template System Help
0
1
2
3
4
5
6
7
8
9
A
START
STOP
UP
DOWN
AUTO
DEVICES
Root sheet 1
No Messages
Ask me anything
ENG
02:34
24-04-2017


FCS - ISIS Professional
File View Edit Tools Design Graph Source Debug Library Template System Help
D
DAC1
PA0
PA1
PA2
PA3
PA4
PA5
PA6
PA7
D0
D1
D2
D3
D4
D5
D6
D7
LE
VOUT
VREF+
VREF-
DAC_8
<TEXT>
DAC1(VREF+)
<TEXT>
kRPM
DEVICES
Root sheet 1
No Messages
Ask me anything
ENG
02:34
24-04-2017


FCS - ISIS Professional
File View Edit Tools Design Graph Source Debug Library Template System Help
74LS373
U5
U6
U7
74LS245
AD[0..15]
D[0..15]
D[0..7]
DEVICES
Root sheet 1
No Messages
Ask me anything
ENG
02:34
24-04-2017


FCS - ISIS Professional
File View Edit Tools Design Graph Source Debug Library Template System Help
U7
U8
U9
U10
U14
U15
U16
U17
U18
U19
U20
M/IO
A13
A0
BHE
RD
NOT
OR
2732
Root sheet 1
No Messages
Ask me anything
ENG
02:35
24-04-2017


FCS - ISIS Professional
File View Edit Tools Design Graph Source Debug Library Template System Help
U10
U11
U20
U21
A0
WR
NOT
OR
2732
6116
Root sheet 1
No Messages
Ask me anything
ENG
02:35
24-04-2017