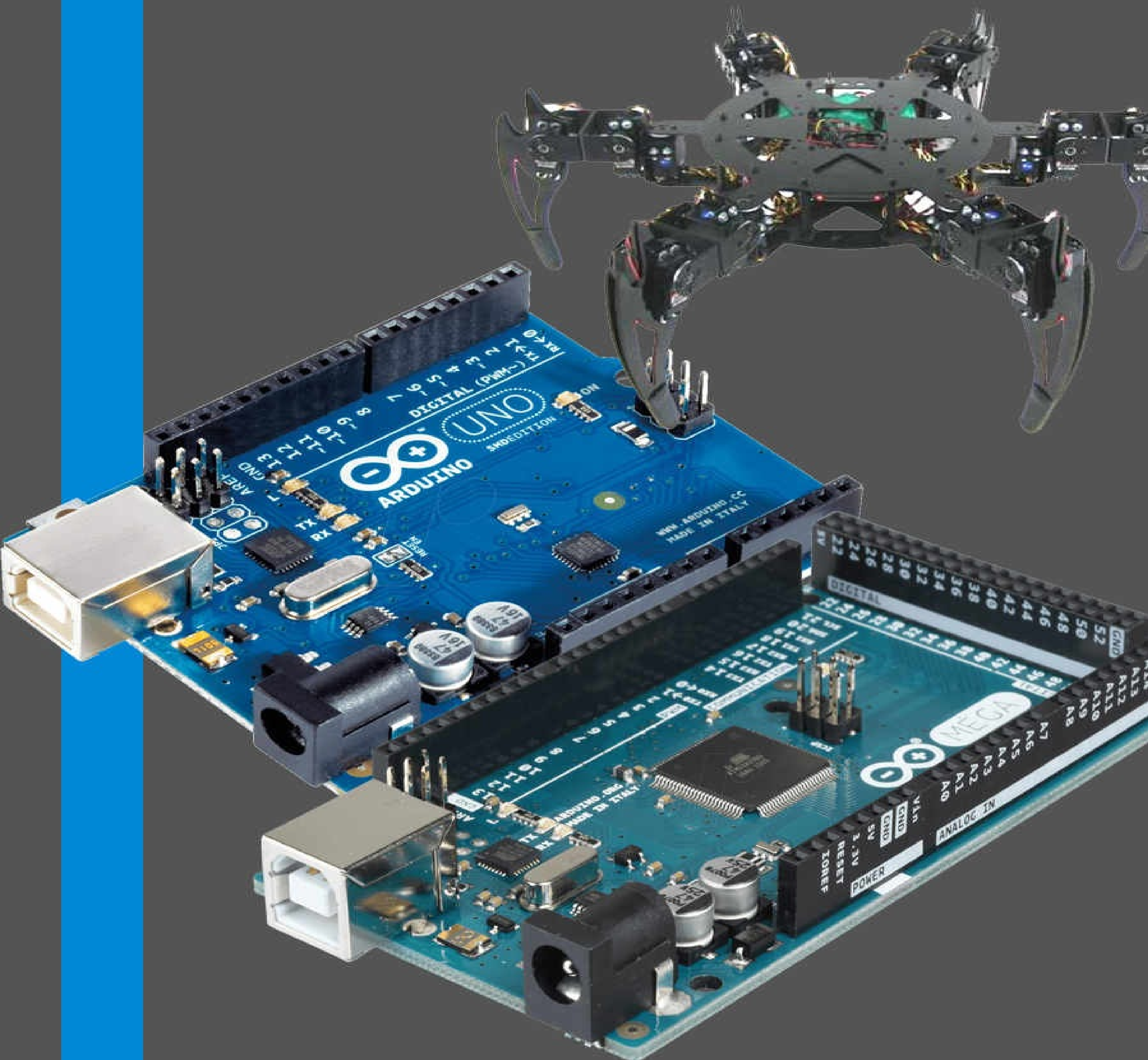
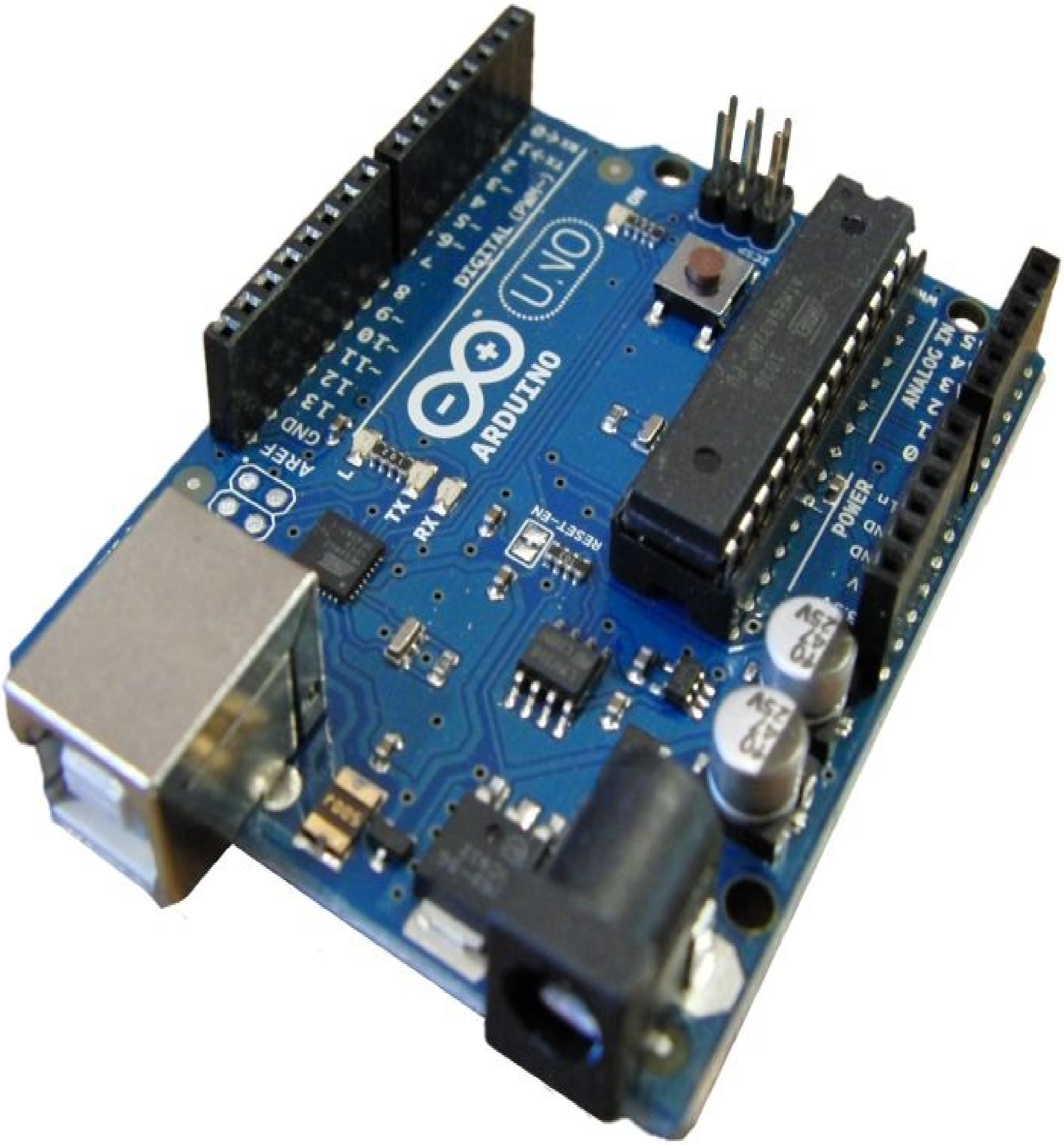
Best Arduino Project For Beginner And Advanced



Best Arduino Project For Beginner And Advanced

Index Of Contents

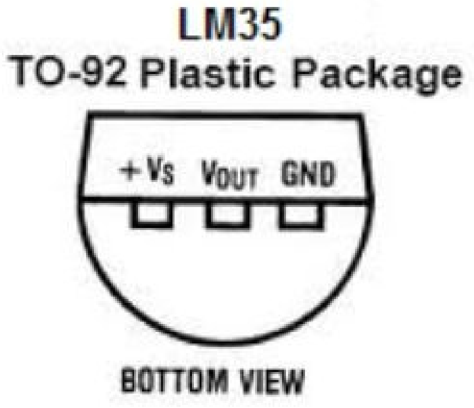
In this text, we compile all of the great and exciting projects we've developed the use of Arduino. nowadays Arduino tasks are used for educational route success as properly. In any case, Arduino is a completely exciting platform to construct actual international initiatives.



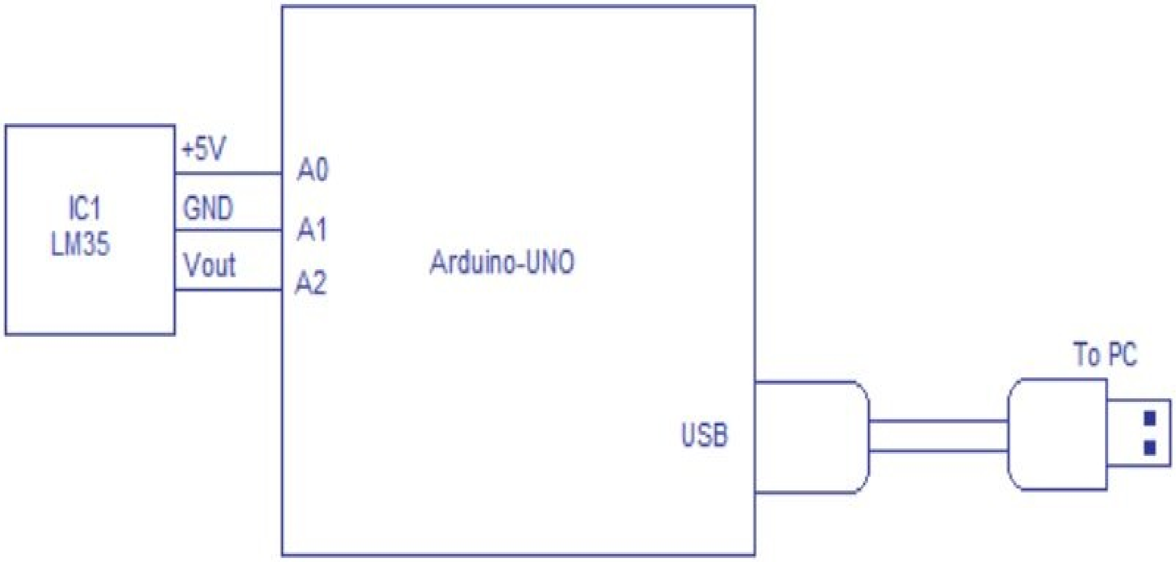
Simple Temperature Logger (°C & °F).

This challenge is about a easy USB temperature logging machine the usage of arduino uno and the serial screen function inside the arduino IDE. The gadget monitors the temperature every 2 seconds and suggests it on the arduino serial reveal. The temperature is proven in °Celsius and °Fahrenheit. The device is interfaced to the laptop via the USB port. LM35 is used because the temperature sensor.

LM35 is 3 terminal linear temperature sensor from countrywide semiconductors. it may measure temperature from-55c to +150C. The voltage output of the LM35 will increase 10mV per diploma Celsius rise in temperature. LM35 can be operated from a 5V supply and the stand by using present day is less than 60uA. The pin out of LM35 is shown within the figure under.



Circuit diagram.



Temperature logger using Arduino www.circuitstoday.com

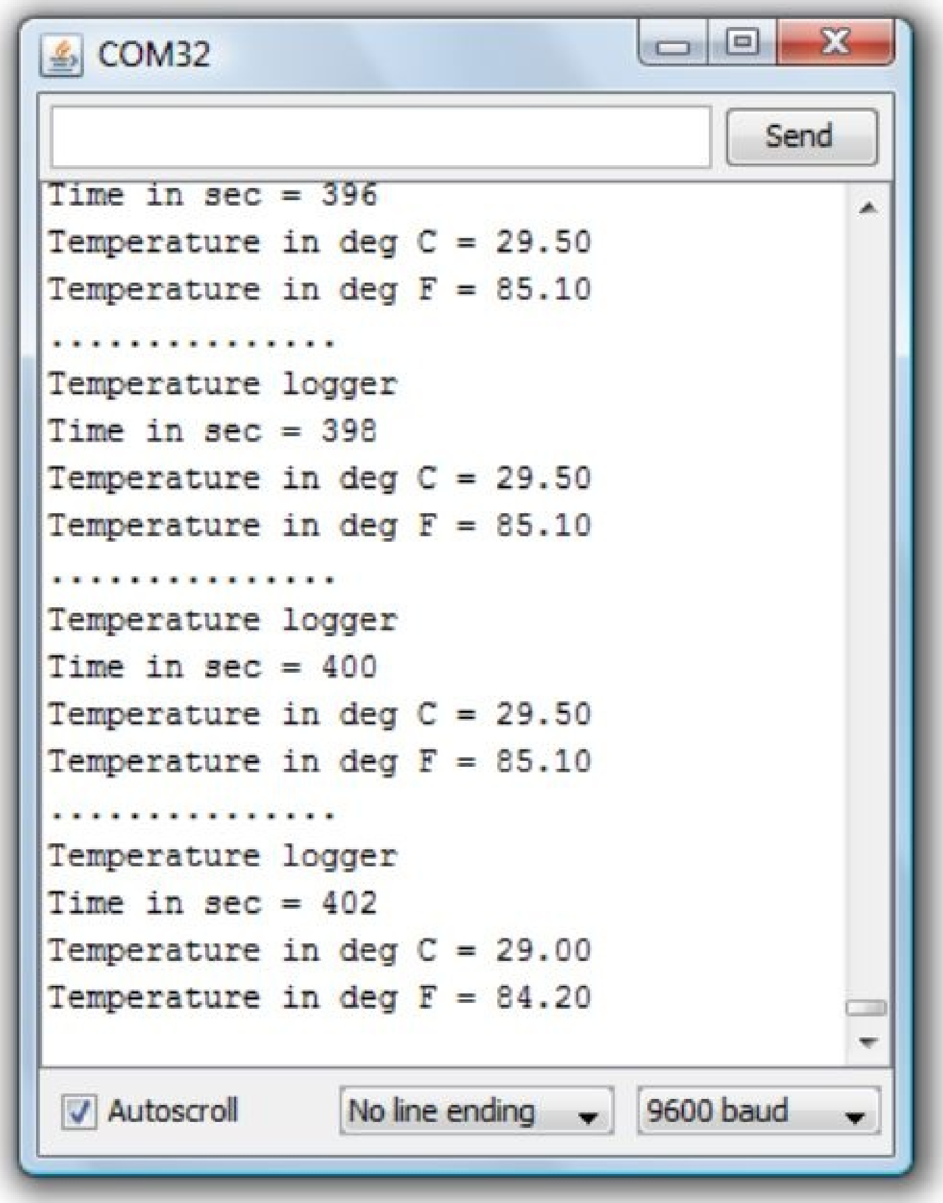
Temperature sensor LM35 is interfaced to the Arduino thru the analog enter pins A0, A1 and A2. Analog enter pin A0 is made excessive and it acts because the 5V supply pin for the LM35. Analog enter pin A2 is made low and it acts as the ground pin for the LM35. Analog input pin A1 is ready as an enter and the voltage output of LM35 is coupled to the arduino via this pin. This scheme could be very useful due to the fact you could plug the LM35 directly into the analog enter female connector and no external connection wires are needed. The arduino board is powered by the computer thru the USB cable and no external strength supply is wanted in this circuit. The USB port also serves because the medium for communication among arduino and laptop.

Program.

```
int t=0;
int vcc=A0; // sets analog input A0 as +5V source for LM35
int sensor=A1; // sets A1 as the sensor input
int gnd=A2; // sets analog input A2 as ground for LM35
float temp;
float tempc;
float tempf;
void setup()
{
pinMode(vcc,OUTPUT);
pinMode(gnd,OUTPUT);
pinMode(sensor,INPUT);
digitalWrite(vcc,HIGH); // sets analog input A0 HIGH
digitalWrite(gnd,LOW); // sets analog input A2 LOW
Serial.begin(9600); // sets the baud rate at 9600

}
void loop()
{ delay(2000); // calls a 2 second delay
t=t+2; // increments the time by 2 every two seconds
temp=analogRead(sensor); // reads the LM35 output
tempc=(temp*5)/10; // converts the digital value into temperature degree C
tempf=(tempc*1.8)+32; // converts degree C to degree F
Serial.println("...............");
Serial.println("Temperature logger");
Serial.print("Time in sec = "); // prints the time on serial monitor window
Serial.println(t);
Serial.print("Temperature in deg C = "); // prints the temperature in degreeC
Serial.println(tempc);
Serial.print("Temperature in deg F = "); // prints the temperature in degreeF
Serial.println(tempf);
}
```

About the program.

The voltage output of LM35 is hooked up to the analog input A1 of the arduino. The voltage at this pin could be proportional to the temperature and this voltage is read the usage of analogRead feature. The analogRead feature will examine the voltage (in a range 0 to five) at a selected analog enter pin and converts it right into a digital fee between zero and 1023. as an example, if 29°C is the temperature, the output of LM35 will be 290mV. The end result of the analogRead feature will be 290mV/(5/1023) =fifty nine. There should be a few way to convert this 59 to 29.0 for showing in the serial screen window. this is done by way of multiplying 59 by way of 5 and then dividing the end result with the aid of 10. The result might be the temperature in °C and it's miles displayed the use of Serial.print function. Then it's far converted to °F using the following system: °F= (°C*1.eight)+32. The temperature in °F is likewise displayed. The serial reveal may be accessed from the equipment tab in the arduino IDE. The shortcut for serial reveal is ctrl+shift+M. The photograph of the serial reveal window is shown in the parent under.



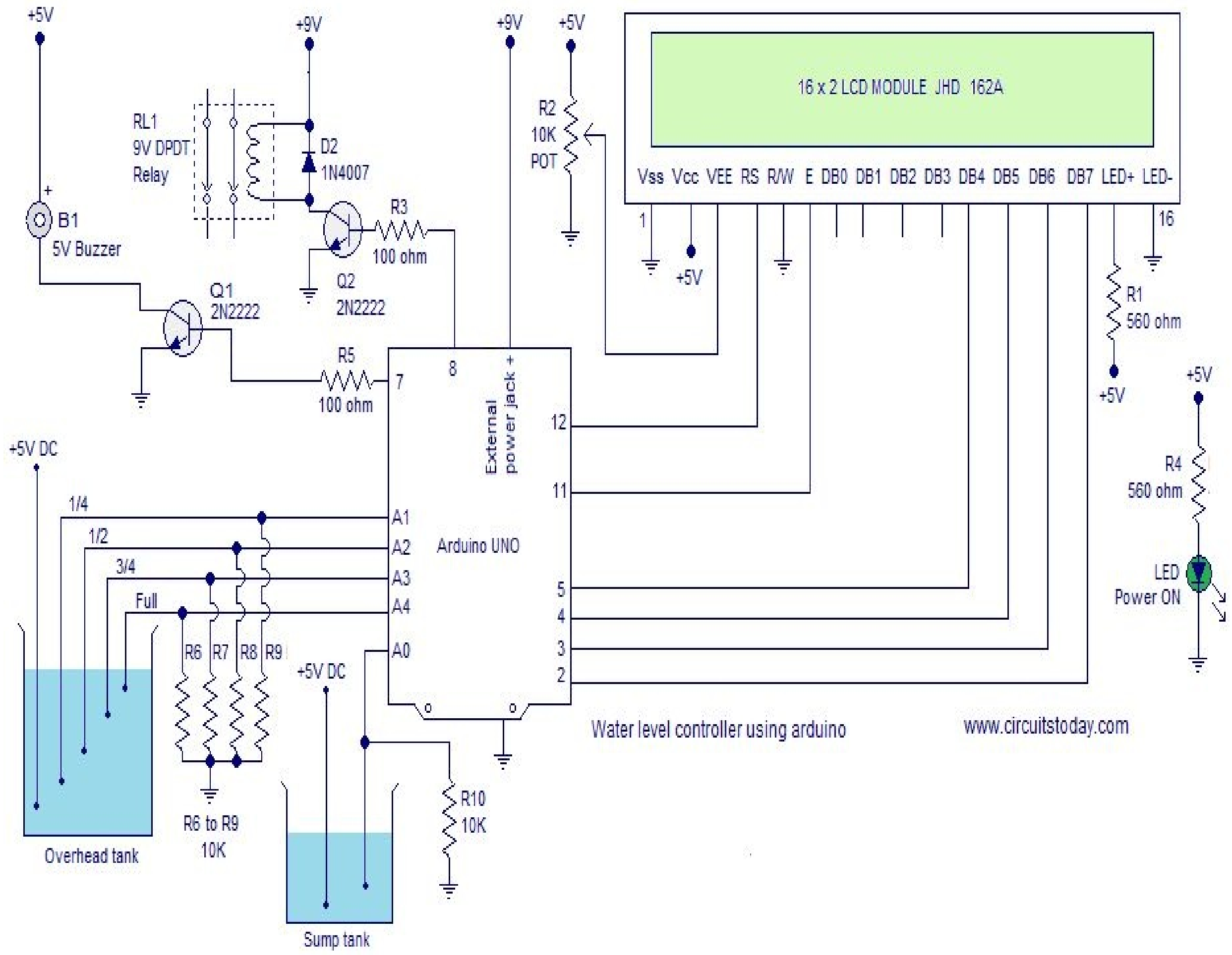
```
Time in sec = 396
Temperature in deg C = 29.50
Temperature in deg F = 85.10
...............
Temperature logger
Time in sec = 398
Temperature in deg C = 29.50
Temperature in deg F = 85.10
...............
Temperature logger
Time in sec = 400
Temperature in deg C = 29.50
Temperature in deg F = 85.10
...............
Temperature logger
Time in sec = 402
Temperature in deg C = 29.00
Temperature in deg F = 84.20
```

☑ Autoscroll   No line ending   9600 baud

Arduino Water Level Indicator.

This text is a about a totally purposeful water degree controller using Arduino. The circuit shows the degree of water within the tank and switches the motor ON when the water level is going beneath a predetermined level. The circuit robotically switches the motor OFF while the tank is full. The water level and other critical statistics are displayed on a sixteen×2 lcd display. The circuit also monitors the level of water within the sump tank (source tank). If the level in aspect the sump tank is low, the motor will now not be switched ON and this protects the motor from dry strolling. A beep sound is generated whilst the level in the sump tank is low or if there's any fault with the sensors.

Circuit diagram.



The circuit diagram of the water stage controller the use of Arduino is proven above. Conductive technique is used to measure the extent. The sensor meeting consists of 4 aluminum wires arranged at 1/4, 1/2, 3/four and complete ranges in the tank. The dry ends of these wires are related to analog input pins A1, A2, A3 and A4 of the Arduino respectively. A 5th twine is positioned at the lowest of the tank. Resistors R6 to R9 are pull down resistors.The dry give up of this twine is hooked up to +5V DC. while the water touches a particular probe, electric connection is mounted among that probe and the +5V probe because water has slight conductivity. As a result contemporary flows thru that probe and this present day is converted right into a proportional voltage with the aid of the pull down resistor. Arduino reads the voltage dropped across every pull down resistor for sensing the level of water inside the tank. equal method is used for measuring the extent of water in the sump tank.

Digital pin 7 of the Arduino controls the buzzer and virtual pin 8 controls the motor. Transistor Q1 drives the buzzer and resistor R5 limits the base current of Q1. Transistor Q2 drives the relay. Resistor R3 limits the bottom present day of Q2. D2 is a freewheeling diode. POT R2 is used to modify the evaluation of the liquid crystal display. resistor R1 limits the current via the back light LED. Resistor R4 limits the modern-day via the electricity ON LED. whole software for the water degree controller using Arduino is given below.

Program.

```
#include <LiquidCrystal.h>
int sump=A0;
int qut=A1;
int hlf=A2;
int thf=A3;
int ful=A4;
int motor=8;
int buz=7;
int s;
int q;
int h;
int t;
int f;
int i; //motor status flag
int v=100; //comparison variable(needs some adjustment)
int b=0; //buzzer flag
int m=0; //motor flag
int c=0; //sump flag

LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

void setup()
{

pinMode(qut,INPUT);
pinMode(hlf,INPUT);
pinMode(qut,INPUT);
pinMode(ful,INPUT);
pinMode(sump,INPUT);
pinMode(motor,OUTPUT);
pinMode(buz,OUTPUT);
lcd.begin(16, 2);
digitalWrite(buz,LOW);
}

void loop()
{

i=digitalRead(motor);
s=analogRead(sump);
q=analogRead(qut);
h=analogRead(hlf);
t=analogRead(thf);
f=analogRead(ful);
lcd.clear();

if(f>v && t>v && h>v && q>v )
{
lcd.setCursor(0,0);
lcd.print(char(219));
lcd.print(char(219));
lcd.print(char(219));
lcd.print(char(219));
lcd.setCursor(5,0);
lcd.print("FULL");
m=0;
b=0;
}
else
{
if(f<v && t>v && h>v && q>v)
{
lcd.setCursor(0,0);
lcd.print(char(219));
lcd.print(char(219));
lcd.print(char(219));
lcd.print("_");
lcd.setCursor(5,0);
lcd.print("3/4th");
b=0;
}
else
{
if(f<v && t<v && h>v && q>v)
{
lcd.setCursor(0,0);
lcd.print(char(219));
lcd.print(char(219));
lcd.print("_");
lcd.print("_");
lcd.setCursor(5,0);
lcd.print("HALF");
m=1;
b=0;
}
else
if(f<v && t<v && h<v && q>v)
{
lcd.setCursor(0,0);
lcd.print(char(219));
lcd.print("_");
lcd.print("_");
lcd.print("_");
lcd.setCursor(5,0);
lcd.print("1/4th");
b=0;
}
else
{
if(f<v && t<v && h<v && q<v)
{
lcd.setCursor(0,0);
lcd.print("_");
lcd.print("_");
lcd.print("_");
lcd.print("_");
lcd.setCursor(5,0);
lcd.print("LOW");
b=0;
}
else

{
digitalWrite(motor,LOW);
lcd.setCursor(0,0);
lcd.print("ERROR!");
b=1;
}
}}}
if(i==HIGH)
{
lcd.setCursor(0,1);
lcd.print("Motor ON");
}
else
{
lcd.setCursor(0,1);
lcd.print("Motor OFF");
}
```

```
if(s>v && m==1)
{
digitalWrite(motor,HIGH);
}
if(s<v)
{
digitalWrite(motor,LOW);
lcd.setCursor(11,0);
lcd.print("Low");
lcd.setCursor(11,1);
lcd.print("Sump");
c=1;
}
if(s>v)
{
c=0;
}

if(m==0)
{
digitalWrite(motor,LOW);
}

if(b==1 || c==1)
{
digitalWrite(buz,HIGH);
delay(500);
digitalWrite(buz,LOW);
}
else
{
digitalWrite(buz,LOW);
}
delay(100);
lcd.clear();
}
```

About the program.

The Arduino reads the sensor output thru the analog enter pins the usage of analogRead characteristic. as an instance q=analogRead(qut); converts the voltage (in the variety 0 to 5V) at the “sector” probe into various (within the range 0 to 1023) and saves it into the variable “q”. This way the voltage at every prob is scanned to corresponding variables. The those variables are as compared to a fixed quantity (a hundred right here) for identifying the cutting-edge circumstance. honestly one hundred is the equal of zero.forty eight volts and if the voltage at a particular sensor is greater than this, it is considered as an electrical continuity and water is believed to be touching the probe. The vale of the fixed number (comparison variable”v”) wishes a few adjustment because the resistivity of water modifications from region to place and the gap among the sensor probes may be different in different tanks.

Notes.

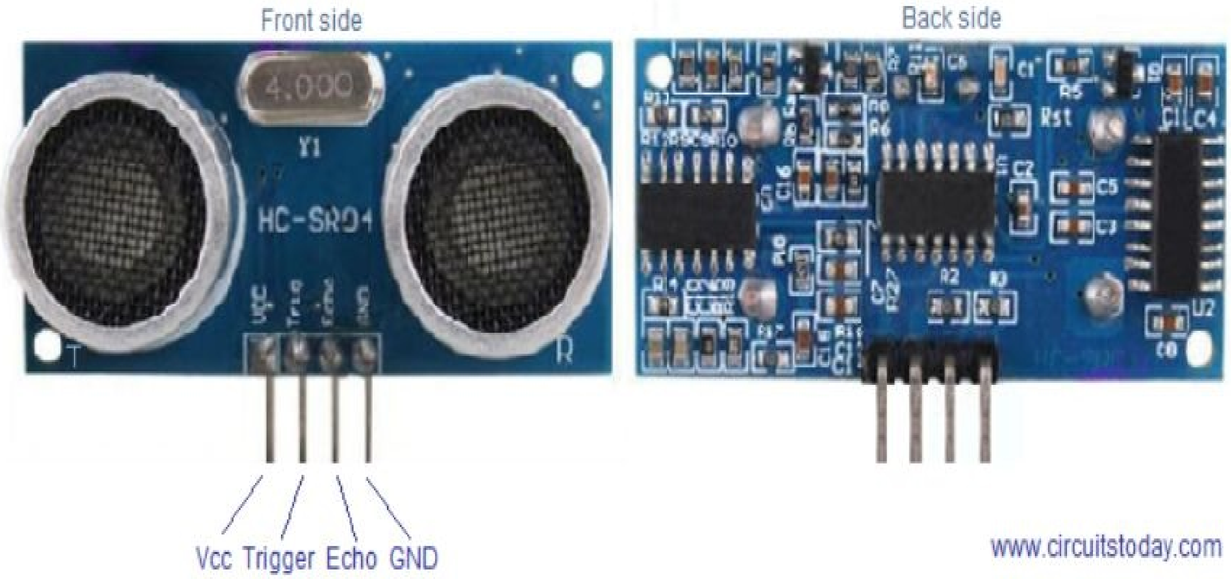
- The circuit is powered thru the 9V outside energy jack on the arduino board.
- 5V needed at extraordinary factors within the circuit can be tapped from the 5V output on the arduino board.
- Use suitable excellent aluminum wires for probe. Do not use copper wires.

Ultrasonic Range Finder.

Ultrasonic variety finder the use of 8051 mictrocontroller has been already published by way of me on this internet site. This time it's miles an ultrasonic variety finder using arduino. HC-SR04 ultrasonic range finder module is used because the sensor right here. The show includes a 3 digit multiplexed seven segment show. This variety finder can measure as much as 200 cm and has an accuracy of 1cm. there is an choice for showing the space in inch additionally. traditional packages of this range finder are parking sensors, impediment warning system, degree controllers, terrain monitoring gadgets and so forth. lets have a study the HC-SR04 ultrasonic module first.

HC SR04 ultrasonic module.

HC SR04 is an ultrasonic variety locating module with an accuracy of zero.3cm. The sensing variety of this module is from 2cm to 5 meter. operating contemporary of this sensor is 15mA and the measuring angle is 15°. The photograph of front and back side of the HC-SR04 sensor is shown within the figure below.

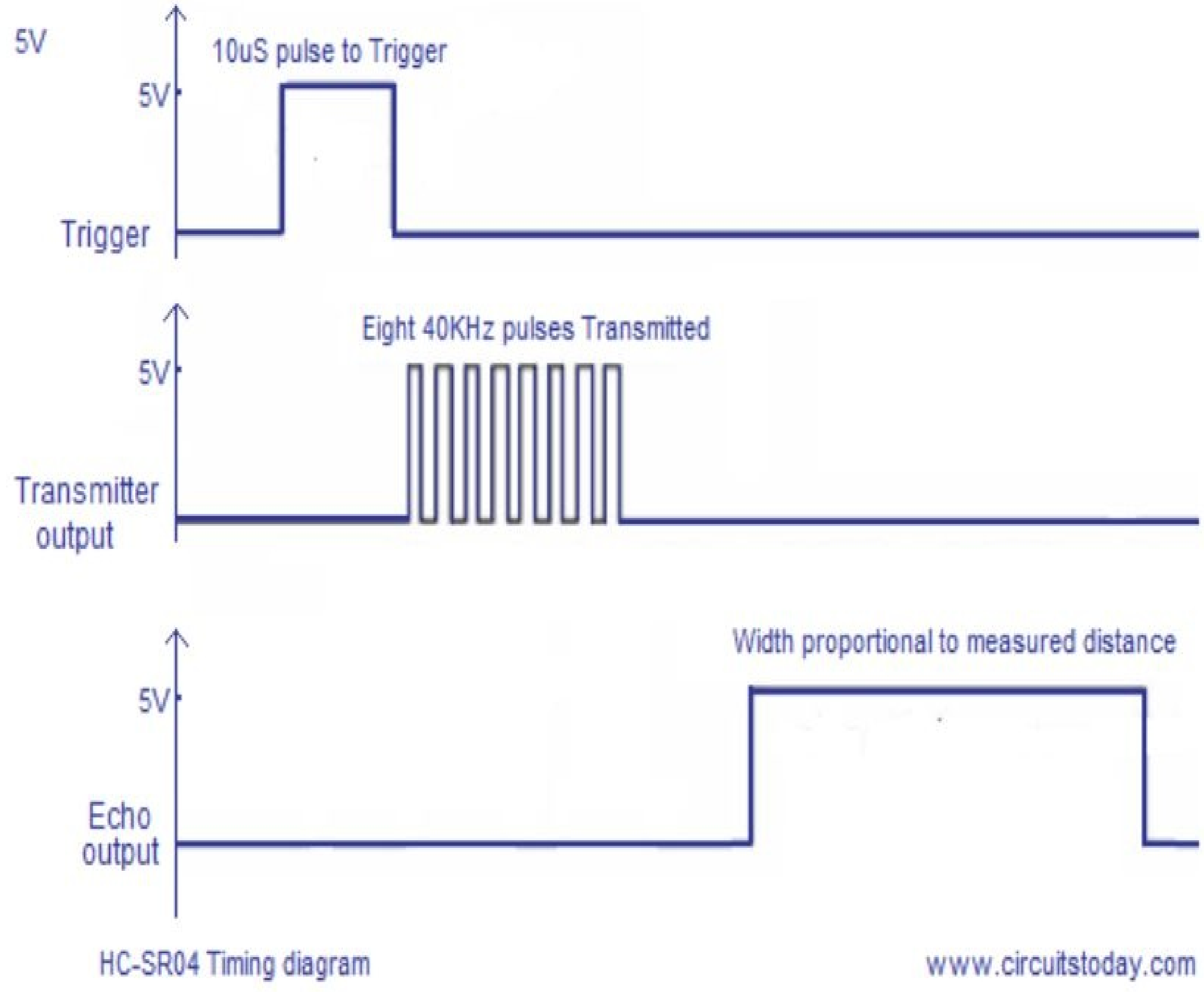


HC-SR04 has four pins. Their names and capabilities are defined underneath.

- Vcc: 5V deliver voltage is given to this pin.
- cause: A 10uS long pulse is given to this pin for triggering the transmission. Upon receiving a valid cause pulse, the HR-SR04 troubles eight 40KHz pulses. Time taken by way of these pulses to mirror lower back is measured and the space is calculated from it.

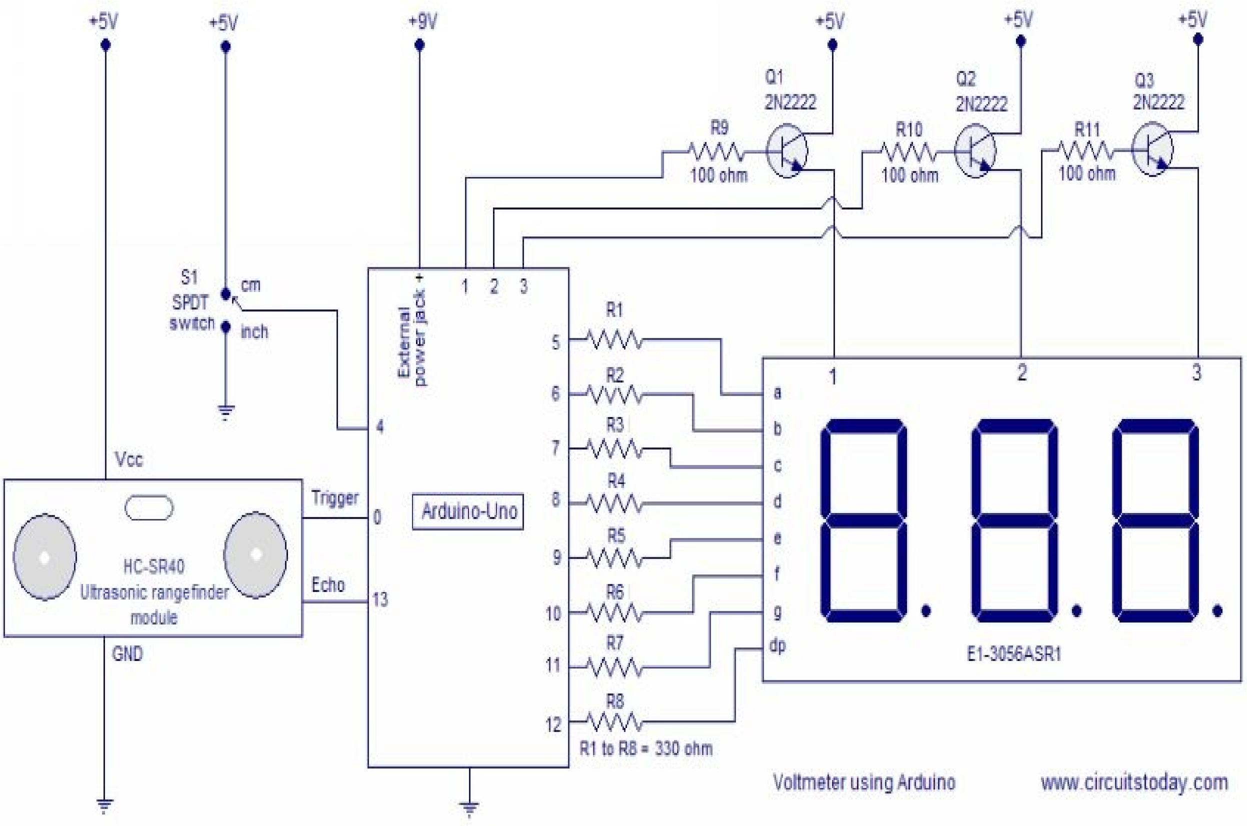
- Echo: At this pin the HC-SR04 outputs a sign whose high time is proportional to the variety.
- ground : ground is connected to this pin.

The timing diagram of HC-SR04 is proven inside the discern beneath.



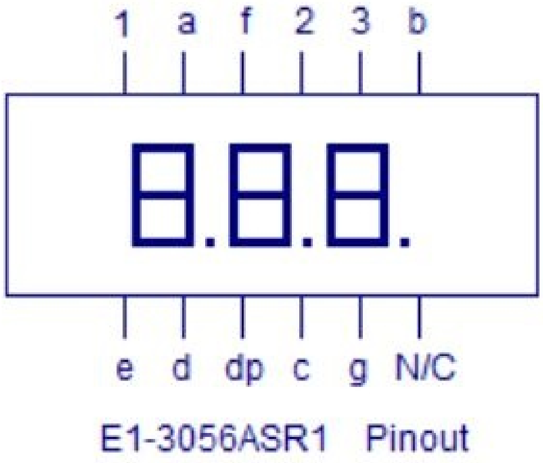
Circuit diagram.

Complete circuit diagram of the ultrasonic variety finder using arduino is proven in the discern beneath.



Cause pin of the ultrasonic variety finder module is attached to virtual pin 0 of the arduino. Echo pin of the ultrasonic module is hooked up to the digital pin thirteen of the arduino. SPDT transfer S1 is used to choose the unit of the dimension proven within the show. Pole of the SPDT switch S1 is attached to virtual pin four of the arduino. If virtual pin 4 is held high, the output can be in centimeters and if the digital pin 4 is held low, the output may be in inches. Digit driving force transistor Q1, Q2 and Q3 of the arduino are interfaced to virtual pins 1, 2 and 3 of the arduino. Multiplexed segments a to dp are interfaced to virtual pins 5 to twelve of the arduino.

The arduino board may be powered thru the +9V jack given at the board. 5V supply wanted in a few different components of the circuit may be obtained from the 5V supply available in the arduino board. Resistors R9, R10 and R11 limits the base contemporary of the corresponding transistors. 330 ohm resistors R1 to R8 limits the present day via the corresponding segments. Pin out of an E1-3056ASR1 3 digit MUX seven segment show in shown inside the discern below.



Program.

```
#include <NewPing.h>
#define trig 0
#define echo 13
#define maximum 200
int a;
int unit;
int usec;
int input=4;
int disp1=1;
int disp2=2;
int disp3=3;
int segA=5;
int segB=6;
int segC=7;
int segD=8;
int segE=9;
int segF=10;
int segG=11;
int segDP=12;
NewPing sonar(trig, echo, maximum);
void setup()
{
pinMode(disp1, OUTPUT);
pinMode(disp2, OUTPUT);
pinMode(disp3, OUTPUT);
pinMode(segA, OUTPUT);
pinMode(segB, OUTPUT);
pinMode(segC, OUTPUT);
pinMode(segD, OUTPUT);
pinMode(segE, OUTPUT);
pinMode(segF, OUTPUT);
pinMode(segG, OUTPUT);
pinMode(segDP, OUTPUT);
pinMode(input, INPUT);
}
```

```
void loop()
{
delay(20);
usec=sonar.ping(); //sends ping and estimates the duration of echo in uS
unit= digitalRead(input); //reads the status of cm/inch selector switch
if(unit==1)
{
usec=usec/58; // distance in cm
}
else
{
    usec=usec/148; // distance in inch
}
a=usec%10;
digitalWrite(disp1,LOW);
digitalWrite(disp2,LOW);
digitalWrite(disp3, HIGH);
digitalWrite(segDP,HIGH);
display(a);
delay(4);
usec = usec/10;
a = usec%10;
digitalWrite(disp3,LOW);
digitalWrite(disp2,HIGH);
digitalWrite(segDP,HIGH);
display(a);
delay(4);
usec=usec/10;
a=usec;
digitalWrite(disp2,LOW);
digitalWrite(disp1,HIGH);
digitalWrite(segDP,HIGH);
display(a);
delay(4);
}
int display (int a)
{
switch (a)
{
    case 0:
    digitalWrite(segA, LOW);
    digitalWrite(segB, LOW);
    digitalWrite(segC, LOW);
    digitalWrite(segD, LOW);
    digitalWrite(segE, LOW);
    digitalWrite(segF, LOW);
    digitalWrite(segG, HIGH);
    break;


    case 1:
    digitalWrite(segA, HIGH);
    digitalWrite(segB, LOW);
    digitalWrite(segC, LOW);
    digitalWrite(segD, HIGH);
    digitalWrite(segE, HIGH);
    digitalWrite(segF, HIGH);
    digitalWrite(segG, HIGH);
    break;

    case 2:
    digitalWrite(segA, LOW);
    digitalWrite(segB, LOW);
    digitalWrite(segC, HIGH);
    digitalWrite(segD, LOW);
    digitalWrite(segE, LOW);
    digitalWrite(segF, HIGH);
    digitalWrite(segG, LOW);
    break;

    case 3:
    digitalWrite(segA, LOW);
    digitalWrite(segB, LOW);
    digitalWrite(segC, LOW);
    digitalWrite(segD, LOW);
    digitalWrite(segE, HIGH);
    digitalWrite(segF, HIGH);
    digitalWrite(segG, LOW);
    break;

    case 4:
    digitalWrite(segA, HIGH);
    digitalWrite(segB, LOW);
    digitalWrite(segC, LOW);
    digitalWrite(segD, HIGH);
    digitalWrite(segE, HIGH);
    digitalWrite(segF, LOW);
    digitalWrite(segG, LOW);
    break;

    case 5:
    digitalWrite(segA, LOW);
    digitalWrite(segB, HIGH);
    digitalWrite(segC, LOW);
    digitalWrite(segD, LOW);
    digitalWrite(segE, HIGH);
    digitalWrite(segF, LOW);
    digitalWrite(segG, LOW);
    break;

    case 6:
    digitalWrite(segA, LOW);
    digitalWrite(segB, HIGH);
    digitalWrite(segC, LOW);
```

```
    digitalWrite(segD, LOW);
    digitalWrite(segE, LOW);
    digitalWrite(segF, LOW);
    digitalWrite(segG, LOW);
    break;

    case 7:
    digitalWrite(segA, LOW);
    digitalWrite(segB, LOW);
    digitalWrite(segC, LOW);
    digitalWrite(segD, HIGH);
    digitalWrite(segE, HIGH);
    digitalWrite(segF, HIGH);
    digitalWrite(segG, HIGH);
    break;

    case 8:
    digitalWrite(segA, LOW);
    digitalWrite(segB, LOW);
    digitalWrite(segC, LOW);
    digitalWrite(segD, LOW);
    digitalWrite(segE, LOW);
    digitalWrite(segF, LOW);
    digitalWrite(segG, LOW);
    break;

    case 9:
    digitalWrite(segA, LOW);
    digitalWrite(segB, LOW);
    digitalWrite(segC, LOW);
    digitalWrite(segD, LOW);
    digitalWrite(segE, HIGH);
    digitalWrite(segF, LOW);
    digitalWrite(segG, LOW);
    break;
}}
```

About the program.

For speaking with the ultrasonic range finder module, library feature is used. The job of sending the 10uS trigger pulse, waiting for the echo and measuring the width of the echo and so forth are performed by the library function. just one line of code usec=sonar.ping()will make the arduino to do all the jobs stated above and the width of the echo pulse in micro seconds will be saved inside the variable usec. Dividing the heartbeat width in uS by 58 will deliver the distance in cm and dividing the heartbeat width in uS through 148 will give the gap in inch. An “if – else” loop is used for selecting the unit according to the location of the SPDT selector transfer(S1). showing the gap on the three digit 7 phase show is performed by way of the approach utilized in the sooner undertaking

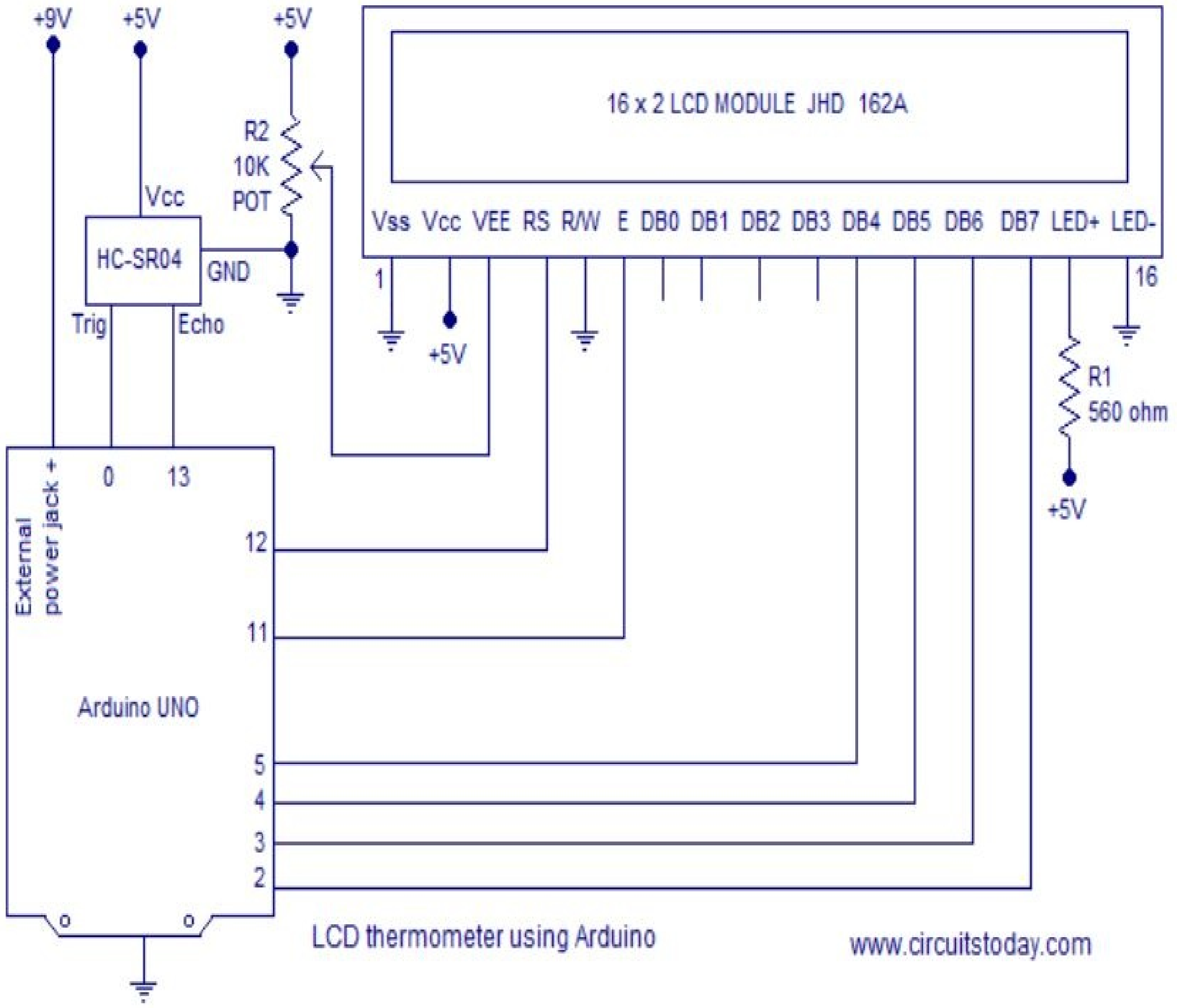
Voltmeter the use of arduino.

The library may be downloaded from here: NewPing_v1.5. down load this zip report, unzip it into a folder, name it NewPing or something and replica it into the …….program files/Arduino/Library folder.

Ultrasonic Range Finder With LCD.

That is just the lcd model of the above mission. The working precept of this circuit is equal as that of the 7-segment LED version. most effective trade is at the display device. in this circuit a 16×2 lcd display is used for showing the gap. the space in cm and inch are displayed concurrently at the liquid crystal display display screen. earlier than attempting the liquid crystal display version, undergo this text: Interfacing lcd to Arduino. Circuit diagram of the liquid crystal display variety finder is proven underneath.

Circuit diagram: LCD range finder.



Program: LCD range finder.

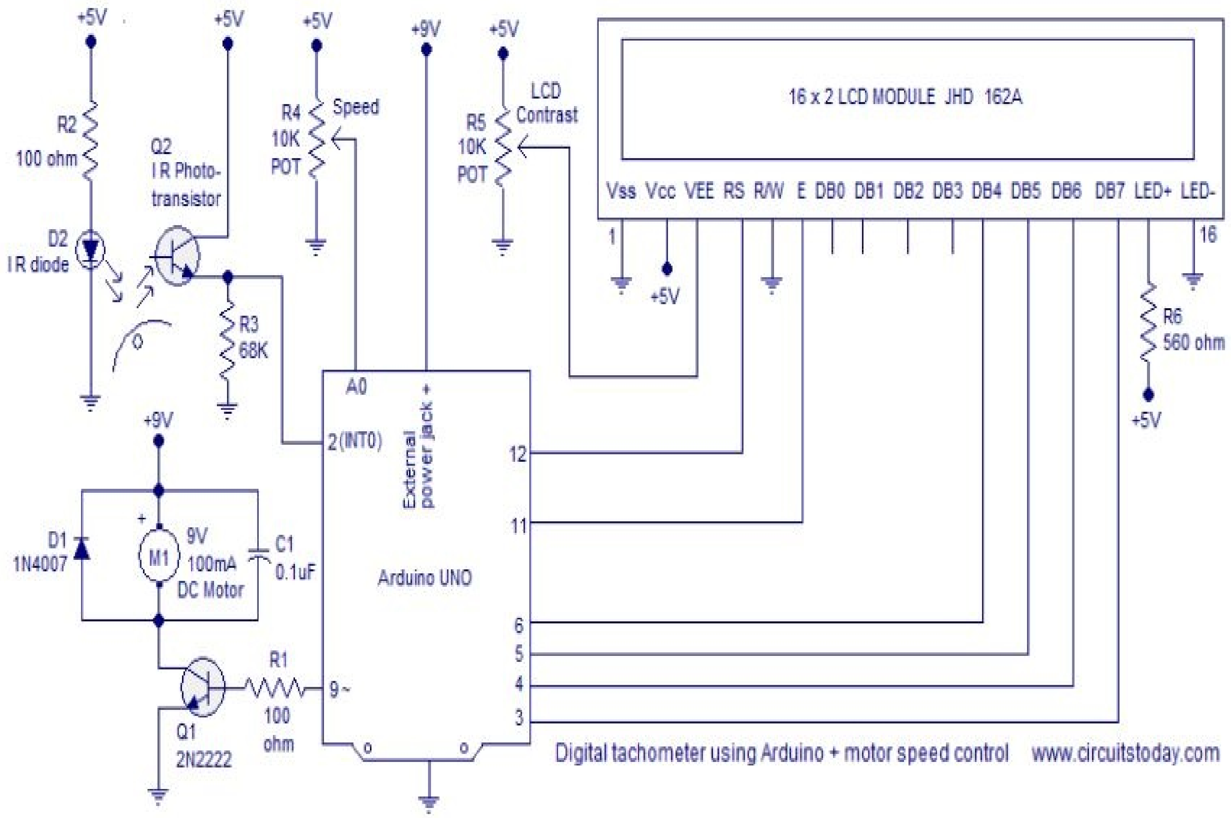
```
#include<NewPing.h>
#include<LiquidCrystal.h>
#define trig 0
#define echo 13
```

```
#define maximum 200
int usec;
int cm;
float inch;
NewPing sonar(trig, echo, maximum);
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);
void setup()
{
lcd.begin(16,2);

}
void loop()
{ lcd.clear();
lcd.setCursor(2,0);
lcd.print("Range Finder");
usec=sonar.ping();
cm=usec/58;
inch=usec/58/2.54;
lcd.setCursor(0,1);
lcd.print(cm);
lcd.print("cm");
lcd.setCursor(7,1);
lcd.print(inch);
lcd.print("inch");
delay(250);

}
```

Digital tachometer Plus Speed Control.

Tachometer is a tool used for measuring the number of revolutions of an object in a given c program language period of time. commonly it's far expressed in revolutions in step with minute or RPM. earlier tachometers only mechanical wherein the revolution is transferred to the tachometer via mechanical coupling (cable or shaft) , the rpm is determined the usage of a equipment mechanism and it's miles displayed on a dial. With the arrival of modern electronics, the tachometers have changed plenty. this text is about a contactless virtual tachometer the usage of arduino. the velocity of the motor may be also controlled the use of the same circuit. The RPM and all of the other informations are displayed on a 16×2 lcd display. The circuit diagram of the digital tachometer using arduino is shown under.

Circuit diagram.

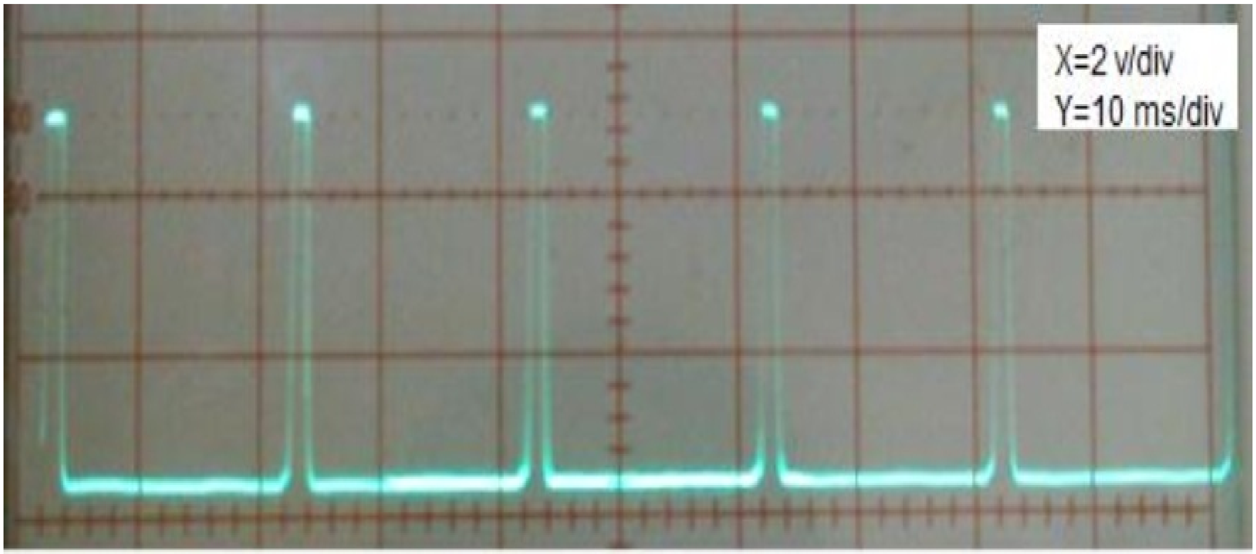


Digital tachometer using Arduino + motor speed control www.circuitstoday.com

RPM Sensor.

An IR picture transistor and IR LED forms the sensor. IR photo transistor is a type of photograph transistor which responds to infra-pink waves best. the usage of IR phototransistor avoids different light interferences from the environment. The photograph transistor and IR diode are aligned aspect through facet. Resistor

R2 limits the present day thru the IR diode. A reflective strip is glued at the rotating item (shaft, disc or fan) consistent with the sensor. I used a 9V/100mA cooling fan. The clearence among the sensor and reflective strip must be much less than 1cm. when the reflective strip passes in front of the sensor, IR waves are meditated again to the photograph transistor. The photo transistor conducts extra at this moment and as a end result the voltage across R3(68K resistor) shoots up at this second. The result can be a waveform like what proven underneath on the emitter of the photograph transistor. RPM can be decided via counting the wide variety of upward shoots in a given c programming language of time.



Counting the RPM.

Arduino is used for counting the RPM and showing it at the lcd display. Emitter of the photograph transistor is connected to the Interrupt zero (virtual pin 2) of the arduino. The arduino interrupt is configured to be rising aspect brought on. As a result the might be an interrupt for every upward shoot within the emitter waveform. The number of interrupts happened in a given time is counted through incrementing a varible the use of the interrupt carrier recurring. The time elapsed throughout te counting cycle is determined using the millis() characteristic. The millis() characteristic returns the range of milli seconds handed because the arduino board is switched ON. Calling the millis() function earlier than and after the counting cycle and the taking their distinction gives the times surpassed for the duration of the counting cycle. The (variety of interrupts/time in milliseconds)*60000 will give the revolutions in step with minute (RPM).

Controlling The Speed Of Motor.

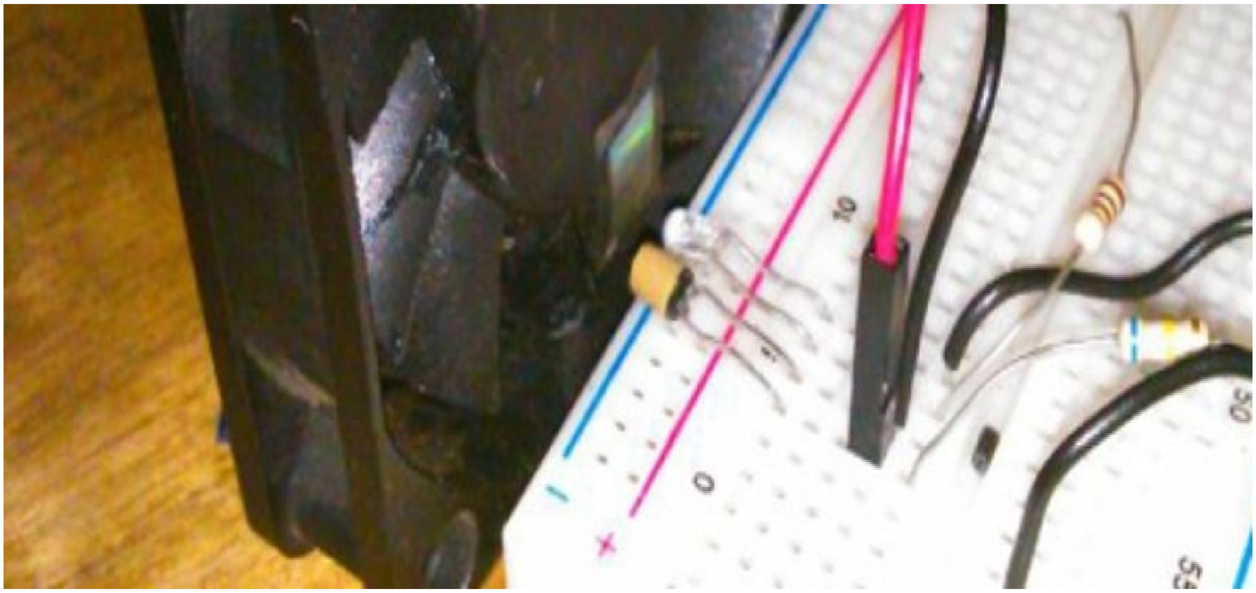
A provision for controlling the motor speed using a potentiometer is also protected within the circuit. Transistor Q1 is used for driving the motor. Its base is connected to pwm pin 9 of the arduino thru the present day limiting resistor R1. Wiper of the velocity manipulate POT R4 is hooked up to anlog pin A0 of the arduino. The voltage at this pin is converted into a price between zero and 1023 the use of the anlogRead feature. Then this fee is split by using four to in shape it into the zero to 255 range. Then this fee is written to the PWM pin 9 using the anlogWrite characteristic. The end result will be a square wave at pin nine whose responsibility cycle is proportional to the value written the usage of the analogWrite feature. as an example if the cost is 255, the duty cycle could be 100% and if the value is 127, the duty cycle can be around 50%. D1 is a unfastened wheeling diode and C1 is a noise by-bypass capacitor(de coupler). The rpm and duty cycle are displayed at the liquid crystal display display using the usual LiquidCrystal library. study this newsletter: Interfacing liquid crystal display to Arduino. full application for the digital tachometer using arduino is shown underneath.

Program.

```
#include<LiquidCrystal.h>
LiquidCrystal lcd(12,11,6,5,4,3);
int pwm=9;
int pot=A0;
float value=0;
int percent;
float rev=0;
int rpm;
int oldtime=0;
int time;

void isr() //interrupt service routine
{
rev++;
}

void setup()
{
lcd.begin(16,2); //initialize LCD
attachInterrupt(0,isr,RISING); //attaching the interrupt
}

void loop()
{
delay(1000);
detachInterrupt(0); //detaches the interrupt
time=millis()-oldtime; //finds the time
rpm=(rev/time)*60000; //calculates rpm
oldtime=millis(); //saves the current time
rev=0;
value=analogRead(pot); //reads the speed control POT
value=value/4;
analogWrite(pwm,value); //sets the desired speed
percent=(value/255)*100; //finds the duty cycle %
lcd.clear();
lcd.setCursor(0,0);
lcd.print("___TACHOMETER___");
lcd.setCursor(0,1);
lcd.print(rpm);
lcd.print(" RPM");
lcd.print(" ");
lcd.print(percent);
lcd.print("%");
attachInterrupt(0,isr,RISING);

}
```

Notes.

- The arduino board can be powered the use of a 9V supply thru the external power jack.
- The 5V wanted at some parts of the circuit can be tapped from the 5V supply at the arduino board.
- The fan I used was rated 9V/100mA. The transistor 2N2222 can deal with most effective upto 800mA. preserve this in thoughts whilst deciding on the burden.
- The lcd module used was JHD162A.
- POT R5 may be used to regulate the assessment of the lcd show. when connected first, the liquid crystal display may not display up whatever. modify the R5 until you get the show. The top of the line voltage at the wiper of R5 is between 0.4 to 1V.
- The IR picture transistor and the IR diode both were taken from an LTH-1550 photo interrupter module.
- The lateral surface of the picture transistor have to be masked the usage of a tape.
- The sensor arrangement is proven within the figure underneath.

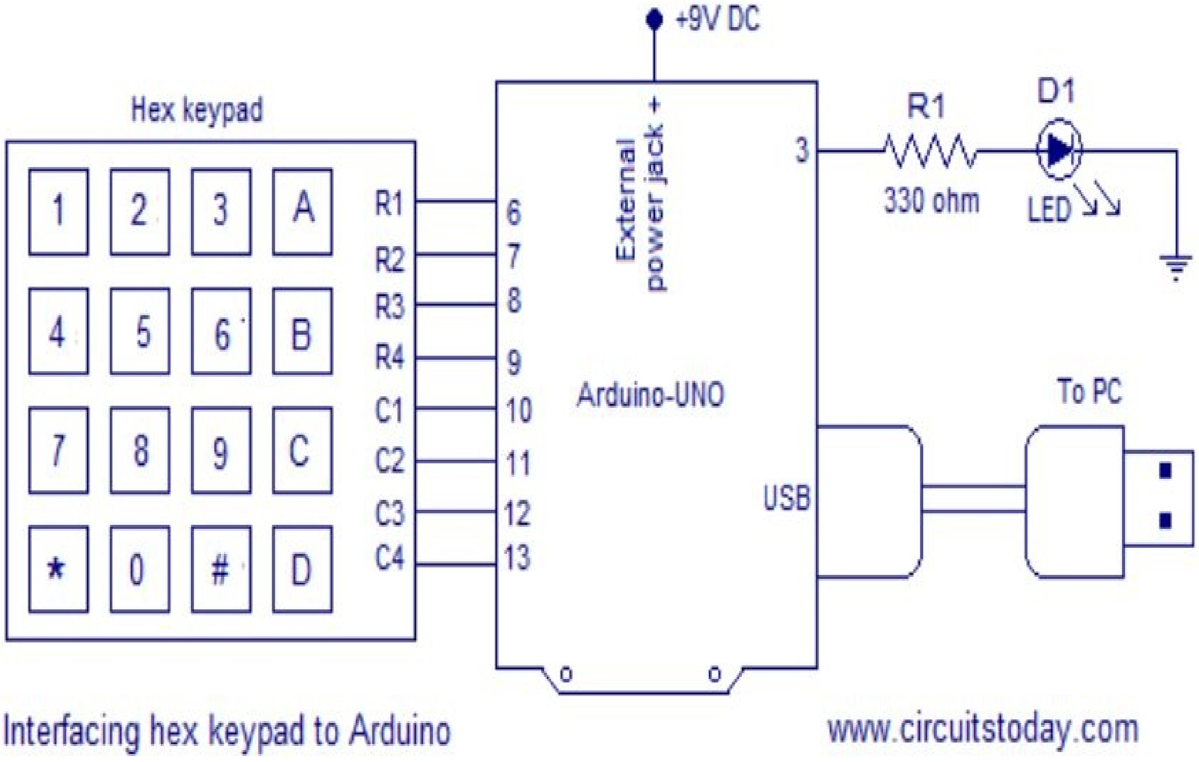


Simple Digital Code Lock.

Virtual code lock or virtual mixture lock are a kind of digital locks in which a aggregate of digits/characters or both are used for unlocking the lock. this article is about a simple digital code lock the use of arduino. right here the code includes a mixture of digits from 1 to 6. There are separate keys for locking and unlocking the device. The device can be unlocked via urgent the free up button after getting into the right aggregate of digits. A hex key pad is used because the input device. handiest the primary two rows of key (1, 2, 3, A, four, five, 6, B) are used in this assignment. A is used for locking the gadget and B is used for unlocking the gadget. read this article Interfacing hex keypad to arduino for understanding more approximately hex keypad and its interfacing to the arduino. The circuit diagram of the digital code lock using arduino is proven inside the discern underneath.

We've advanced an advanced digital Code Lock using Arduino – that's a far advanced model of this mission. The superior model comes with an option to enter user defined Password at installation. The Lock is interfaced to liquid crystal display module to output repute. The password input at set up may be modified any time later on a single key press.

Circuit diagram.



Row pins R1 to r4 are interfaced to digital pins 6 to 9 of the arduino. Column pins C1 to C4 are interfaced to virtual pins 10 to thirteen of the arduino. digital pin three of the arduino is configured as the output pin for handing over the manage sign for the solenoid lock. this system and circuit is designed on the belief that the solenoid lock will "lock" for a low sign at its input and "unlock" for a excessive signal at its input. in this circuit the solenoid is represented by the LED D1. LED ON way "unlocked" and LED OFF method "locked". 330 ohm resistor R1 limits the modern-day thru the LED.

Program.

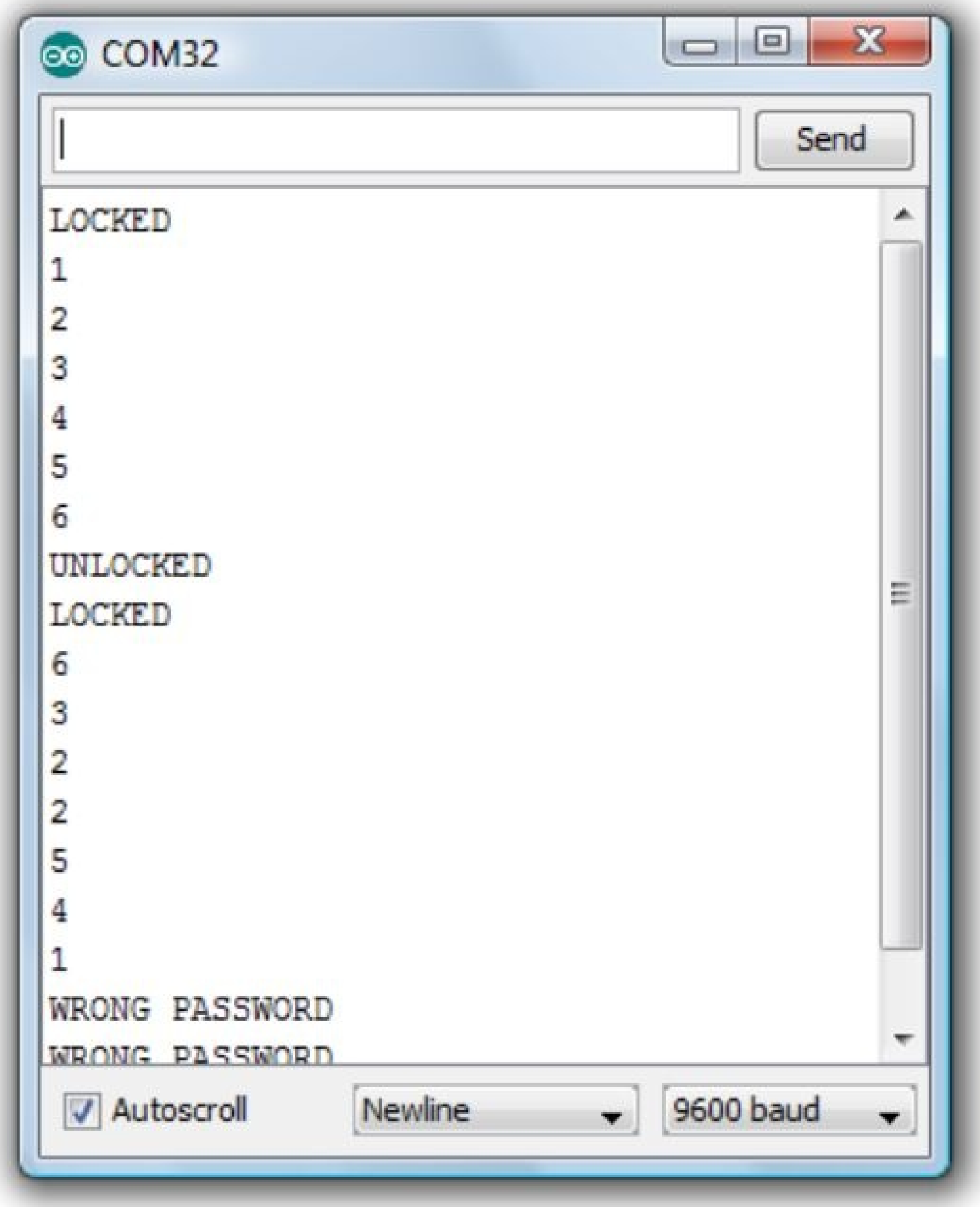
```
int p[6]; //array for storing the password
int c[6]; // array for storing the input code
int n;
int a=0;
int i=0;
int lock=3;
int r1=6;
int r2=7;
int r3=8;
int r4=9;
int c1=10;
int c2=11;
int c3=12;
int c4=13;
int colm1;
int colm2;
int colm3;
int colm4;

void setup()
{
pinMode(r1,OUTPUT);
pinMode(r2,OUTPUT);
pinMode(r3,OUTPUT);
pinMode(r4,OUTPUT);
pinMode(c1,INPUT);
pinMode(c2,INPUT);
pinMode(c3,INPUT);
pinMode(c4,INPUT);
pinMode(lock,OUTPUT);
Serial.begin(9600); //sets the baud rate at 9600
digitalWrite(c1,HIGH);
digitalWrite(c2,HIGH);
digitalWrite(c3,HIGH);
digitalWrite(c4,HIGH);
digitalWrite(lock,LOW);
p[0]=1; //sets 1st digit of the password
p[1]=2; // sets 2nd digit of the password
p[2]=3; // sets 3rd digit of the password
p[3]=4; // sets 4th digit of the password
```

```
p[4]=5; // sets 5th digit of the password
p[5]=6; // sets 6th digit of the password
}
void loop()
{
digitalWrite(r1,LOW);
digitalWrite(r2,HIGH);
digitalWrite(r3,HIGH);
digitalWrite(r4,HIGH);
colm1=digitalRead(c1);
colm2=digitalRead(c2);
colm3=digitalRead(c3);
colm4=digitalRead(c4);
if(colm1==LOW)
{ n=1;
    a=1;
    Serial.println("1");
    delay(200);}
else
{
    if(colm2==LOW)
    { n=2;
    a=1;
    Serial.println("2");
    delay(200);}
    else
    {
    if(colm3==LOW)
    {Serial.println("3");
    n=3;
    a=1;
    delay(200);}
    else
    {
    if(colm4==LOW)
    {Serial.println("LOCKED");
    digitalWrite(lock,LOW); //locks
    i=0;
    delay(200);}
    }}}

digitalWrite(r1,HIGH);
digitalWrite(r2,LOW);
digitalWrite(r3,HIGH);
digitalWrite(r4,HIGH);
colm1=digitalRead(c1);
colm2=digitalRead(c2);
colm3=digitalRead(c3);
colm4=digitalRead(c4);
if(colm1==LOW)
{Serial.println("4");
    n=4;
    a=1;
delay(200);}
else
{
    if(colm2==LOW)
    {Serial.println("5");
    n=5;
    a=1;
delay(200);}
    else
    {
    if(colm3==LOW)
    {Serial.println("6");
    n=6;
    a=1;
    delay(200);}
    else
    {
    if(colm4==LOW)
    {
    if(c[0]==p[0]&&c[1]==p[1]&&c[2]==p[2]&&c[3]==p[3]&&c[4]==p[4]&&c[5]==p[5])
{digitalWrite(lock,HIGH); //unlocks
    Serial.println("UNLOCKED");
    c[5]=9;} //corrupts the code in array c
    else
    {Serial.println("WRONG PASSWORD");}
    delay(200);}
    }}}
    if(a==1) // test whether a digit key is pressed
    {
    c[i]=n; // saves the current digit pressed to array c
    i=i+1;
    a=0;}
    }
```

About the program.

The password that is “123456” is stored within the array “p”. while ever the digit keys are pressed, they are saved inside the array “c”. while ever the unencumber button is pressed, the contents in the both array are as compared and if they may be equal then virtual pin three is made excessive. After this the content material of array “c” corrupted by means of this system. that is finished to prevent the proper code from ultimate within the reminiscence. If it is not achieved the system will liberate just on the clicking of the release button(B) after another lock cycle. urgent the lock button(A) will make the virtual pin low. The lock button has to be pressed earlier than you input the password on every occasion.

The device may be linked to the pc via the USB and the pressed keys may be viewed through the serial display window of the arduino. The display screen shot of the serial monitor window of this mission is shown in the figure below.



```
LOCKED
1
2
3
4
5
6
UNLOCKED
LOCKED
6
3
2
2
5
4
1
WRONG PASSWORD
WRONG PASSWORD
```

Whilst lock button (key A within the hex keypad) is pressed the serial monitor window will display “LOCKED”. The code entered may be additionally displayed on the window. while unlock button (key b inside the hex keypad) is pressed the serial screen window will show “UNLOCKED”. If the code entered is incorrect the serial screen window will show”wrong PASSWORD”.

Notes.

- For the prevailing configuration ie; no solenoid, there may be no need for the 9V outside deliver. The board may be powered by way of the pc through the USB.

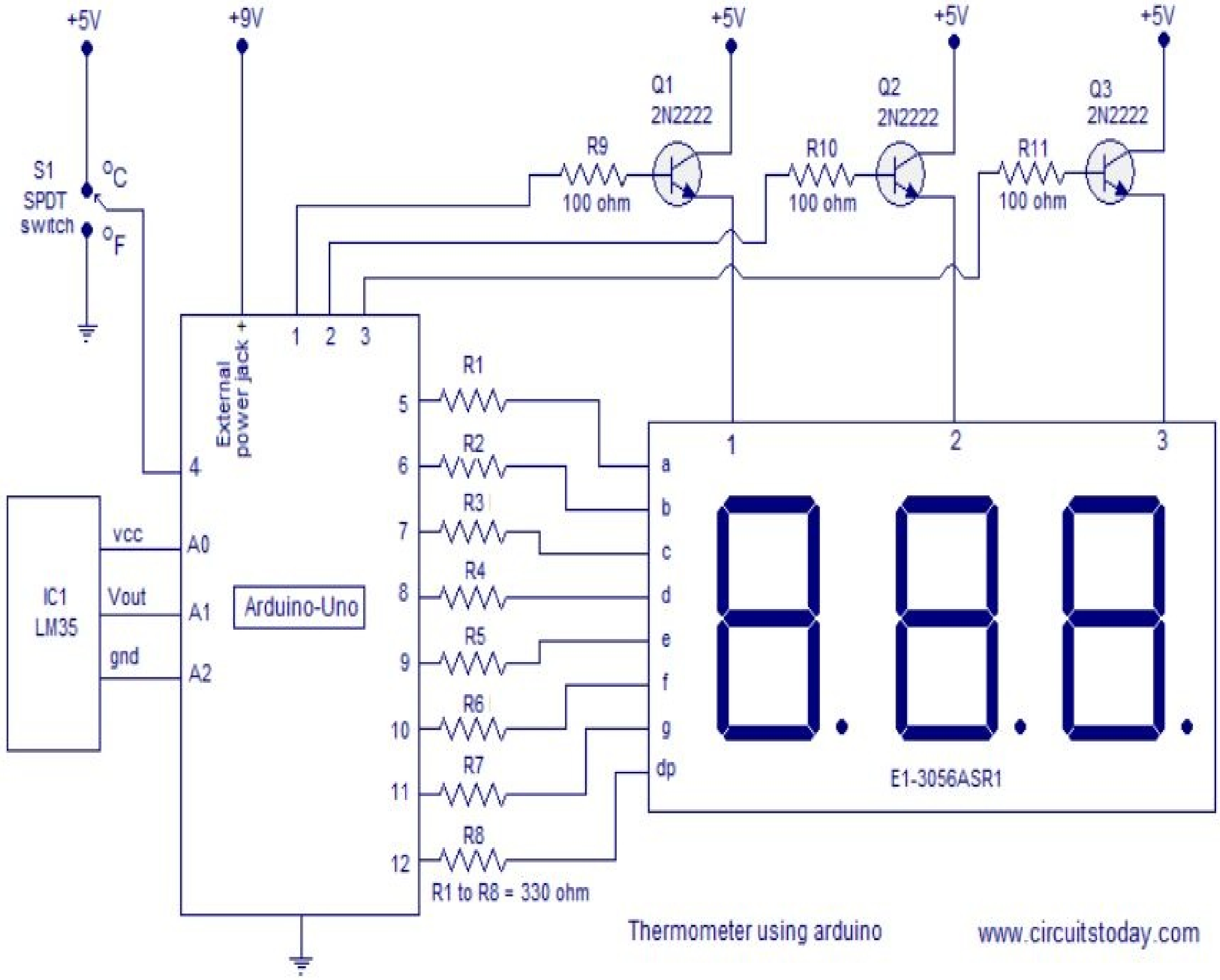
- The solenoid will eat a very good amount of cutting-edge and the pc’s USB port can be unable deliver it. So while you are using a solenoid a separate outside deliver for powering it is required. The arduino board can be also powered from this external supply if it's miles 9V. test this hyperlink Arduino UNO for expertise greater approximately powering the arduino uno board.
- The variety of digits inside the password can be improved by modifying this system.
- i have not proven the solenoid due to the fact I do not have one right now. i will upload the up to date circuit diagram and software as soon as i get one.

Digital Thermometer Using LM35.

This article is ready a easy 3 digit digital thermometer the usage of arduino. variety of this thermometer is from 0°C to 99.9°C. there may be also a provision for showing the temperature in °F scale. three terminal analog temperature sensor LM35 is used because the sensor right here. LM35 can measure temperatures between -fifty five°C to +one hundred fifty five°C. The deliver voltage variety is from 4V to 30V DC and the cutting-edge drain is 60uA. The LM35 is available in TO-92 bundle and it's far very smooth to apply.

The output voltage of the arduino will increase 10mV in step with °C rise in temperature. that means if 25 °C is the temperature, then output voltage of the sensor may be 250mV. Circuit diagram of the digital thermometer the usage of arduino and LM35 is proven within the parent underneath.

Circuit diagram.



Temperature sensor LM35 is attached to the arduino thru the analog input pins.

A0 pin of the arduino serves because the deliver voltage supply for LM35 and A2 pin of the arduino serves as the ground. Arduino reads the voltage output of the LM35 thru the analog input pin A1. virtual pin 4 is used for interfacing the °C/°F selector switch to the arduino. Digit driving force transistors Q1, Q2 and Q3 are interfaced to the digital pins 1, 2 and three of the arduino respectively. Multiplexed phase strains a to dp are interfaced to the virtual pins five to twelve of the arduino respectively. Resistors R9, R10 and R11 limits the base present day of the corresponding transistors. Resistors R1 to R8 limits the contemporary through the corresponding segments.

Program.

```
int i;
int backup;
int a;
int unit;
int value;
int vcc=A0;
int sensor=A1;
int gnd=A2;
int select=4;
int disp1=1;
int disp2=2;
int disp3=3;
int segA=5;
int segB=6;
int segC=7;
int segD=8;
int segE=9;
int segF=10;
int segG=11;
int segDP=12;
void setup()
{
pinMode(disp1, OUTPUT);
pinMode(disp2, OUTPUT);
pinMode(disp3, OUTPUT);
pinMode(segA, OUTPUT);
pinMode(segB, OUTPUT);
pinMode(segC, OUTPUT);
pinMode(segD, OUTPUT);
pinMode(segE, OUTPUT);
pinMode(segF, OUTPUT);
pinMode(segG, OUTPUT);
pinMode(segDP, OUTPUT);
pinMode(sensor, INPUT);
pinMode(vcc, OUTPUT);
pinMode(gnd, OUTPUT);
pinMode(select, INPUT);
digitalWrite(vcc, HIGH);
digitalWrite(gnd, LOW);

}
void loop()
{
value=analogRead(sensor); //Reads the sensor LM35
value=value*5;
unit=digitalRead(select); //Reads the selector switch
if(unit==1)
{value=value;} // Output in celcius
else
{value=(value*1.8)+320;} //Output in Fahrenheit
backup=value; //Backs up the content in variable value

for(i=0;i<100;i++) //Loops the display loop 100 times(steadies the display)
{value=backup;
a=value%10;
digitalWrite(disp1,LOW);
digitalWrite(disp2,LOW);
digitalWrite(disp3, HIGH);
digitalWrite(segDP,HIGH);
display(a);
delay(4);
value = value/10;
a = value%10;
digitalWrite(disp3,LOW);
digitalWrite(disp2,HIGH);
digitalWrite(segDP,LOW);
display(a);
delay(4);
value=value/10;
a=value;
digitalWrite(disp2,LOW);
digitalWrite(disp1,HIGH);
digitalWrite(segDP,HIGH);
display(a);
delay(4);
}
}
int display (int a)
{
switch (a)
{
    case 0:
    digitalWrite(segA, LOW);
    digitalWrite(segB, LOW);
    digitalWrite(segC, LOW);
    digitalWrite(segD, LOW);
    digitalWrite(segE, LOW);
    digitalWrite(segF, LOW);
    digitalWrite(segG, HIGH);
    break;

    case 1:
    digitalWrite(segA, HIGH);
    digitalWrite(segB, LOW);
    digitalWrite(segC, LOW);
    digitalWrite(segD, HIGH);
    digitalWrite(segE, HIGH);
    digitalWrite(segF, HIGH);
    digitalWrite(segG, HIGH);
    break;

    case 2:
    digitalWrite(segA, LOW);
    digitalWrite(segB, LOW);
    digitalWrite(segC, HIGH);
    digitalWrite(segD, LOW);
    digitalWrite(segE, LOW);
    digitalWrite(segF, HIGH);
    digitalWrite(segG, LOW);
    break;

    case 3:
    digitalWrite(segA, LOW);
    digitalWrite(segB, LOW);
    digitalWrite(segC, LOW);
    digitalWrite(segD, LOW);
    digitalWrite(segE, HIGH);
    digitalWrite(segF, HIGH);
    digitalWrite(segG, LOW);
    break;

    case 4:
    digitalWrite(segA, HIGH);
    digitalWrite(segB, LOW);
    digitalWrite(segC, LOW);
    digitalWrite(segD, HIGH);
    digitalWrite(segE, HIGH);
    digitalWrite(segF, LOW);
    digitalWrite(segG, LOW);
    break;

    case 5:
    digitalWrite(segA, LOW);
    digitalWrite(segB, HIGH);
    digitalWrite(segC, LOW);
    digitalWrite(segD, LOW);
    digitalWrite(segE, HIGH);
    digitalWrite(segF, LOW);
```

```
    digitalWrite(segG, LOW);
    break;

    case 6:
    digitalWrite(segA, LOW);
    digitalWrite(segB, HIGH);
    digitalWrite(segC, LOW);
    digitalWrite(segD, LOW);
    digitalWrite(segE, LOW);
    digitalWrite(segF, LOW);
    digitalWrite(segG, LOW);
    break;

    case 7:
    digitalWrite(segA, LOW);
    digitalWrite(segB, LOW);
    digitalWrite(segC, LOW);
    digitalWrite(segD, HIGH);
    digitalWrite(segE, HIGH);
    digitalWrite(segF, HIGH);
    digitalWrite(segG, HIGH);
    break;

    case 8:
    digitalWrite(segA, LOW);
    digitalWrite(segB, LOW);
    digitalWrite(segC, LOW);
    digitalWrite(segD, LOW);
    digitalWrite(segE, LOW);
    digitalWrite(segF, LOW);
    digitalWrite(segG, LOW);
    break;

    case 9:
    digitalWrite(segA, LOW);
    digitalWrite(segB, LOW);
    digitalWrite(segC, LOW);
    digitalWrite(segD, LOW);
    digitalWrite(segE, HIGH);
    digitalWrite(segF, LOW);
    digitalWrite(segG, LOW);
    break;
}}
```

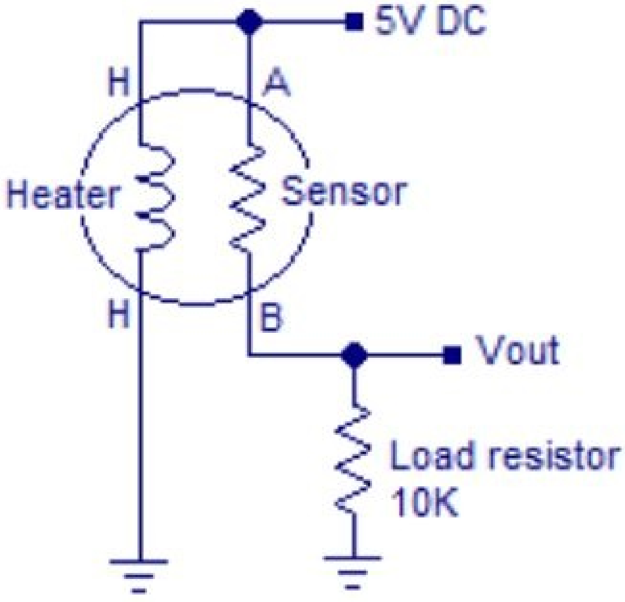
About the program.

The voltage output of the LM35 is scanned using the analogRead feature. The analogRead function will study the voltage at a specific analog enter pin and converts it right into a virtual price among zero and 1023. If 30°C is the temperature, the LM35 output will be 300mV. The code fee=digitalRead(sensor) will examine this voltage, convert it right into a digital fee and stores it within the variable “cost”. So, for 30°C the quantity saved in variable “value” could be 3oomV/(5/1023)=sixty one. This number is extended by using 5 to get 305 a decimal point is region before the final digit at the same time as showing on the seven segment display.The end result will be 30.5°C.

this system additionally exams the popularity of the unit selector switch through studying the digital pin four. If this pin is held excessive the temperature is displayed in °Celsius and if this pin is low the temperature in °Celsius scale is transformed to °Fahrenheit and then displayed. The technique used for showing the variety on the 3 digit seven segment display is identical as that of used in Voltmeter the use of arduino.

LPG Sensor With Alarm And Cutoff.

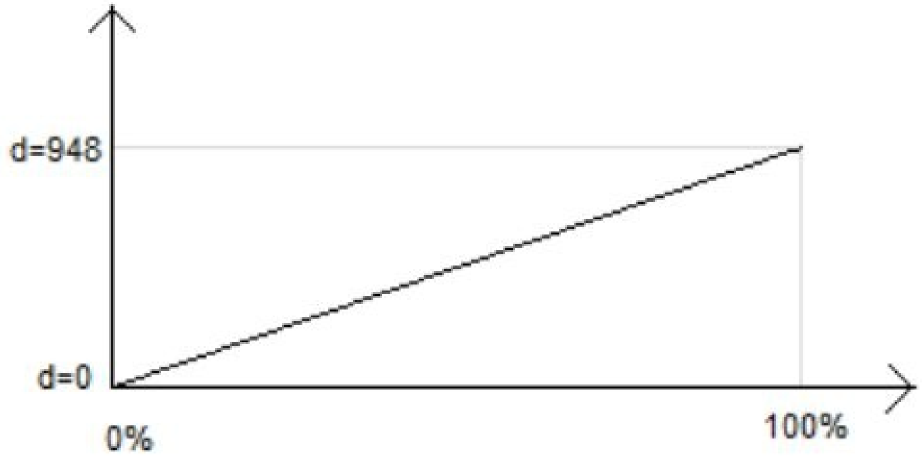
A easy LPG sensor using arduino is proven in this article. This circuit shows the amount of LPG in the air. The circuit sounds an alarm and journeys a relay whilst the attention is above a predetermined stage. MQ2 is the gas sensor used in this mission. MQ2 is an SnO2 based gasoline sensor which could feel gases like methane, propane, butane, alcohol, smoke, hydrogen and so forth. because LPG often includes propane and butane, MQ2 sensor may be used for sensing LPG. The discern below indicates the schematic and association of an MQ2 gas sensor.



MQ2 gas sensor connection diagram

MQ2 sensor senses the flammable gases with the aid of the growth in temperature whilst they are oxidized by means of the heating element. consider the figure given above. If there is any flammable gasoline gift within the pattern , the oxidization of the identical gasoline outcomes in elevated temperature and the resistance of the sensor resistor will drop. which means more modern will go with the flow via the load resistor and so the voltage across it will shoot up.

At regular situations(no LPG within the air), the sensor resistor can be very excessive round 850K . So the voltage drop Vout throughout the burden resistor will be round 0. whilst the sensor is absolutely exposed to LPG the sensor resistance drops to round 800 ohms and the voltage drop across the burden resistance will be round four.62 volts. After conversion by the ADC, the virtual equivalent of four.sixty two volt could be 948 and it's far saved within the variable “d” (refer this system). determine below suggests a graph plotted from the found parameters.

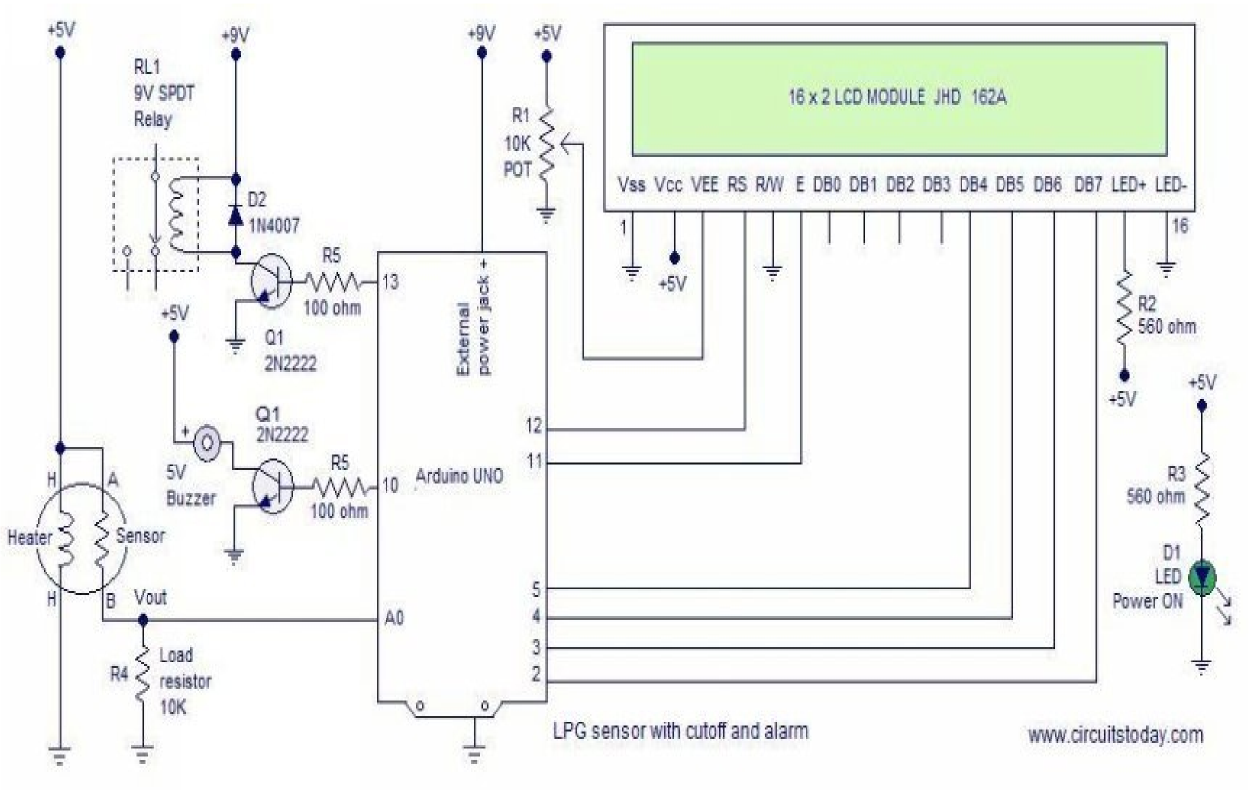


LPG gas sensor using arduino (MQ2 sensor)

Clearly the graph won't be a instantly line. however right here we must count on it to be a instantly line because it is not possible to simulate LPG concentrations apart from zero% and 100% with our constrained lab centers. For correct calibration of the sensor, we need some means to recognise the best attention of the fuel inside the given environment. anyway what we've got is quite sufficient for our cause.

The awareness percent for a given digital output of the ADC may be decided using the following equation. p=d/9.forty eight wherein d is the digital output of the ADC and p is the share. The equation is acquired by way of finding the equation of the above graph inside the widespread form y=mx+c. wherein m is the slope and c is the y intercept. the overall circuit diagram of the LPG sensor the usage of arduino is proven below.

Circuit diagram.



LPG sensor with cutoff and alarm

www.circuitstoday.com

Output of the fuel sensor is connected to the analog input pin A0 of the arduino. digital pin 10 of the arduino is used for controlling the buzzer and virtual pin thirteen used for controlling the relay. The relay used here's a SPDT relay and so it is able to be used for switching ON or OFF the goal device whilst there is a fuel leak. right here the trip threshold is ready to be 30%. you may set your threshold point inside the program.

Whilst deciding on the threshold point, temperature and humidity effects must be additionally taken into consideration because MQ2 sensor has accurate dependence on both. Refer the datasheet of MQ2 fuel sensor for more concept. The whole program of the LPG sensor the use of arduino is shown under.

Program.

```
#include<LiquidCrystal.h>
int mq2=A0;
int rel=13;
int buz=10;
int d;
float p;

LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

void setup()
{
pinMode(rel,OUTPUT);
pinMode(buz,OUTPUT);
digitalWrite(rel,LOW);
digitalWrite(buz,LOW);
lcd.begin(16,2);
}
void loop()
{
d=analogRead(mq2);
lcd.setCursor(0,0);
lcd.print("LPG SENSOR");

if(d<60)
{
p=0;
}
else
{
p=(d-60)/9.64;
}
lcd.setCursor(0,1);
lcd.print(p);
lcd.setCursor(5,1);
lcd.print("%");
if(p>=30)
{
digitalWrite(rel,LOW);
digitalWrite(buz,HIGH);
lcd.setCursor(9,1);
lcd.print("TRIP");
}
else
{
digitalWrite(rel,HIGH);
digitalWrite(buz,LOW);
}
delay(500);
lcd.clear();
}
```

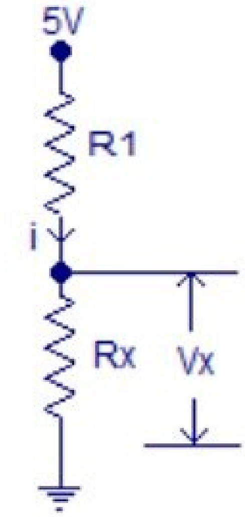
Notes.

- MQ2 sensor calls for a 24 hour preheat for strong operation.
- The heating coil of the MQ2consumes around150mA and so it's far smart to electricity the coil from a separate supply.
- 5V required at different components of the circuit can be tapped from the arduino board.
- The arduino board can be powered via the 9V electricity supply jack.

Auto Ranging Ohmmeter.

This text is set a easy vehicle ranging ohmmeter the usage of arduino. The measured resistance is displayed the use of a sixteen×2 lcd display. The circuit is satisfactorily correct and makes use of minimum wide variety of external additives feasible. before going into the information of this challenge, we could have a have a look at the fundamental resistance measurement method.

Resistance measurement.



The discern above indicates the circuit diagram of a easy resistance dimension scheme. Rx is the resistance to be measured. R1 is the enter resistance. i is the modern passing via the loop and 5V is the supply voltage. To find the unknown resistance Rx, the voltage across Rx is measured first. let the voltage across R1 be VR1. Then VR1=5-Vx. The modern i=VR1/R1=(five-Vx)/R1. considering that R1 and Rx are connected in series, the current via them may be same. So the unknown resistance Rx= Vx/i. The voltage across the unknown resistance is measured the use of the ADC of the arduino. To be precise, analog channel A5.

Besides this method have a downside. If there is extremely good difference between the input resistance and the Rx, the result can be extraordinarily faulty. that is due to the fact almost all the enter voltage will drop throughout the larger resistance and this presents very much less facts.

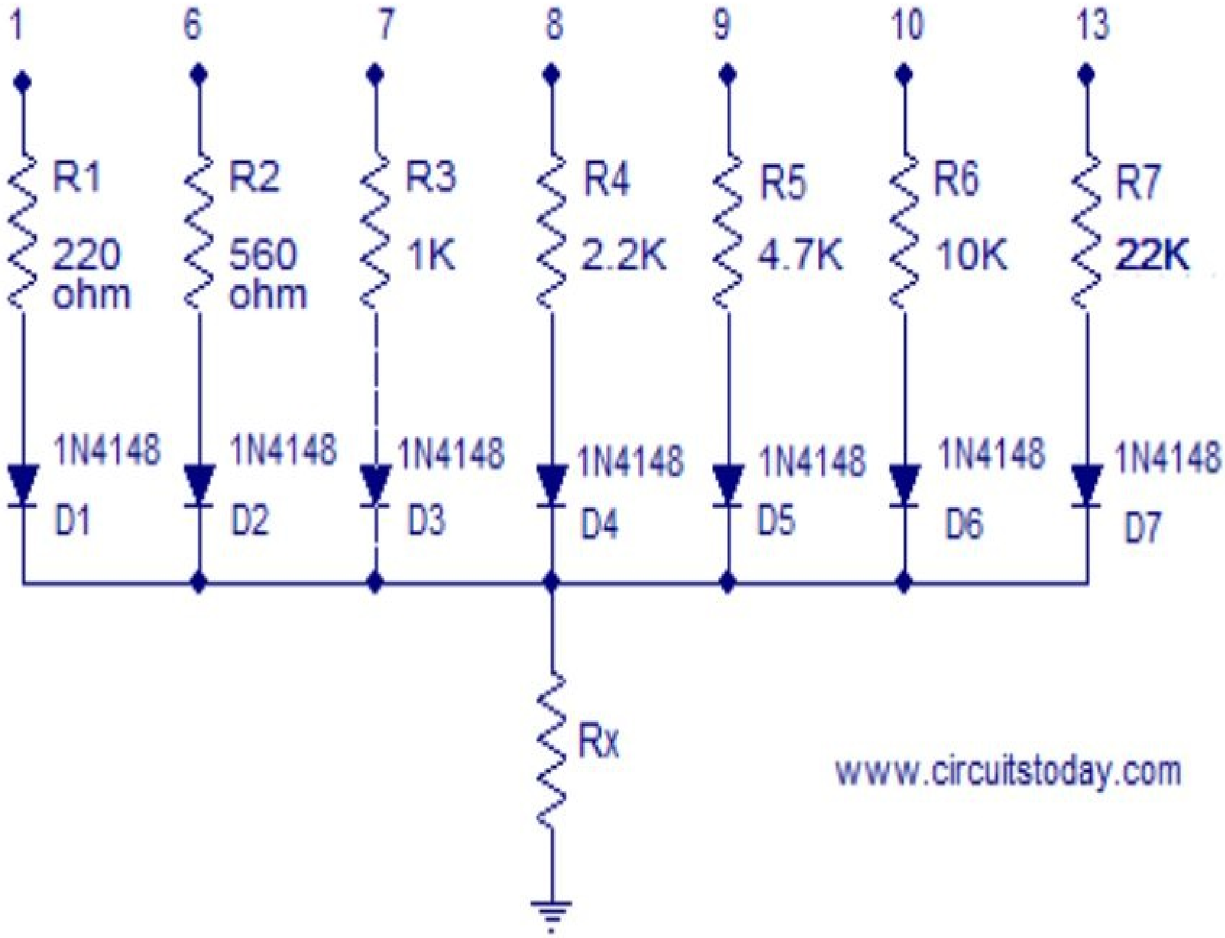
- Suppose R1=10K and Rx=a hundred ohm. Then the voltage throughout R1 can be four.95v and voltage throughout Rx might be 50mV and this offers less records. The sensitivity of the arduino is 4.889mV. So while we study 50mV using the arduino ADC the end result can be 10. when converted it into voltage the result will be 10 x four.889mV =forty eight.89mV. Then Rx= zero.0488/((5V-forty eight.89mV)/ten thousand) = ninety eight.7 ohm.

- Think R1=10 and Rx=220 ohm. Then the voltage throughout R1 may be 4.89V and voltage throughout Rx might be 107mV. The corresponding virtual analyzing will be 21. when we convert it into voltage the end result will be 21 x 4.889mV=102mv. Following the calculations used within the previous case, Rx=208 ohm.

In the above instances you could see accuracy issues. The maximum correct end result occurs when the Rx and R1 are as near as viable.

Auto ranging.

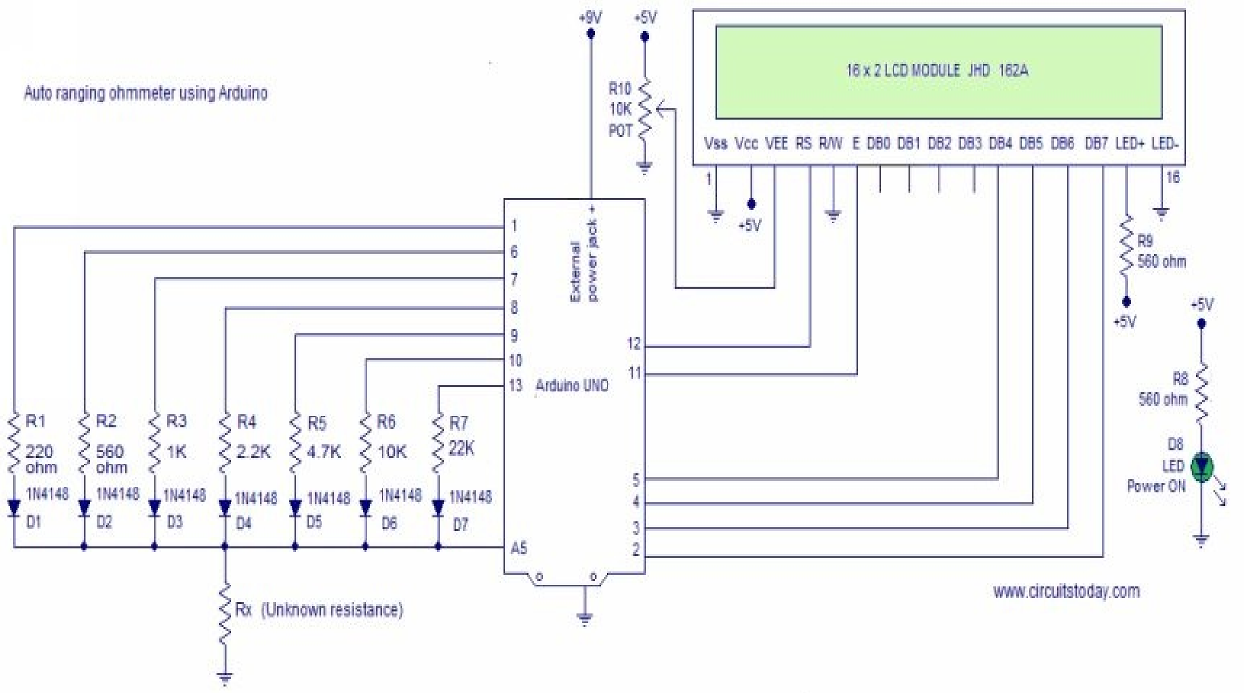
A scheme for estimating the cost of Rx roughly after which putting a matching resistor in vicinity of R1 is what we need here and this method is known as auto ranging. The circuit given underneath demonstrates automobile ranging.



Resistances R1 to R7 are the input resistors. on this scheme the free cease of one resistor is held high and the loose ends of other resistors are held low. The the voltage across the unknown resistance Rx is measured. Diodes D1 to D7 are used to save you the back flow of modern toward the low ends. suppose free quit of R1 is held low. If R1 and Rx are same, then the voltage drop across Rx will be (five-zero.7)/2 = 2.15 wherein zero.7 is the diode drop.

If the voltage throughout Rx is much less than or identical to two.15, we are able to assume that Rx is much less than or equal to 220 ohms. The closest price feasible for the enter resistance is 220 ohms and so this loop is considered for calculation. If the above condition is not satisfied, the above steps are repeated with the succeeding input resistors till we get a solution.

Circuit diagram.



Complete circuit diagram of the car ranging ohmmeter using arduino is proven in the figure above. digital pins 1, 6, 7, 8, nine, 10, thirteen of the arduino are used to switch the input resistors R1, R2, R3, R4, R5, R6, R7 respectively. Resistors D1 to D7 are used to save you the back float of current via the corresponding path. D8 is the energy ON indicator LED. POT R10 is used for comparison adjustment of the lcd. Resistor R9 limits the back light LED modern.

Program.

```
#include<LiquidCrystal.h>
int vin=A5;


int t=1;
int u=6;
int v=7;
int w=8;
int x=9;
int y=10;
int z=13;

int at;
int au;
int av;
int aw;
int ax;
int ay;
int az;
int a;
double vx;
float rx;
double i;
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);
void setup()
{
pinMode(vin,INPUT);
lcd.begin(16,2);

pinMode(t,OUTPUT);
pinMode(u,OUTPUT);
pinMode(v,OUTPUT);
pinMode(w,OUTPUT);
pinMode(x,OUTPUT);
pinMode(y,OUTPUT);
pinMode(z,OUTPUT);

digitalWrite(t,LOW);
digitalWrite(u,LOW);
digitalWrite(v,LOW);
digitalWrite(w,LOW);
digitalWrite(x,LOW);
digitalWrite(y,LOW);
digitalWrite(z,LOW);
}
void loop()
{

digitalWrite(t,HIGH);
digitalWrite(u,LOW);
digitalWrite(v,LOW);
digitalWrite(w,LOW);
digitalWrite(x,LOW);
digitalWrite(y,LOW);
digitalWrite(z,LOW);
delay(100);
at=analogRead(vin);
```

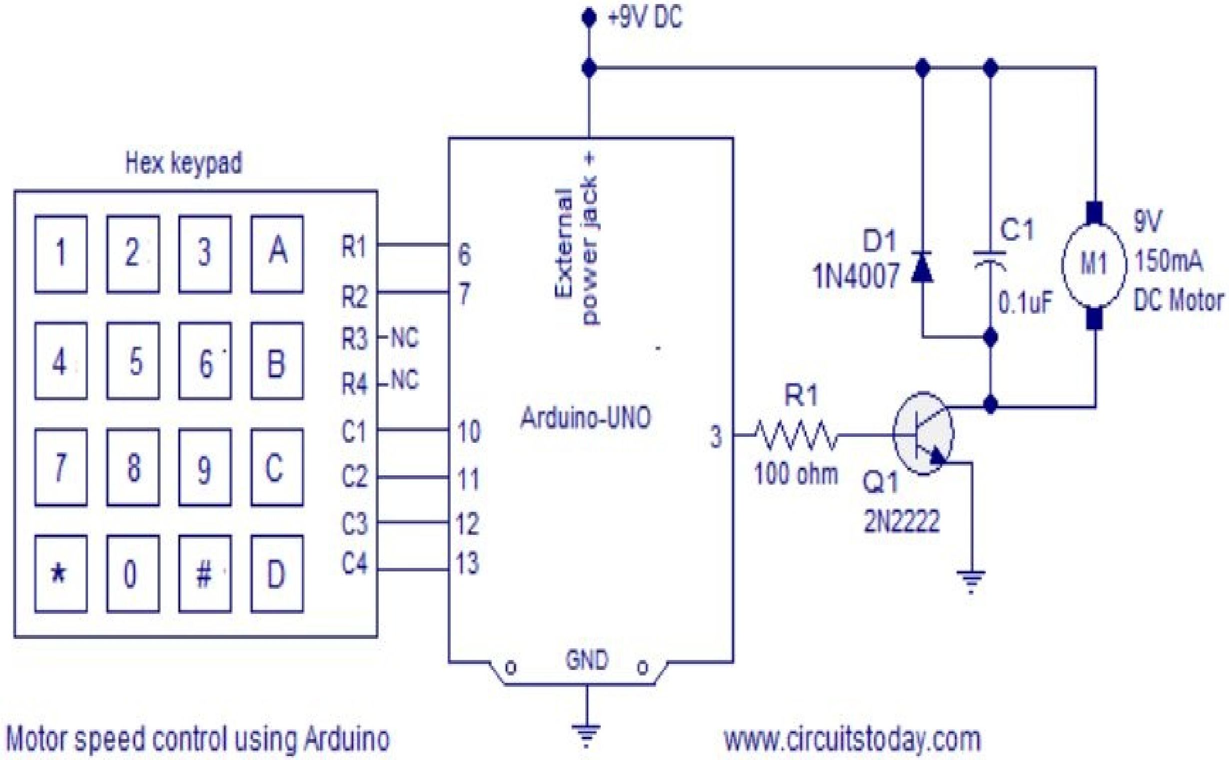
```
digitalWrite(t,LOW);
digitalWrite(u,HIGH);
digitalWrite(v,LOW);
digitalWrite(w,LOW);
digitalWrite(x,LOW);
digitalWrite(y,LOW);
digitalWrite(z,LOW);
delay(100);
au=analogRead(vin);
digitalWrite(t,LOW);
digitalWrite(u,LOW);
digitalWrite(v,HIGH);
digitalWrite(w,LOW);
digitalWrite(x,LOW);
digitalWrite(y,LOW);
digitalWrite(z,LOW);
delay(100);
av=analogRead(vin);



digitalWrite(t,LOW);
digitalWrite(u,LOW);
digitalWrite(v,LOW);
digitalWrite(w,HIGH);
digitalWrite(x,LOW);
digitalWrite(y,LOW);
digitalWrite(z,LOW);
delay(100);
aw=analogRead(vin);


digitalWrite(t,LOW);
digitalWrite(u,LOW);
digitalWrite(v,LOW);
digitalWrite(w,LOW);
digitalWrite(x,HIGH);
digitalWrite(y,LOW);
digitalWrite(z,LOW);
delay(100);
ax=analogRead(vin);


digitalWrite(t,LOW);
digitalWrite(u,LOW);
digitalWrite(v,LOW);
digitalWrite(w,LOW);
digitalWrite(x,LOW);
digitalWrite(y,HIGH);
digitalWrite(z,LOW);
delay(100);
ay=analogRead(vin);




digitalWrite(t,LOW);
digitalWrite(u,LOW);
digitalWrite(v,LOW);
digitalWrite(w,LOW);
digitalWrite(x,LOW);
digitalWrite(y,LOW);
digitalWrite(z,HIGH);
delay(100);
az=analogRead(vin);

if(az>=450)
{
vx=az*0.00489;
i=(5-vx-0.55)/22000;
rx=(vx/i);
}
if(ay>=450 && az<450)
{
vx=ay*0.00489;
i=(5-vx-0.55)/10000;
rx=(vx/i);
}
if(ax>=448 && ay<448 && az<448)
{
vx=ax*0.00489;
i=(5-vx-0.55)/4700;
rx=(vx/i);
}

if(aw>=439 && ax<439 && ay<439 && az<439)
{
vx=aw*0.00489;
i=(5-vx-0.55)/2200;
rx=(vx/i);
}

if(av>=439 && aw<439 && ax<439 && ay<439 && az<439) {
vx=av*0.00489;
i=(4.8-vx-0.55)/1000;
rx=(vx/i);
}

if(au>=430 && av<430 && aw<430 && ax<430 && ay<430 && az<430) {
vx=au*0.00489;
i=(4.5-vx-0.55)/560;
rx=(vx/i);
}

if(at>=430 && au<430 && av<430 && aw<430 && ax<430 && ay<430 && az<430 ) {
vx=at*0.00489;
i=(4.5-vx-0.55)/220;
rx=(vx/i);
}

if(at<430 && au<430 && av<430 && aw<430 && ax<430 && ay<430 && az<430 ) {
vx=at*0.00489;
i=(4.5-vx-0.55)/220;
rx=(vx/i);
}
lcd.setCursor(0,0);

if(vx>4.8)
{
lcd.clear();
lcd.setCursor(0,0);
lcd.print("----INFINITY----");
}
else
{
if(rx<1000)
{
lcd.clear();
lcd.setCursor(0,0);
lcd.print(rx);
lcd.setCursor(7,0);
lcd.print((char)244);
}
else
{
lcd.clear();
rx=rx/1000;
lcd.setCursor(0,0);
lcd.print(rx);
lcd.setCursor(6,0);
lcd.print("k");
lcd.print((char)244);
}
}
lcd.setCursor(0,1);
lcd.print("Arduino Ohmmeter");
```

```
}
```

PWM Motor Speed Control.

PWM or pulse width modulation is a very common method used for controlling the energy across gadgets like motor, light etc. In PWM method the power throughout the weight is controlled by means of various the obligation cycle of the force sign. more the obligation cycle more strength is introduced throughout the load and less the duty cycle, less power is delivered throughout the burden. A hex keypad is used for controlling the velocity. the speed may be various in seven steps the usage of the hex keypad. Arduino UNO is the kind os arduino improvement board used on this circuit. The circuit diagram of the PWM motor velocity control the use of arduino is proven in the discern beneath.

Circuit diagram.



Motor speed control using Arduino

www.circuitstoday.com

Row pins R1 and R2 of the hex keypad are interfaced to virtual pins 6 and seven of the arduino. Column pins C1, C2, C3 and C4 are interfaced to the digital pind 10, 11, 12 and thirteen of the arduino. the key pressed on the hex keypad is diagnosed using the column scanning method and it's miles explained in element in this text. Interfacing hex keypad to arduino. The digital pins of the arduino can supply or sink only as much as 4omA of modern-day. So the virtual pin three can not pressure the motor at once. To remedy this hassle an NPN transistor

(2N2222) is used to pressure the motor according the the PWM signal to be had at virtual pin three. a hundred ohm resistor R1 is used to limit the base present day of the transistor. The motor is hooked up as a collector load to the transistor. The zero.1uF capacitor C1 related throughout the motor is used to by using-bypass the voltage spikes and noises produced all through the switching of the motor.

The arduino board is powered through the external power jack supplied on the board. The arduino board may be also powered by means of the pc through USB but there should be an extra external source for powering the motor. The entire application for PWM motor velocity manage using arduino is given below. clarification of the program is given underneath the "approximately the program" heading.

Program.

```
int pwm=3; // declares digital pin 3 as PWM output
int r1=6;
int r2=7;
int c1=10;
int c2=11;
int c3=12;
int c4=13;
int colm1;
int colm2;
int colm3;
int colm4;

void setup()
{
pinMode(r1,OUTPUT);
pinMode(r2,OUTPUT);
pinMode(c1,INPUT);
pinMode(c2,INPUT);
pinMode(c3,INPUT);
pinMode(c4,INPUT);
pinMode(pwm,OUTPUT);
digitalWrite(c1,HIGH);
digitalWrite(c2,HIGH);
digitalWrite(c3,HIGH);
digitalWrite(c4,HIGH);
digitalWrite(pwm,LOW);
}
void loop()
{
digitalWrite(r1,LOW);
digitalWrite(r2,HIGH);
colm1=digitalRead(c1);
colm2=digitalRead(c2);
colm3=digitalRead(c3);
colm4=digitalRead(c4);
if(colm1==LOW) //checks whether key "1" is pressed.
{ analogWrite(pwm,42); // writes "42" (duty cycle 16%).
    delay(200);}
else
{
    if(colm2==LOW) //checks whether key "2" is pressed.
    { analogWrite(pwm,84); // writes "84" (duty cycle 32%).
    delay(200);}
    else
    {
    if(colm3==LOW) //checks whether key "3" is pressed
    {analogWrite(pwm,126); // writes "126" (duty cycle 48%).
    delay(200);}
    else
    {
    if(colm4==LOW) // checks whether key"A" is pressed.
    {digitalWrite(pwm,LOW); // makes pin 3 LOW (duty cycle 0%).Motor OFF.
    delay(200);}
    }}}

digitalWrite(r1,HIGH);
digitalWrite(r2,LOW);
colm1=digitalRead(c1);
colm2=digitalRead(c2);
colm3=digitalRead(c3);
colm4=digitalRead(c4);
if(colm1==LOW) // checks whether key "4" is pressed.
{analogWrite(pwm,168); //writes "168" (duty cycle 64%).
delay(200);}
else
{
    if(colm2==LOW) // checks whether key "5" is pressed.
    {analogWrite(pwm,202); // writes "202" (duty cycle 80%).
delay(200);}
    else
    {
    if(colm3==LOW) // checks whether key "6" is pressed.
    {analogWrite(pwm,244); // writes "244" (duty cycle 96%).
    delay(200);}
    else
    {
    if(colm4==LOW) // checks whether key "B" is pressed.
    {digitalWrite(pwm,HIGH);//makes pin 3 HIGH (duty cycle 100%). FULL POWER
    delay(200); }


    }}}}
```

About the program.

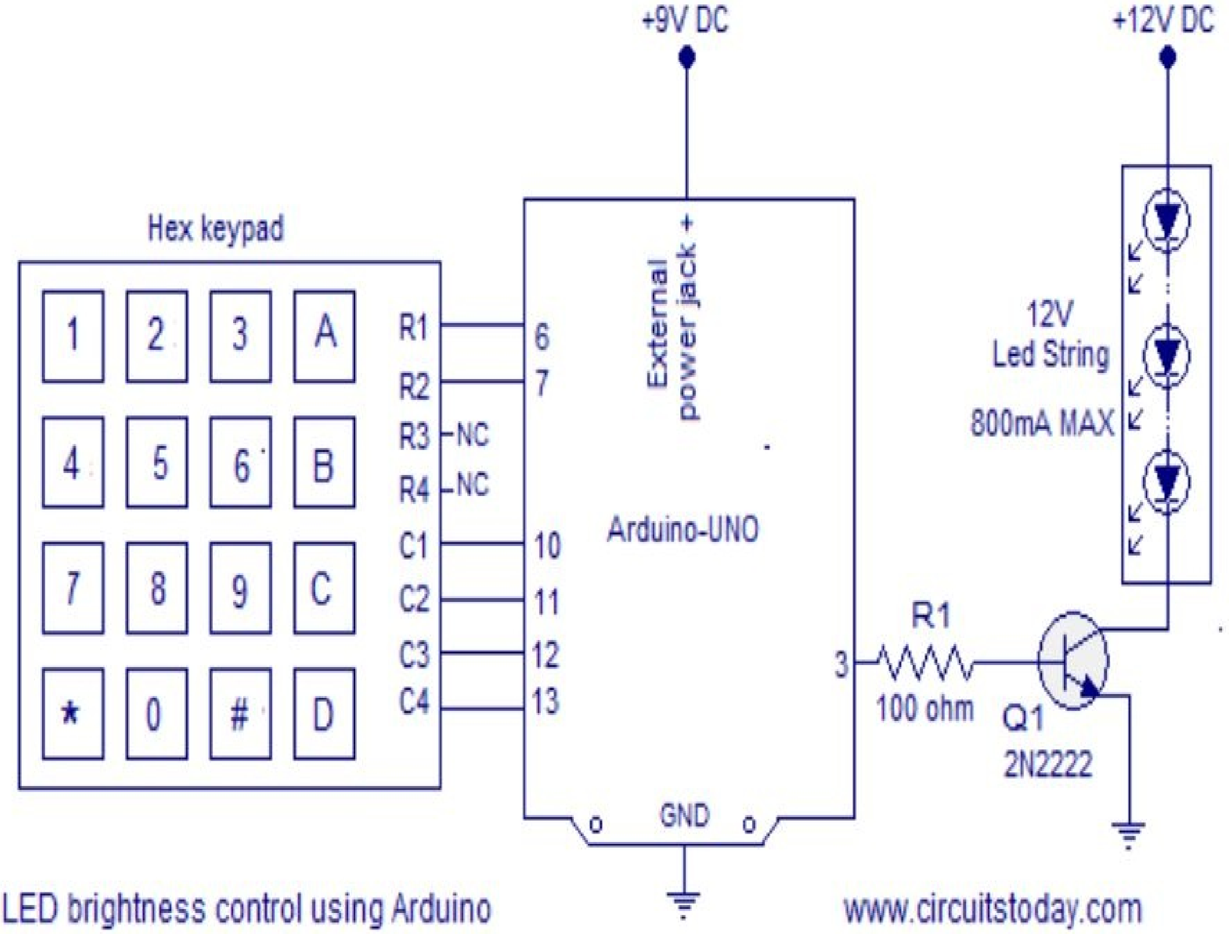
The responsibility cycle of the PWM control sign is numerous by various the value written to the output pin 3 using the analogWrite() characteristic. The range of the cost that may be written is among 0 and 255. The anlogWrite() function can be employed on pins 3, 5, 6, nine, 10 and eleven inside the Arduino UNO board. In most of the arduino boards the frequency of the PWM signal may be round 490Hz. The responsibility cycle of the PWM sign is proportional to the price written the use of the analogWrite() feature. Few examples using the analogWrite() feature are shown underneath.

- analogWrite(pwm,255) will generate a pwm wave of 100% duty cycle (complete energy) at the pin denoted via the variable “pwm”.
- analogWrite(pwm,128) will generate a pwm wave of 50% duty cycle (1/2 strength) at the pin denote via the variable “pwm”.
- analogWrite(pwm,0) will generate a pwm wave of 0% responsibility cycle (no strength) on the pin denoted via the variable “pwm”.

Inside the application the digital pin 3 is configured as the PWM output pin. Keys 1 to six at the hex keypad are used for growing the energy in steps of “forty two” in terms of the fee written the usage of the analogWrite() feature or sixteen% in phrases of duty cycle. Key “A” on the hex keypad is used for switching the motor OFF and it's far done using the command “digitalWrite(pwm,LOW);. Key “B” at the hex keypad is used for placing the motor at maximum pace an it's miles completed using the command “digitalWrite(pwm,HIGH);.

Notes.

As a substitute the motor you may additionally use the same circuit for various the brightness of an LED string. Any manner the weight modern need to be within the safe limits of transistor 2N2222 and it's far 800mA. additionally the outside power supply ought to be powerful sufficient to pressure the LED string. The circuit diagram of PWM brightness control of LED the use of arduino is shown within the discern underneath.



LED brightness control using Arduino

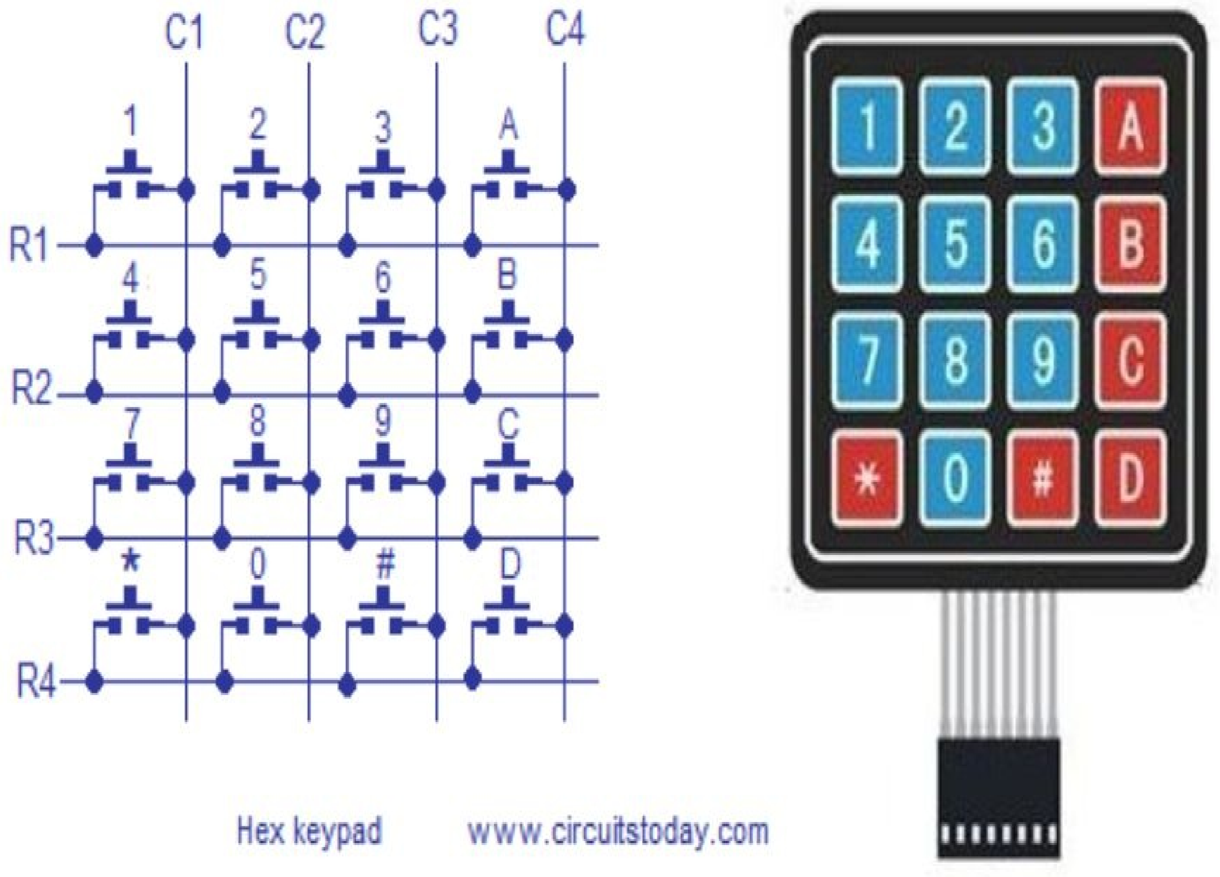
www.circuitstoday.com

Interfacing Hex Keypad With Arduino Uno.

This article is ready a way to interface a hex keypad to arduino. Hex keypad is a very vital element in embedded systems and the typical applications are code locks, calculators, automation systems or really any aspect that requires a individual or numeric input. This venture will display the pressed key inside the serial display window of the arduino IDE. The equal challenge can be modified to display the pressed key on 7-segment LED show or an liquid crystal display show. before going into the info, have a observe the hex keypad.

Hex keypad.

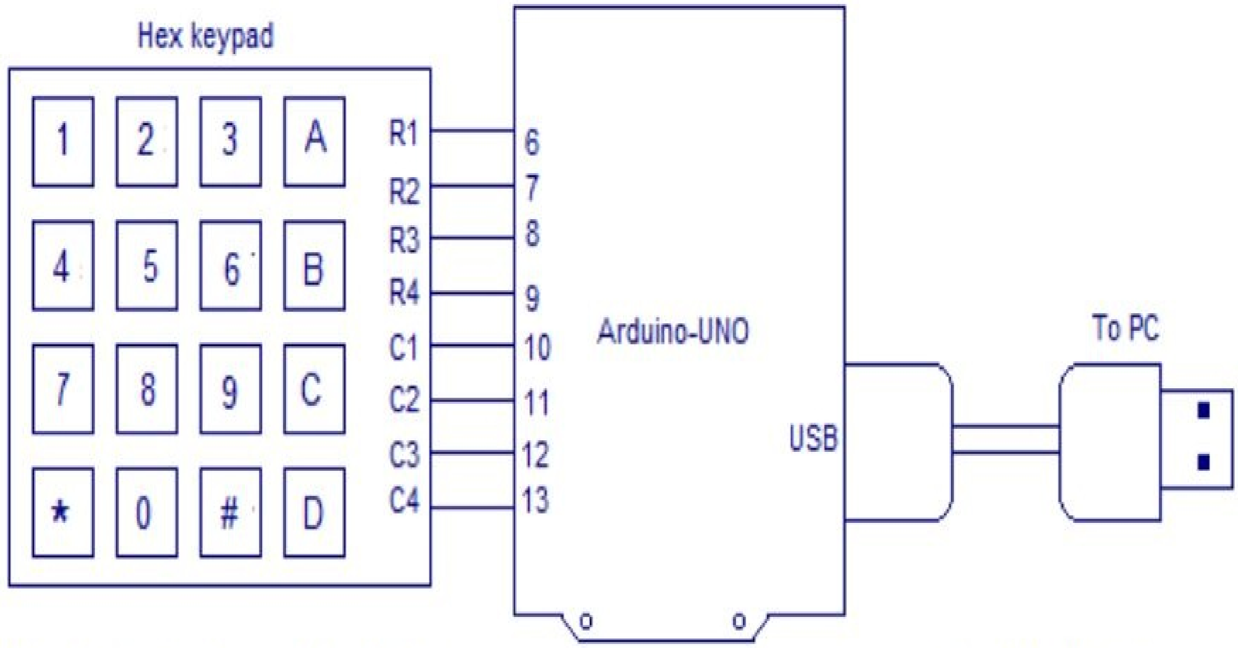
Hex key pad is honestly an association 0f sixteen push button switches in a 4X4 matrix form. usually a hex keypad could have keys for quantity 0, 1, 2, 3, four, five, 6, 7, eight, 9 and letters A, B, C, D, *, #. The hex keypad may have 8 connection wires particularly R1, R2, R3, R4 and C1, C2, C3, C4 representing the rows and columns respectively. The schematic diagram and image of a regular hex keypad is shown within the discern under.



Hex keypad www.circuitstoday.com

The program identifies the pressed key by means of a method referred to as column scanning. on this method a specific row is stored low and other rows are held high. The the logic fame of every column line is scanned. If a specific column is discovered low, then meaning the important thing that comes in among that column and row is short(pressed). Then this system registers that key being pressed. Then the same technique is applied for the following rows and the complete system is repeated. for instance if row 1 is stored low and column 1 is discovered low for the duration of scanning, then which means key”1″ is pressed. full circuit diagram of interfacing hex keypad is proven under.

Circuit diagram.



Interfacing hex keypad to Arduino

www.circuitstoday.com

Rows R1, R2, R3 and R4 are interfaced to virtual pins 6, 7, eight and 9 pins of the arduino respectively. Columns C1, C2, C3 and C4 are interfaced to the digital pins 10, eleven, 12, thirteen of the arduino. The arduino is hooked up to pc via the USB port. The circuit is powered from the USB itself and no external energy deliver is wanted. the entire application for interfacing hex keypad to arduino is given under.

Program (Version 1)

```
int r1=6;
int r2=7;
int r3=8;
int r4=9;
int c1=10;
int c2=11;
int c3=12;
int c4=13;
int colm1;
int colm2;
int colm3;
int colm4;

void setup()
{
pinMode(r1,OUTPUT);
pinMode(r2,OUTPUT);
pinMode(r3,OUTPUT);
pinMode(r4,OUTPUT);
pinMode(c1,INPUT);
pinMode(c2,INPUT);
pinMode(c3,INPUT);
pinMode(c4,INPUT);
Serial.begin(9600);
digitalWrite(c1,HIGH);
digitalWrite(c2,HIGH);
digitalWrite(c3,HIGH);
digitalWrite(c4,HIGH);
}
void loop()
{
digitalWrite(r1,LOW);
digitalWrite(r2,HIGH);
digitalWrite(r3,HIGH);
digitalWrite(r4,HIGH);
colm1=digitalRead(c1);
colm2=digitalRead(c2);
colm3=digitalRead(c3);
colm4=digitalRead(c4);
if(colm1==LOW)
{Serial.println("1");
    delay(200);}
else
{
    if(colm2==LOW)
    {Serial.println("2");
    delay(200);}
    else
    {
    if(colm3==LOW)
    {Serial.println("3");
    delay(200);}
    else
    {
    if(colm4==LOW)
    {Serial.println("A");
    delay(200);}
    }}}
```

```
digitalWrite(r1,HIGH);
digitalWrite(r2,LOW);
digitalWrite(r3,HIGH);
digitalWrite(r4,HIGH);
colm1=digitalRead(c1);
colm2=digitalRead(c2);
colm3=digitalRead(c3);
colm4=digitalRead(c4);
if(colm1==LOW)
{Serial.println("4");
    delay(200);}
else
{
    if(colm2==LOW)
    {Serial.println("5");
    delay(200);}
    else
    {
    if(colm3==LOW)
    {Serial.println("6");
    delay(200);}
    else
    {
    if(colm4==LOW)
    {Serial.println("B");
    delay(200);}
    }}}

digitalWrite(r1,HIGH);
digitalWrite(r2,HIGH);
digitalWrite(r3,LOW);
digitalWrite(r4,HIGH);
colm1=digitalRead(c1);
colm2=digitalRead(c2);
colm3=digitalRead(c3);
colm4=digitalRead(c4);
if(colm1==LOW)
{Serial.println("7");
    delay(200);}
else
{
    if(colm2==LOW)
    {Serial.println("8");
    delay(200);}
    else
    {
    if(colm3==LOW)
    {Serial.println("9");
    delay(200);}
    else
    {
    if(colm4==LOW)
    {Serial.println("C");
    delay(200);}
    }}}
digitalWrite(r1,HIGH);
digitalWrite(r2,HIGH);
digitalWrite(r3,HIGH);
digitalWrite(r4,LOW);
colm1=digitalRead(c1);
colm2=digitalRead(c2);
colm3=digitalRead(c3);
colm4=digitalRead(c4);
if(colm1==LOW)
{Serial.println("*");
    delay(200);}
else
{
    if(colm2==LOW)
    {Serial.println("0");
    delay(200);}
    else
    {
    if(colm3==LOW)
    {Serial.println("#");
    delay(200);}
    else
    {
    if(colm4==LOW)
    {Serial.println("D");
    delay(200);}

    }}}

}
```

About the program.

Code Serial.begin(9600); sets the baud fee for serial verbal exchange at 9600. Baud price is the number of signal modifications taking place in a 2d in a digitally modulated transmission line. first of all row1 is held excessive and different rows are held low using digitalWrite command. Then the status of each column is read using digitalRead command. Then each column is checked for low the usage of an if-else loop. If a selected column in discovered low, the important thing intersecting that column and row1 is believed to be pressed and the call of that secret is displayed at the serial screen window the usage of Serial.print command. A put off of 200ms is given between every situation checking loops with a view to avoid a couple of key registering in the course of a single key press.

If this postpone is expanded similarly more the reaction of the keypad could be reduced. After a few trial and mistakes, i found the top of the line fee for my scheme to be 200mS. Then row 2 is made low and other rows are kept high. The column scanning the usage of if loop-else loop is performed again. Then the equal thing is achieved for row three after which for row 4. The the complete cycle is repeated through the years. The result can be the name of the pressed key displayed at the serial monitor any time.

Simpler Version of Above Program (Version 2)

We are able to without problems simplify the program written above with smart utilization of Arrays and For Loops. we're including a simplified version of the above application (that's half of the wide variety of traces of code above). you can evaluate each codes for knowledge the subject.

```
int row[]={6,7,8,9};// Defining row pins of keypad connected to Aeduino pins
int col[]={10,11,12,13};//Defining column pins of keypad connected to Arduino
int i,j; // Two counter variables to count inside for loop
int col_scan; // Variable to hold value of scanned columns
void setup()
{
Serial.begin(9600);
for(i=0;i<=3;i++)
{
pinMode(row[i],OUTPUT);
pinMode(col[i],INPUT);
digitalWrite(col[i],HIGH);
} }
void loop()
{
for(i=0; i<=3; i++)
{
digitalWrite(row[0],HIGH);
digitalWrite(row[1],HIGH);
digitalWrite(row[2],HIGH);
digitalWrite(row[3],HIGH);
digitalWrite(row[i],LOW);
for(j=0; j<=3; j++)
{
col_scan=digitalRead(col[j]);
if(col_scan==LOW)
{
keypress(i,j);
delay(300);
}}
}}
void keypress(int i, int j)
{
if(i==0&&j==0)
Serial.println("1");
if(i==0&&j==1)
Serial.println("2");
if(i==0&&j==2)
Serial.println("3");
if(i==0&&j==3)
Serial.println("A");
if(i==1&&j==0)
Serial.println("4");
if(i==1&&j==1)
Serial.println("5");
if(i==1&&j==2)
Serial.println("6");
if(i==1&&j==3)
Serial.println("B");
if(i==2&&j==0)
Serial.println("7");
if(i==2&&j==1)
Serial.println("8");
if(i==2&&j==2)
Serial.println("9");
if(i==2&&j==3)
Serial.println("C");
if(i==3&&j==0)
Serial.println("*");
if(i==3&&j==1)
Serial.println("0");
if(i==3&&j==2)
Serial.println("#");
if(i==3&&j==3)
Serial.println("D");
}
```

Build Arduino Weather Station Using DHT11, Soil Sensor, And Nokia Display

On this challenge tutorial, we're going to make an Arduino weather station the usage of 2 sensors; FC-28 soil moisture sensor to degree the moisture and the DHT22 sensor to measure the temperature, humidity and the warmth index. all of the measured parameters (with the assist of sensors) might be displayed using Nokia 5110 liquid crystal display.

Before we proceed in addition building our Arduino weather station, allow's see a few tutorials which is a prerequisite to efficiently enforce this mission. you can discover ways to use Soil moisture sensor with Arduino and also you could learn the way DHT11 is interfaced with Arduino. finally, learn how to use Nokia5110 liquid crystal display with Arduino. when you cowl all the ones three tutorials, you are equipped to build a climate station the usage of Arduino.

Components Required

the specified components for this challenge are as follows

- Arduino Uno
- Nokia 5110 liquid crystal display
- DHT22 Temperature and humidity sensor
- FC-28 Soil moisture sensor
- Breadboard
- Connecting wires
- 1k potentiometer
- four X 10k resistors
- 1k resistor
- 330 ohm resistor

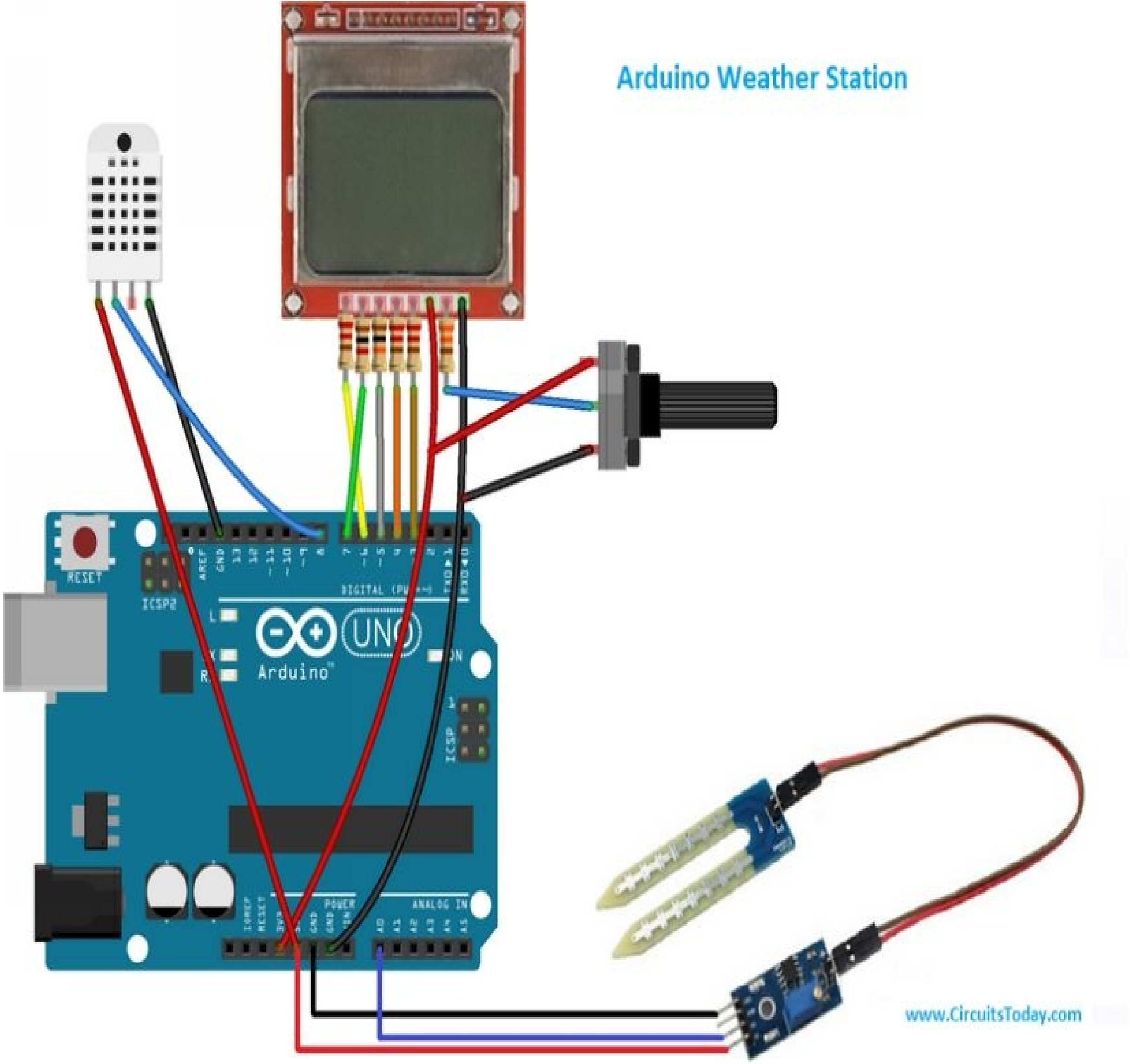
Working: Weather Station

On this climate station project, we are going to use sensors. One is for measuring the moisture of the soil and the alternative is for analyzing the temperature, humidity and the heat index.

The FC- 28 soil moisture sensor gives us analyzing in the form of analog voltage from zero to 1023. when the soil is dry, the output price of FC-28 sensor might be towards 1023 and whilst the soil is moist, the cost may be towards zero. Moisture is measured in percent, so we need to convert those values to a brand new scale measuring zer0 to one hundred. This percentage is then be displayed on the Nokia 5110 liquid crystal display.

The DHT22 sensor can measure temperature, humidity, and warmth index. The DHT22 sensor gives output inside the virtual shape and is fed without delay through the digital I/O pins of Arduino. The Arduino reads this output and calculates temperature, humidity, and the heat index. those values are then displayed at the Nokia 5110 lcd.

Circuit Diagram



Arduino Weather Station – Circuit Diagram

First of all, let’s connect the Nokia 5110 LCD with the Arduino. Follow the instructions below.

- Connect the RST pin of Nokia 5110 LCD to the pin 6 of Arduino through the 10k resistor

- Connect the SCE pin of Nokia 5110 LCD to the pin 7 of Arduino through the 1k resistor
- Connect the D/C pin of Nokia 5110 LCD to the pin 5 of Arduino through the 10k resistor
- Connect the DIN pin of Nokia 5110 LCD to the pin 4 of Arduino through the 10k resistor
- Connect the CLK pin of Nokia 5110 LCD to the pin 3 of Arduino through the 10k resistor
- Connect the VCC pin of Nokia 5110 LCD to the 3.3V pin of Arduino
- Connect the LED pin of Nokia 5110 LCD to the middle pin of 1k potentiometer through the 330 ohm resistor and connect the other two ends of the potentiometer to the VCC and the GND.
- Connect the GND pin of Nokia 5110 LCD to the GND pin 6 of Arduino

After that, make the connections of the DHT22 sensor and FC-28 soil moisture with the Arduino.

Connections for DHT22 with Arduino

- Connect pin 1 (VCC pin) of DHT22 sensor to the 5V pin of Arduino
- Connect pin 2 (Data pin) of DHT22 to the pin 8 of Arduino
- Connect pin 4 (GND pin) of DHT22 to the GND pin of Arduino

Connections for FC-28 with Arduino

- Connect VCC pin of FC-28 to 5V of Arduino
- Connect A0 pin of FC-28 to A0 of Arduino
- Connect GND pin of FC-28 to GND of Arduino

Code Explanation

Download Program – Arduino Weather Station Project

First of all, we need to consist of the libraries for the Nokia 5110 lcd and the DHT22 sensor. After including the libraries, we initialized a variable named ‘lcd’ of kind PCD8544 and a variable named ‘temp_hum_sensor’ of type DHT. Then we declared the pin eight for DHT22.

```
#include <PCD8544.h>

PCD8544 lcd;

#include "DHT.h"
```

```
#define DHTPIN 8

#define DHTTYPE DHT22

DHT temp_hum_sensor(DHTPIN, DHTTYPE);
```

Inside the setup characteristic, we commenced the conversation with the Nokia 5110 lcd at ’84 X forty eight’ dimensions and revealed “analyzing From the Sensors” at the liquid crystal display display. Then we started the conversation with the DHT22 sensor.

```
lcd.begin(84, 48);

lcd.print("Reading From the Sensors");

temp_hum_sensor.begin();
```

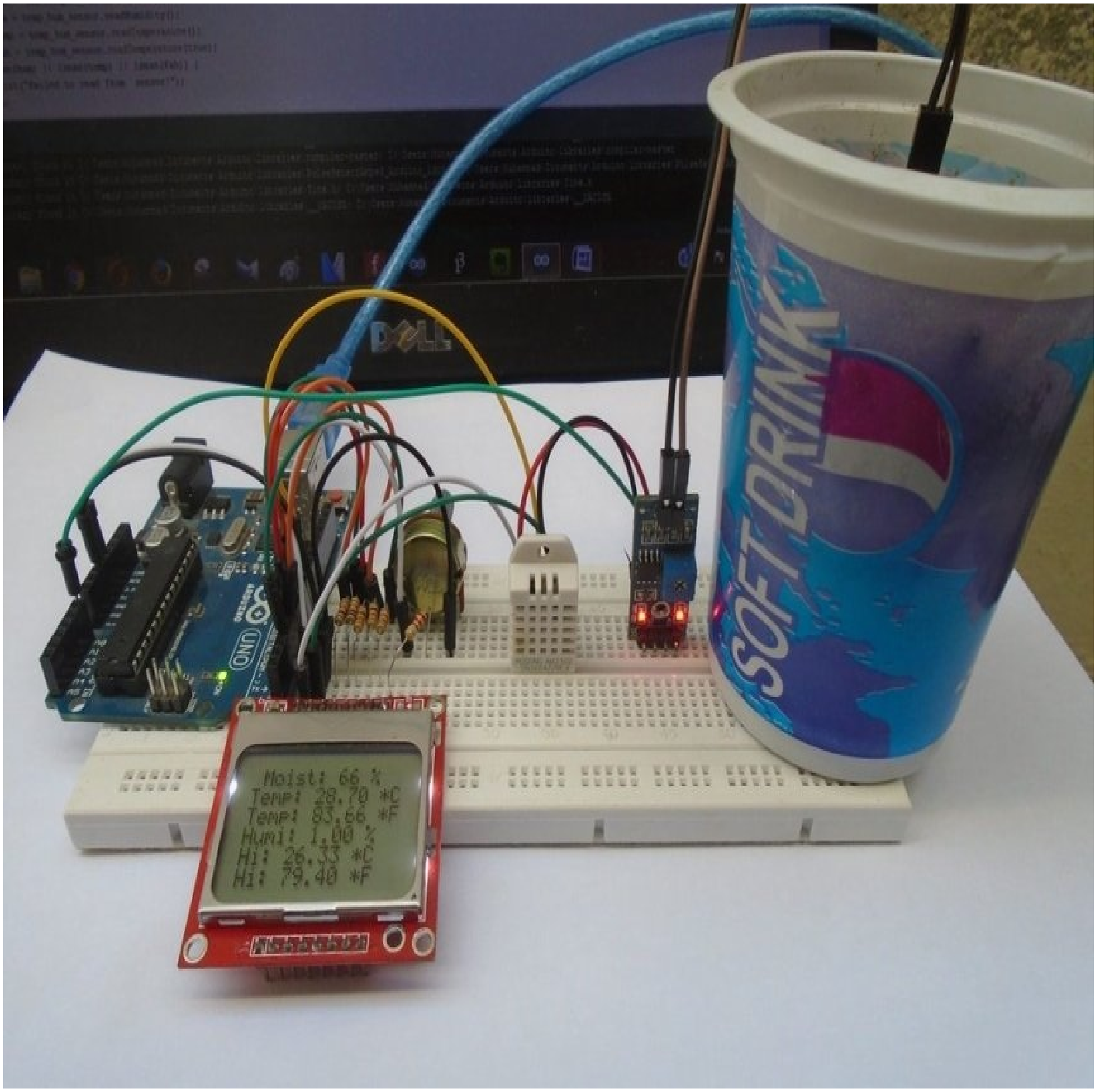
In the loop feature, we took the values from the soil moisture sensor and stored within the variable. The soil moisture sensor will give us values from 0 to 1023, we want to convert these into percentage to measure the moisture. So, we converted the values from 550 to forty into the 0 to a hundred. We transformed the values from 550 to zero due to the fact 550 is the cost that sensor gives us when the soil changed into dry and when the soil turned into absolutely wet, when it gives us value of forty.

```
output_value= analogRead(sensor_pin);

output_value = map(output_value, 550, 40, 0, 100);
```

After that, we took the studying of the temperature, humidity and heat index from the DHT22 temperature and humidity sensor and saved these inside the variable. Then, we confirmed these line via line on Nokia 5110 liquid crystal display.

```
float hum = temp_hum_sensor.readHumidity();

float temp = temp_hum_sensor.readTemperature();

float fah = temp_hum_sensor.readTemperature(true);

lcd.print("Hi: ");

lcd.print(heat_index);

lcd.println(" *F ");
```

Photographs



```
Moist: 66 %
Temp: 28.70 *C
Temp: 83.66 *F
Humi: 1.00 %
Hi: 26.33 *C
Hi: 79.40 *F
```

in order that’s it! we've got completed our educational on building an Arduino climate station task the usage of two sensors (FC-28 soil moisture sensor and the DHT11 humidity/temperature sensor) and the Nokia 5110 liquid crystal display display. Do the mission yourself and spot the way it is going!

Automatic Irrigation System.

in this venture, we are going to build an automatic irrigation system the use of Arduino which senses the moisture of the soil and opens or closes the valve in step with the moisture fee. The moisture fee and the valve popularity is shown on the Nokia 5110 liquid crystal display display.

Before we proceed, permit's take a look at another task – a plant watering system using Arduino, which is very similar to our application. you can also study our other tutorials on – interfacing Nokia lcd to Arduino and our tutorial on interfacing Soil Moisture Sensor to Arduino.

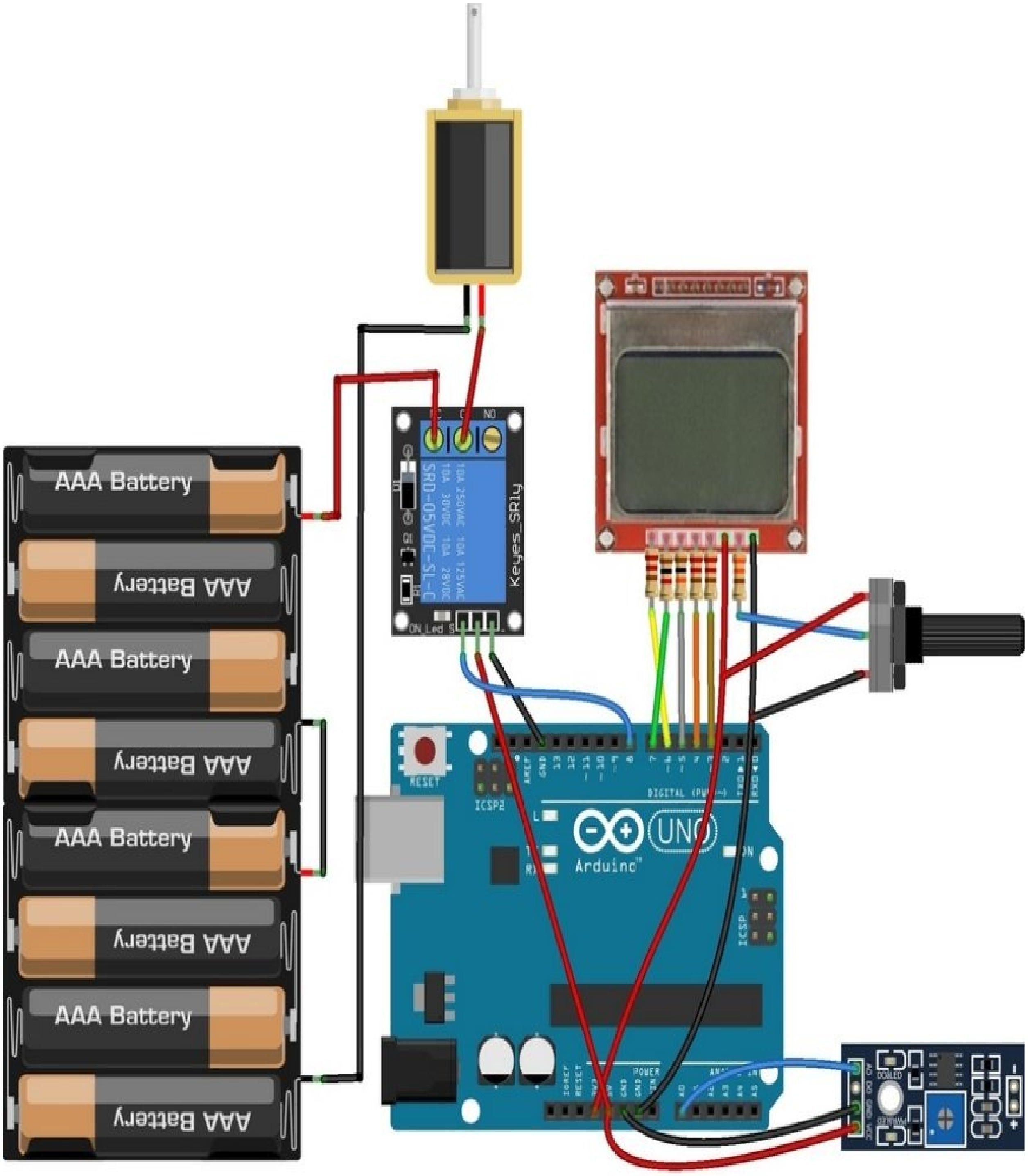
Required Components

The components required for this venture are as follows

- Arduino
- Nokia 5110 liquid crystal display
- FC-28 Soil Moisture Sensor
- single Relay Module
- 1K potentiometer
- four X 10K resistors
- 1K resistor
- 330 ohm resistor

Circuit Diagram – Automatic Irrigation System

Nokia 5110 lcd operates on three.3V, and if we join it without delay to the records pins of Arduino the existence time of Nokia 5110 lcd will lower (word that the lcd will nevertheless work even in case you join without delay without current restricting resistors). So, if you need to apply it for a longer span, it's good to connect through resistors as we've got proven on this project.



To connect the Nokia 5110 lcd with the Arduino, make the connections for the Nokia 5110 liquid crystal display with the Arduino as follows

- join the pin 1 of Nokia 5110 lcd that is the reset Pin to the pin 6 of Arduino through the 10K resistor.

- join the pin 2 of Nokia 5110 lcd which is the chip pick out pin to the pin 7 of Arduino thru the 1K resistor.
- join the pin 3 of Nokia 5110 liquid crystal display that's the facts or command pin to the pin five of Arduino via the 10K resistor.
- join the pin four of Nokia 5110 lcd which is the information input pin to the pin four of Arduino thru the 10K resistor.
- connect the pin 5 of Nokia 5110 liquid crystal display that is the clock pin to the pin three of Arduino through the 10K resistor.
- join the pin 6 of Nokia 5110 liquid crystal display that is the VCC pin to the three.3V pin of Arduino.
- join the pin 7 of Nokia 5110 lcd that is the LED pin to the center pin of 1k potentiometer via 330 ohm resistor and connect the opposite pins to the VCC and the floor.
- join the pin eight of Nokia 5110 liquid crystal display that's the ground pin to the GND of Arduino.

The potentiometer used right here is for growing or reducing the backlight evaluation of lcd. in case you want complete lower back mild, then you may join it without delay to 5V or if you need no backlight, then join it to ground.

After that, join the relay module with the Arduino as follows

- join the VCC pin of relay to the 5V of Arduino.
- join the floor pin of the relay to the GND of Arduino.
- connect the IN pin to the pin eight of Arduino.

On the opposite aspect of the relay, connect the tremendous of the battery to the NC terminal (No Connection) of the relay and the C terminal of the relay to the superb of the solenoid valve. in the end, connect the poor of the battery to the terrible of the solenoid valve.

- In last, connect the FC-28 soil moisture sensor to the Arduino as follows
- connect the VCC pin of FC-28 to the 5V pin of Arduino.
- connect the A0 pin of FC-28 to the A0 of Arduino.

Working – Automatic Irrigation System

The soil moisture sensor senses the moisture present within the soil and offers the output to Arduino. The output of the sensor can be from 0-1023. The moisture is measured in percentage; therefore we should map those values to zero-one hundred.

every time the moisture price is lower than the cost we've set internal our code as the threshold price, the relay will switch on and the valve will flip open and on every occasion the moisture fee is lower than this threshold fee, the relay will relay will turn OFF and the valve gets closed.

Program/Code

```
#include <PCD8544.h>

PCD8544 lcd;

int sensor_pin = A0 ;

int output_value ;

int valve_pin = 8 ;

void setup ( ) {

lcd.begin (84, 48);

lcd.print ("Reading From the Sensor ... " );

pinMode (valve_pin, OUTPUT);

delay (2000);

}


void loop ( ) {

output_value = analogRead (sensor_pin);

output_value = map (output_value , 1023 , 0 , 0 , 100);

lcd.clear ( );

lcd.setCursor (0, 0);

lcd.print (" Moisture = ");

lcd.print (output_value);

lcd.setCursor (0, 1);

if (output_value < 40)
{
lcd.print (" Valve: ON ");
digitalWrite (valve_pin , LOW); //Relay operates on opposite signal
delay (1000);
}
else
{
lcd.print (" Valve: OFF ");
digitalWrite (valve_pin , HIGH); //Relay operates on opposite signal
delay (1000);
}
delay(1000);
}
```

Code Explanation

First, we covered the library for Nokia 5110 lcd and declared a constructor variable named ‘lcd’ for the use of the Nokia 5110 liquid crystal display library functions. Then we initialized different variables to examine from the sensors and to show ON or OFF the valve.

```
#include <PCD8544.h>
PCD8544 lcd;
int sensor_pin = A0 ;
int output_value ;
int valve_pin = 8 ;
```

In the loop function, we first get the readings from the soil moisture sensor FC-28 and saved those readings inside the ‘output_value’ variable. The output values of the sensor could be from 0 to 1023 but we want the moisture cost that is in percent. So, we will map the output values of the sensor from 1023-zero to zero-100.

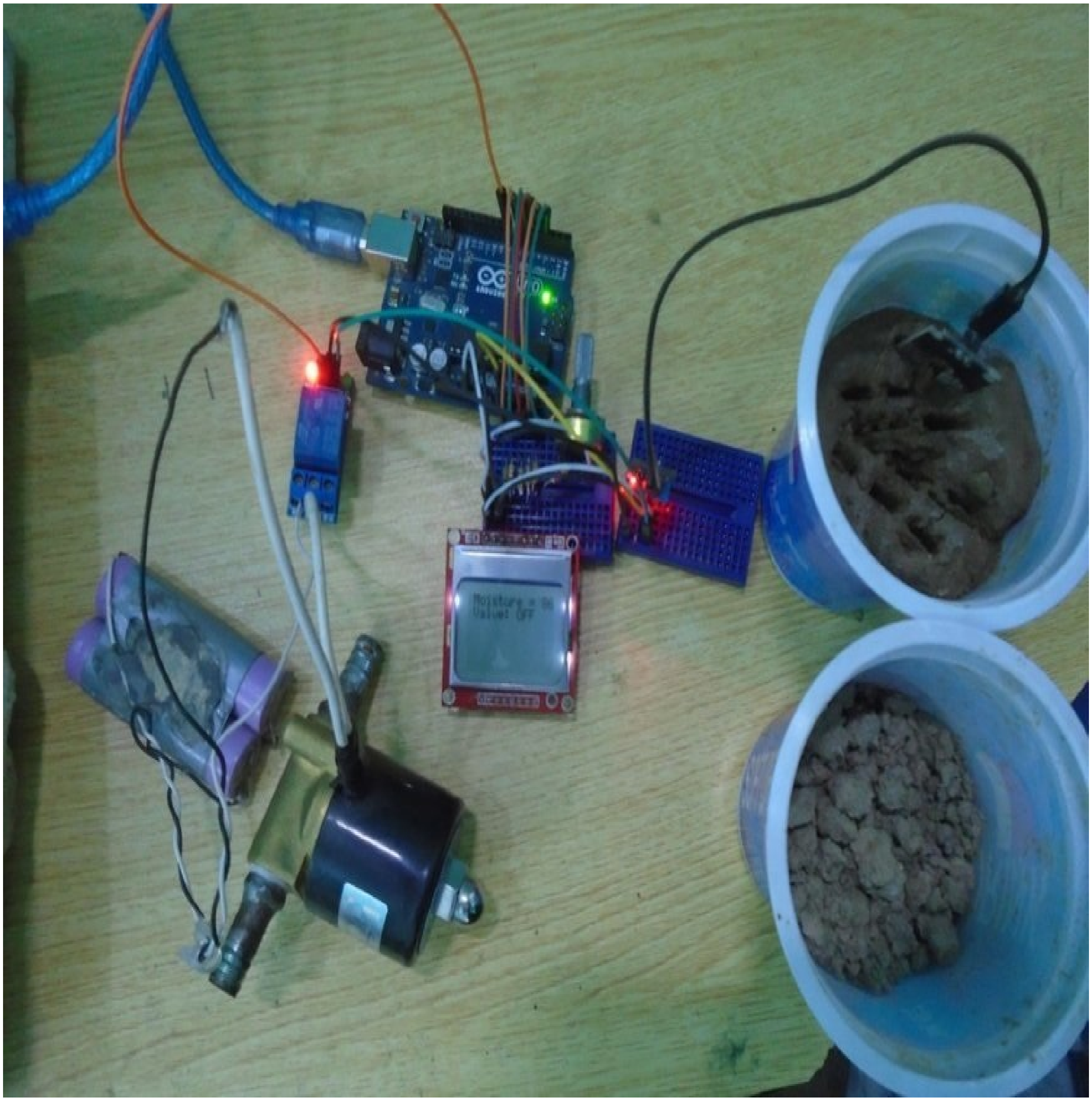
```
output_value = analogRead (sensor_pin);

output_value = map (output_value , 1023 , 0 , 0 , 100);
```

Then we are able to print these values at the Nokia 5110 liquid crystal display and we will activate or off the valve in step with it. If the moisture price may be decrease than 40, then we will make the valve pin low so one can turn on the valve. If the moisture fee could be better than 40, then we can make the valve pin excessive a good way to turn off the valve.

```
lcd.print (" Moisture = ");

lcd.print (output_value);

lcd.setCursor (0, 1);

if (output_value < 40)

{

lcd.print (" Valve: ON ");

digitalWrite (valve_pin , LOW); //Relay operates on opposite signal

delay (1000);

}

else

{

lcd.print (" Valve: OFF ");

digitalWrite (valve_pin , HIGH); //Relay operates on opposite signal
```

So, we finished building an automatic irrigation gadget the usage of Arduino. you could do this DIY irrigation system on your fields close by and see how it is going!

Output Photographs


Moisture = 96
Valve: OFF

Rotating LED Display Using Arduino POV

On this challenge, we're going to show you a way to make a easy “Rotating LED show” (also popularly referred to as Spinning LED display) with Arduino. the inducement to make this assignment got here to my thoughts after I noticed a product in the marketplace, in which a text message changed into sincerely seen on a “Rotating LED Strip”.

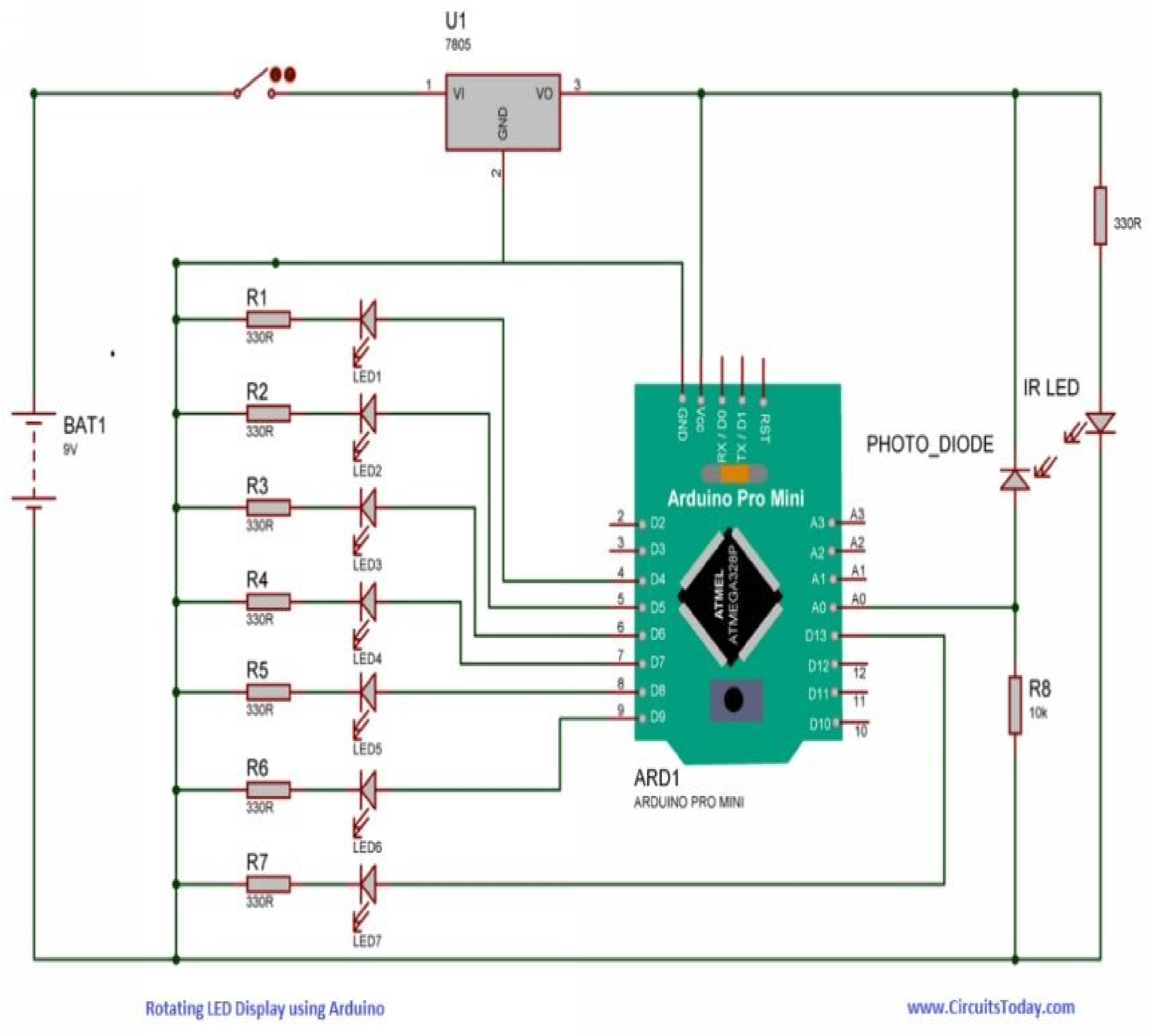
After that, I planned to make this assignment myself and Arduino is a superb preference to build this spinning led display.

For this mission, I designed PCB the use of Eagle CAD software. The motive to pick Eagle became to build a lightweight PCB, as there's excessive speed rotation worried within the project, and the PCB will be part of the rotating module. in this circuit, “Arduino pro mini” is used due to the fact it's far very small and powerful board.

Seven LED’s, a pair of IR LED and a Photodiode is also used. IR LED and image diode are used to layout a comments machine; which offers the real position of the rotor to Arduino. LED’s are located in a single line and are connected to the Arduino board. A small DC gear motor is used to rotate this circuit; and while the circuit rotates the LED’s make a blink sample that shows English alphabets. The code may be changed for showing other languages.

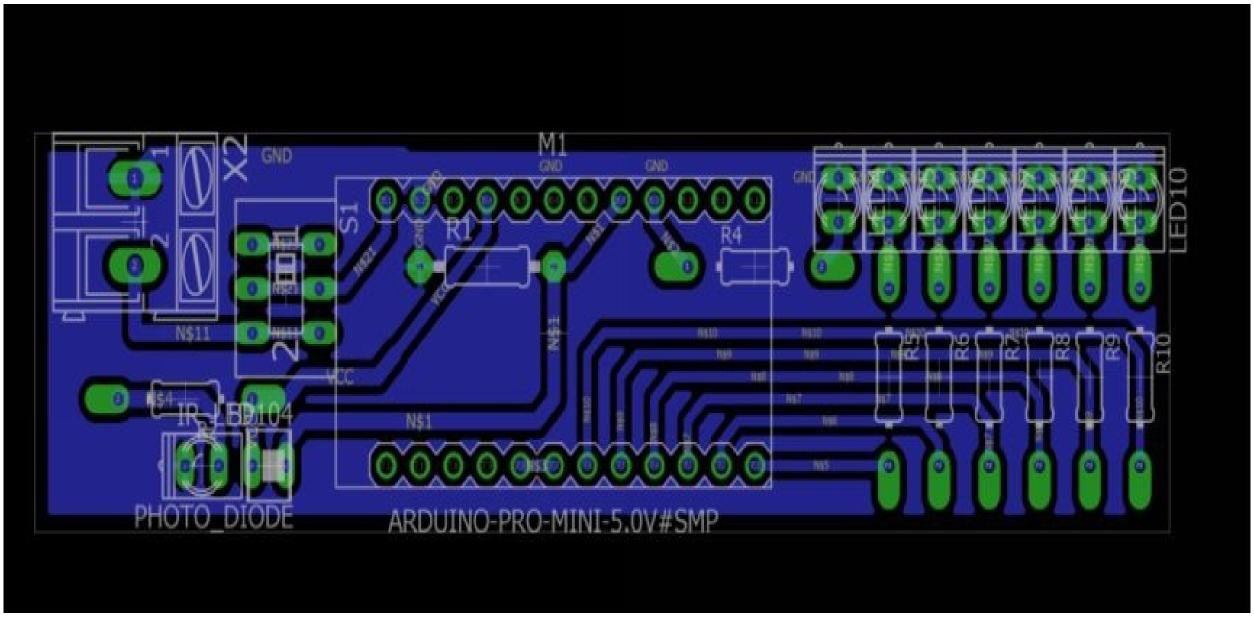
The operating of Rotating LED display is very exciting and it is able to be used for decoration purpose, where you can make beautiful designs and images the usage of this notable venture. This display also can be used in public places like railway/ bus station and airport as a signal board, as it's far very price effective to build. right protection is essential for the equipment motor and rotating assembly.

Circuit Diagram

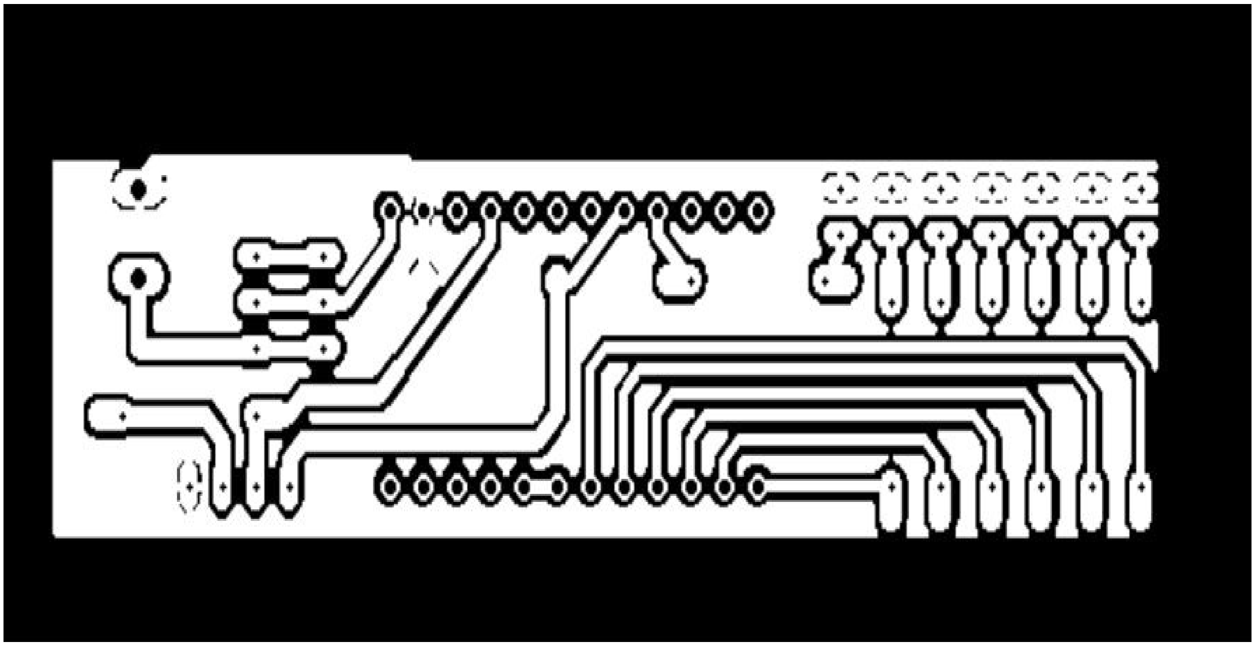


Spinning LED Display

Circuit diagram is proven above. I also designed a PCB on EAGLE and the documents are given below. it'd be better to use etched PCB. PCB Etching technique is a simple venture. For PCB etching refers the figure given under.

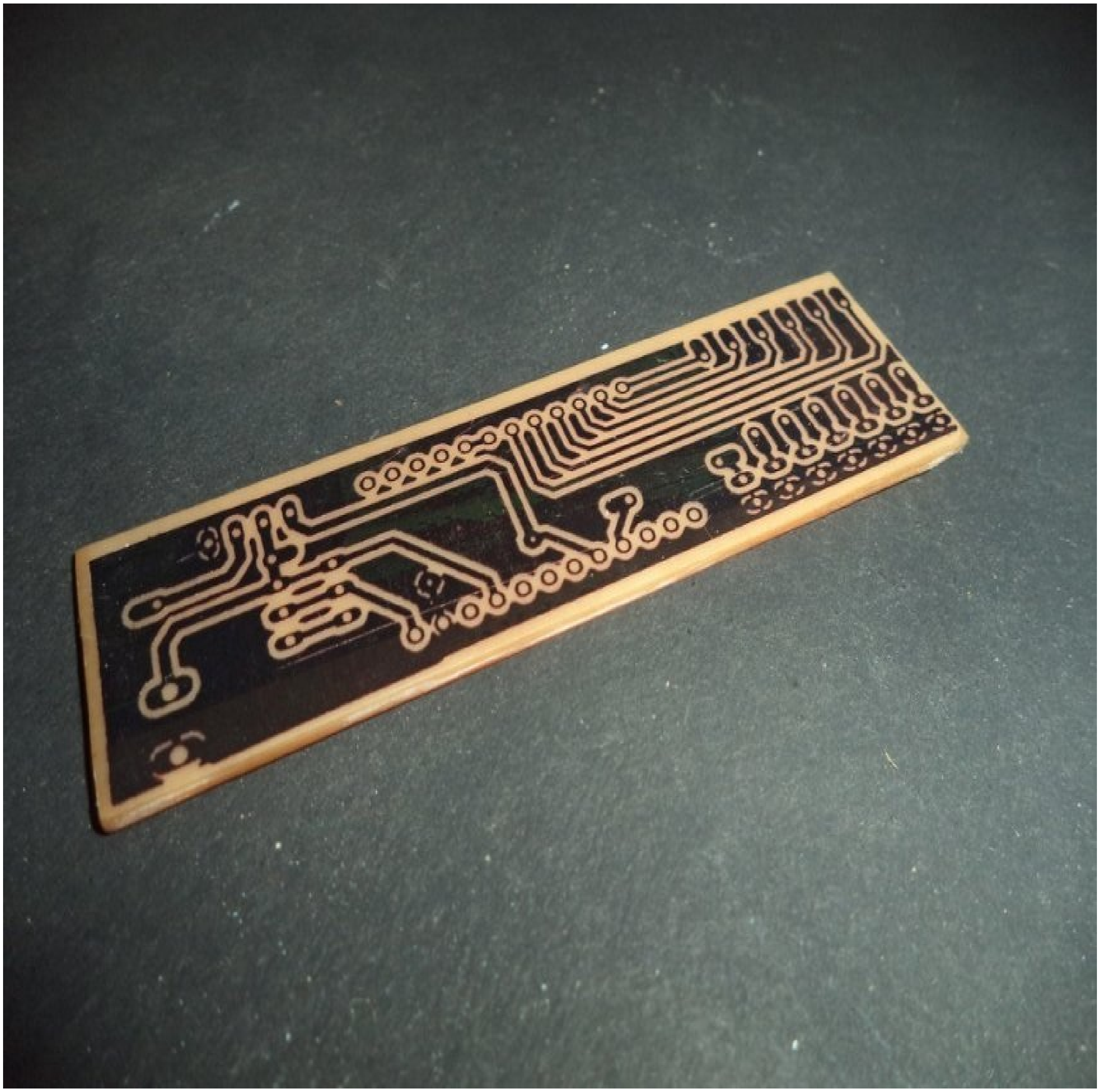


PCB Layout



PCB Layout for etching

In case you do no longer need to etch the PCB, simply use the general motive PCB (GPP), which is without difficulty to be had within the marketplace.

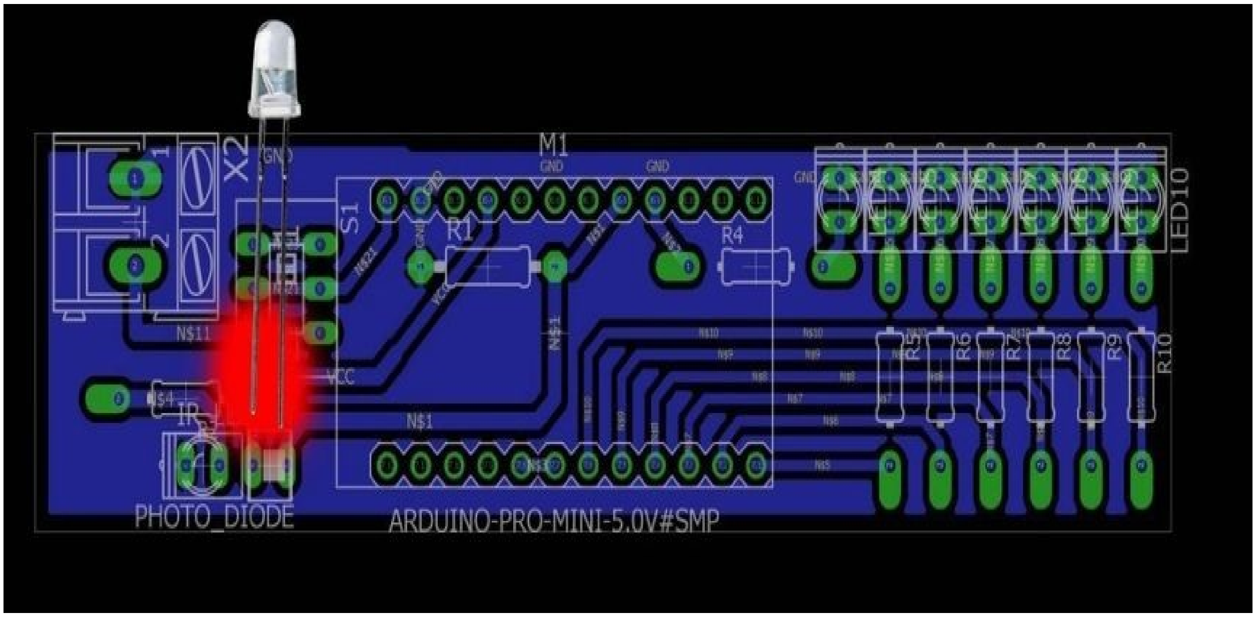


As you could see, circuit is simple however here i am explaining some key points that have to be saved in mind.

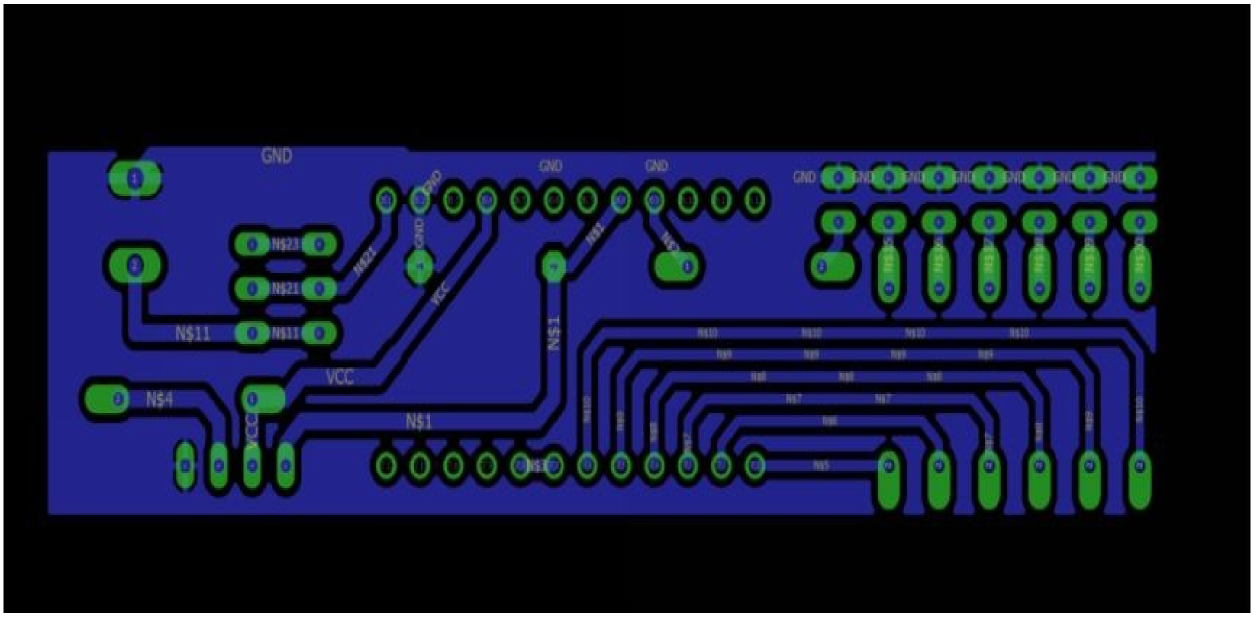
Use rectangular LED’s; because round LED’s consume more space, and the format designed by me is based on square LED’s.

After making the relationship, take a look at the IR LED. IR ray is not seen to our eyes, so open the digicam of your clever cellphone and observe the IR LED through your clever phones display screen. If mild is seen on IR LED, then the connection of LED is satisfactory.

Connection of Photodiode is a bit complicated, due to the fact Photodiode works in reverse bias. for proper connection simply see the discern below.

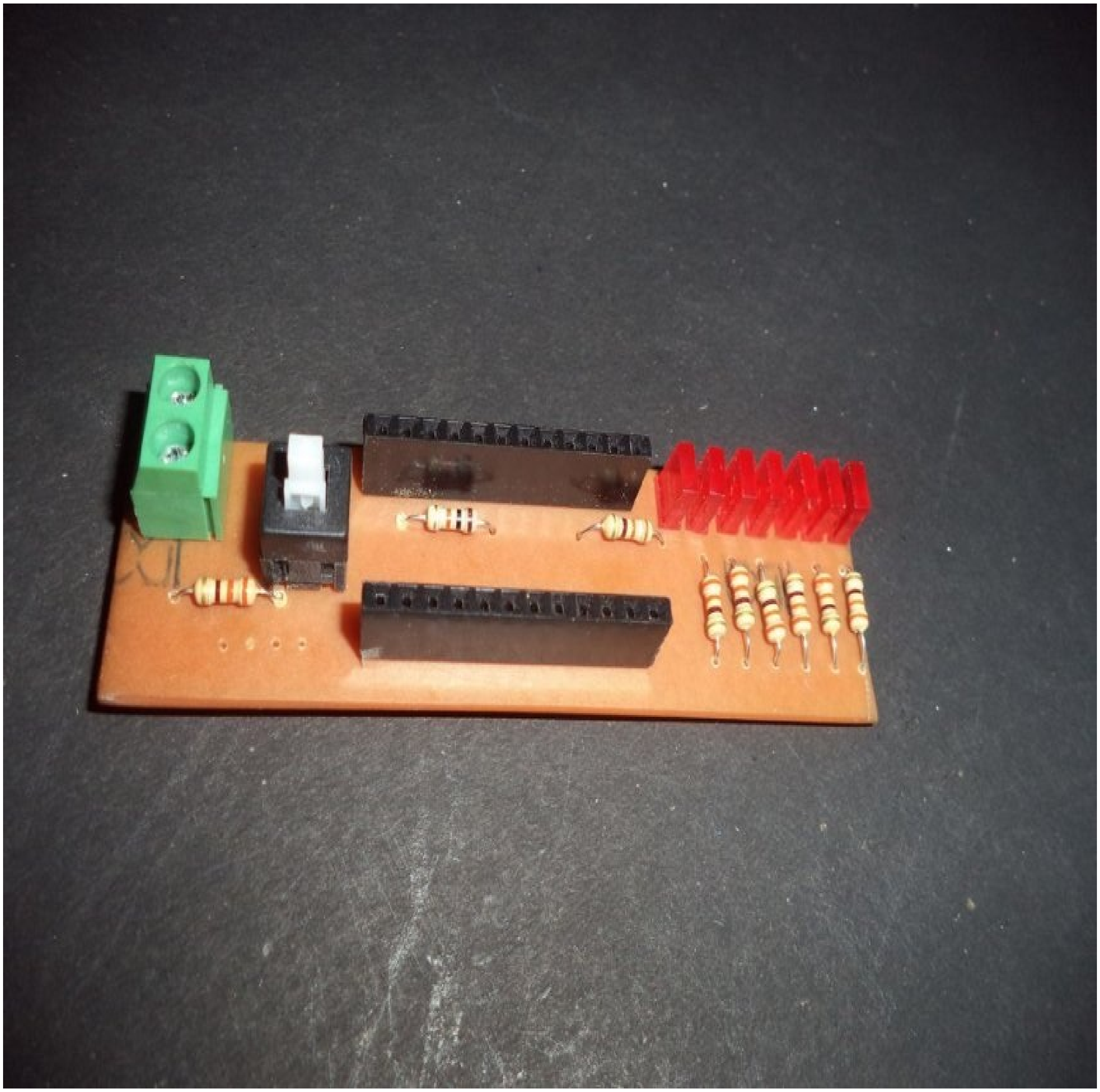


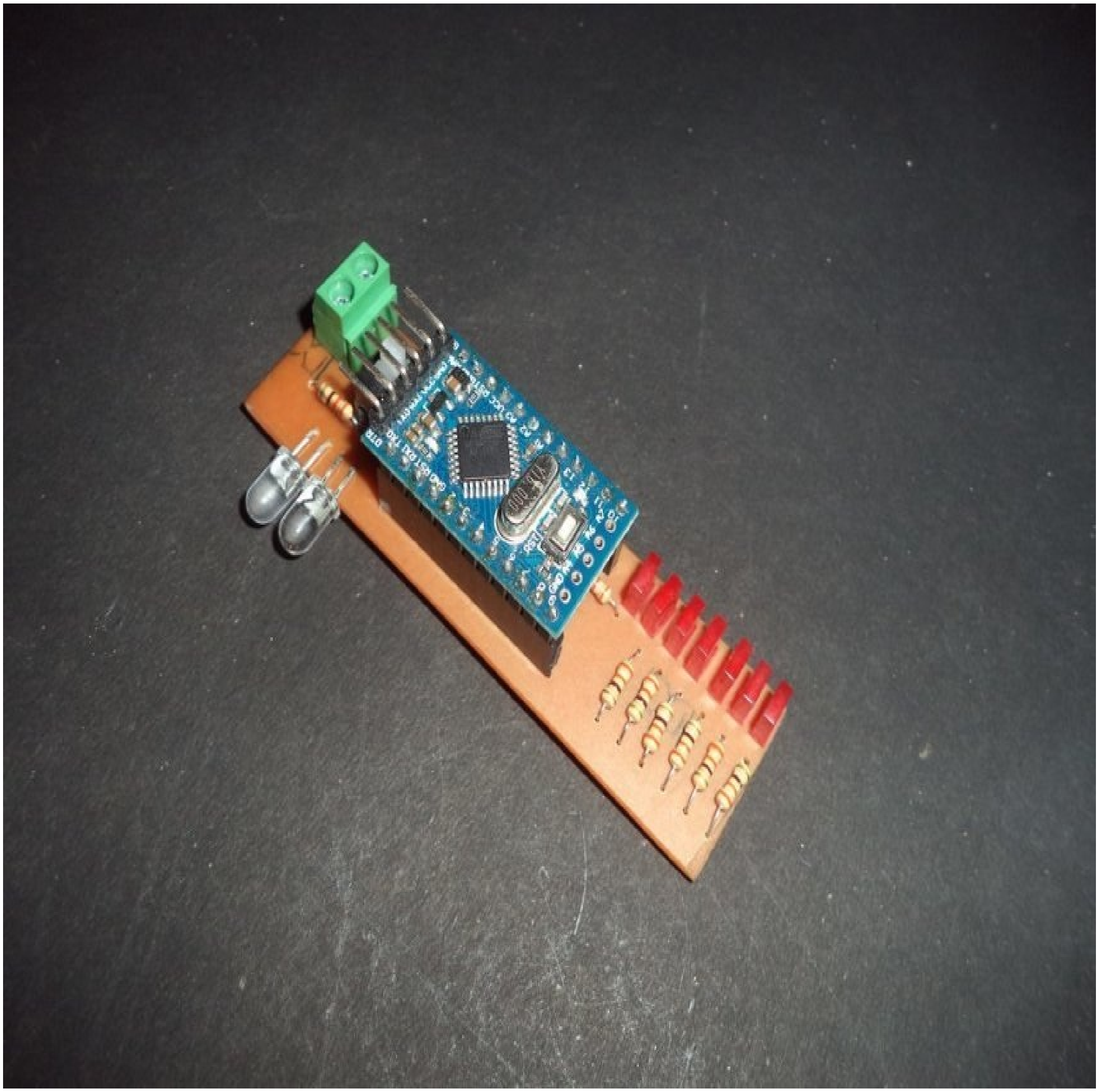
Photodiode Connection



This circuit can be powered with a 9-volt battery, join the battery with electricity connector the usage of battery cap.

After soldering all of the components, just turn on the circuit and take a look at the energy LED on Arduino. If the LED is glowing, you could add the code.



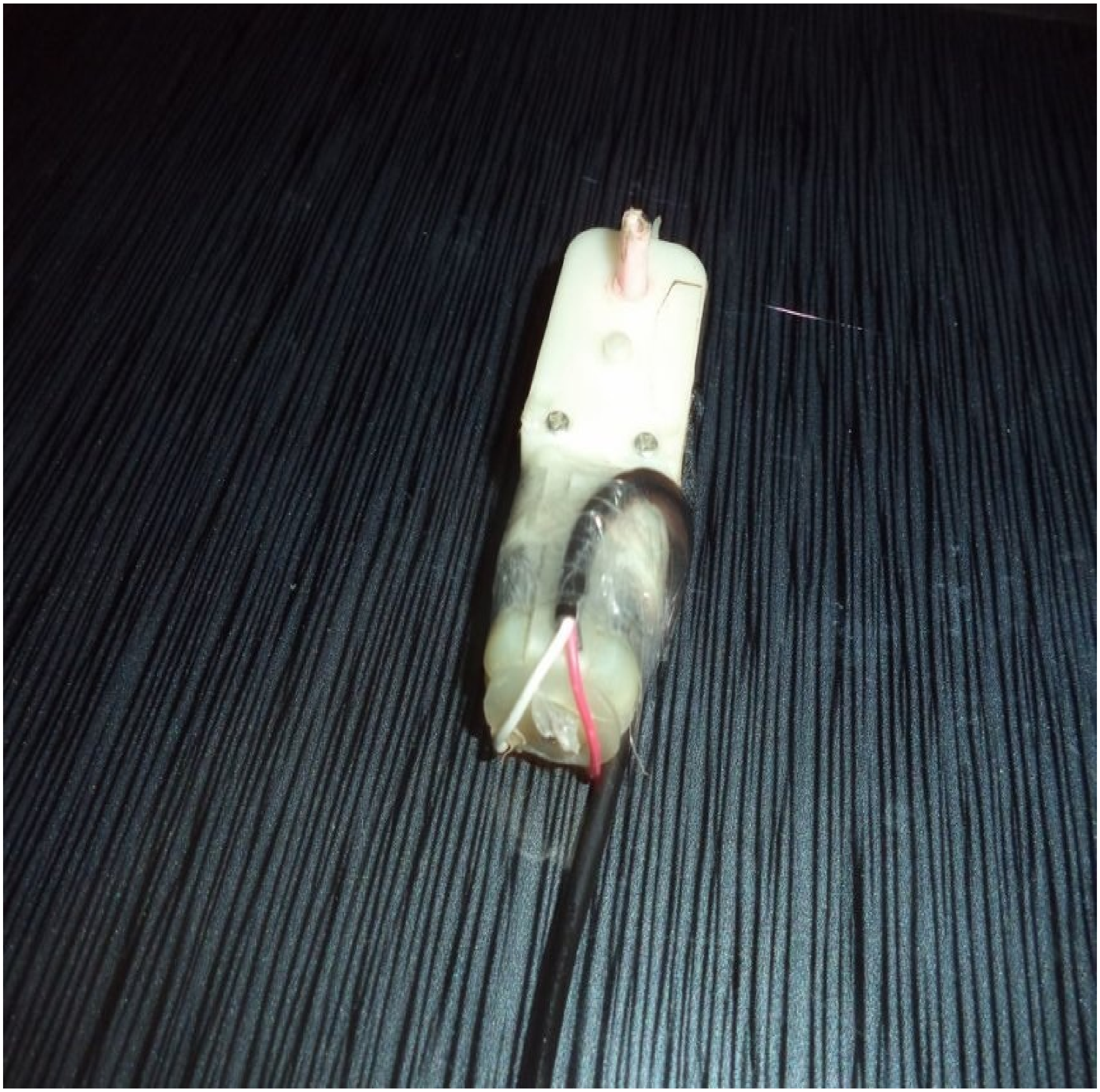


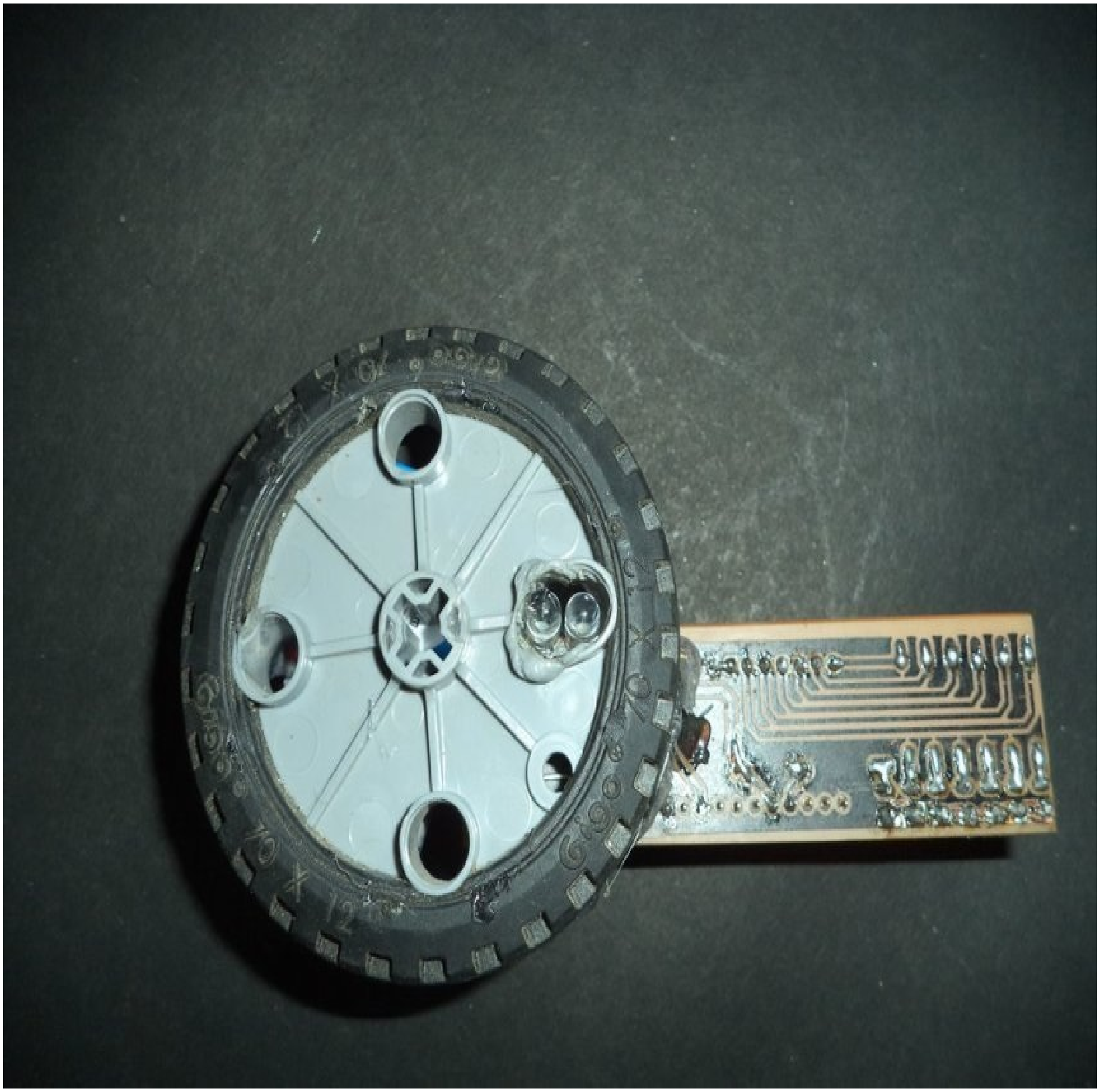
S. No.	Name	Quantity	Value	Specification
1.	Arduino Pro Mini	1		
2.	Copper Clad PCB	1		
3.	DC Geared Motor	1		More Then 300 RPM
4.	Wheel	1		For Motor
5.	Adapter	1	12 V	
6.	Battery	1	9 V	
7.	Battery Cap	1		

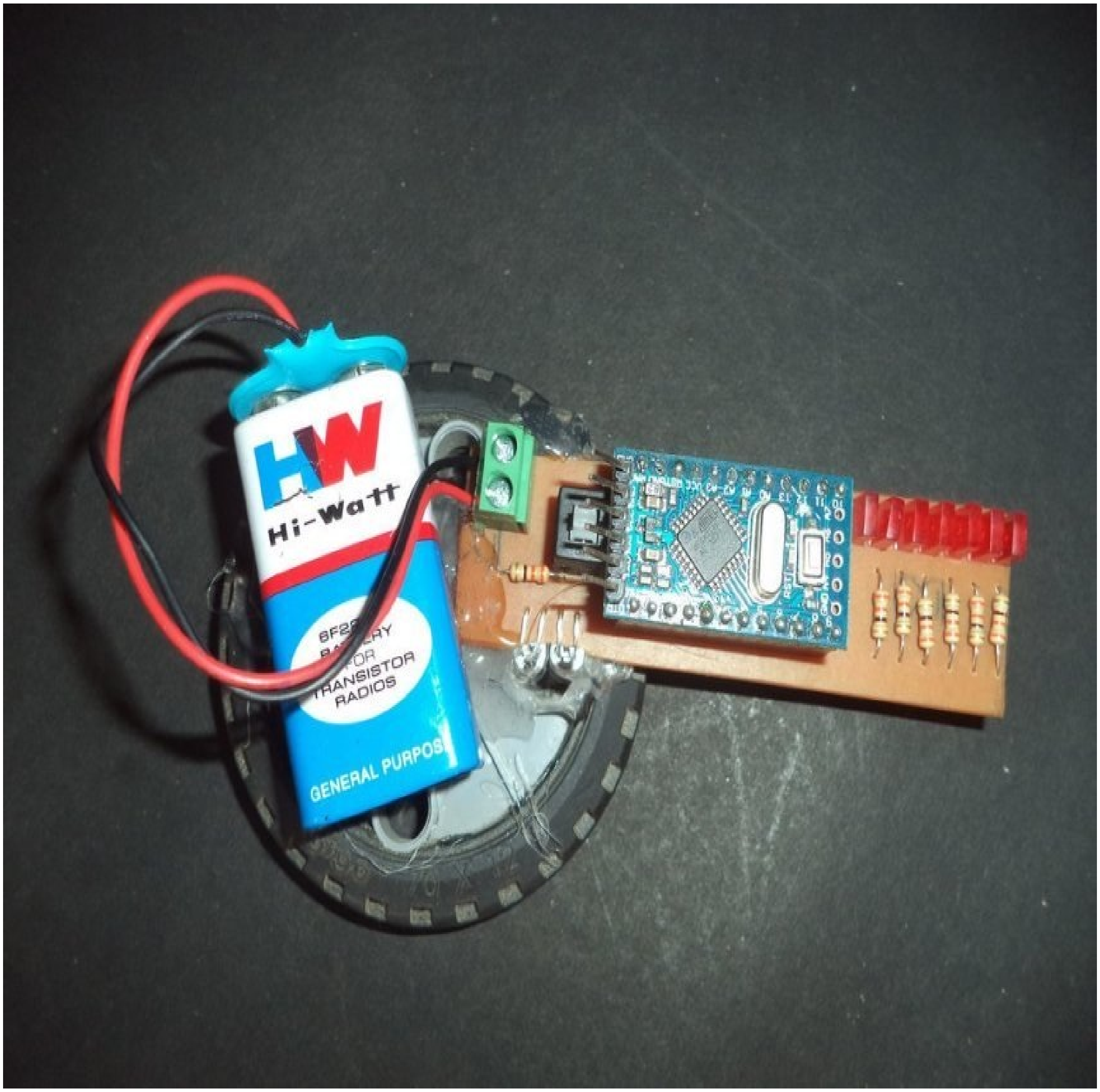
8.	LED	7		RED, Rectangular
9.	Resistance	8	330 ohm	
10.	Resistance	1	10K ohm	
11.	IR LED	1		
12.	Photo Diode	1		
13	DPDT Switch	1		

Motor

I am the usage of a 12 volt DC geared BO motor whose RPM is 300. right here I selected BO motor due to the fact it's far very easy to put in. you can use another motor based on availability, but do not use very small cars as it can't bring weight of the circuit. RPM of the motor need to be extra the 300.




Gigo 70 X 12



Program/Code

[Download Program](#)

Arduino seasoned Mini can be programmed the use of FTDI chip, so join the chip with Arduino, just copy and paste the code and upload.

earlier than making the motor assembly, test the circuit. For checking, positioned a white (reflective) paper in front of the IR LED and Photodiode. LED’s begin blinking and blinking will stop if we use black (non reflective) paper.

Photodiode is an analog sensor. So, it is connected to the analog pin A0 of Arduino. that is defined by using an integer StartPin. Seven LED’s are connected to 7 pins of Arduino. they're defined by way of seven integers PIN1 to PIN7.

Inside the “void setup()” A0 pin is defined as input, and PIN pins are defined as OUTPUT.

In the void loop(), an integer value is defined that's same to “analogRead” cost of StartPin, if the price val is extra than 200 then it goes to internal loop.

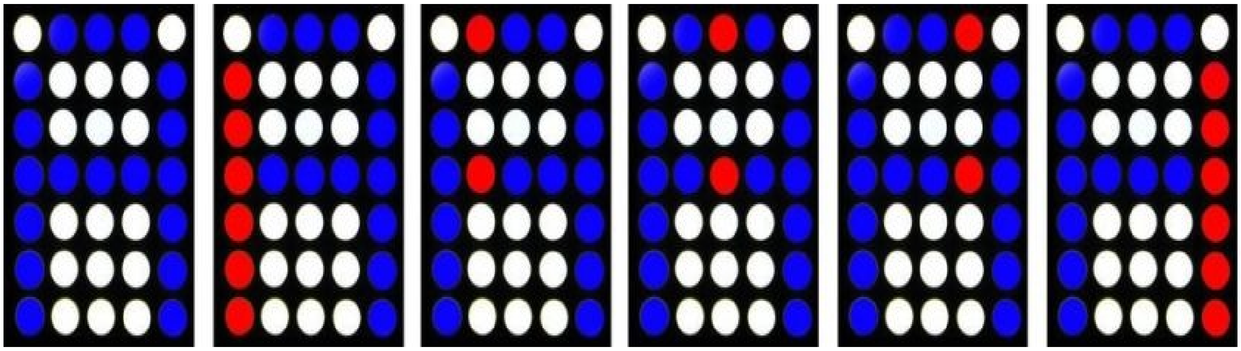
Within the “if” declaration, each alphabet of text is written, accompanied by means of Sp();

Within the subsequent steps, feature of all the alphabets are described. as an instance, void A(), in which the first line is signal(zero,1,1,1,1,1,1) – that's the first column of alphabet ‘A’. within the second line, sign(1,zero,0,1,zero,zero,0) – that is the second column of alphabet ‘A’. And on this manner all of the columns of each alphabets are described the use of sign function.

In the end, signal feature is defined with the aid of void signal in which D1 to D7 are described as high or LOW in step with ‘1’ or ‘0’. Following that, a postpone of one milli 2nd is given. in case you are using a motor of higher speed, lower the postpone.

Working of Project

This mission is using POV (endurance of vision). suppose we want to reveal alphabet A, then you can see as in determine given below, at a time LED’s of handiest one column are blinking.

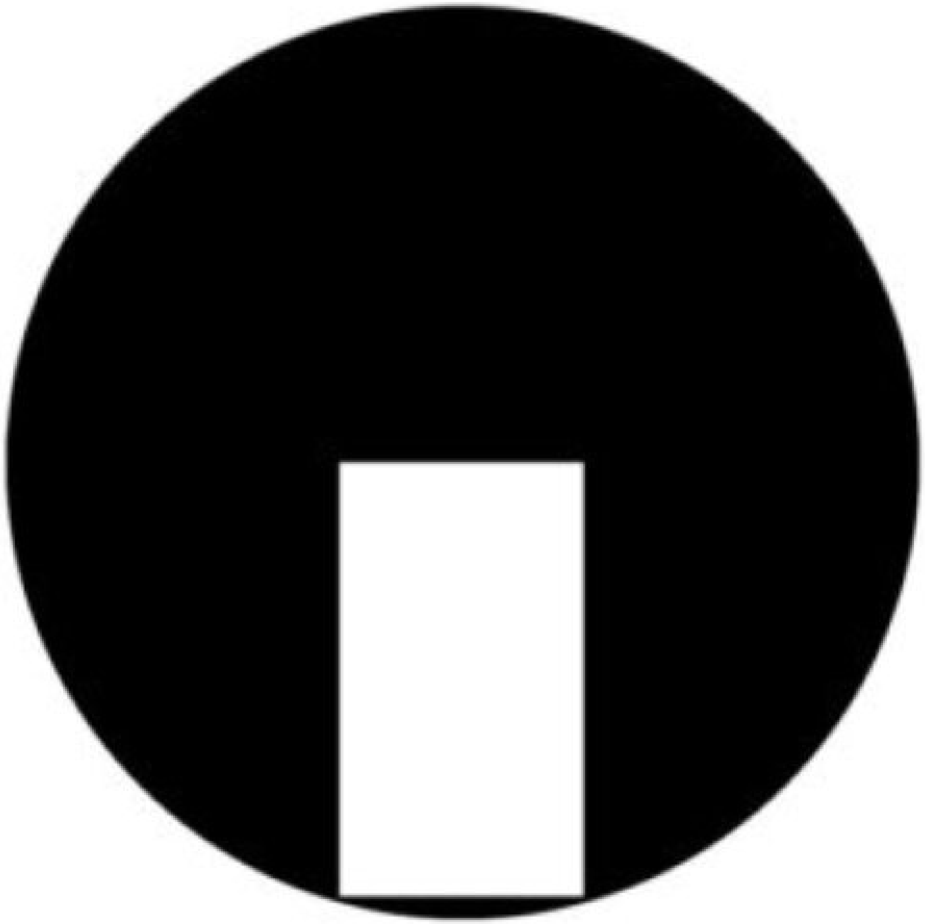


POV – Persistence of Vision

To begin with LED’s of column 1 blinks, and next time column 2 and so on. in the code, delay of some milli seconds should be carried out among the blinking of columns, and in our case it's miles one millisecond.

This whole state of affairs (blinking of all columns one after the other) makes a entire alphabet. speed of motor is excessive enough that our eyes cannot sense the blinking of LED’s (due to persistence of imaginative and prescient).

In the circuit, a couple of IR LED and a Photodiode is likewise used. when this pair pass over the white strip (as proven in figure beneath), the microcontroller (arduino) identifies this factor as the starting function and LED’s begin blinking according the textual content. on this way LED’s blinks in the same pattern all the time, so we will see the text displayed on it.



Troubleshooting

If the led’s aren't blinking while white paper is placed in front of the IR LED and photo diode, change the price in line 23 of code.

If led’s are blinking while black paper is positioned in the front of IR LED and PD, exchange the fee in line 23 of code.