Problem in brief

- Direct exposure to solar radiation reduces the quality.
- Drying rate is very slow.
- The drying can only be carried out during sun shine hours.
- Economic losses due to having to sell at a low price due to low quality.
- Cannot be carried out in dust, rainy weather.
- Excessive wastage during grain drying in the presence of sunlight.

Project Aim & Objectives

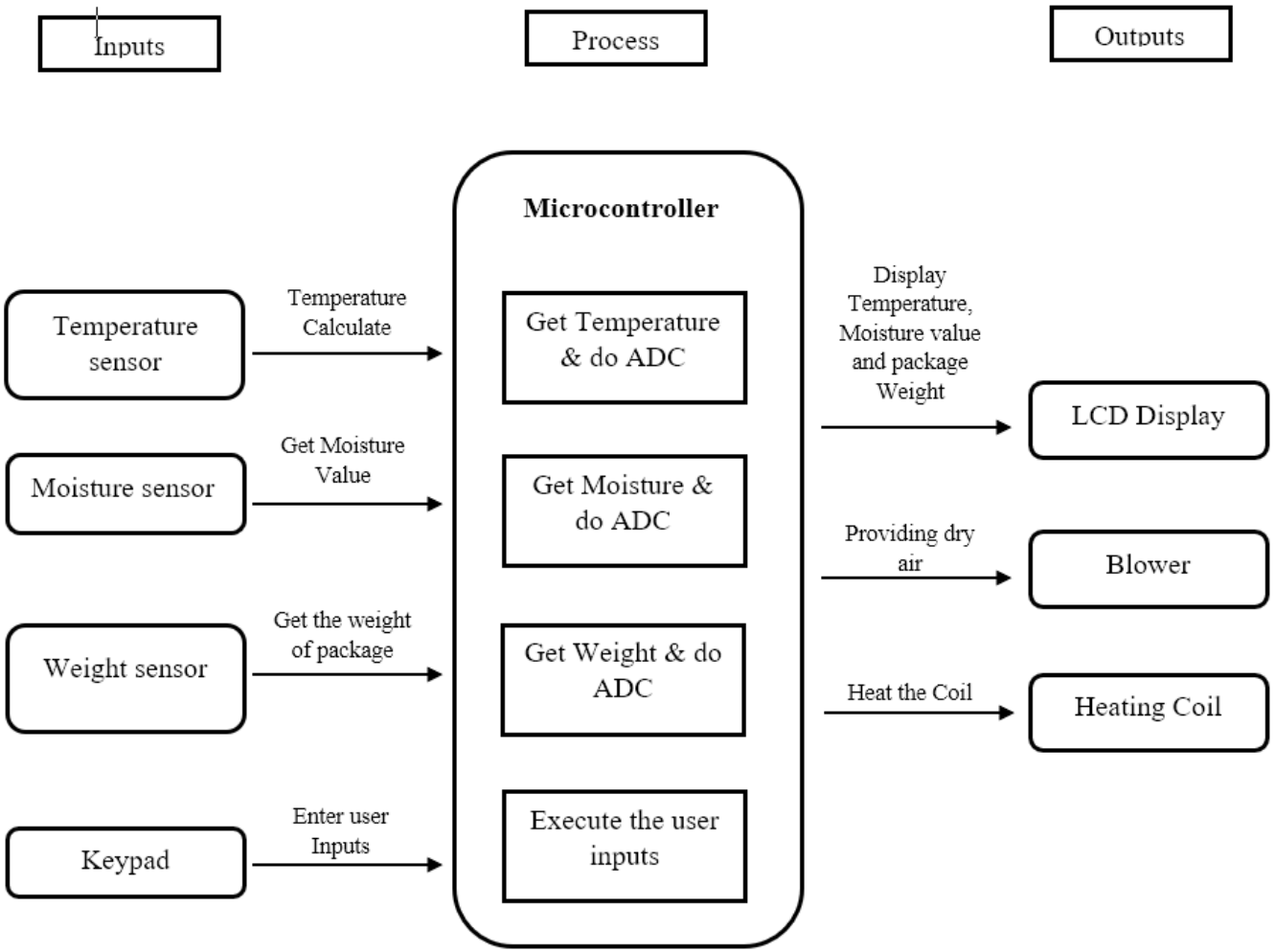
Aim:

- Drying of grain with proper quality and quality even in the absence of sunlight.

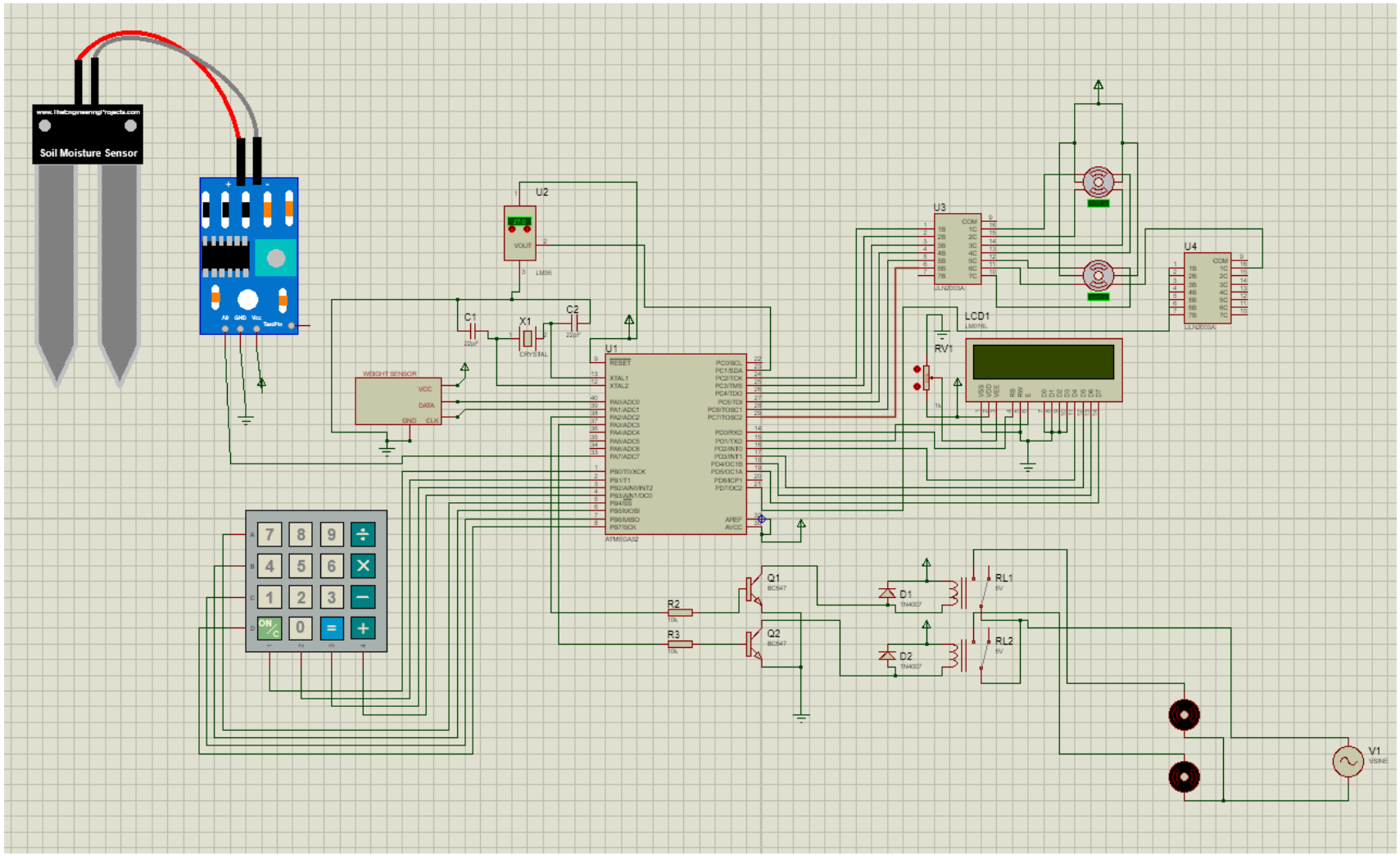
Objectives:

- To Minimize Excessive wastage during grain drying in the presence.
- To Minimize Economic losses due to having to sell at a lower price due to lower quality.
- To accelerate drying process.
- To dry grains regardless the weather.
- To Maintaining proper quality of grain.

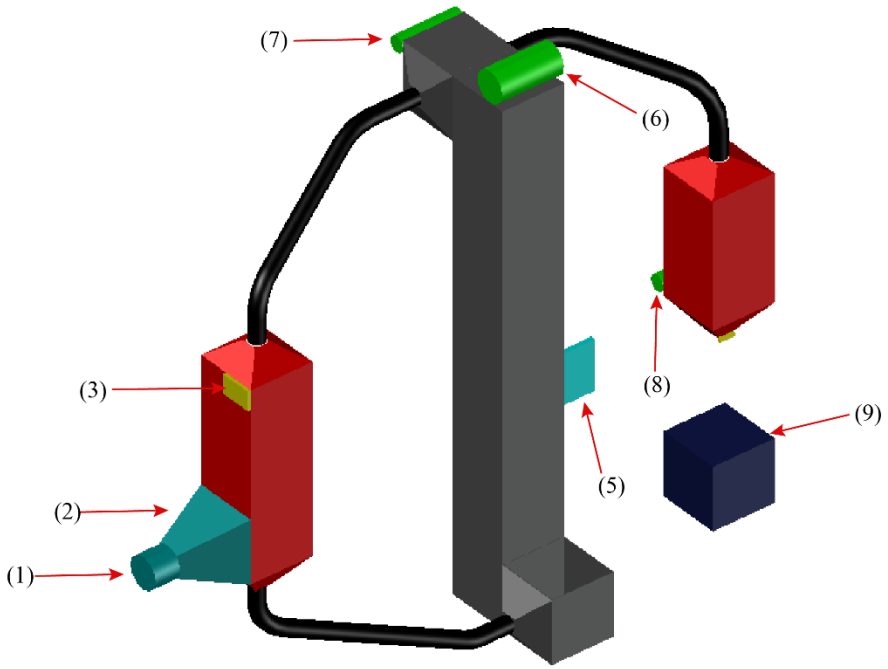
Proposed Solution



Circuit Diagram

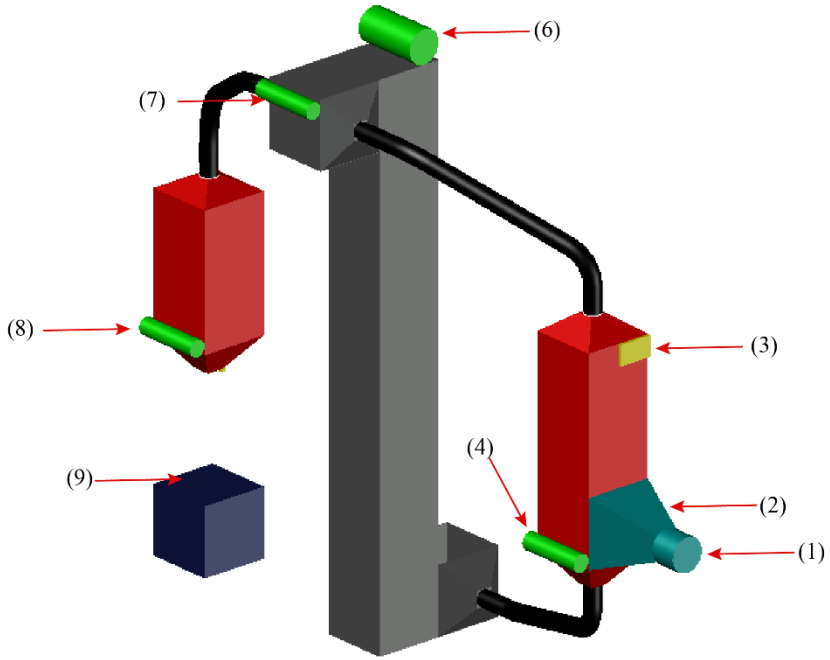


3D Diagram & 3D animation



(1)-Blower
(2)-Heating Element
(3)-Temperature Sensor & Moisture Sensor
(4)-Motor for Dryer Gate
(5)-Keypad and LCD Display
(6)-Motor for Elevator
(7)-Motor for Bypass Selector
(8)-Motor for Discharge Door
(9)-Weight Sensor

3D Diagram & 3D animation



(1)-Blower
(2)-Heating Element
(3)-Temperature Sensor & Moisture Sensor
(4)-Motor for Dryer Gate
(5)-Keypad and LCD Display
(6)-Motor for Elevator
(7)-Motor for Bypass Selector
(8)-Motor for Discharge Door
(9)-Weight Sensor

Animation link :
https://drive.google.com/file/d/1mzPMt5gEuvJjW0YeT08v5LOPZedchIF8/view?usp=sharing

Resource Required

Software: Atmel Studio

Hardware:

Sensors: Temperature Sensor
Weight Sensor
Moisture Sensor

Microcontroller: Atmega32
Stepper Motors
DC Motor
Gear Motors
Keypad
LCD Display
Iron Sheets
Blower
Breadboard
Wires
Heating Element

Cost Estimation

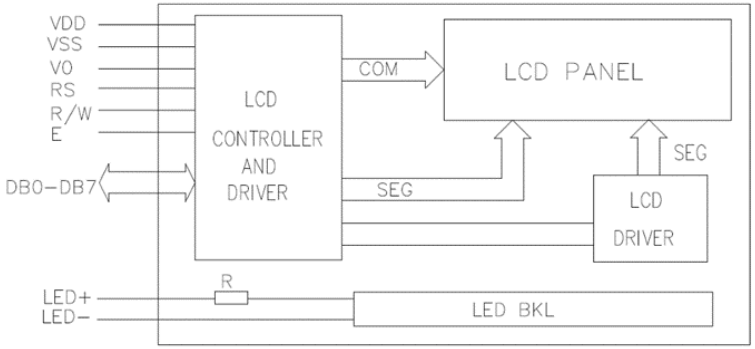
Name	Unit Price (Rs)	Quantity (Rs)	Amount (Rs)
Temperature Sensor	600	1	600
Moisture Sensor	800	1	800
Atmega32 Microcontroller	500	1	500
Weight Sensor	1500	1	1500
Keypad	200	1	200
Blower	3000	1	3000
LCD Display	650	1	650
Heating coil	4500	1	4500
Iron Sheets	5000	1	5000
Stepper Motors	1800	3	5400
Breadboard	250	1	250
Gear motor	2000	1	2000
Other parts	3000	1	3000
Total			27400

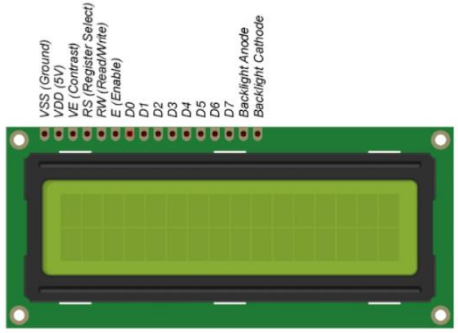
Test and Implementation

Member 1
Responsibilities :

- Find out about LCD display and how to get output from it.
- Study the process of a temperature sensor unit and find out how the resulting readings are displayed on the LCD display.
- Create Video

01. 16*2 LCD Display





LCD Display

Maximum voltage - 5V

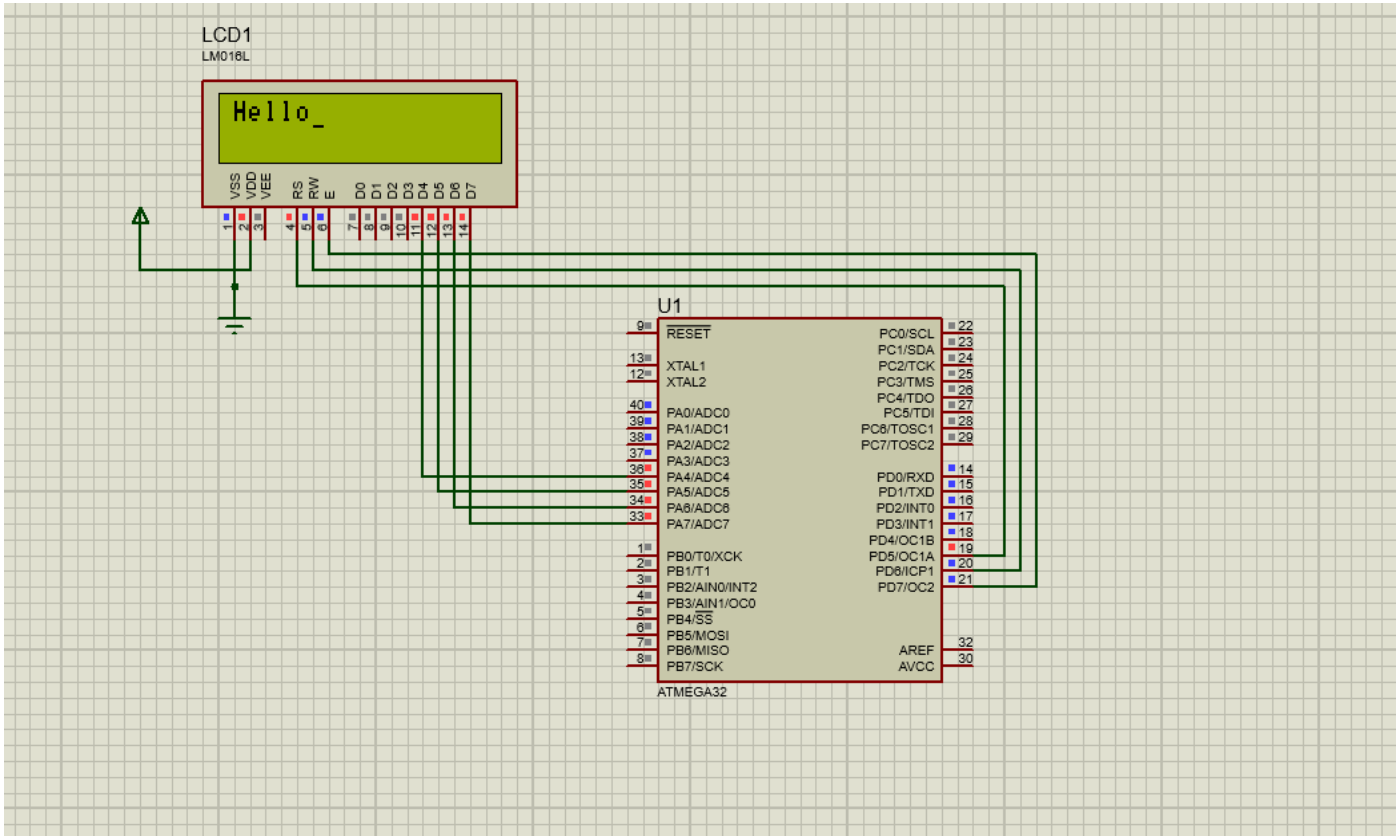
Maximum current - 20mA

16 characters and 2 lines display

4 – bit or 8 – bit MPU Interfaces

Test and Implementation

Circuit Diagram :



Test and Implementation

Compiled Code :

```
#include <avr/io.h>
#define F_CPU 16000000UL
#include <avr/io.h>
#include <util/delay.h>
#define LCD PORTA
#define EN 7
#define RW 6
#define RS 5
void lcdcmd(unsigned char cmd)
{
                        PORTD &= ~(1<<RS);
                        PORTD &= ~(1<<RW);
                        LCD =cmd & 0xF0;
                        PORTD |= (1<<EN);
                        _delay_ms(1);
                        PORTD &= ~(1<<EN);
                        LCD =cmd<<4;
                        PORTD |= (1<<EN);
                        _delay_ms(1);
                        PORTD &= ~(1<<EN);
}
```

Test and Implementation

```
void lcddata(unsigned char data)
{
                    PORTD |= (1<<RS);
                    PORTD &= ~ (1<<RW);
                    LCD = data & 0xF0;
                    PORTD |= (1<<EN);
                    _delay_ms(1);
                    PORTD &= ~(1<<EN);
                    LCD =data<<4;
                    PORTD |= (1<<EN);
                    _delay_ms(1);
                    PORTD &= ~(1<<EN);
}void lcd_init()
{
                    DDRA = 0XFF;
                    DDRD =0xFF;
                    PORTD &= ~(1<<EN);
                    lcdcmd(0x33);
                    lcdcmd(0x32);
                    lcdcmd(0x28);
                    lcdcmd(0x0E);
                    lcdcmd(0x01);
                    _delay_ms(2);
}
```

Test and Implementation

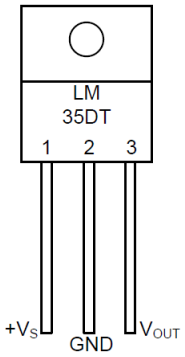
```
void lcd_print(char *str)
{
                unsigned char i=0;
                while(str[i]!=0)
                {
                                lcddata(str[i]);
                                i++;
                }
}

int main(void) //main function
{
                lcd_init();
                lcd_print("Hello");
                while(1);
                return 0;
}
```

Link to refer Speciation Data Sheet :

https://www.sparkfun.com/datasheets/LCD/ADM1602K-NSW-FBS-3.3v.pdf

Test and Implementation

02. LM 35 Temperature Sensor



LM 35 Temperature Sensor

LM 35 Temperature Sensor

Supply voltage : 4V to 30V

Maximum current : 0.06mA

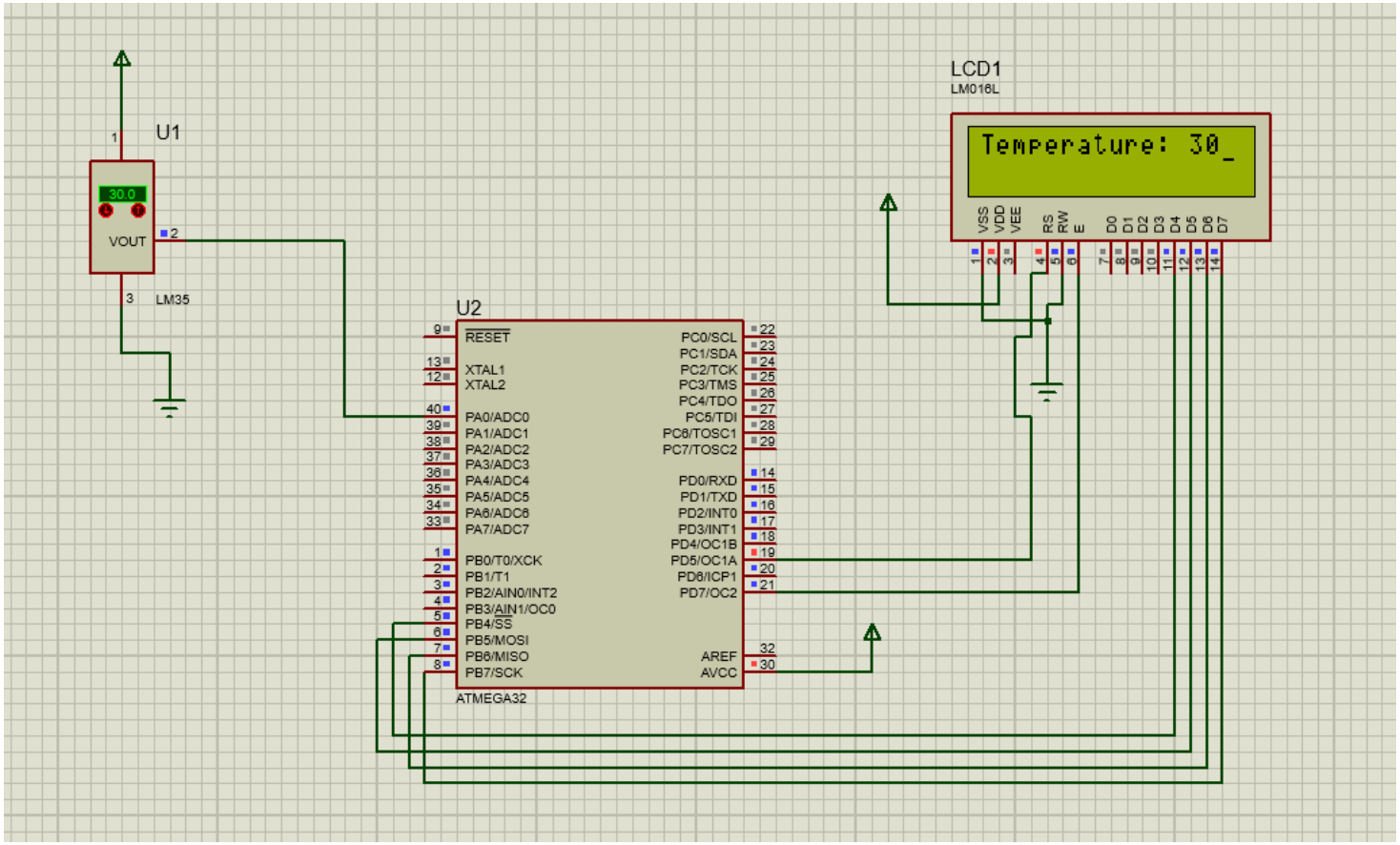
Accuracy at 25°C : ±0.5°C

Interface type : Analog

Range : −55°C to 150°C

Test and Implementation

Circuit Diagram :



Test and Implementation

Compiled Code :

```
#define F_CPU 16000000UL
#include <avr/io.h>
#include <util/delay.h>
#define LCD PORTB
#define EN 7
#define RS 5
#define RW 6
unsigned char data;
void lcdcmd(unsigned char cmd)
{
                        PORTD &= ~(1<<RS); // RS = 0 for command
                        PORTD &= ~(1<<RW); //RW=0 for write
                        LCD =cmd & 0xF0; // send upper nibble
                        PORTD |= (1<<EN); //EN = 1 for H to L pulse
                        _delay_ms(1);
                        PORTD &= ~(1<<EN); // EN = 0 for H to L pulse
                        LCD =cmd<<4;//send low nibble
                        PORTD |= (1<<EN); //EN = 1 for H to L pulse
                        _delay_ms(1);
                        PORTD &= ~(1<<EN);
}
```

Test and Implementation

```
void lcddata(unsigned char data)
{
                    PORTD |= (1<<RS); // RS = 1 for data
                    PORTD &= ~ (1<<RW); //RW=0 for write
                    LCD = data & 0xF0; // send upper nibble
                    PORTD |= (1<<EN); //EN = 1 for H to L pulse
                    _delay_ms(1);
                    PORTD &= ~(1<<EN); // EN = 0 for H to L pulse
                    LCD =data<<4;//send low nibble
                    PORTD |= (1<<EN); //EN = 1 for H to L pulse
                    _delay_ms(1);
PORTD &= ~(1<<EN); }
void lcd_init()
{
                    DDRB = 0XFF; //define output port
                    DDRD =0xFF; //((1<<RS) | (1<<RW)| (1<<EN));
                    PORTD &= ~(1<<EN); // initialize en = 0
                    lcdcmd(0x33);
                    lcdcmd(0x32);
                    lcdcmd(0x28); // LCD in 4 bit mode
                    lcdcmd(0x0E); // display on cursor on
                    lcdcmd(0x01); // clear LCD
                    _delay_ms(2);
}
```

Test and Implementation

```
void lcd_print(char *str)
{
		unsigned char i=0;
		while(str[i]!=0)
		{
				lcddata(str[i]);
				i++;
		}
}
int main(void)
{
		lcd_init();
		lcd_print("Temperature:");
		DDRA|= (1<<0); // make PA0 an i/p for ADC
		ADCSRA= 0x87; // make ADC enable and ck/128
		ADMUX = 0xE0; //2.56 V Vref and ADC0 single ended data will be left justified
		while(1)
		{
				ADCSRA |= (1<<ADSC);
				while ((ADCSRA&(1<<ADIF))==0);
				data=ADCH;
				convertndispaly(data);
				_delay_ms(500);
		}
		return 0;
}
```

Test and Implementation

```
void convertndispaly(unsigned char value)
{
                unsigned char x,d1,d2;
                x=value/10; // X = 32/10 = 3
                d1=x; // d1 = 3
                d2=value%10; // d2 = 32 % 10 = 2
                lcdcmd(0x8D); // Move the cursor to the 13th position on the first row
                lcddata(d1+0x30); //Move 0X33 ASCII to LCD that will display 3 on LCD
                lcddata(d2+0x30); //Move data 0X32 ASCII to LCD that will display 2 on LCD
}
```

Link to refer Speciation Data Sheet :

https://www.ti.com/lit/ds/symlink/lm35.pdf?ts=1638742405874&ref_url=https%253A%252F%252Fwww.ti.com%252Fproduct%252FLM35

Test and Implementation

Member 2 -

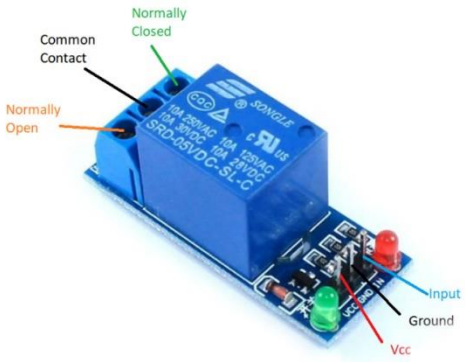
Responsibilities – AC Fan Motor & Heating Coil & Relays

- Information gathered and learned,
- About the process of a blower motor and heating coil. How it connect to the micro controller .
- How connect Relay and control the blower motor and heating coil.

AC Blower Motor

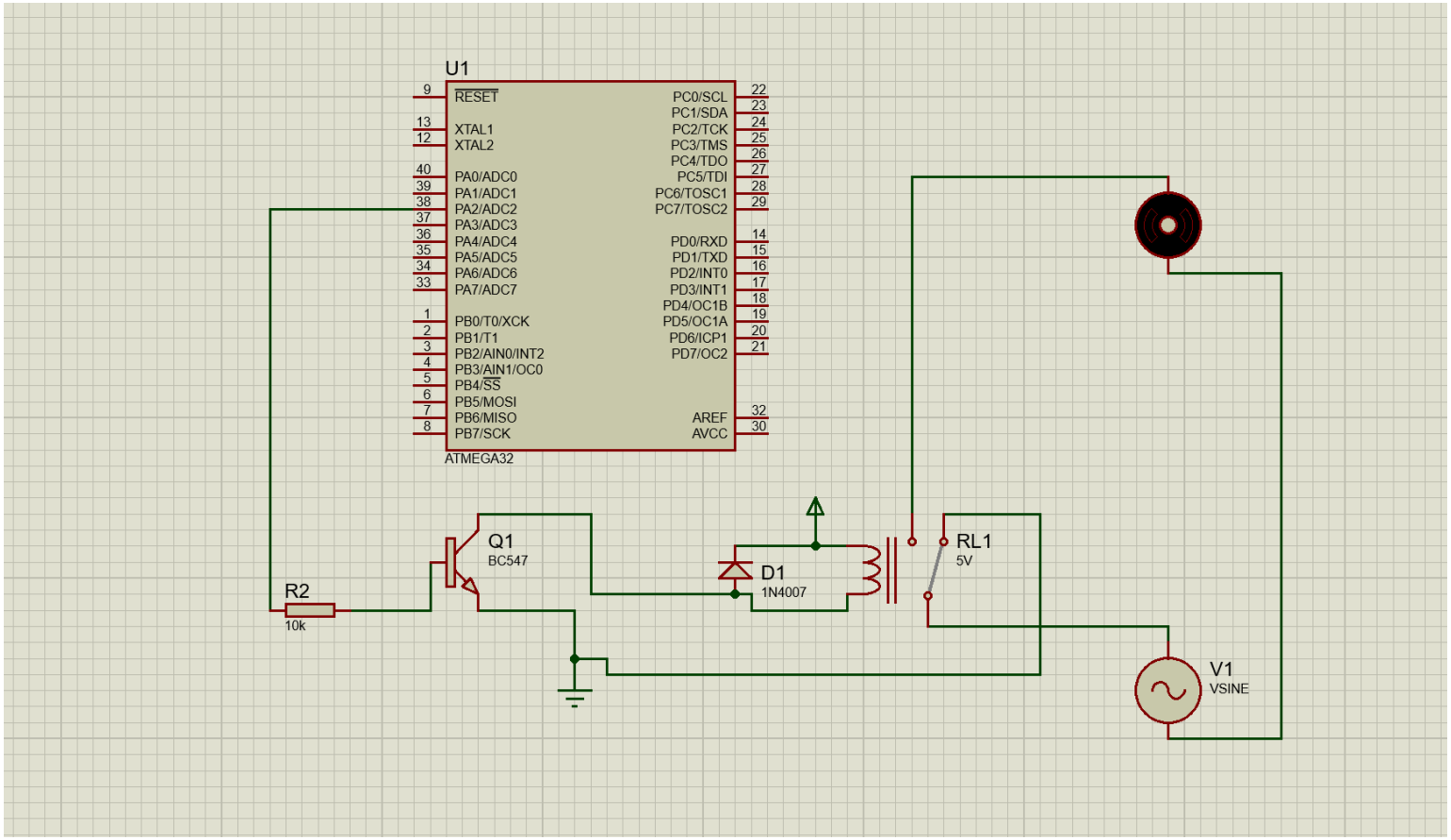
Voltage : 230 V(AC)
Rpm : 900
Maximum torque : 9.94 N·m
Power : 20W

Relay (Single channel relay module)



Voltage : 5V (Dc)
Normal current : 70mA
Load current max : 10A 250V (Ac)
Work as a automatic switch.

Circuit Diagram



Link referred to data sheet

https://components101.com/sites/default/files/component_datasheet/5V%20Relay%20Datasheet.pdf

Compile code

```
#include <avr/io.h>
#include <util/delay.h>
 #define Relay PA0

 int main()
 {
/* Configure the port A0 as Output */
DDRA = (1 << Relay);
while(1)
 {
PORTA = (1 << Relay);; /* Turn ON the Relay */ _delay_ms(5000);
 PORTA = (0 << Relay);; /* Turn OFF the Relay */ _delay_ms(5000);
}
return (0);
 }
```

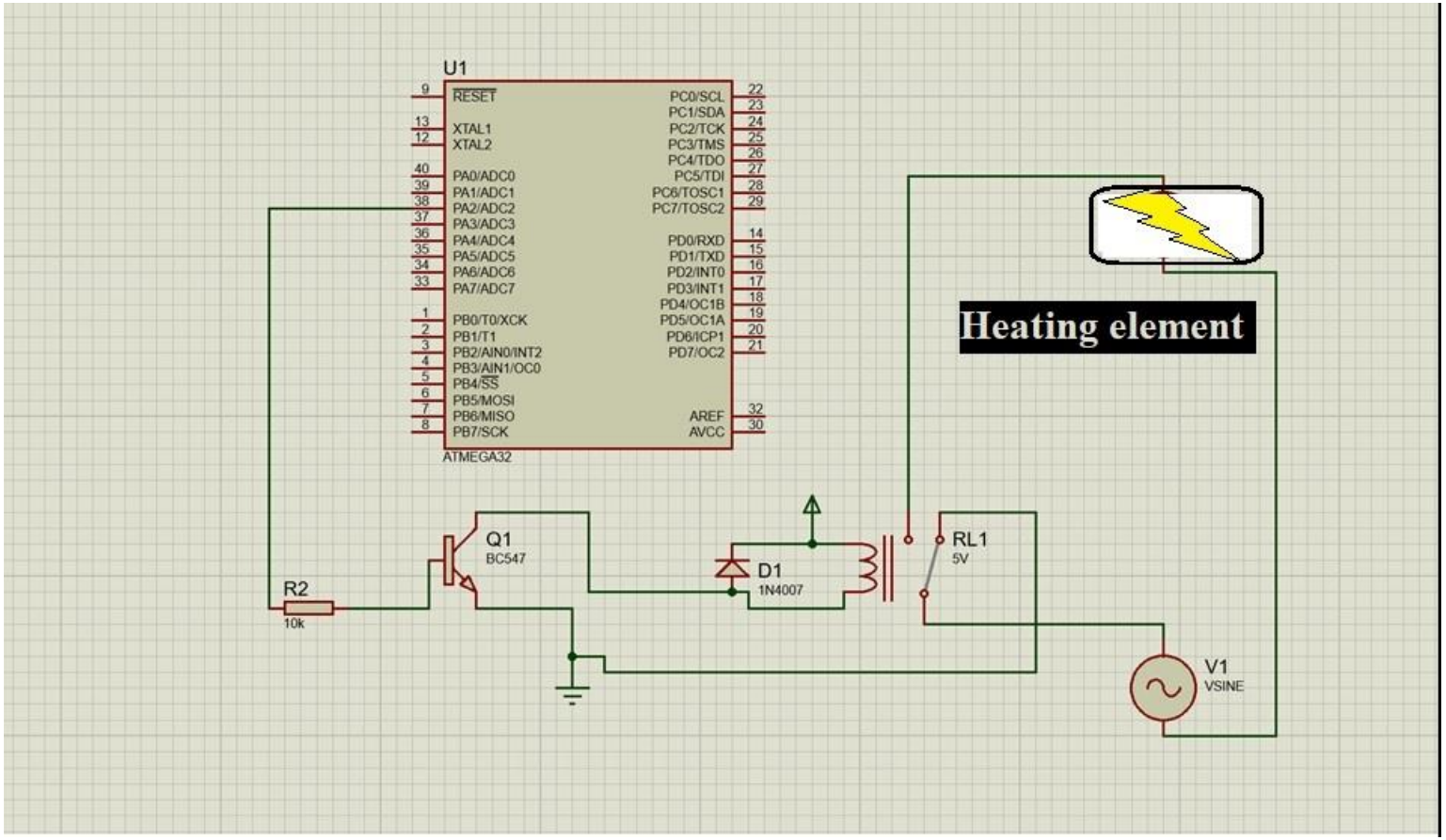
Heating elements

Voltage: 220V AC
Wattage: 1900W
Tube diameter ： 6.6mm
Diameter: 190mm

Tube Material: SUS304

Application:For Air Heater

Circuit Diagram



Link referred to data sheet
https://components101.com/sites/default/files/component_datasheet/5V%20Relay%20Datasheet.pdf

Test and Implementation

Member 3 -

Responsibilities - Power Supply and Moisture Sensor

Information gathered and learned,

- How to make a power supply and what are components used to make a power supply
- About the process of a moisture sensor unit and how to connect to the microcontroller

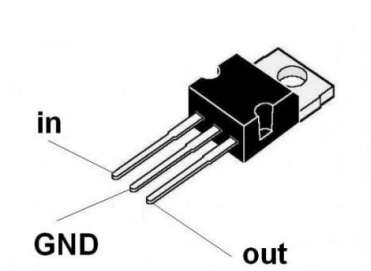
I. Power Supply

Resources Required

1. L7812, L7809, L7805 Regulators
2. Electrolytic Capacitor (1000µf/25V)
3. 4 Diodes (1N4007)
4. Transformer (230/12V)

Test and Implementation

L7812, L7809, L7805 Regulators



L7805 Specification
Maximum Input Voltage - 35V
Maximum output current - 1A

L7809 Specification
Maximum Input Voltage - 35V
Maximum output current - 1A

L7812 Specification
Maximum Input Voltage - 35V
Maximum output current – 1A

Link to refer specification data sheet

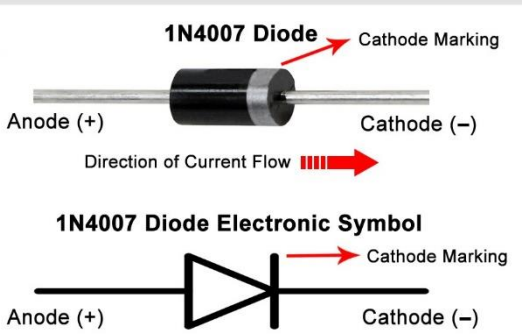
Electrolytic Capacitor

Specification
Value-1000uF
Rated Voltage-25V

Link to refer specification of data sheet.

Test and Implementation

Diodes (1N4007)



Specification
Maximum voltage - 1000V
Maximum current - 30A

Link to refer specification of data sheet.

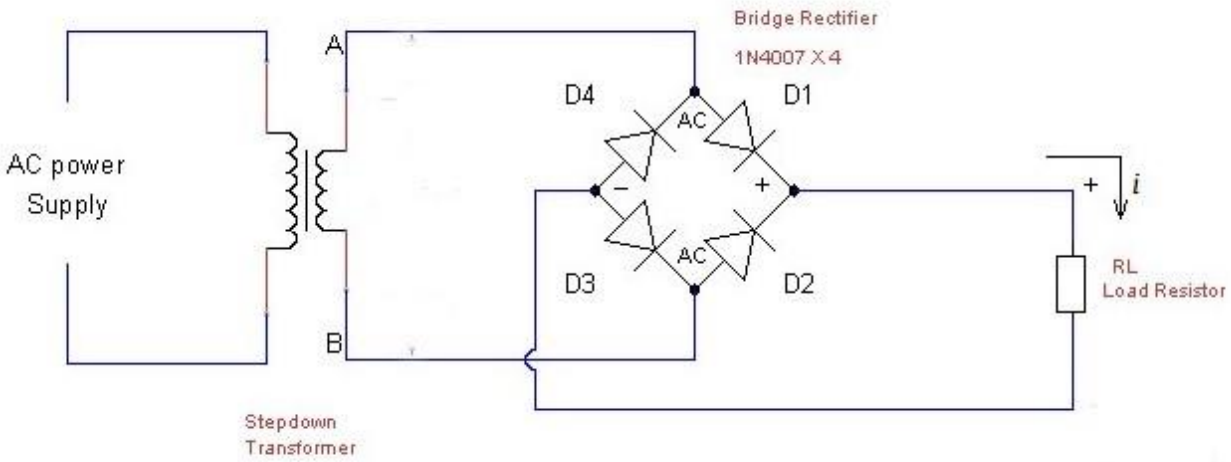
Transformer (230/12V)

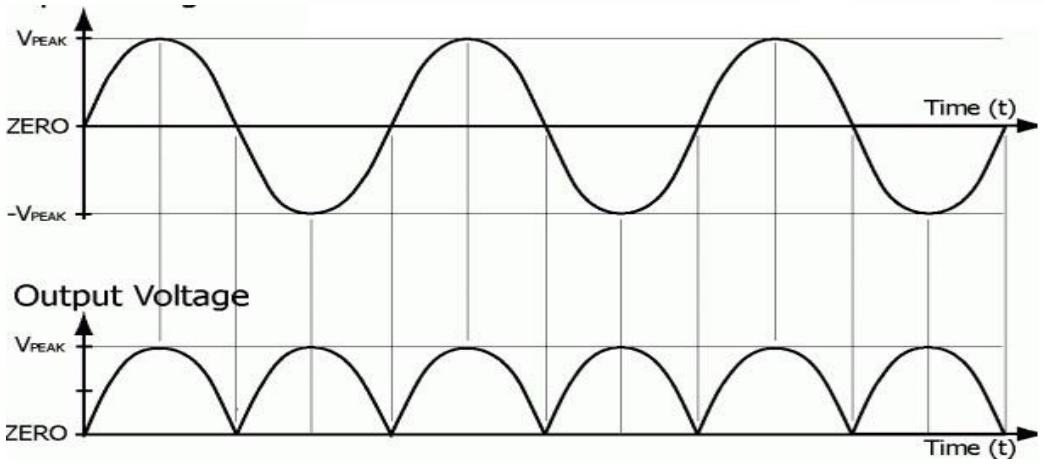
Specification
Maximum voltage - 230V
Maximum current - 30A

Link to refer specification of data sheet.

Test and Implementation

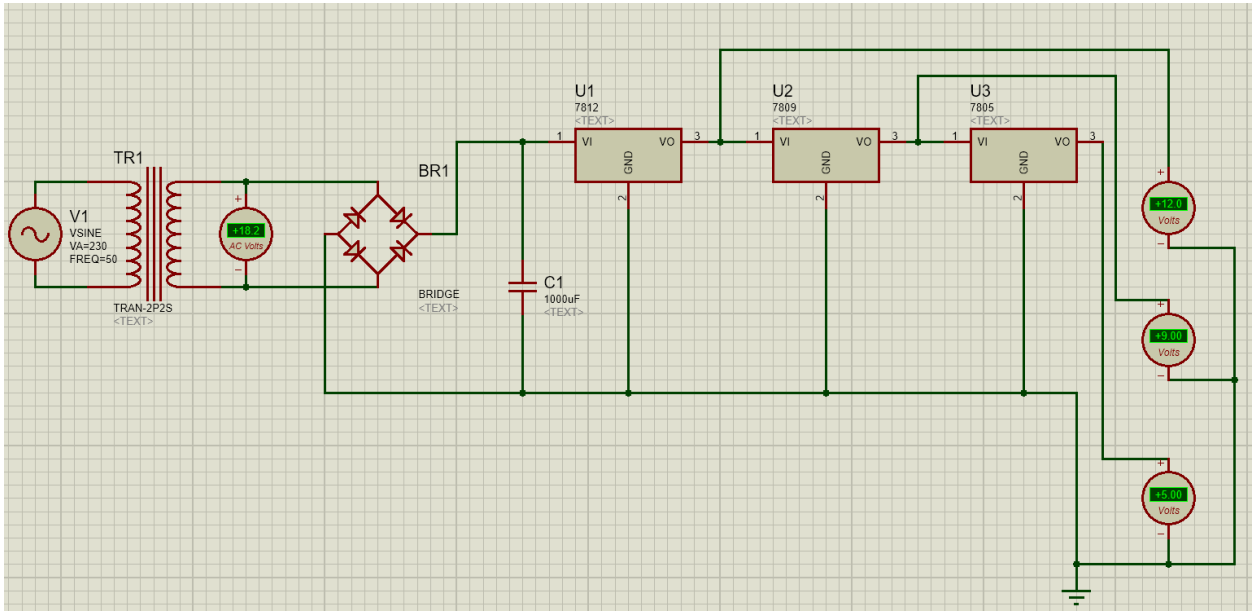
Rectification

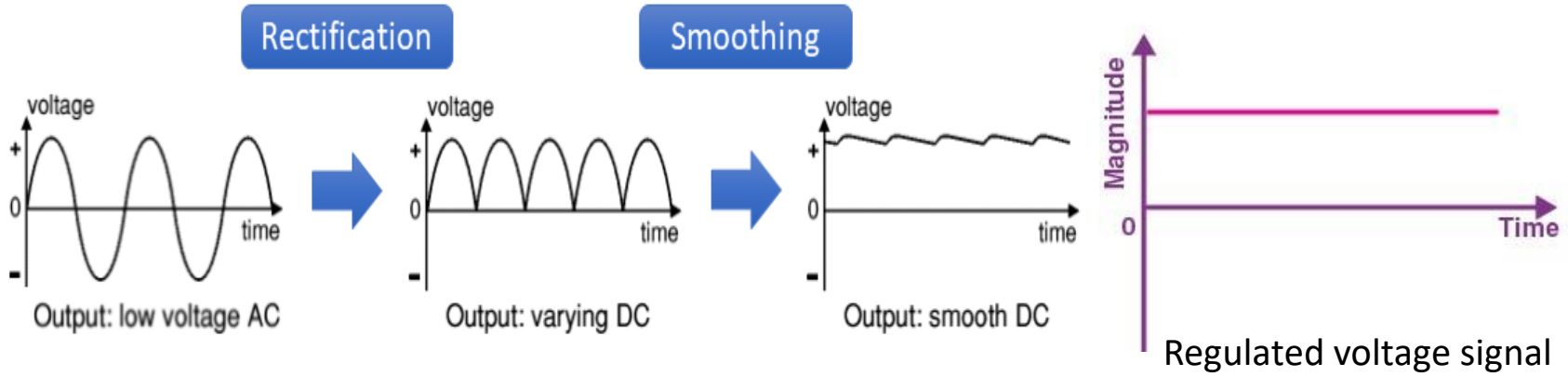




Test and Implementation

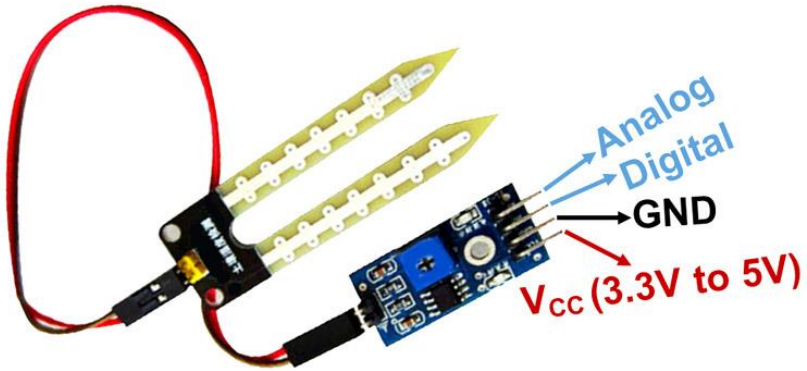
1. Power Supply Circuit Diagram





Test and Implementation

2. Moisture Sensor

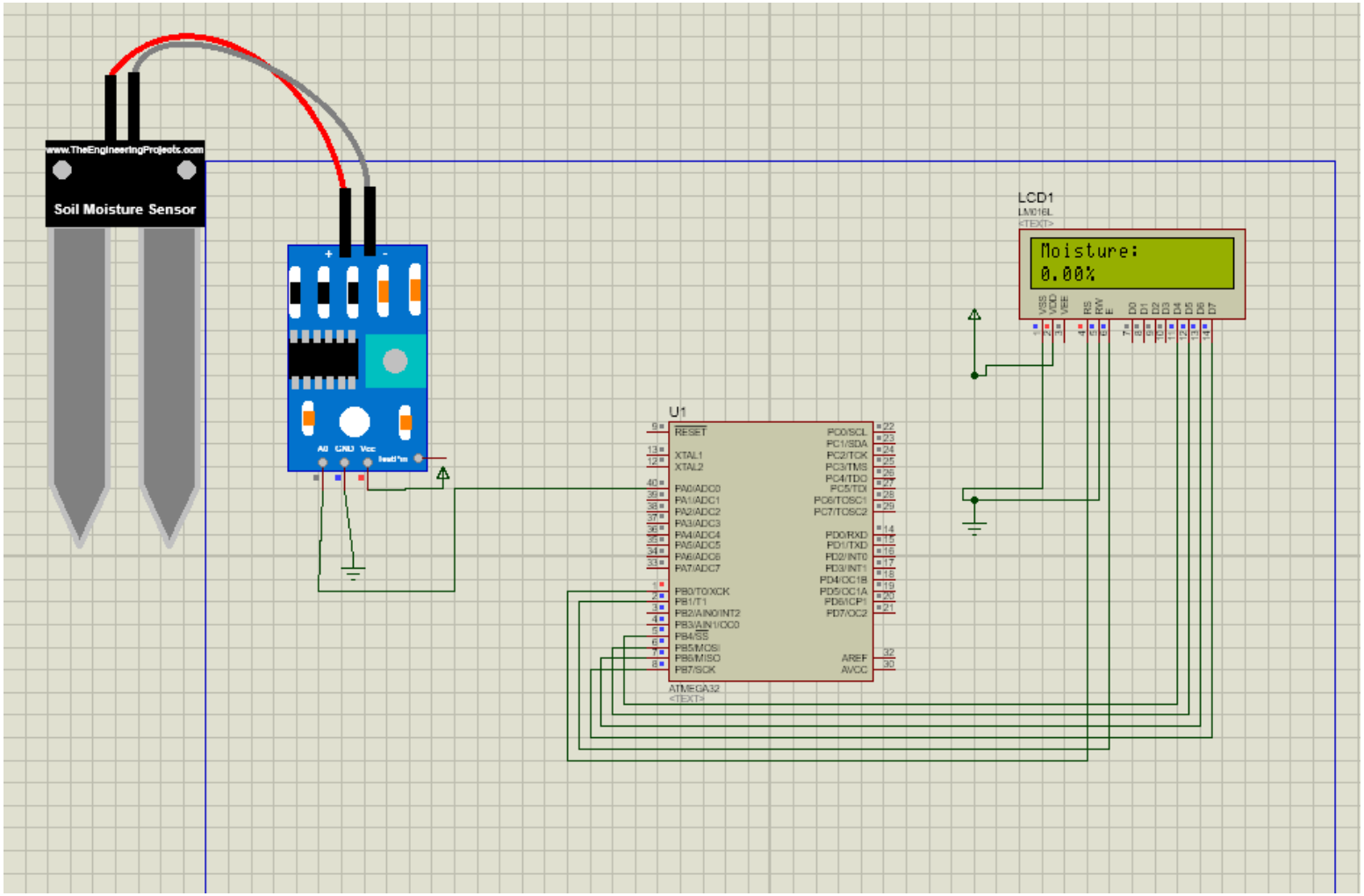


Maximum voltage - 5V
Maximum current - 35mA

Link to refer specification of data sheet.

Test and Implementation

Circuit Diagram :



Test and Implementation

Compile code:

```
#include <avr/io.h>
#include "LCD16x2_4bit.h"
#include <util/delay.h>
#include <stdlib.h>
#include <string.h>

 void ADC_Init()
{
          DDRA=0x0
         ADCSRA = 0x87;
 }
int ADC_Read()
{
         ADMUX = 0x40;
         ADCSRA |= (1<<ADSC);
        while ((ADCSRA
&(1<<ADIF))==0);
         ADCSRA |=(1<<ADIF);
         return(ADCW
}
```

```
int main(void)
{
          lcdinit();
          lcd_clear();
          ADC_Init();
          char array[10];
          int adc_value;
         float moisture;
 while(1)
    {
         adc_value = ADC_Read();
         moisture = 100(adc_value*100.00)/1023.00;
         lcd_gotoxy(0,0); lcd_print("Moisture: ");
         dtostrf(moisture,3,2,array);
         strcat(array,"% ");
         Lcd_gotoxy(0,1);
         lcd_print(array);
        memset(array,0,10);
         _delay_ms(500);
      }
 }
```

Test and Implementation

Member 4 -

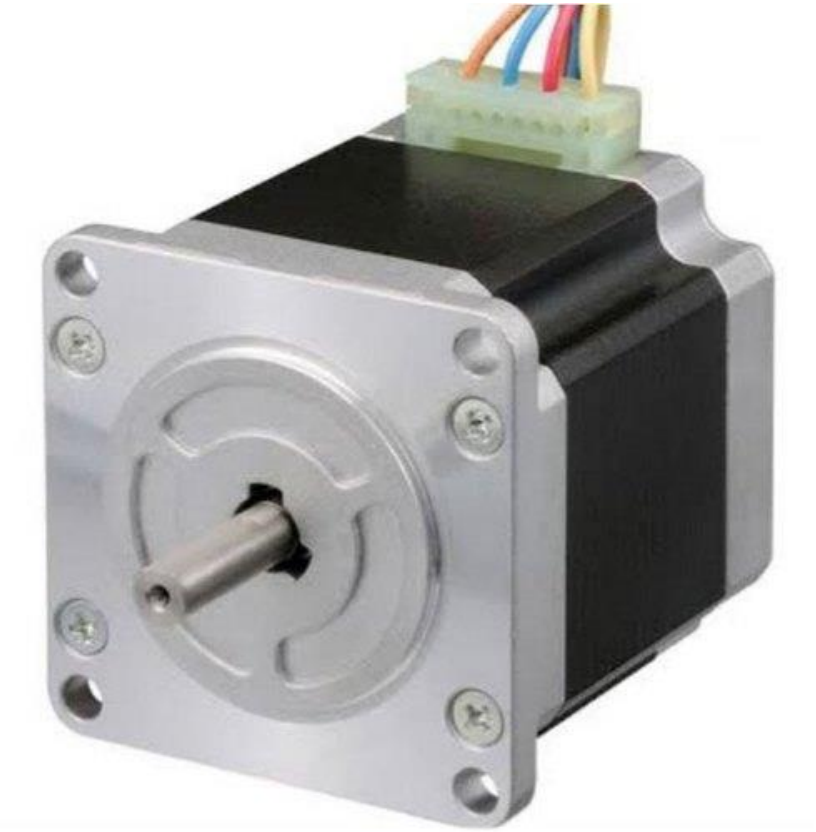
Responsibilities

Gear Motor and Stepper Motor

Information gathered and learned,

- About Gear motors and Stepper Motor how it works
- How to connect Gear Motor and Stepper Motor to atmega32 microcontroller
- Practiced the code needed for the process
- Creating animations

Stepper Motor



Specification

Voltage : 24V

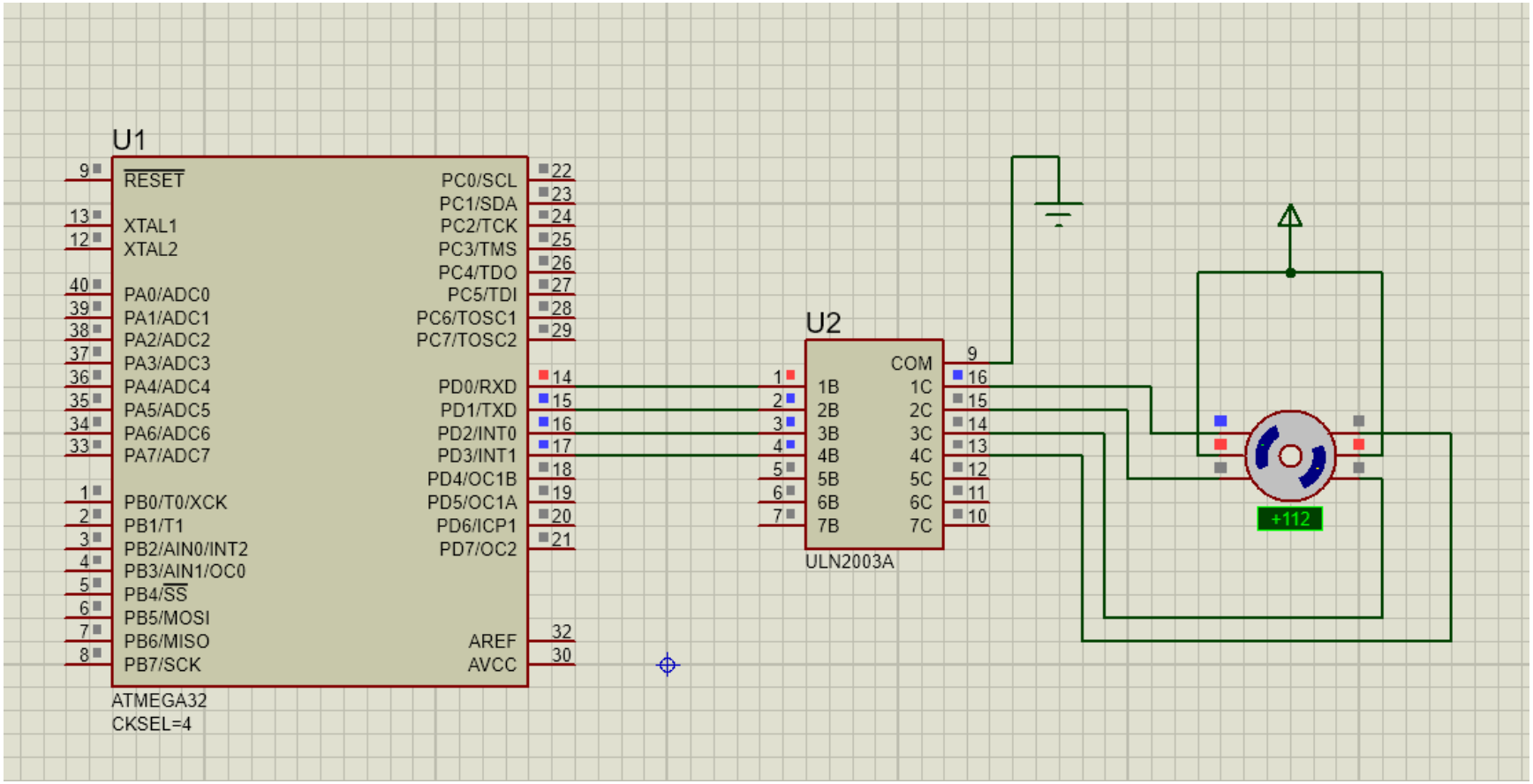
Current Rating : 1.5A

Step angle :1.8 Degree

Torque : 1.2Nm

Link to refer specification of data sheet.

Circuit Diagram



ULN2003A Motor Driver

Specifications:

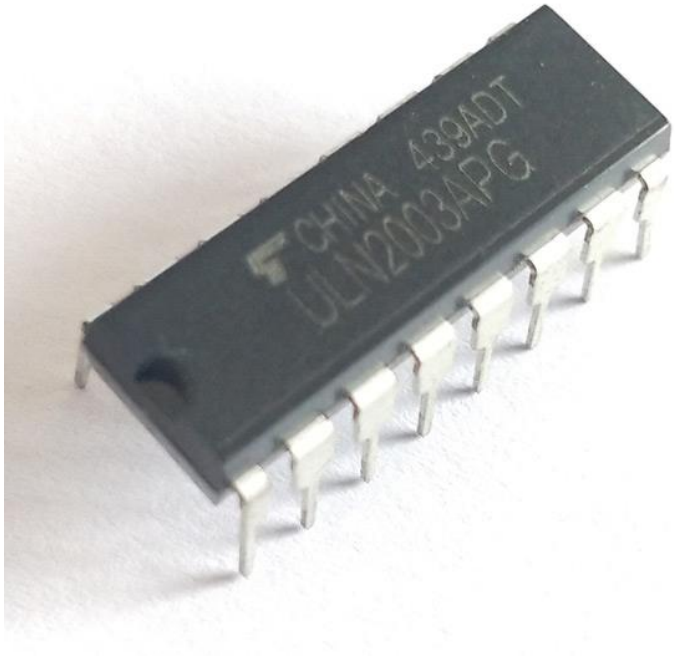
Input voltage: 30V

Output current: 500mA

Operating temperature: 150 C

Number of Outputs: Single Output

Number of pin: 16



Link to the Data sheet:

https://components101.com/sites/default/files/component_datasheet/IC%20ULN2003%20Datasheet_0.pdf

Compile code

```
#define F_CPU 8000000UL     /* Define CPU Frequency 8MHz */
#include <avr/io.h>     /* Include AVR std. library file */
#include <util/delay.h>     /* Include delay header file */


int main(void)
{
    int period;
    DDRD = 0x0F;        /* Make PORTD lower pins as output */
    period = 100;       /* Set period in between two steps */
    while (1)
    {
        /* Rotate Stepper Motor clockwise with Half step sequence */
        for(int i=0;i<12;i++)
        {
            PORTD = 0x09;
            _delay_ms(period);
            PORTD = 0x08;
            _delay_ms(period);
            PORTD = 0x0C;
            _delay_ms(period);
            PORTD = 0x04;
            _delay_ms(period);
            PORTD = 0x06;
            _delay_ms(period);
            PORTD = 0x02;
            _delay_ms(period);
            PORTD = 0x03;
            _delay_ms(period);
            PORTD = 0x01;
            _delay_ms(period);
        }
        PORTD = 0x09;       /* Last step to initial position */
        _delay_ms(period);
        _delay_ms(1000);
```

```
        /* Rotate Stepper Motor Anticlockwise with Full step sequence */
        for(int i=0;i<12;i++)
        {
            PORTD = 0x09;
            _delay_ms(period);
            PORTD = 0x03;
            _delay_ms(period);
            PORTD = 0x06;
            _delay_ms(period);
            PORTD = 0x0C;
            _delay_ms(period);
        }
        PORTD = 0x09;
        _delay_ms(period);
        _delay_ms(1000);
    }
}
```

Gear Motor

Specification

Voltage : 12V

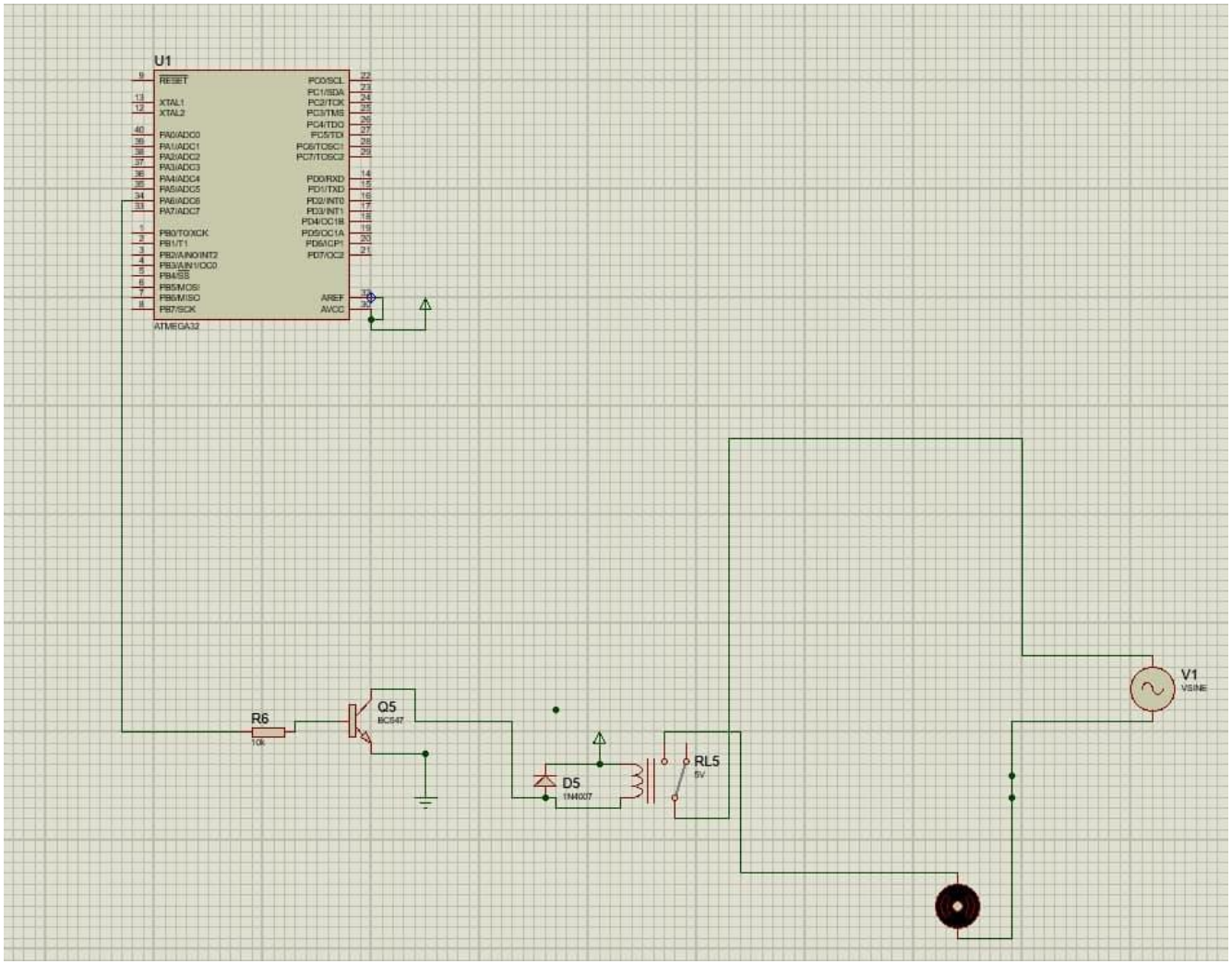
Speed: 200 RPM

Current Rating : 1A

Torque : 16kgcm

Link to refer specification of data sheet.

Circuit Diagram



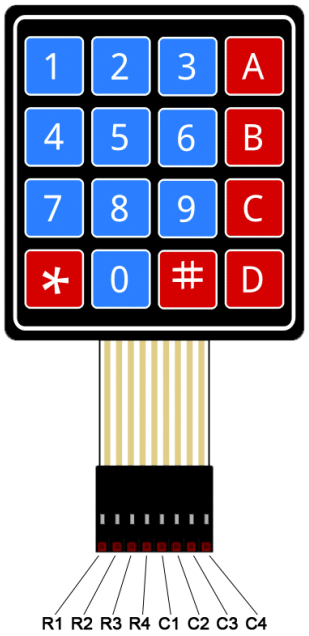
Test and Implementation

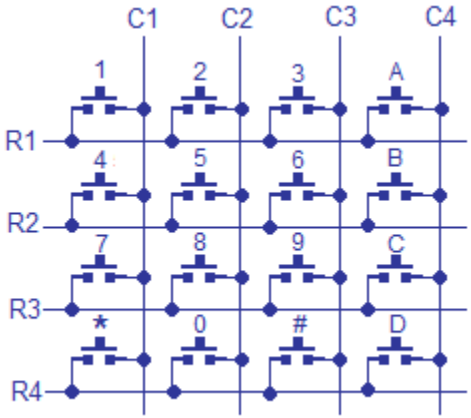
Member 5 –
Responsibilities - Weight Sensor and Keypad

Information gathered and learned,

- About weight sensor, keypad and how it works
- How to connect weight sensor and keypad to atmega32 microcontroller
- Practiced the code needed for the process
- Creating animations

4x4 Matrix Keypad



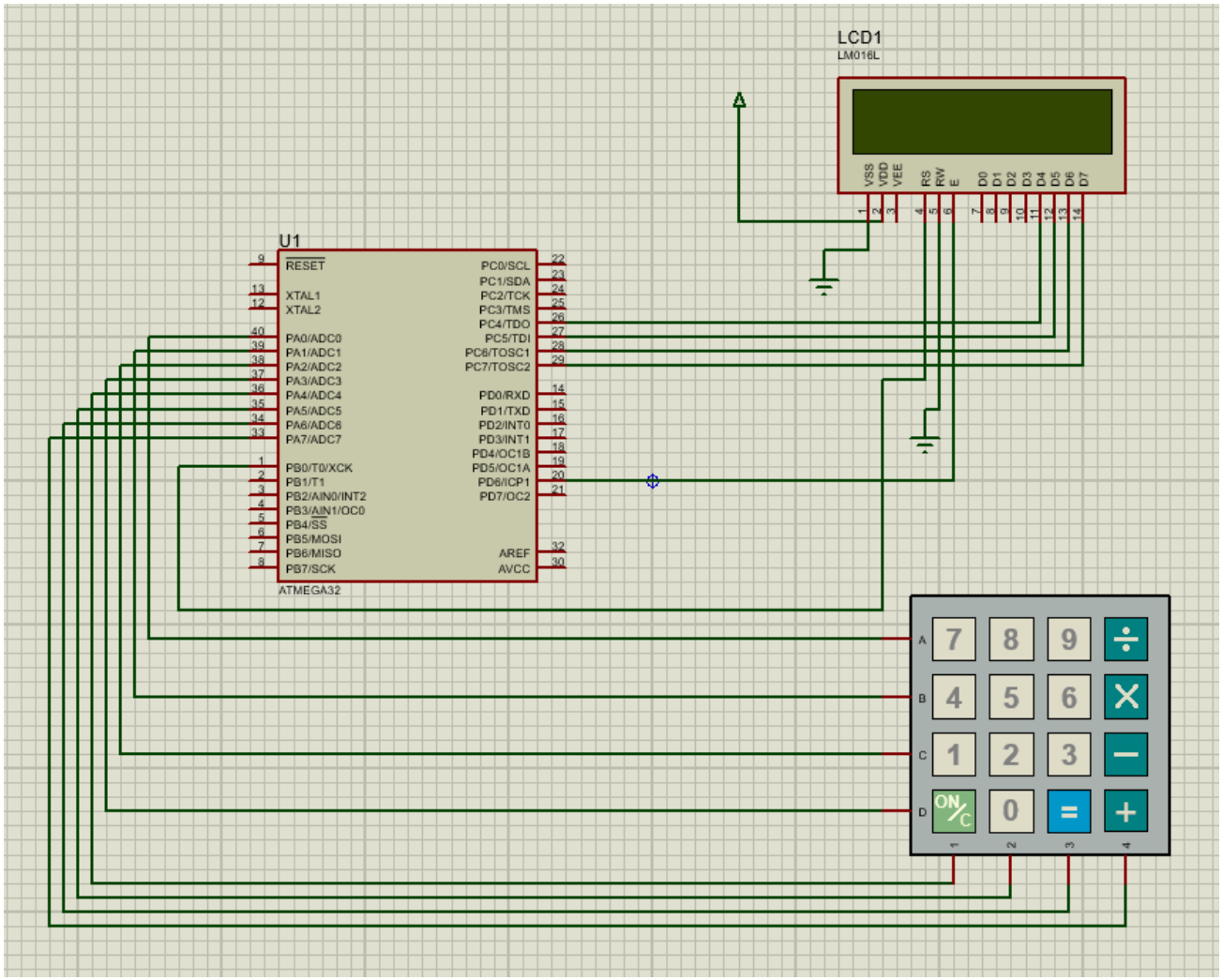


Specification

Max Voltage- 24V
Max Current- 30mA

Link to the datasheet

Circuit Diagram



Compile code

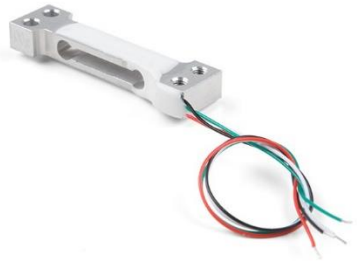
```
#define F_CPU 16000000UL
#include <avr/io.h>
#include <util/delay.h>

#define LCD PORTC // LCD data port connected to PORTC
#define EN 6
#define RS 0
#define RW 1

unsigned char keypad();

void lcdcmd(unsigned char cmd)
{
    PORTB |=~(1<<RS);   //RS=0 for command
    PORTB &=~(1<<RW);   //RW=0 for write
    LCD=cmd & 0xF0;     //Send upper nibble
    PORTD |=(1<<EN);    //EN=1 for H to L pulse
    _delay_ms(1);
    PORTD &=~(1<<EN);   //EN=0 for H to L pulse

    LCD=cmd<<4;         //Send low nibble
    PORTD |=(1<<EN);    //EN=1 for H to L pulse
    _delay_ms(1);
    PORTD &=~(1<<EN);

}

void lcddata(unsigned char data)
{
    PORTB |=(1<<RS);    //RS=1 for data
    PORTB &=~(1<<RW);   //RW=0 for write
    LCD=data & 0xF0;    //send upper nibble
    PORTD |=(1<<EN);    //EN=1 for H to L pulse
    _delay_ms(1);
    PORTD &=~(1<<EN);   //EN=0 for H to L pulse

    LCD=data<<4;        //send low nibble
    PORTD |=(1<<EN);    //EN=1 for H to L pulse
    _delay_ms(1);
    PORTD &=~(1<<EN);
}

void lcd_init()
{
    DDRC=0xFF;          //define output LCD port
    DDRD|=(1<<EN);      //define EN pin as output
    DDRB=0xFF;          //define RS and RW pin as output
    PORTD &=~(1<<EN);   //initialization en=0
    lcdcmd(0x33);
    lcdcmd(0x32);
    lcdcmd(0x28);       //LCD in 4 bit mode
    lcdcmd(0x0E);       //display on cursor on
    lcdcmd(0x01);       //clear LCD
    _delay_ms(2;)

}

int main() //main function
{
    unsigned char x;
    DDRA=0X0F;      //make PA0 to PA3 =O/P and PA4 to PA7 =I/P
    lcd_init();
    while(1)
    {
        PORTA=0xF0; //make all 4 column and all 4 rows 0
        if (PINA!=0xF0)
        {
            x=keypad();
            lcddata(x);

        }
    }
    return 0;
}

unsigned char keypad()
{
    PORTA=0b11111110;       //make first row 0

    if((PINA &(1<<PINA4))==0)
    {
        _delay_ms(3);
```

Compile code

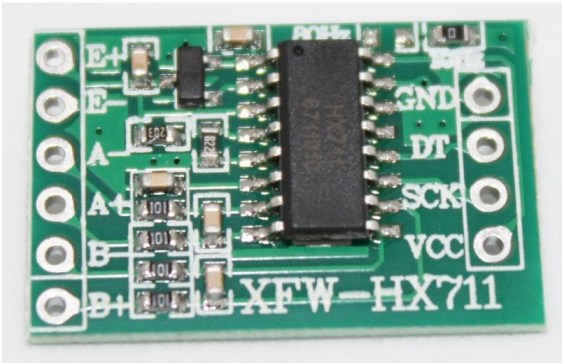
```
        return '7';
    }
    else if((PINA &(1<<PINA5))==0)
    {
        _delay_ms(3);
        return '8';
    }
    else if((PINA &(1<<PINA6))==0)
    {
        _delay_ms(3);
        return '9';
    }
    else if((PINA &(1<<PINA7))==0)
    {
        _delay_ms(3);
        return '/';
    }

    PORTA=0b11111101;       //make second row 0

    if((PINA &(1<<PINA4))==0)
    {
        _delay_ms(3);
        return '4';
    }
    else if((PINA &(1<<PINA5))==0)
    {
        _delay_ms(3);
        return '3';
    }
    else if((PINA &(1<<PINA6))==0)
    {
        _delay_ms(3);
        return '6';
    }
    else if((PINA &(1<<PINA7))==0)
    {
        _delay_ms(3);
        return '*';
    }
```

```
    }

    PORTA=0b11111011;       //make third row 0

    if((PINA &(1<<PINA4))==0)
    {
        _delay_ms(3);
        return '1';
    }
    else if((PINA &(1<<PINA5))==0)
    {
        _delay_ms(3);
        return '2';
    }
    else if((PINA &(1<<PINA6))==0)
    {
        _delay_ms(3);
        return '3';
    }
    else if((PINA &(1<<PINA7))==0)
    {
        _delay_ms(3);
        return '-';
    }

    PORTA=0b11110111;       //make fourth row 0

    if((PINA &(1<<PINA4))==0)
    {
        _delay_ms(3);
        return 'C';
    }
    else if((PINA &(1<<PINA5))==0)
    {
        _delay_ms(3);
        return '0';
    }
    else if((PINA &(1<<PINA6))==0)
    {
        _delay_ms(3);
        return '=';
    }
    else if((PINA &(1<<PINA7))==0)
    {
        _delay_ms(3);
        return '+';
    }

    return 0;
}
```

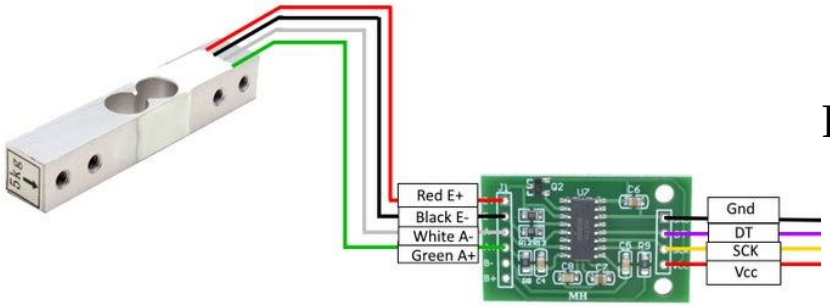
Weight Sensor (Load cell+HX711)



Load cell



HX711 Analog to Digital Converter

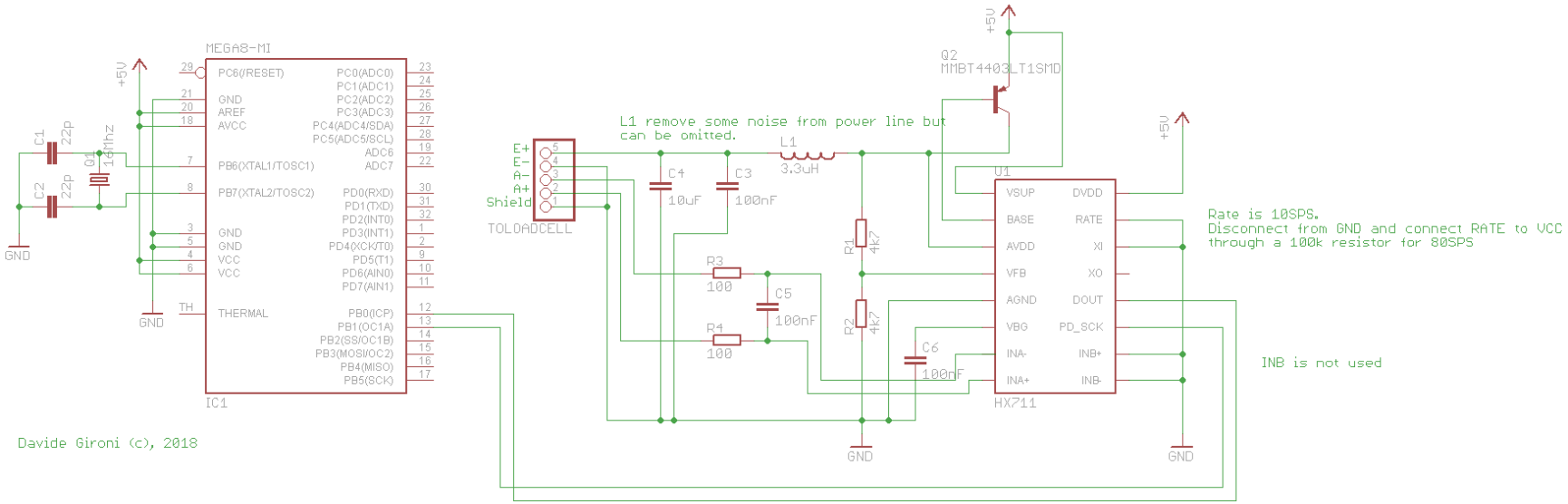


Pin diagram

Specifications

- Load cell
  - Maximum Voltage- 5.5V
  - Maximum Current- 1.5mA
  - Datasheet
- HX711 ADC
  - Maximum Voltage- 5.5V
  - Maximum Current- 0.8mA
  - Datasheet

Circuit diagram



Compile code

```
//hx711 lib example

#include <stdio.h>
#include <stdlib.h>
#include <avr/io.h>
#include <avr/interrupt.h>
#include <util/delay.h>

//include uart
#define UART_BAUD_RATE 4800
#include "uart/uart.h"

#include "hx711/hx711.h"

//defines running modes
#define HX711_MODERUNNING 1
#define HX711_MODECALIBRATION1OF2 2
#define HX711_MODECALIBRATION2OF2 3
#define HX711_MODECURRENT 1

//2 step calibration procedure
//set the mode to calibration step 1
//read the calibration offset leaving the load cell with no weight
//set the offset to value read
//put a know weight on the load cell and set calibrationweight value
//run the calibration stes 2 of 2
//read the calibration scale
//set the scale to value read

//set the gain
int8_t gain = HX711_GAINCHANNELA128;

#if HX711_MODECURRENT == HX711_MODERUNNING
//set the offset
int32_t offset = 8389246;
//set the scale
double scale = 15797.8;
#elif HX711_MODECURRENT == HX711_MODECALIBRATION1OF2
//set the offset
```

Compile code

```
int32_t offset = 8389246;
//set the scale
double scale = HX711_SCALEDEFAULT;
//set the calibration weight
double calibrationweight = 0.082;
#endif


int main(void) {
    //init uart
    uart_init(UART_BAUD_SELECT(UART_BAUD_RATE,F_CPU));

    sei();

    char printbuff[100];

    //init hx711
    hx711_init(gain, scale, offset);


#if HX711_MODECURRENT == HX711_MODERUNNING
    for(;;) {
        //get read
        int32_t read = hx711_read();
        ltoa(read, printbuff, 10);
        uart_puts("Read: "); uart_puts(printbuff); uart_puts("\r\n");

        //get weight
        double weight = hx711_getweight();
        dtostrf(weight, 3, 3, printbuff);
        uart_puts("Weight: "); uart_puts(printbuff); uart_puts("units"); uart_puts("\r\n");

        uart_puts("\r\n");

        _delay_ms(500);
    }
#elif HX711_MODECURRENT == HX711_MODECALIBRATION1OF2
    for(;;) {
```

Compile code

```
        //set calibration offset
        hx711_calibrate1setoffset();
        int32_t calibrationoffset = hx711_getoffset();
        ltoa(calibrationoffset, printbuff, 10);
        uart_puts("Calibration offset: "); uart_puts(printbuff); uart_puts("\r\n");

        uart_puts("\r\n");

        _delay_ms(500);
    }
#elif HX711_MODECURRENT == HX711_MODECALIBRATION2OF2
    for(;;) {
        //calibrate
        hx711_calibrate2setscale(calibrationweight);

        //get scale
        double scale = hx711_getscale();
        dtostrf(scale, 3, 3, printbuff);
        uart_puts("Calibration scale: "); uart_puts(printbuff); uart_puts("\r\n");

        uart_puts("\r\n");

        _delay_ms(500);
    }
#endif
}
```

Thank You

Questions