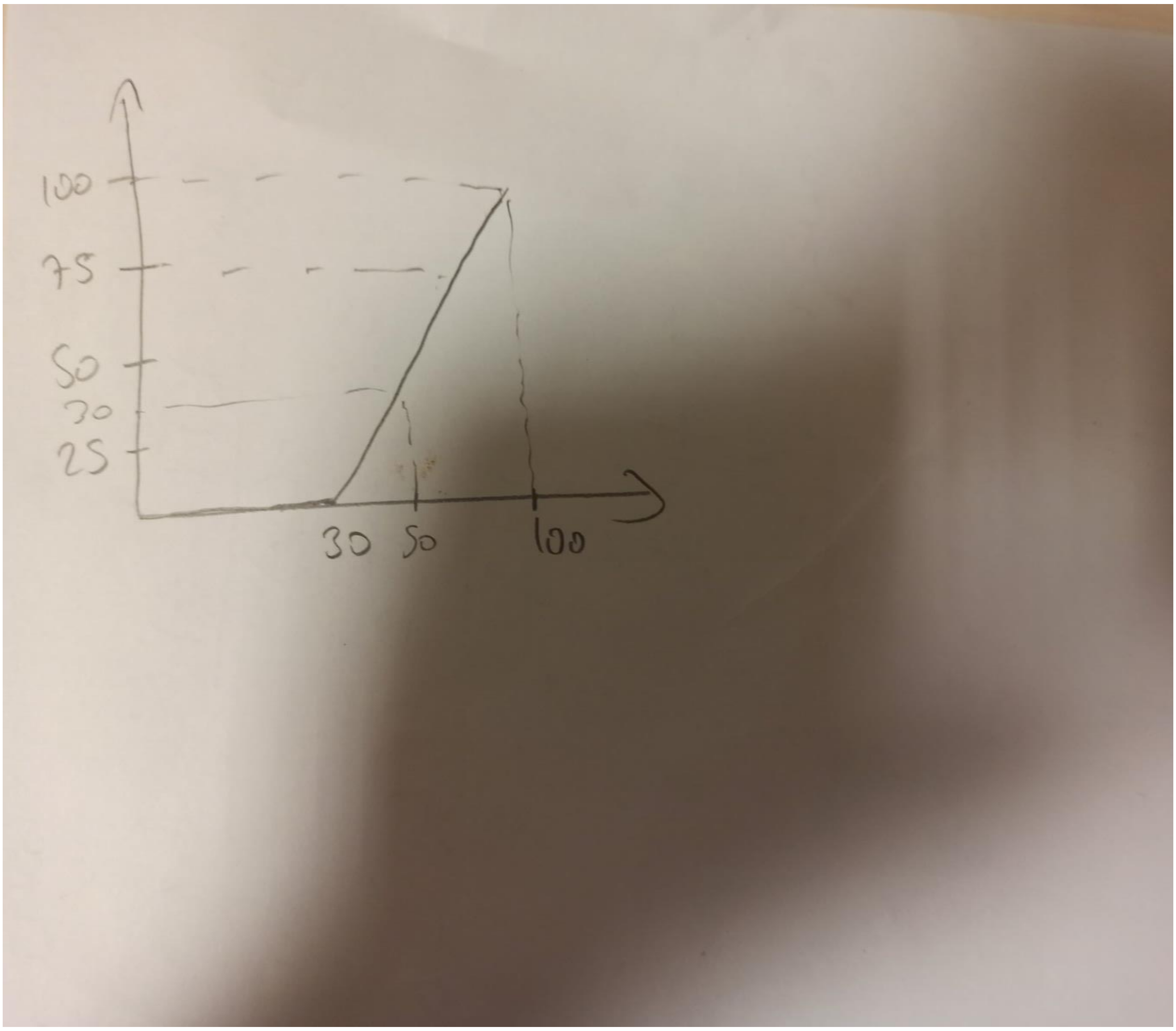
Cng 336

Lab Final Prework

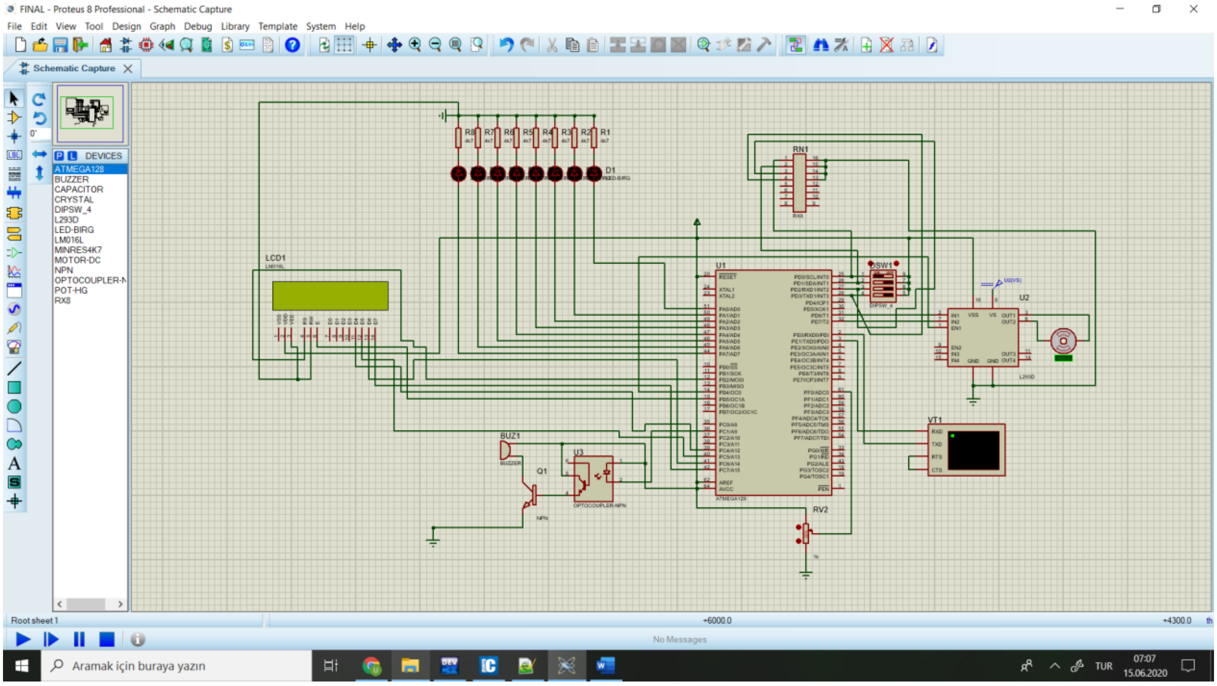
Ulaş Cem Erten - 2256006

6.2.3)



In above picture y axis shows the fan speed, x axis shows the temperature. Under 30 C, fan speed is zero. And Its working with max speed in 100 C with 100 percent fan speed.

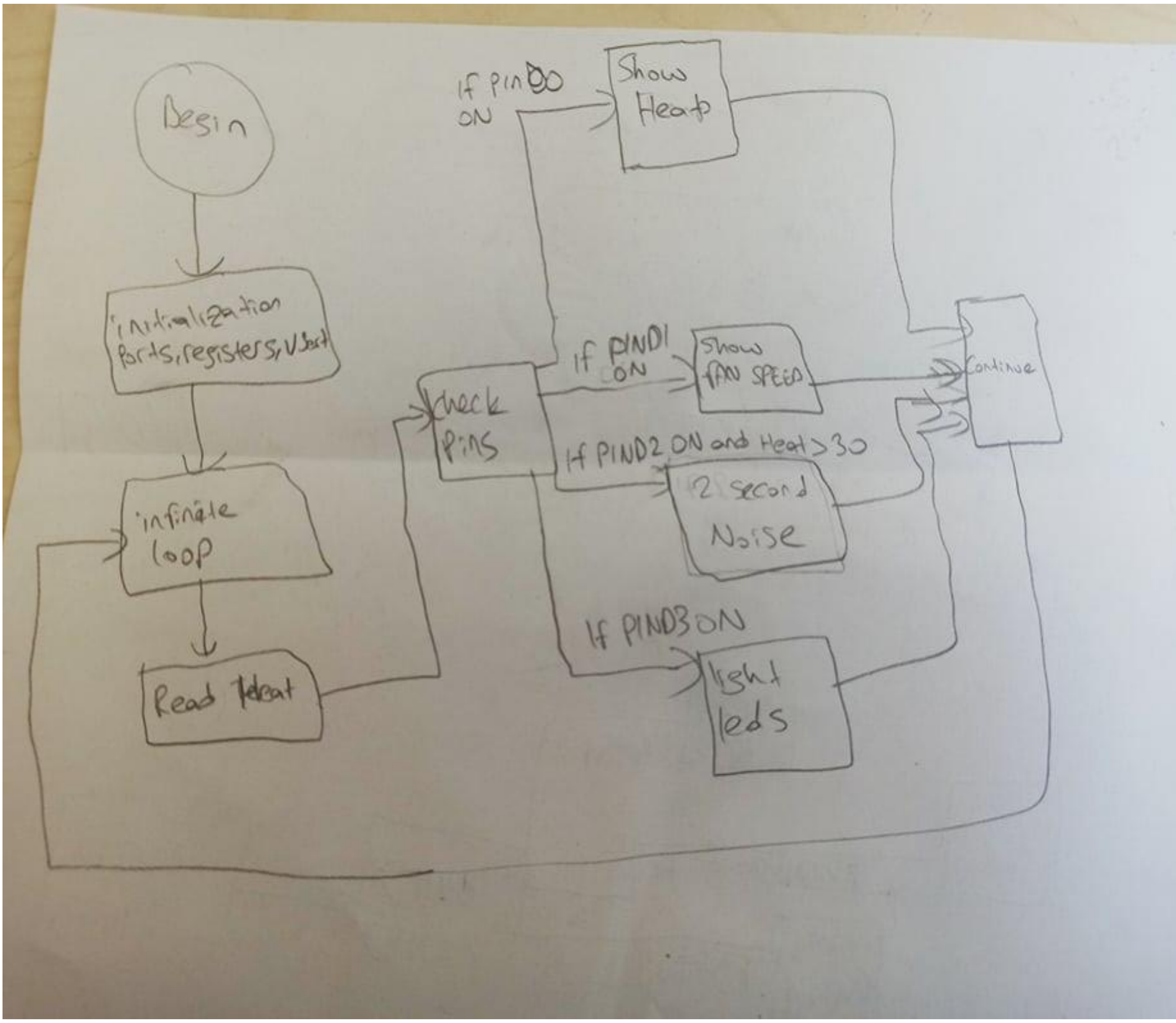
Schematic



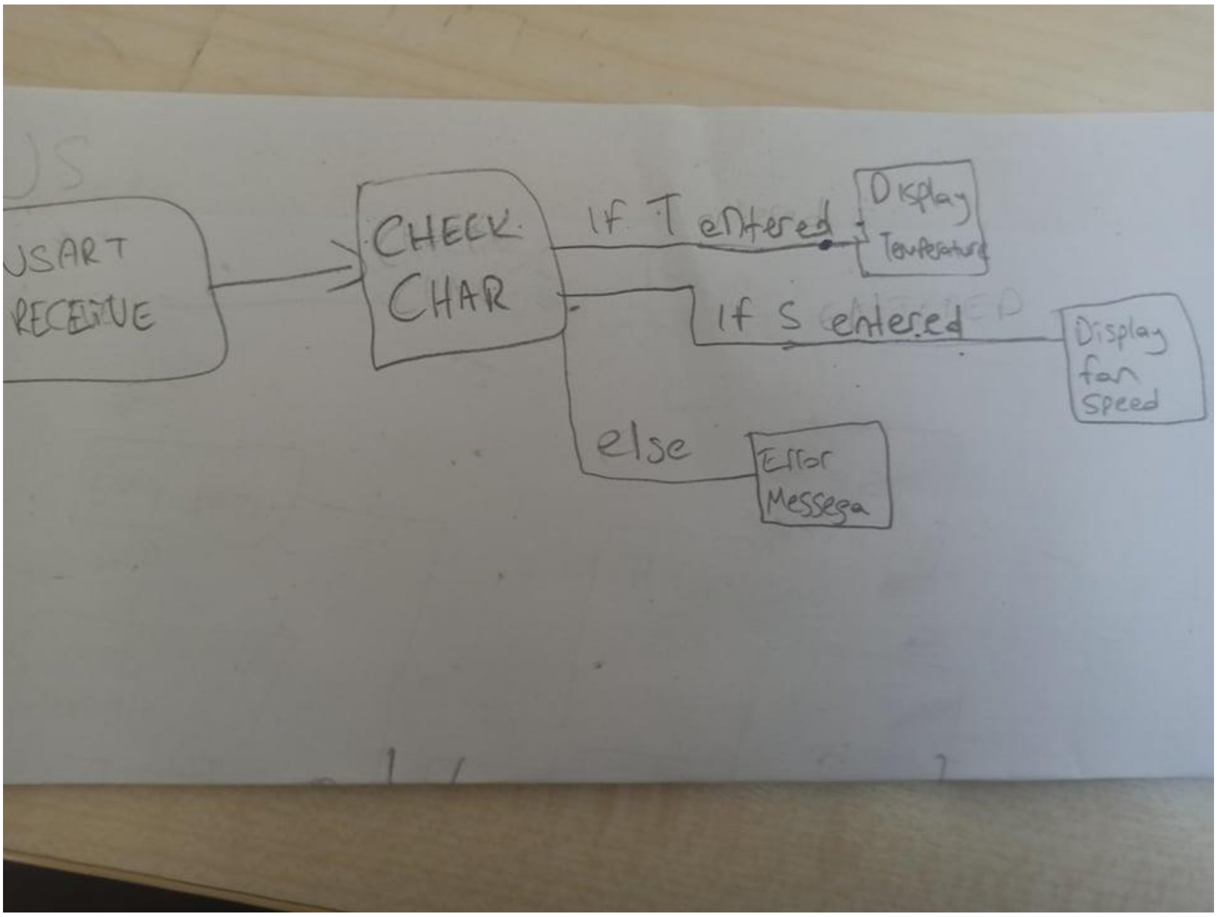
I used PE0 and PE1 to communicate with terminal. I used PD0 for LCD Line 1 select for Temperature display on LCD, PD1 for LCD Line 2 Select for Speed Display on LCD, PD3 for LEDs Display select on PORTA and PD2 for Buzzer noise select. I used PC1 for Buzzer. I used PB4 to deliver PWM signal to the motor driver. I used PB2 for LCD RS, PB5 for LCD En, PC4 to PC7 for LCD Data delivery.

Flow Diagram

Main Module



Usart Module



Source Code

```
sbit LCD_RS at PORTB2_bit;

sbit LCD_EN at PORTB5_bit;

sbit LCD_D4 at PORTC4_bit;

sbit LCD_D5 at PORTC5_bit;

sbit LCD_D6 at PORTC6_bit; // lcd init

sbit LCD_D7 at PORTC7_bit;

sbit LCD_RS_Direction at DDB2_bit;
sbit LCD_EN_Direction at DDB5_bit;
sbit LCD_D4_Direction at DDC4_bit;
sbit LCD_D5_Direction at DDC5_bit;
sbit LCD_D6_Direction at DDC6_bit;
sbit LCD_D7_Direction at DDC7_bit;



void send_char(unsigned char my_char){ // send char function


    while (!((UCSR0A) & (1 << (UDRE0))));
     UDR0 = my_char;
    }


void send_string(unsigned char* my_string){ // send string function


     while(*my_string)
      send_char(*my_string++);
     }


unsigned char* my_data = 0x500;
void clear_every_thing(unsigned char* my_data, unsigned int ADDR)
   {
          int i = 0;  // it clears everything in the given adress i used to clear usart
          memset(my_data, 0, 0x50);
   my_data = ADDR;
          }


unsigned char* int_to_String(unsigned int x) { //int to string convertion function
   unsigned char converted[4]="";
          unsigned int initial_value = x;
          char temp_val;
```

```
if(initial_value==100){ // look up table for 3 digit number and 1 digit number
        return "100";
          }
          if(initial_value==0){
        return "0";
          }
          if(initial_value==1){
        return "1";
          }
          if(initial_value==2){
        return "2";
          }
          if(initial_value==3){
        return "3";
          }
          if(initial_value==4){
        return "4";
          }
          if(initial_value==5){
        return "5";
          }
          if(initial_value==6){
        return "6";
          }
           if(initial_value==7){
        return "7";
          }
          if(initial_value==8){
        return "8";
          }
```

```
      if(initial_value==9){
     return "9";
      }
     if(x==0)
{
     *converted = '0';
*(converted + 1) = '\0';
return "";
 }
else
 {
      unsigned int i = 0;
 while(x)
 {
      unsigned char d = x % 10;
 x /= 10;
 if(d>9)
 converted[i++]=(d + 55);// to find its ascii if it is a letter
 else
 converted[i++]=(d + 48);
}

      converted[i] = '\0';



             temp_val = converted[0]; //swap operation because now it is in opposite order
             converted[0] = converted[1];
             converted[1] = temp_val;



     return converted;
```

```
   }
}

 unsigned char waves = 0;
 void start_noise() iv 0x001E{ // create square wave on portb 4 for making noise if timer 0 compare
handle

     PORTB.B4 = waves ^ 1;

            }

unsigned int sum_twos = 0;
 void stop_noise() iv 0x0014 // for stopping noise when timer2 overflow
   {++sum_twos; // increment until it becomes 2 sec

    }

unsigned char temp;
int my_flag;
unsigned char text_for_usart[0x50];
unsigned char temp_for_usart[4];
unsigned int unit_select;
unsigned int fan_speed = 0;
unsigned int heat_value = 0;
void usart() iv 0x0024{
            unsigned int enter = 0;
            temp = UDR0;

   if(temp == '\n' || temp == '\r') // check if enter is pressed
      enter = 1;
   else
   {if((unsigned int) my_data < 0x530) {
     *my_data++ = temp;
     *my_data = '\0';}
}
UDR0 = temp;



if(enter)
  {my_data = 0x500;


               my_flag=1; // flag for adding the unit  % or C


                        if(!(strcmp(my_data,"T"))){ // if equals will return 0
                                          unit_select=1; //choose % or C
           strcpy(text_for_usart, "TEMPERATURE: "); // basic copy string
           strcpy(temp_for_usart,int_to_String(heat_value));
           strcpy(text_for_usart + strlen("TEMPERATURE: "), temp_for_usart);



                        }
                        else if(!(strcmp(my_data,"S"))){
                         unit_select=2;//same as above
           strcpy(text_for_usart, "FAN SPEED: ");
           strcpy(temp_for_usart,int_to_String(fan_speed));
           strcpy(text_for_usart + strlen("FAN SPEED: "), temp_for_usart);



                                    }
           else{


               send_string("Wrong input entered, Please "); // another input
```

```
                        my_flag=0; // flag will choose wrong input message
                                                }
                                                if(my_flag==1){
                                                        if(unit_select==1)
                                                        strcpy(text_for_usart + strlen("TEMPERATURE: ") +
strlen(temp_for_usart), "C\r"); //explained above


                                                        if(unit_select==2)
                                                        strcpy(text_for_usart + strlen("FAN SPEED: ") +
strlen(temp_for_usart), "%\r"); //explained above


                                                        send_string(text_for_usart);
                                                }




                                                clear_every_thing(my_data, 0x500); // clear the data so last
data value it wont be seen while asking input
                               send_string("Enter S for % fan speed and T for temperature:");





    }
    enter = 0;



      }


void active_switches();
unsigned char len_temperature;
unsigned char* lcd_up = 0x300;
unsigned char* lcd_down = 0x320;
unsigned char buzzer_1 = 0;
unsigned char buzzer_2 = 0;
unsigned char motor = 0;
unsigned int second_adc = 0;
unsigned char my_led = 0;
unsigned int first_adc = 0;
void all_calculations() iv 0x002A //adc function
   {
   second_adc = ADCL;
   second_adc += (ADCH & 0x03) << 8; // hence value max 10 bit we should take first 2 bits of high and shift it 8 bits left
   if(first_adc != second_adc) // if them changed
        {first_adc = second_adc; //store the


          heat_value = (second_adc * 0.098); // heat = 5/1023/50 mV * (10 bit value)
         if(heat_value < 30) // if new heat value <30
          {
                                buzzer_2 = 0; // buzzer will not sound
          fan_speed = 0; //fan speed will be 0
          if(motor) // if motor already active
          {
                                ((TIMSK) &= ~(1 << (OCIE0))); // enable interrupt for timer0
           waves = 0; // make waves variable 0 so when next time interrupt it will create waves on portb4
           PORTB.B4 = 0; // but no need because heat < 30
           motor = 0; // motor shouldnt work under 30 C
                                 }
          if(buzzer_1) //this condition is for high to low temp which means buzzer1 is already set but now we are under 30 C
            {((TIMSK) &= ~(1 << (TOIE2))); // enable timer2 interrupt
            TCNT2 = 0; //load 0 to timer 2
            buzzer_1 = 0; // now buzzer1 should be 0
```

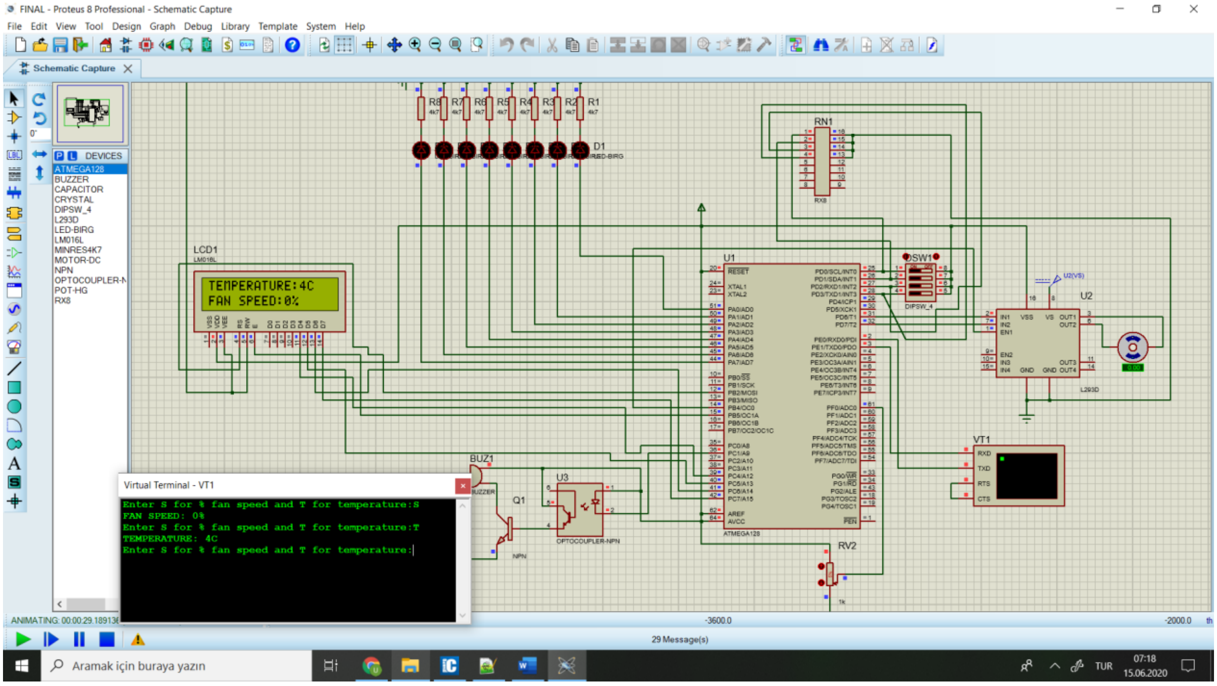
```
                                    }
      }
      else //if heat > 30
        {
                            fan_speed = 100*(heat_value-30)/70; // calculate fan speed
        if(!motor) //if low to high
        TIMSK |= 1 << (OCIE0); // start timer 0 to make noise
        motor = 1; //now motor is active
                            }

    OCR0 = (fan_speed * 255) / 100; // load needed value to ocr0
    strcpy(lcd_up, "TEMPERATURE:");
    strcpy(temp_for_usart,int_to_String(heat_value));
    strcpy(lcd_up + 12, temp_for_usart);
    len_temperature = strlen(temp_for_usart);
    lcd_up[12 + len_temperature] = 'C';
    memset(lcd_up + 13 + len_temperature, ' ', 11 - len_temperature);
    strcpy(lcd_down, "FAN SPEED:");
    strcpy(temp_for_usart,int_to_String(fan_speed));
    strcpy(lcd_down + 10, temp_for_usart);
    len_temperature = strlen(temp_for_usart);
    lcd_down[10 + strlen(temp_for_usart)] = '%';
    memset(lcd_down + 11 + len_temperature, ' ', 11 - len_temperature);
    my_led = ((1 << (second_adc / 127)) - 1) & 0xFF; // led values on port a 5V 100 -> 0.625 V for 12.5 for each led
    }

              active_switches();
}
void active_switches(){ // to check which switches are active
        if(PIND.B0==1) //display temperature on first line
    Lcd_Out(1, 1, lcd_up);
```

```
else
   {unsigned char temp[0x10]; // else display empty lines


   memset(temp, ' ', 0x10);
   Lcd_Out(1, 1, temp);}


if(PIND.B1==1) //same as above
   Lcd_Out(2, 1, lcd_down);
else
   {unsigned char temp[0x10];


   memset(temp, ' ', 0x10);
   Lcd_Out(2, 1, temp);}


if(PIND.B2==0) //if buzzer noise switch not active
   {
                    buzzer_2 = 0; // make buzzer2 0 for future
   if(buzzer_1) // even if it should have been make noise
   {((TIMSK) &= ~(1 << (TOIE2))); // disable timer2
   TCNT2 = 0; //load 0 to timer 2
   ((PORTC) |= (1 << (1))); //portc1 is 1
   buzzer_1 = 0;}} //make buzzer1 0 for future
else if(!buzzer_1 && heat_value > 30 && !buzzer_2) //so prev buzzers are 0 and heat > 30
   {((TIMSK) |= (1 << (TOIE2))); //enable timer2 overflow interrupt
   ((PORTC) &= ~(1 << (1)));
    buzzer_1 = 1; //made noise become 1
    buzzer_2 = 1;}




if(PIND.B3==0) //if led display pin inactive
   PORTA = 0; // no display
   else
       PORTA = my_led; //else display 1 bit for each 12.5 C
}
void restart_timer(){
                  sum_twos = 0; //make it 0
            TIMSK &= ~(1 << TOIE2); // Timer2 overflow interrupt enable
          TCNT2 = 0; //timer2 is again 0
          PORTC |= 1 << 1; // make PORTC.1 to 1 again
          buzzer_1 = 0; // buzzer_1 will become 0 so it will not start
}

void initialize_ports_and_registers(){

            DDRA = 0xFF; // A for led outputs

            DDRC = 0x02; // C1 connection for buzzer

   DDRD = 0xF0; // D7 and D6 for motor connection output

   UCSR0C |= (1 << UCSZ01)|(1 << UCSZ00); // UCR0C init

            UCSR0B |= (1 << RXEN0)| (1 << RXCIE0) | (1 <<TXEN0)  ; // UCSR0B init


    DDRB |=  1 << 4; //portb4 will be connected L293D(motor entegre) EN1
    PORTD &= ~(1 << (7)); //portd4 will be connected L293D(motor entegre) IN2
    PORTD |= 1 << 6; //portd4 will be connected L293D(motor entegre) IN1
    PORTC |= 1 << 1; // portc1 will be connected to buzzer
   Lcd_Init(); // lcd init
   Lcd_Cmd(_LCD_CLEAR); // clear
```

```
    Lcd_Cmd(_LCD_CURSOR_OFF); // if i dont write this it is seen like cloudy
    UBRR0L = 0x40; // baud rate = fclock/(16*UBRR) -1 for async mode, normal speed
    UBRR0H = 0;    // UBRR = 10^7/(9600*16)-1


    TCCR0 |= (1 << CS02) | (1 << CS00) | (1 << WGM00) | (1 << COM01); //prescale 1024 normal mode


    OCR0 = fan_speed * 2.55;
    TCCR2 |= 1 << CS22; //presecaler 128, normal mode
    ADCSRA = 0xEF; // prescelar 64 ,no ADIF and ADPS0, free run mode
}




void enable_interrupt(){//enable interrupt
            SREG_I_bit = 1;
}


void main() {
    initialize_ports_and_registers(); //call function
    enable_interrupt(); // call function
    send_string("Enter S for % fan speed and T for temperature:"); // send this to Terminal for 1 time
    while(1) {
                        //just wait
        if(sum_twos >= 0x92) { // if it reaches 2 second restart
            restart_timer();
        }
    }
}
```

Demonstration

4C

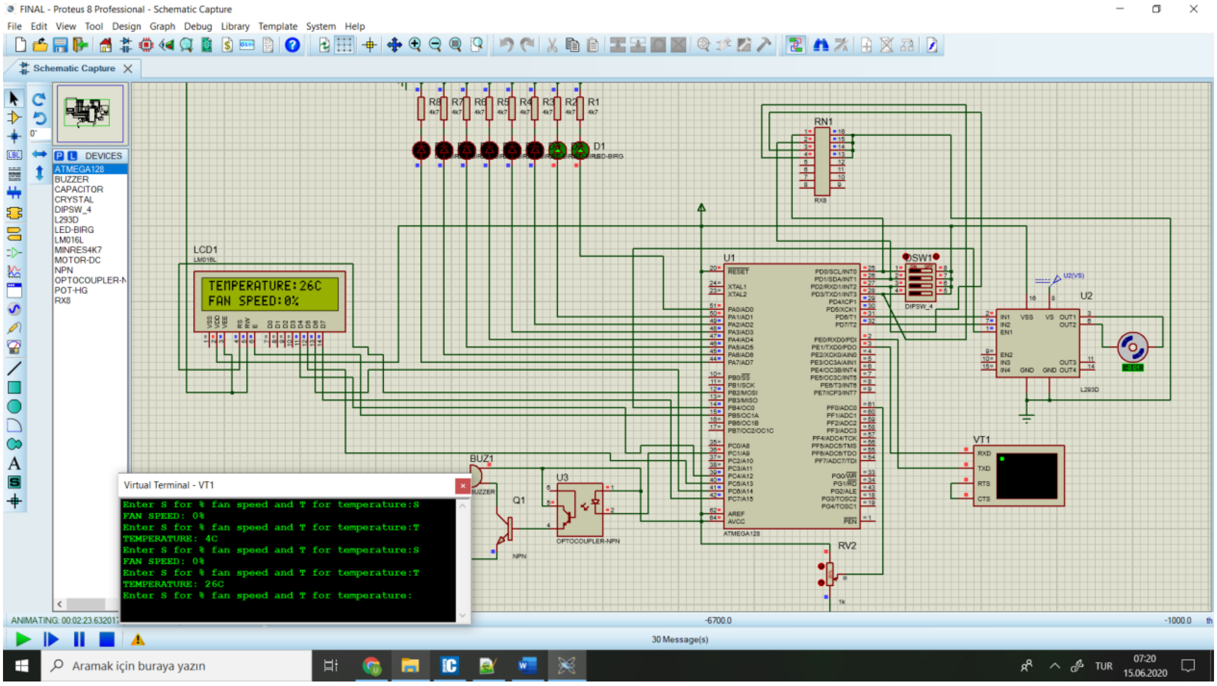


```
TEMPERATURE:4C
FAN SPEED:0%
```

Virtual Terminal - VT1

```
Enter S for % fan speed and T for temperature:S
FAN SPEED: 0%
Enter S for % fan speed and T for temperature:T
TEMPERATURE: 4C
Enter S for % fan speed and T for temperature:
```

26 C

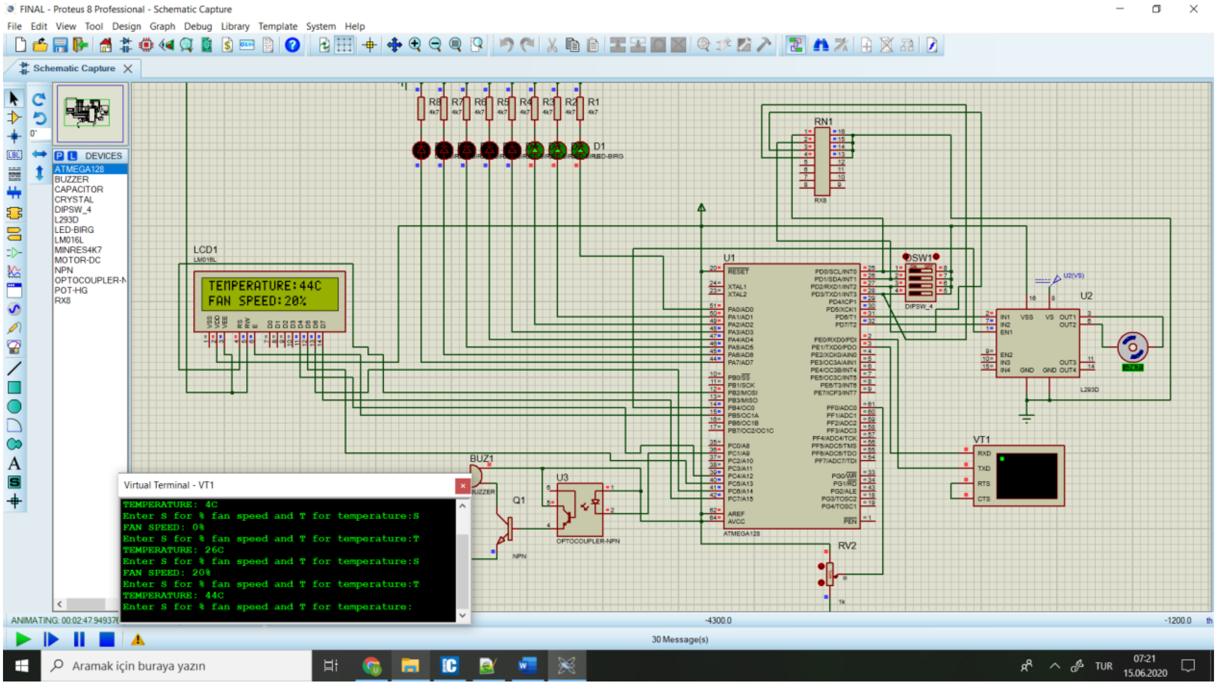


```
TEMPERATURE:26C
FAN SPEED:0%
```

Virtual Terminal - VT1

```
Enter S for % fan speed and T for temperature:S
FAN SPEED: 0%
Enter S for % fan speed and T for temperature:T
TEMPERATURE: 4C
Enter S for % fan speed and T for temperature:S
FAN SPEED: 0%
Enter S for % fan speed and T for temperature:T
TEMPERATURE: 26C
Enter S for % fan speed and T for temperature:
```

44 C

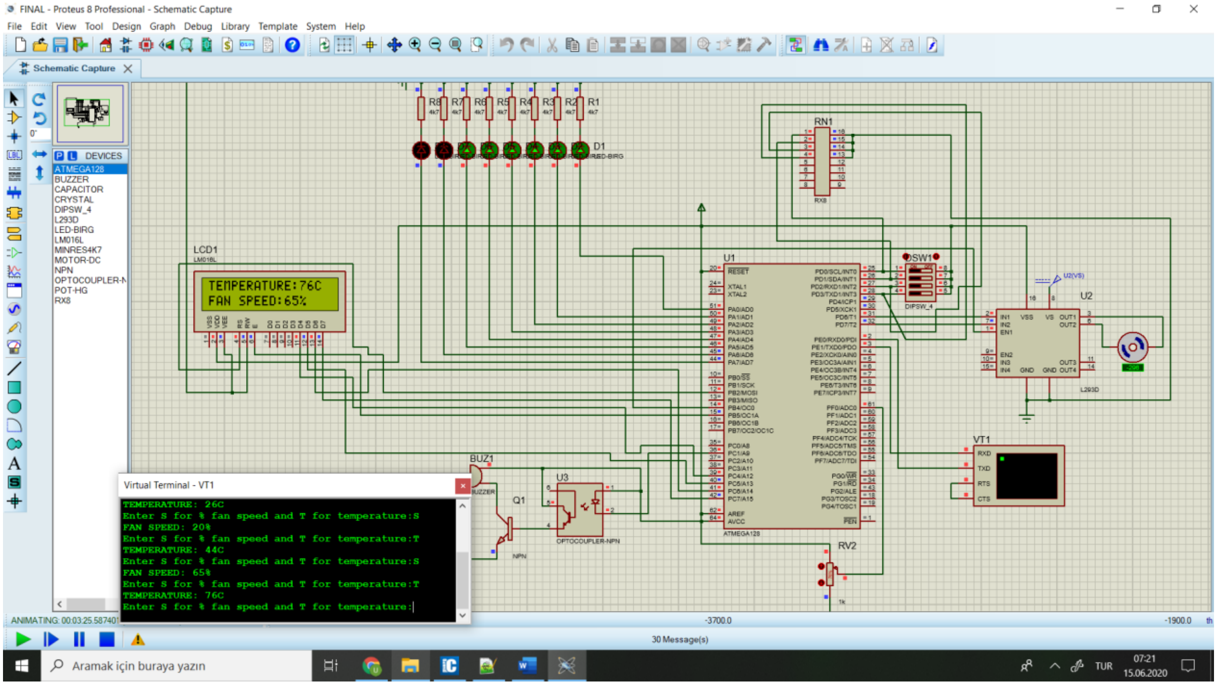


```
TEMPERATURE:44C
FAN SPEED:20%
```

Virtual Terminal - VT1

```
TEMPERATURE: 4C
Enter S for % fan speed and T for temperature:S
FAN SPEED: 0%
Enter S for % fan speed and T for temperature:T
TEMPERATURE: 26C
Enter S for % fan speed and T for temperature:S
FAN SPEED: 20%
Enter S for % fan speed and T for temperature:T
TEMPERATURE: 44C
Enter S for % fan speed and T for temperature:
```

76C

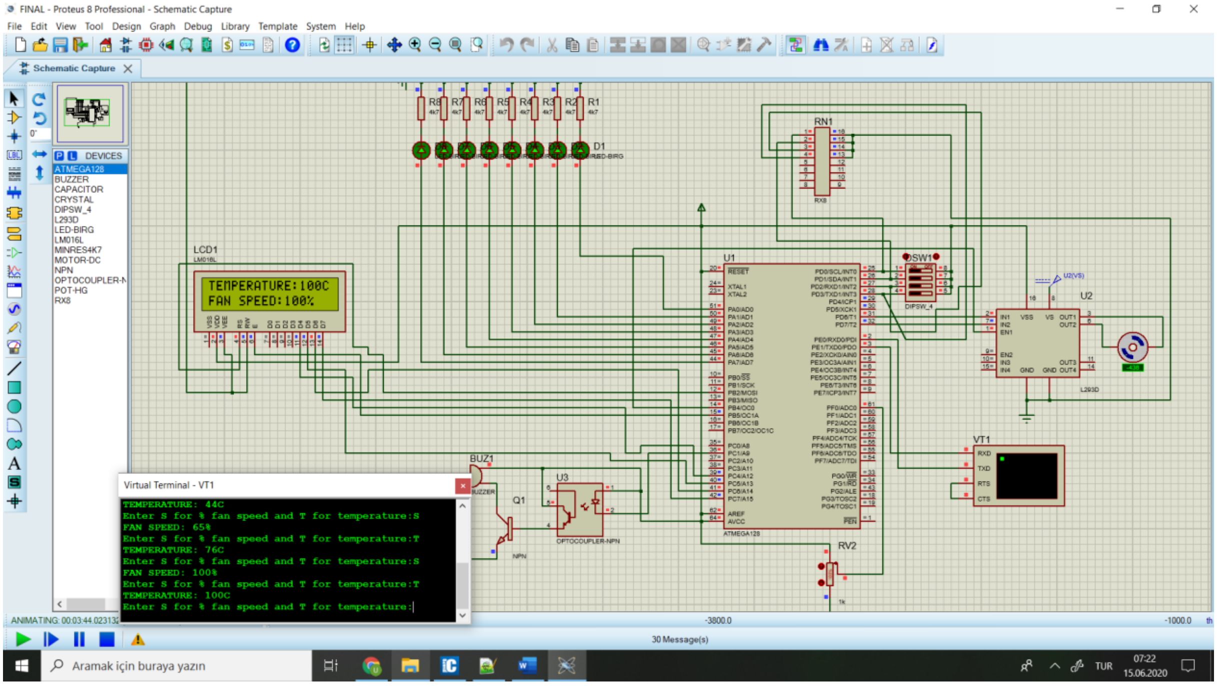


```
TEMPERATURE:76C
FAN SPEED:65%
```

Virtual Terminal - VT1

```
TEMPERATURE: 26C
Enter S for % fan speed and T for temperature:S
FAN SPEED: 20%
Enter S for % fan speed and T for temperature:T
TEMPERATURE: 44C
Enter S for % fan speed and T for temperature:S
FAN SPEED: 65%
Enter S for % fan speed and T for temperature:T
TEMPERATURE: 76C
Enter S for % fan speed and T for temperature:
```

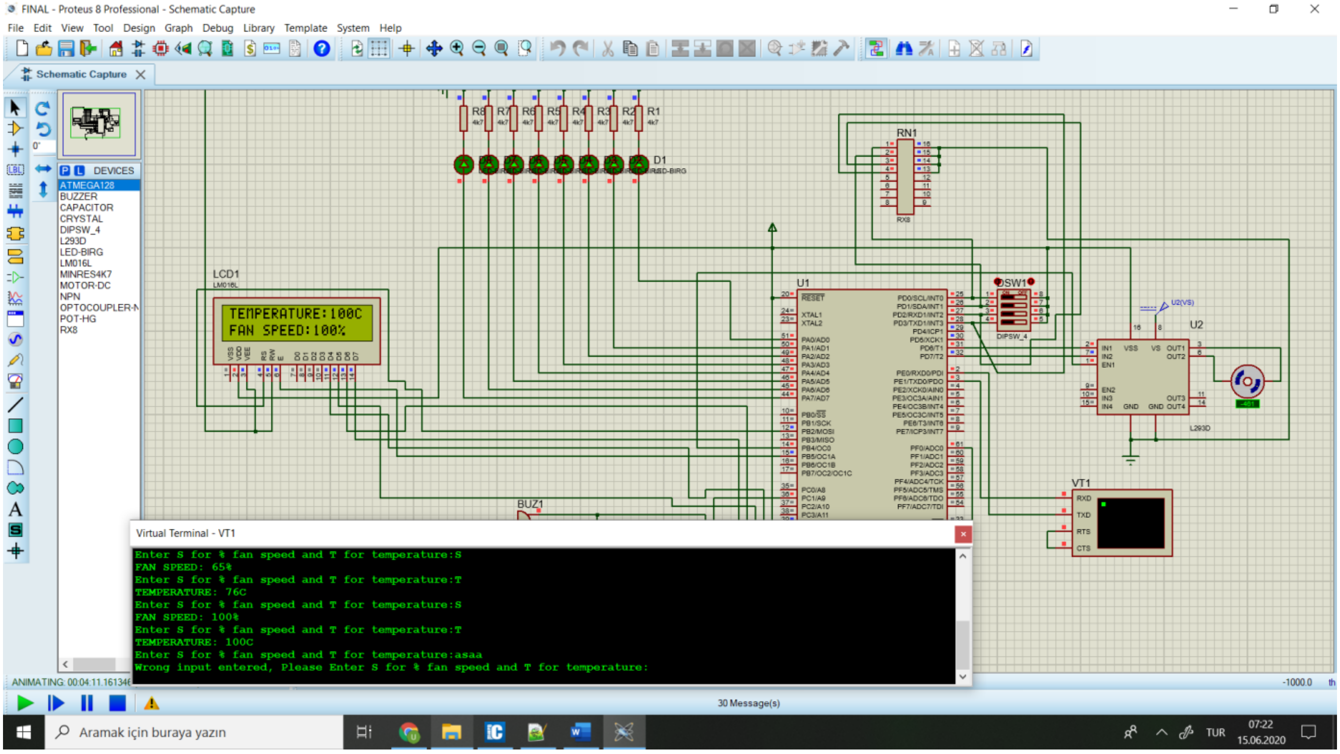
100 C (Highest)



Virtual Terminal - VT1

```
TEMPERATURE: 44C
Enter S for % fan speed and T for temperature:S
FAN SPEED: 65%
Enter S for % fan speed and T for temperature:T
TEMPERATURE: 76C
Enter S for % fan speed and T for temperature:S
FAN SPEED: 100%
Enter S for % fan speed and T for temperature:T
TEMPERATURE: 100C
Enter S for % fan speed and T for temperature:
```

Capture for wrong input in terminal



Virtual Terminal - VT1

```
Enter S for % fan speed and T for temperature:S
FAN SPEED: 65%
Enter S for % fan speed and T for temperature:T
TEMPERATURE: 76C
Enter S for % fan speed and T for temperature:S
FAN SPEED: 100%
Enter S for % fan speed and T for temperature:T
TEMPERATURE: 100C
Enter S for % fan speed and T for temperature:asaa
Wrong input entered, Please Enter S for % fan speed and T for temperature:
```