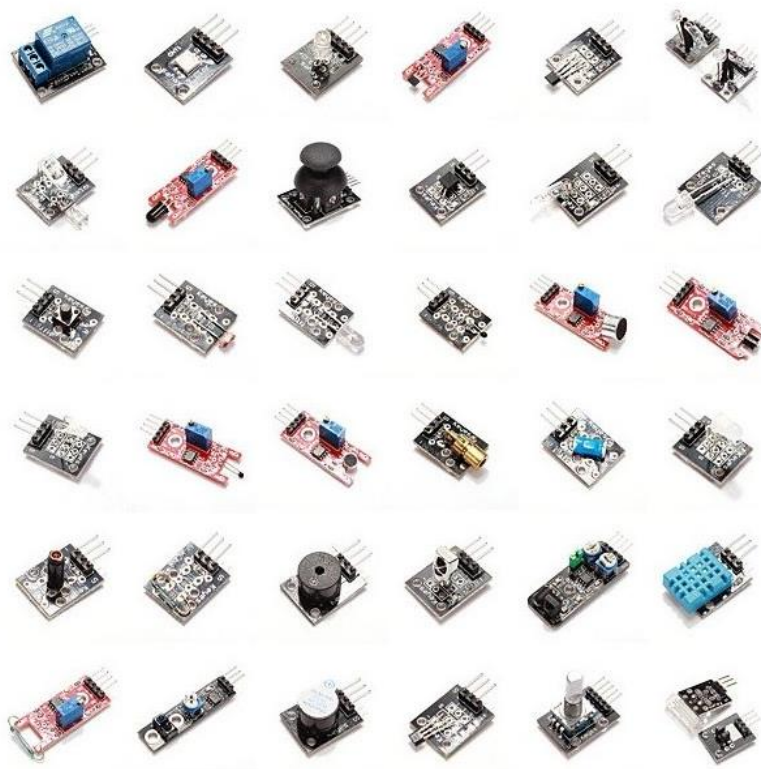


آموزش 37 کیت سنسور برای آردوینو و سایر میکروکنترلرها



[<< لینک خرید مستقیم این بسته آموزشی از فروشگاه الکترو ولت >>](#)

کاری از وبسایت الکترو ولت (Electrovolt.ir)

ویرایش اول - تابستان 1396

در این بسته آموزشی 37 سنسور و ماژول به شرح زیر وجود دارد :

1. JoyStick
2. Flame Sensor
3. RGB LED
4. Heartbeat Sensor
5. Light Cup
6. Hall Magnetic Sensor
7. Relay
8. Linear Hall Sensor
9. SMD RGB
10. 7 Color Flash
11. Tilt Switch
12. Temperature Sensor
13. Big Sound Sensor
14. Touch Sensor
15. Two-Color LED
16. Laser Emitter
17. Ball Switch
18. Analog Temperature Sensor
19. Small Sound Sensor
20. Digital Temperature Sensor
21. Two-Color LED (small)
22. Button
23. Photoresistor
24. IR Emission
25. Tracking Sensor
26. Buzzer
27. Reed Switch
28. Shock Sensor
29. Temperature And Humidity Sensor
30. IR Receiver
31. Avoidance Sensor
32. Passive Buzzer
33. Mini Reed
34. Rotary Encoders
35. Analog Hall Sensor
36. Tap Module
37. Light Blocking

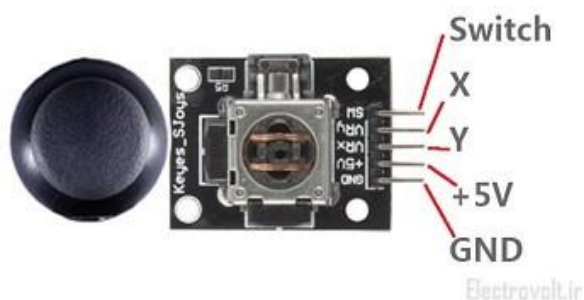
در این آموزش ، هر یک از سنسورها و ماژول های فوق را با هم بررسی می کنیم. در هر بخش ابتدا توضیحات کوتاهی راجع به آن ماژول یا سنسور ارائه خواهیم کرد سپس برنامه آردوینو مربوط به راه اندازی آن را خواهیم نوشت. توجه کنید در برخی از مثال ها یک LED روی پایه 13 وجود دارد.

بخش اول : JoyStick



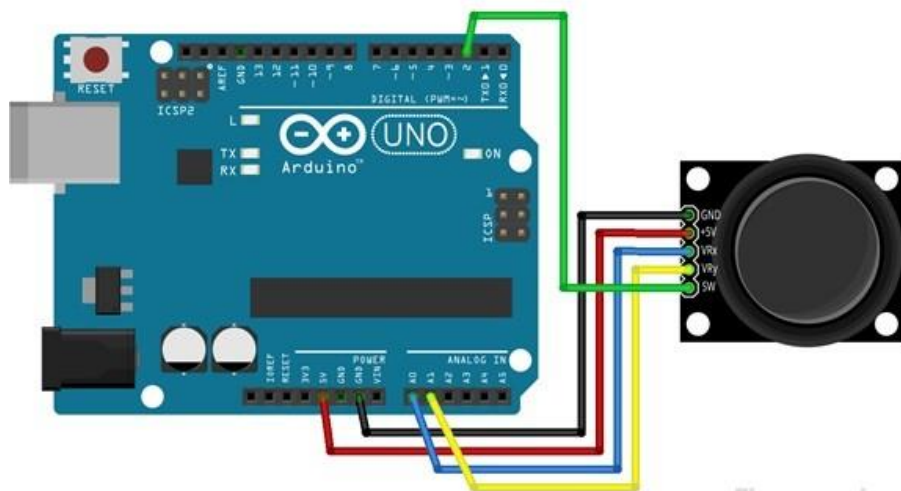
Electrovolt.ir

جوی استیک (JoyStick) یک سنسور آنالوگ می باشد که معمولاً برای کارهای کنترلی مورد استفاده قرار می گیرد. درون جوی استیک دو عدد پتانسیومتر (یکی در راستای محور X و دیگری در راستای محور Y) قرار دارد که با توجه به میزان منحرف شدن از مرکز مقدار مقاومت این دو پتانسیومتر تغییر می کند و با خواندن توسط واحد ADC آردوینو میتوان آن را به دیجیتال تبدیل کرد. همچنین درون جوی استیک یک عدد کلید (Push Button) نیز وجود دارد که در صورت فشردن جوی استیک آن کلید نیز فشرده می شود. در شکل زیر پایه های این سنسور را مشاهده می کنید.



Electrovolt.ir

نحوه اتصال جوی استیک به آردوینو:



Electrovolt.ir

برنامه آردوینو:

```
// Arduino pin numbers

const int SW_pin = 2; // digital pin connected to switch output
const int X_pin = 0; // analog pin connected to X output
const int Y_pin = 1; // analog pin connected to Y output

void setup() {
  pinMode(SW_pin, INPUT);
  digitalWrite(SW_pin, HIGH);
  Serial.begin(115200);
}

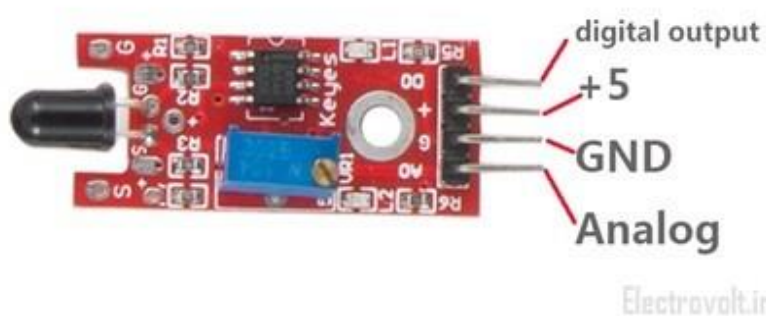
void loop() {
  Serial.print("Switch: ");
  Serial.print(digitalRead(SW_pin));
  Serial.print("\n");
  Serial.print("X-axis: ");
  Serial.print(analogRead(X_pin));
  Serial.print("\n");
  Serial.print("Y-axis: ");
  Serial.println(analogRead(Y_pin));
  Serial.print("\n\n");
  delay(500);
}
```

توضیح برنامه:

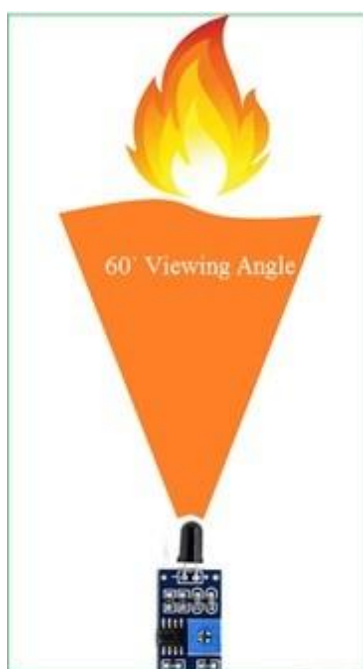
پایه SW به یکی از پایه های دیجیتال (پایه 2) وصل شده است. این پایه ورودی و پول آپ باید باشد. پایه های VRx و VRy به ترتیب به پایه های 1 و 2 آنالوگ متصل می گردند. درون حلقه مقدار های SW و VRx و VRy خوانده می شود و توسط Serial.print به کامپیوتر

ارسال می گردد. درون نرم افزار Arduino IDE با کلیک بر روی سریال مانیتور (Tools -> Serial Monitor) میتوانید تغییرات مقادیر خوانده شده توسط ADC را ببینید.

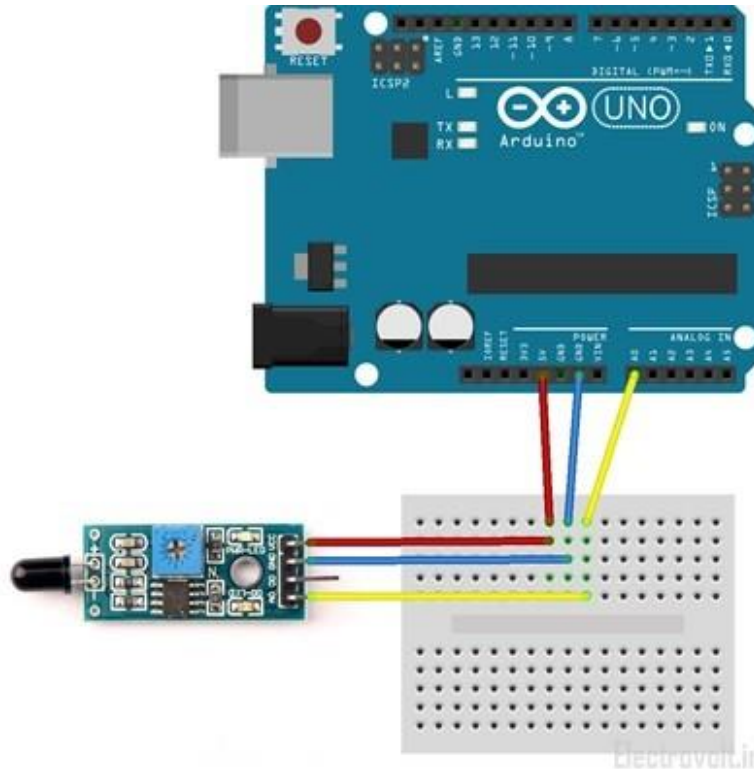
بخش دوم : Flame Sensor



این سنسور برای تشخیص شعله های آتش با فاصله نزدیک می باشد. که میتواند در پروژه هایی که برای نظارت و کنترل ایمنی آتش می باشد استفاده شود. درون این سنسور یک گیرنده مادون قرمز وجود دارد که نورهای با طول موج بین 760 تا 1100 نانومتر را تشخیص می دهد. در صورتی که سنسور آتش را تشخیص دهد پایه digital output آن 1 می شود. همچنین مقدار ولتاژ آنالوگ روی پایه Analog بسته به شدت آتش با رابطه عکس تغییر می کند. این سنسور محدوده عملکرد محدودی دارد که در شکل زیر مشاهده می کنید.



نحوه اتصال سنسور شعله به آردوینو:



برنامه آردوینو:

```
// lowest and highest sensor readings:
const int sensorMin = 0; // sensor minimum
const int sensorMax = 1024; // sensor maximum
void setup() {
// initialize serial communication @ 9600 baud:
Serial.begin(9600);
}

void loop() {
// read the sensor on analog A0:
int sensorReading = analogRead(A0);
```

```
// map the sensor range (four options):
// ex: 'long int map(long int, long int, long int, long int, long int)'
int range = map(sensorReading, sensorMin, sensorMax, 0, 3);

// range value:
switch (range) {
case 0: // A fire closer than 1.5 feet away.
Serial.println("*** Close Fire ***");
break;

case 1: // A fire between 1-3 feet away.
Serial.println("*** Distant Fire ***");
break;

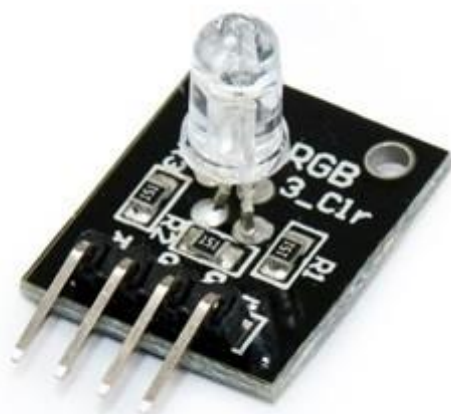
case 2: // No fire detected.
Serial.println("No Fire");
break;
}
delay(1); // delay between reads
}
```

توضیح برنامه:

در ابتدا یک عدد دلخواه برای مینیمم و ماکزیمم مقدار سنسور شعله تعیین می کنیم. سپس مقدار آنالوگ را از سنسور شعله می خوانیم. این مقدار آنالوگ را بوسیله تابع map به تفاضل مینیمم و ماکزیمم تقسیم کرده و به سه عدد 0، 1 یا 2 تبدیل می کنیم. (برای اطلاعات بیشتر راجع به تابع map به << [این لینک](#) >> مراجعه کنید). درنهایت در صورتی که مقدار نهایی سنسور 0 باشد یعنی آتش نزدیک سنسور است. در صورت 1 بودن یعنی آتش دور است (یا ضعیف است) و در صورتی 3 باشد یعنی آتشی در محدوده عملکرد سنسور وجود ندارد.

توجه: در روش دوم برنامه نویسی میتوان با خواندن پایه دیجیتال تنها وجود یا عدم وجود شعله را تشخیص داد.

بخش سوم : RGB LED



Electrovolt.ir

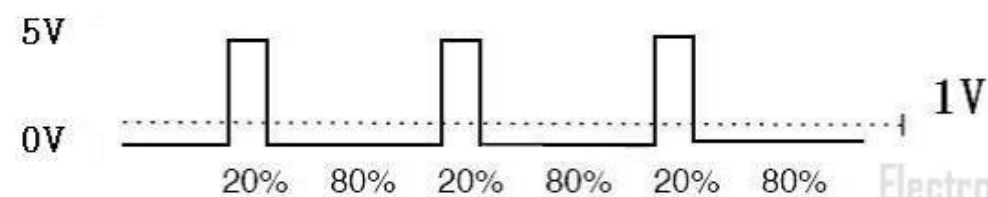
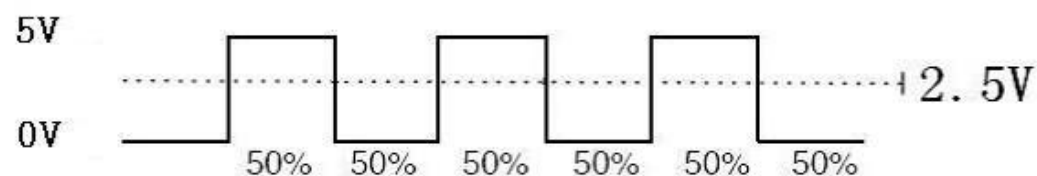
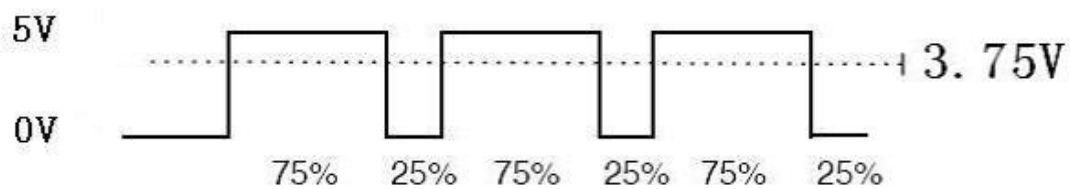
در این ماژول یک دیود نورانی (LED) سه رنگ (قرمز ، سبز ، آبی) وجود دارد. در صورتی که این سه رنگ با شدت های مختلف با هم ترکیب شوند کلیه رنگ های طبیعی موجود را تشکیل می دهد. شکل زیر پایه های این ماژول را نشان می دهد.



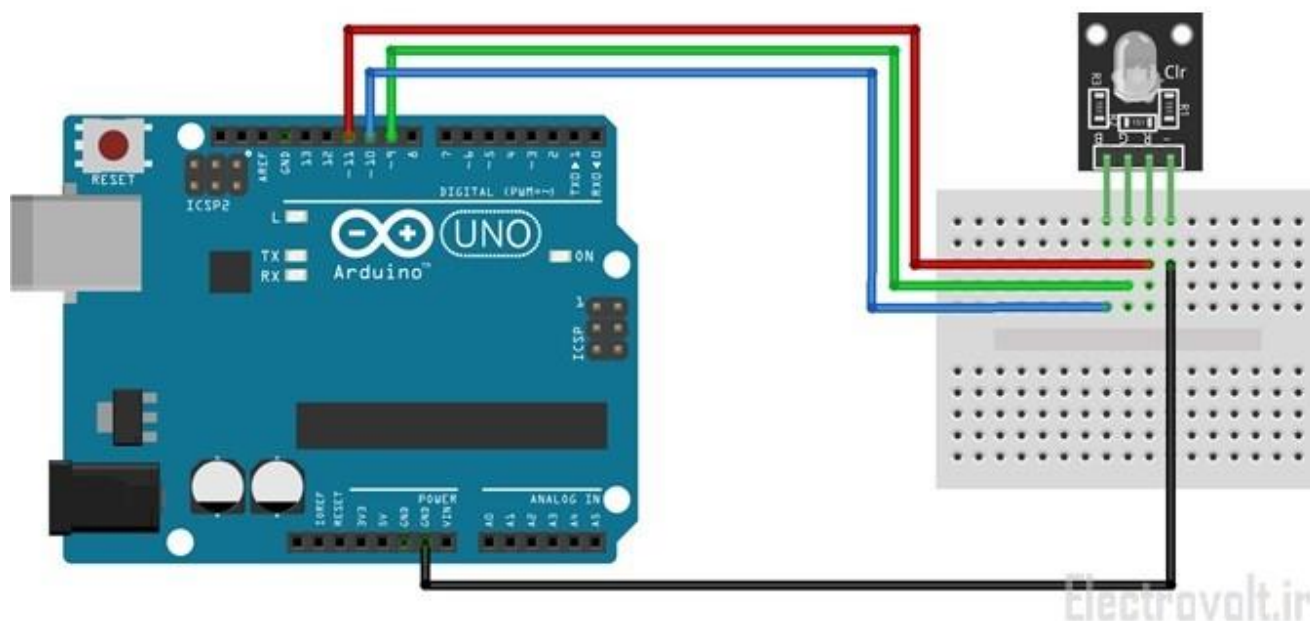
DIP Package

Electrovolt.ir

میزان درخشندگی پایه های R,G,B هر سه توسط PWM کنترل می شود. PWM مخفف Pulse Width Modulation به معنای مدولاسیون عرض پالس می باشد. در این روش برای کنترل ولتاژ اعمالی به LED ها از یک پالس دیجیتال استفاده می گردد که دارای یک فرکانس ثابت اما عرض پالس متغیر می باشد. توسط پالس PWM میتوان میزان توان اعمالی به LED را کنترل کرد. به طوری که هر چه عرض پالس بیشتر شود ولتاژ معادل پالس و در نتیجه توان اعمالی آن بزرگتر می شود. شکل زیر سه پالس PWM با پهنای پالس مختلف و ولتاژ معادل آن ها را نشان می دهد.



نحوه اتصال به آردوینو:



```
//KY016 3-color LED module  
int redpin = 11; // select the pin for the red LED  
int bluepin = 10; // select the pin for the blue LED  
int greenpin = 9 ;// select the pin for the green LED  
int val;
```

```
void setup () {  
  pinMode (redpin, OUTPUT);  
  pinMode (bluepin, OUTPUT);  
  pinMode (greenpin, OUTPUT);  
  Serial.begin (9600);  
}
```

```
void loop () {  
  for (val = 255; val > 0; val --)  
  {  
    analogWrite (11, val);  
    analogWrite (10, 255-val);  
    analogWrite (9, 128-val);  
    delay (10);  
    Serial.println (val, DEC);  
  }  
  for (val = 0; val < 255; val ++)  
  {  
    analogWrite (11, val);  
    analogWrite (10, 255-val);
```

```

analogWrite (9, 128-val);

delay (10);

Serial.println (val, DEC);

}

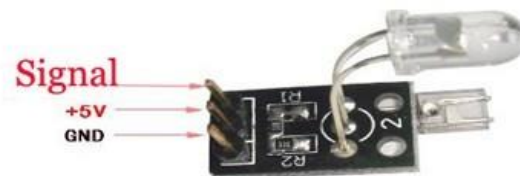
}

```

توضیح برنامه:

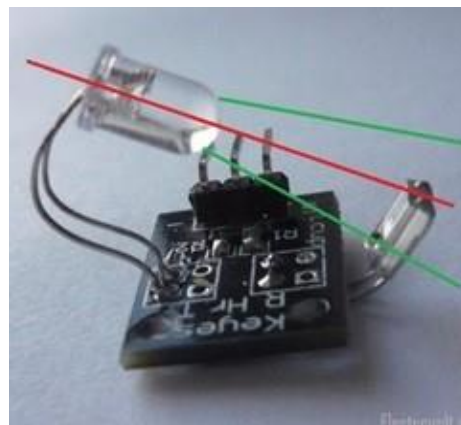
ابتدا هر سه پین R و G و B را خروجی می کنیم. سپس توسط تابع analogWrite میتوان یک موج PWM تولید کرده و به پایه مورد نظر داد. ورودی این تابع میتواند عددی بین 0 تا 255 باشد. هر چه عدد به 255 نزدیک تر باشد رنگ مورد نظر پر رنگ تر می گردد. برای تغییر رنگ از حلقه for و متغیر حلقه (var) استفاده شده است که میتواند انواع رنگ ها را ایجاد کند.

بخش چهارم : Heartbeat Sensor

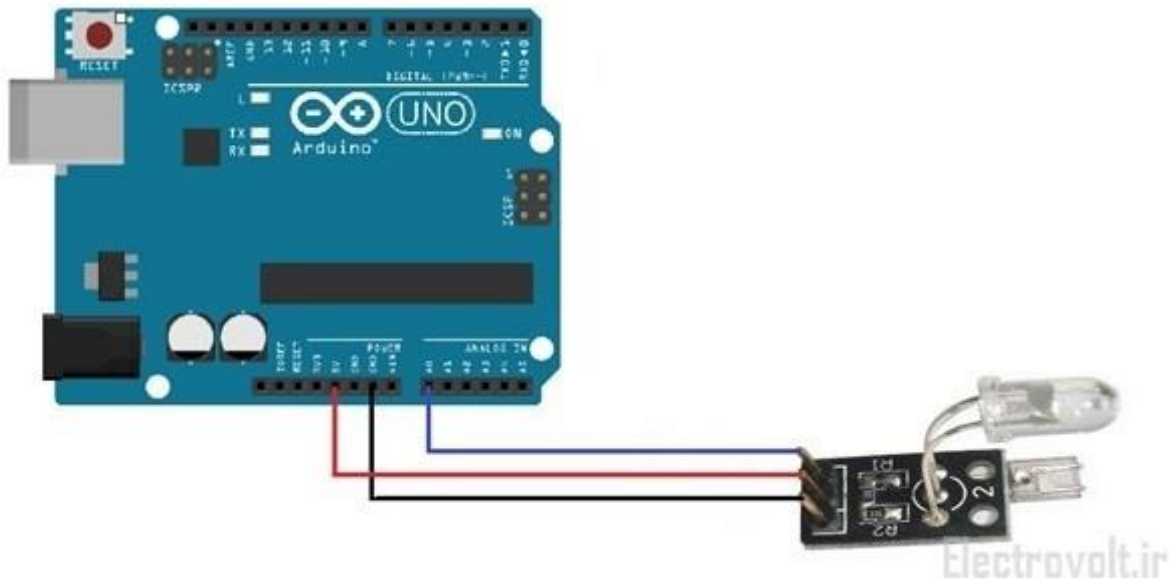


Electrovolt.ir

این سنسور همانطور که از نام آن پیداست برای تشخیص ضربان قلب از روی انگشت استفاده می شود. درون این سنسور یک فرستنده مادون قرمز و یک گیرنده فوتو ترانزیستور استفاده شده است. با تغییر فشار خون مقدار خروجی سنسور نیز تغییر می کند. فرستنده باید در یک طرف انگشت و گیرنده در طرف دیگر انگشت قرار گیرد و دقیقا باید روبروی هم قرار گیرند. شکل زیر این موضوع را نشان می دهد.



نحوه اتصال به آردوینو:



برنامه آردوینو:

```
// Pulse Monitor Test Script
```

```
int sensorPin = 0;
```

```
double alpha = 0.75;
```

```
int period = 100;
```

```
double change = 0.0;
```

```
double minval = 0.0;
```

```
void setup ()
```

```
{
```

```
  Serial.begin (9600);
```

```
}
```

```
void loop ()
```

```
{
```

```
  Electrovolt.ir
```

```
static double oldValue = 0;

static double oldChange = 0;

int rawValue = analogRead (sensorPin);

double value = alpha * oldValue + (1 - alpha) * rawValue;

Serial.print (rawValue);

Serial.print (",");

Serial.println (value);

oldValue = value;

delay (period);

}
```

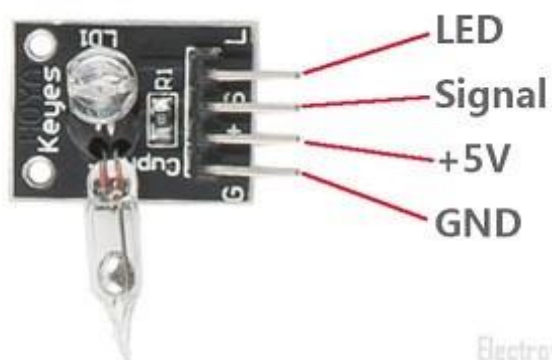
توضیح پر نامہ:

در این برنامه ابتدا مقدار جدیدی از پایه آنالوگ سنسور خوانده می شود و در متغیر `rawValue` ذخیره می گردد. این مقدار با مقدارهای قبلی خوانده شده طبق فرمول موجود جمع شده و درون `Value` ریخته می شود. (نوعی عمل میانگین گیری در هر مرحله صورت می گیرد). در نهایت مقدار `rawValue` و `Value` توسط `Serial.print` به کامپیوتر ارسال می گردد. درون نرم افزار `Arduino IDE` با کلیک بر روی سریال مانیتور (`Tools -> Serial Monitor`) میتوانید تغییرات مقادیر خوانده شده توسط `ADC` را ببینید. هر چه زمان بیشتری بگذرد ، مقدار نشان داده شده به ضربان قلب واقعیتان نزدیک تر می گردد.

Magic Light Cup : بخش پنجم

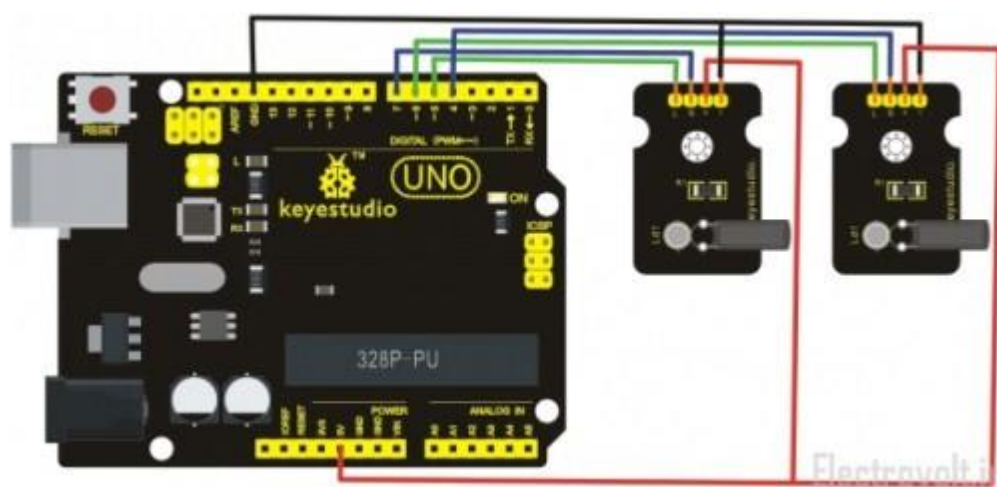


دو عدد ماژول مجیک لایت کاپ (یا جام نور جادویی) مشابه هم در این بسته آموزشی وجود دارد. پایه های این ماژول به شرح زیر است:



هر ماژول از دو بخش تشکیل شده است ، یک LED معمولی قرمز رنگ و یک حباب کوچک که درون آن جیوه وجود دارد. جیوه داخل حباب باعث می شود که همانند یک سوئیچ عمل کند که در زمان حرکت قطع و وصل می شود. این سوئیچ به آردوینو متصل می شود و هنگام هر تکان خوردن چراغ LED را با نور بیشتر یا کمتری روشن می کند.

نحوه اتصال به آردوینو:



برنامه آردوینو:

```
int LedPinA = 5;  
int LedPinB = 6;  
int ButtonPinA = 7;  
int ButtonPinB = 4;
```

```
int buttonStateA = 0;
int buttonStateB = 0;
int brightness = 0;
void setup ()
{
  pinMode (LedPinA, OUTPUT);
  pinMode (LedPinB, OUTPUT);
  pinMode (ButtonPinA, INPUT);
  pinMode (ButtonPinB, INPUT);
}
void loop ()
{
  buttonStateA = digitalRead (ButtonPinA);
  if (buttonStateA == HIGH && brightness! = 255)
  {
    brightness + +;
  }
  buttonStateB = digitalRead (ButtonPinB);
  if (buttonStateB == HIGH && brightness! = 0)
  {
    brightness -;
  }
  analogWrite (LedPinA, brightness); // A few Guan Yuan (ii) ?
  analogWrite (LedPinB, 255 - brightness); // B Yuan (ii) a few Bang ?
  delay (25);
}
```

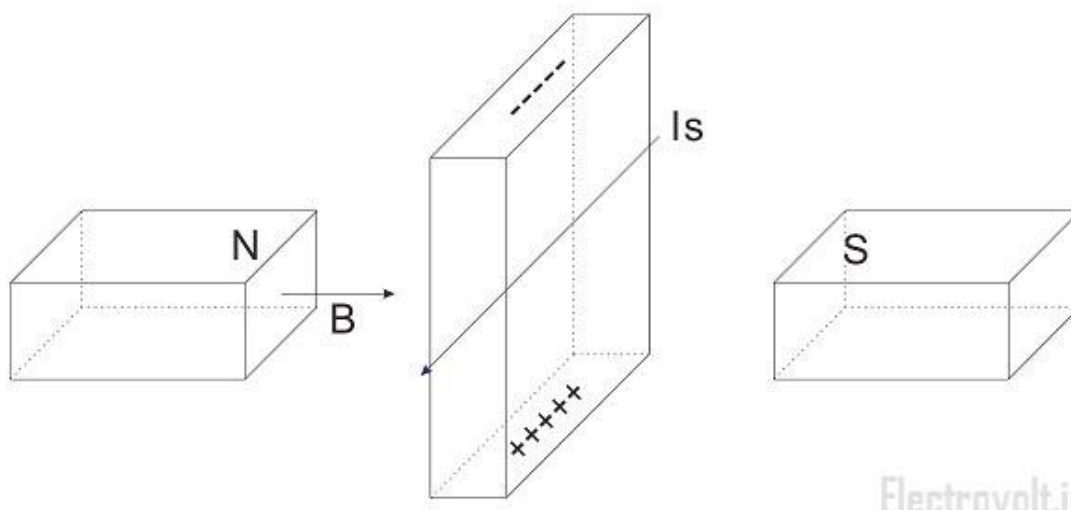
توضیح برنامه:

در این برنامه دو سنسور لایت کاپ متصل شده است (A و B) که در صورت تکان خوردن A ، نور LEDA زیاد و LEDB کم می شود. در صورت تکان خوردن دومی ، برعکس حالت قبل اتفاق می افتد. یعنی نور LEDA کم و نور LEDB زیاد می شود.

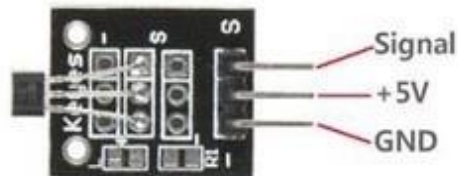
بخش ششم : Switch Hall Magnetic Sensor



اثر هال یک اثر الکترومغناطیسی است که توسط Edwin Hall در سال 1879 کشف شد. طبق این اثر اگر یک هادی در میدان مغناطیسی قرار گیرد ، از طرف میدان مغناطیسی نیرویی وارد بر الکترون های آن هادی می شود که موجب حرکت آن ها و بوجود آمدن اختلاف پتانسیل (ولتاژ) می گردد. همانطور که در شکل زیر مشاهده می کنید ، در یک طرف نیمه هادی قطب N و در طرف دیگر قطب S قرار دادیم و مشاهده می شود که بارهای مثبت و منفی (الکترون) در دو سمت هادی جمع شده و باعث بوجود آمدن اختلاف پتانسیل در هادی شده است.



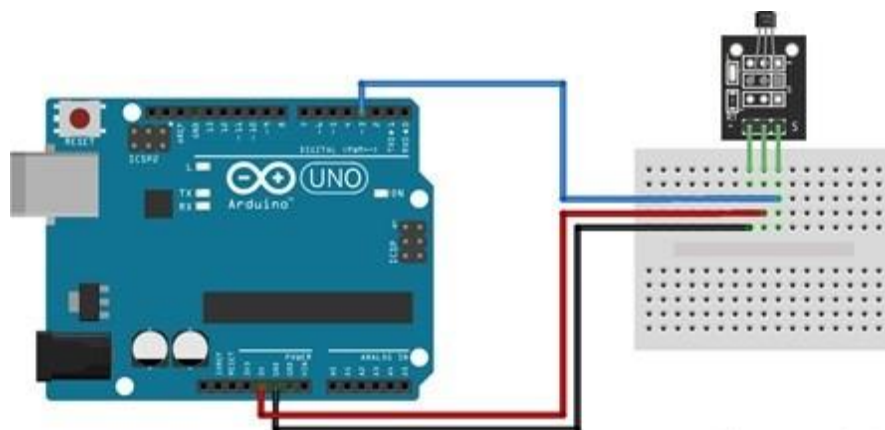
سنسور اثر هال مدل 3144 یک سنسور اثر هال سه پایه از نوع سوئیچی (Switch) است. روی این سنسور عدد 3144 نوشته شده است. اگر هیچ میدانی وجود نداشته باشد ، خروجی سنسور Low می شود و در صورت وجود میدان مغناطیسی قوی اطراف سنسور خروجی آن High می شود. یک LED نیز روی برد قرار دارد که در صورت High شدن ، روشن می گردد. شکل زیر پایه های این سنسور را نشان می دهد.



Switch Hall

Electrovolt.ir

اتصال به آردوینو:



Electrovolt.ir

برنامه آردوینو:

```
/*
KY-003 Hall magnetic switch
*/
int Led = 13 ; // define LED Interface
int SENSOR = 10 ; // define the Hall magnetic sensor interface
int val ; // define numeric variables val

void setup ()
{
```

Electrovolt.ir

```

pinMode (Led, OUTPUT) ; // define LED as output interface
pinMode (SENSOR, INPUT) ; // define the Hall magnetic sensor line as input
}
void loop ()
{
val = digitalRead (SENSOR) ; // read sensor line
if (val == LOW) // when the Hall sensor detects a magnetic field, Arduino LED lights up
{
digitalWrite (Led, HIGH);
}
{
digitalWrite (Led, LOW);
}
}

```

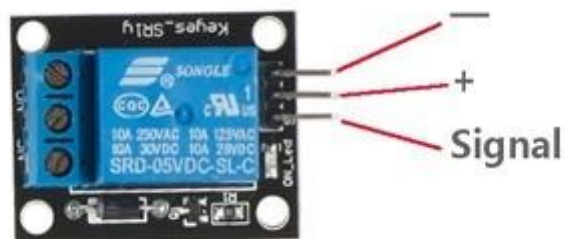
توضیح برنامه:

زمانی که میدان مغناطیسی تشخیص داده شود ، LED روشن و در غیر این صورت خاموش است.

بخش هفتم : Relay

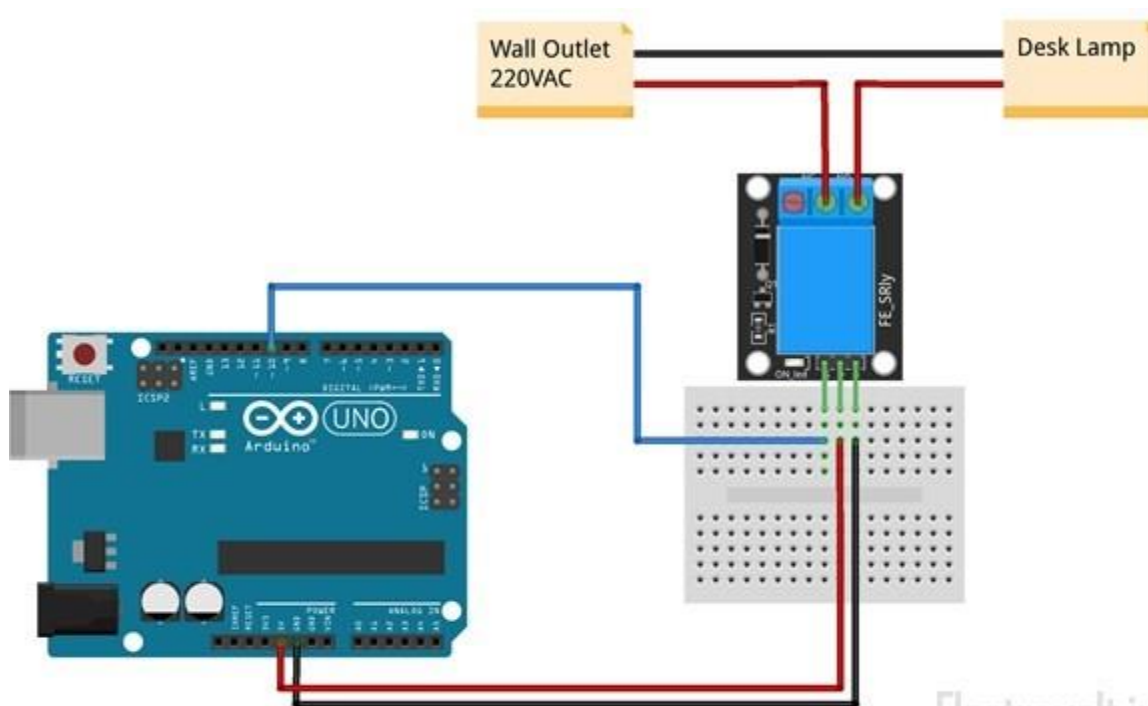


رله نوعی سوئیچ است که میتواند بوسیله ولتاژ پایین (مثلا 5 ولت) ، قطع/وصل بودن ولتاژ بالا (نظیر برق 220 ولت) را کنترل نماید. اگر به طرف ولتاژ پایین رله ولتاژ 5 ولت اعمال شود کنتاکت درون رله برقرار شده و باعث اتصال کوتاه شدن طرف دیگر رله می شود.



Electrovolt.ir

نحوه اتصال رله به آردوینو:



Electrovolt.ir

برنامه آردوینو:

```
//KY019 5V relay module
```

```
int relay = 10; // relay turns trigger signal - active high;
```

Electrovolt.ir

```

void setup ()
{
pinMode (relay, OUTPUT); // Define port attribute is output;
}

void loop ()
{
digitalWrite (relay, HIGH); // relay conduction;

delay (1000);

digitalWrite (relay, LOW); // relay switch is turned off;

delay (1000);
}

```

توضیح برنامه:

هر یک ثانیه رله روشن و خاموش می گردد.

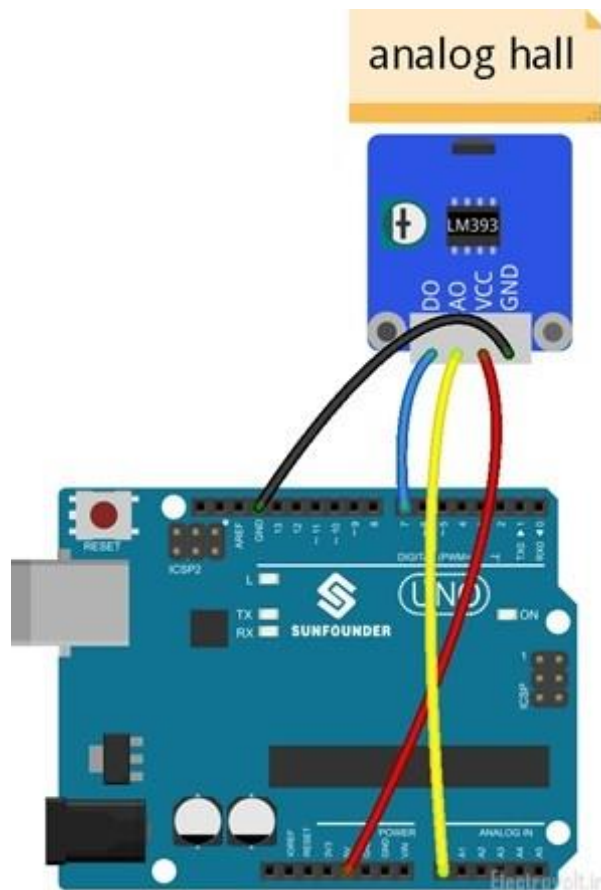
بخش هشتم : Linear Hall Sensor



عملکرد این سنسور همانند بخش ششم می باشد با این تفاوت که علاوه بر خروجی دیجیتال ، دارای خروجی آنالوگ نیز می باشد. در این ماژول از سنسور اثر هال مدل E 49 به همراه مدار جانبی استفاده شده است. این سنسور در هنگام قرار گرفتن در میدان مغناطیسی ولتاژ آنالوگ متناسب با قدرت آن میدان تولید می کند. ولتاژ آنالوگ خروجی پس از دریافت توسط آردوینو به دیجیتال تبدیل می گردد. در صورتی که میدان قوی در اطراف سنسور شناسایی شود ، LED موجود روی آن روشن می گردد. در این ماژول با استفاده از یک مقایسه

کننده (آپ امپ) مدل LM324 علاوه بر خروجی آنالوگ ، خروجی دیجیتال نیز تعبیه شده است. یک پتانسیومتر برای کنترل حساسیت خروجی دیجیتال نیز وجود دارد.

نحوه اتصال به آردوینو:



برنامه آردوینو:

```
//Analog Hall Sensor
const int ledPin = 13;//the led attach to pin13
int sensorPin = A0; // select the input pin for the potentiometer
int digitalPin=7; //DO attach to pin7
int sensorValue = 0;// variable to store the value coming from A0
boolean digitalValue=0;// variable to store the value coming from pin7
void setup()
{
```

```

pinMode(digitalPin,INPUT);//set the state of D0 as INPUT
pinMode(ledPin,OUTPUT);//set the state of pin13 as OUTPUT
Serial.begin(9600); // initialize serial communications at 9600 bps
}

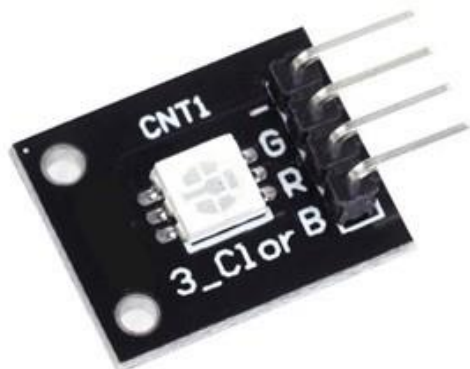
void loop()
{
sensorValue = analogRead(sensorPin); //read the value of A0
digitalValue=digitalRead(digitalPin); //read the value of D0
Serial.print("Sensor Value "); // print label to serial monitor
Serial.println(sensorValue); //print the value of A0
Serial.print("Digital Value "); // print label to serial monitor
Serial.println(digitalValue); //print the value of D0 in the serial
if( digitalValue==HIGH )//if the value of D0 is HIGH
{
digitalWrite(ledPin,LOW);//turn off the led
}
if( digitalValue==LOW)//else
{
digitalWrite(ledPin,HIGH);//turn on the led
}
delay(1000);//delay 200ms
}

```

توضیح برنامه:

مقدار آنالوگ سنسور خوانده شده و پس از تبدیل به دیجیتال ، مقدار عددی آن نمایش داده می شود. همچنین زمانی که میدان مغناطیسی قوی اطراف سنسور تشخیص داده شود ، LED روشن و در غیر این صورت خاموش است. درون نرم افزار Arduino IDE با کلیک بر روی سریال مانیتور (Tools -> Serial Monitor) میتوانید تغییرات مقادیر خوانده شده سنسور ها را توسط آردوینو را ببینید.

بخش نهم : SMD RGB LED



Electrovolt.ir

دو عدد ماژول RGB در این بسته آموزشی وجود دارد که اولی در بخش سوم بررسی شد. تفاوت این دو ماژول تنها در بسته بندی (Package) آن ها است. به طوری که اولی پایه دار و دومی روی سطح برد (یا SMD) می باشد.

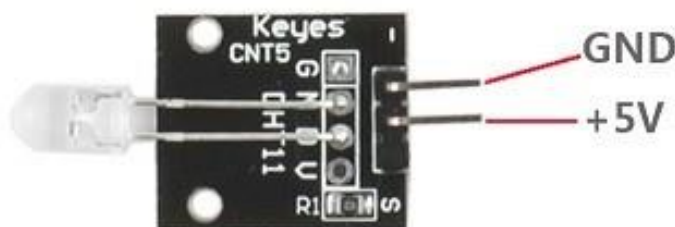
نحوه اتصال به آردوینو:

همانند بخش سوم

برنامه آردوینو:

همانند بخش سوم

بخش دهم : Color Flash LED7



Electrovolt.ir

این ماژول دارای یک LED هفت رنگ می باشد که به طور خودکار تغییر رنگ می دهد. برای راه اندازی این ماژول کافی است پایه VCC آن را به پایه 13 آردوینو متصل کنیم.

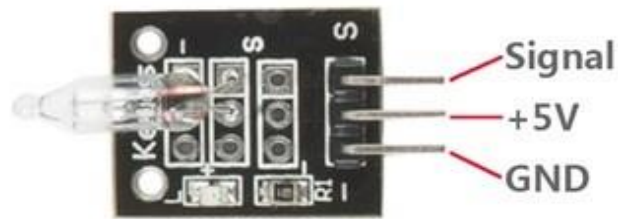
برنامه آردوینو:

```
/*  
Blink  
Turns on an LED on for two second, then off for two second, repeatedly.  
This example code is in the public domain.  
*/  
  
void setup () {  
  
// Initialize the digital pin as an output.  
// Pin 13 has an LED connected on most Arduino boards:  
pinMode (13, OUTPUT);  
  
}  
  
void loop () {  
  
digitalWrite (13, HIGH); // set the LED on  
delay (2000); // wait for a second  
digitalWrite (13, LOW); // set the LED off  
delay (2000); // wait for a second  
  
}
```

توضیح برنامه:

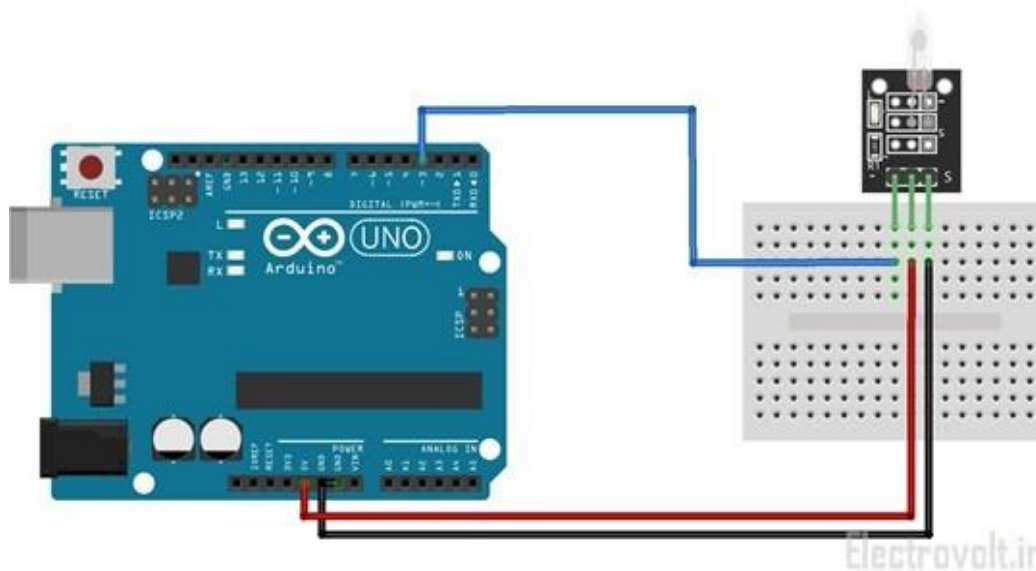
LED به مدت دو ثانیه روشن و سپس دو ثانیه خاموش می شود و...

بخش یازدهم : Mercury Tilt Switch



با این سوئیچ قبلاً در بخش پنجم آشنا شده بودیم. تیلت سوئیچ شامل یک حباب است که داخل آن جیوه وجود دارد. جاذبه زمین باعث کشیده شدن جیوه به پایین ترین سطح ممکن و تماس آن با کنتاکت های موجود می گردد. بنابراین این سوئیچ هنگام حرکت و تغییر زاویه قطع و وصل می گردد.

نحوه اتصال به آردوینو:



برنامه آردوینو:

```
//KY017 Mercury open optical module
int Led = 13 ;// define LED Interface
int buttonpin = 3; // define the mercury tilt switch sensor interface
int val ;// define numeric variables val

void setup ()
{
```

```

pinMode (Led, OUTPUT) ;// define LED as output interface

pinMode (buttonpin, INPUT) ;// define the mercury tilt switch sensor output interface
}

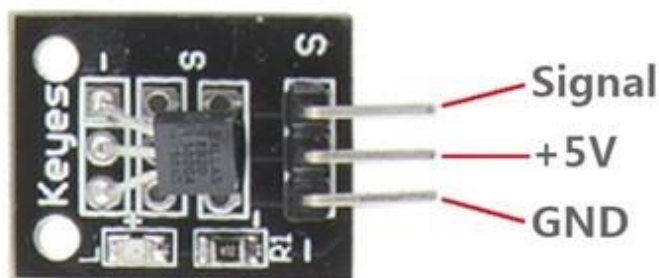
void loop ()
{
val = digitalRead (buttonpin) ;// read the values assigned to the digital interface 3 val
if (val == HIGH) // When the mercury tilt switch sensor detects a signal, LED flashes
{
digitalWrite (Led, HIGH);
}
else
{
digitalWrite (Led, LOW);
}
}

```

توضیح برنامه:

سنسور تیلت سوئیچ را تکان دهید، به محض اینکه سوئیچ بسته باشد، LED موجود روی پایه 13 روشن می گردد.

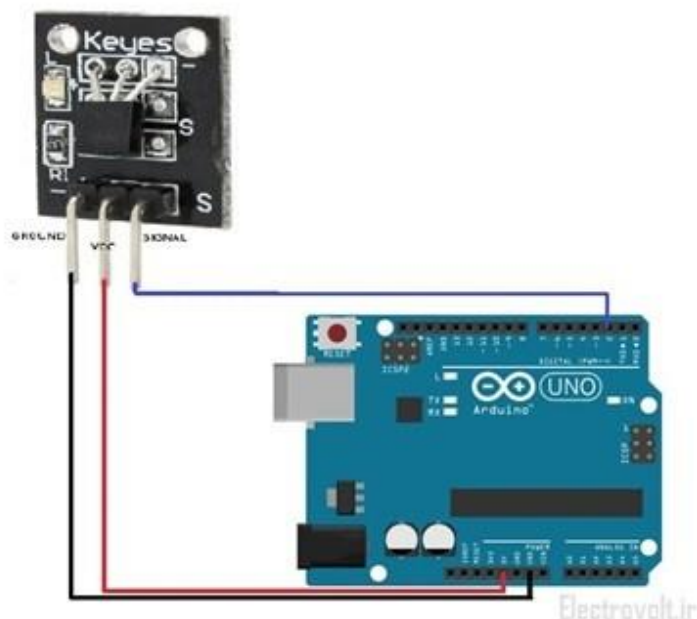
بخش دوازدهم : Temperature Sensor



Electrovolt.ir

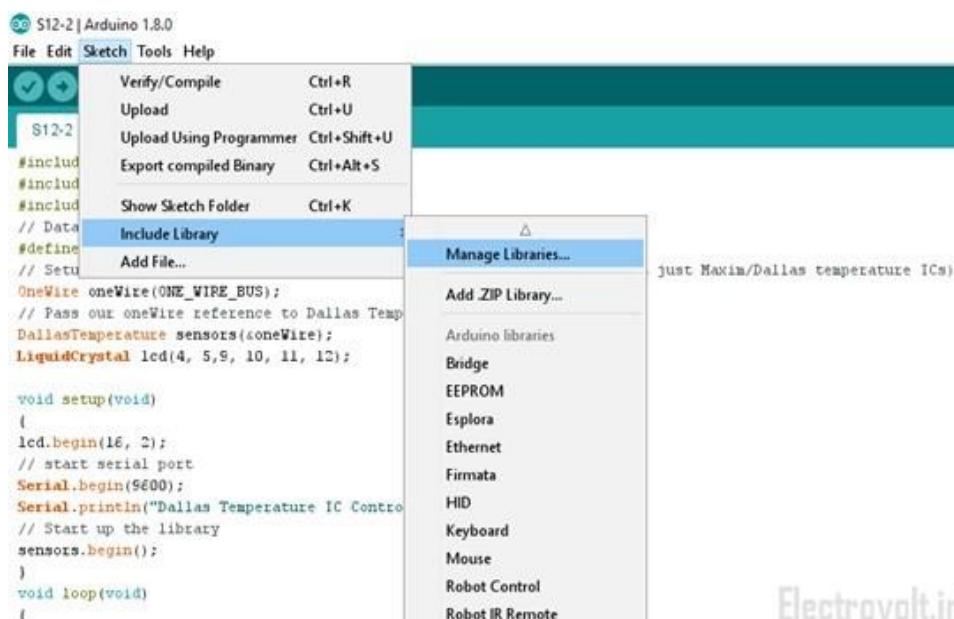
این ماژول یک سنسور دمای دیجیتال مدل DS18B20 می باشد که از پروتکل سریال 1 wire (یک سیمه) استفاده می کند. ابعاد این سنسور بسیار کوچک بوده و میتواند مقدار دما را با دقت 16 Bit (دو بایت) ارسال کند.

نحوه اتصال به آردوینو:



برنامه آردوینو:

برای خواندن دما توسط این سنسور دو نوع برنامه مختلف میتوان نوشت. اولی با استفاده از کتابخانه OneWire و دومی با استفاده از کتابخانه DallasTemperature می باشد. توجه داشته باشید که برای استفاده از این برنامه ها باید ابتدا کتابخانه مربوطه را با سرچ کردن نام آن در منوی Manage Library نصب نمایید. (شکل زیر)



```
#include <OneWire.h>

// DS18S20 Temperature chip i/o
OneWire ds(10); // on pin 10

void setup(void) {

// initialize inputs/outputs
// start serial port
Serial.begin(9600);

}

void loop(void) {

//For conversion of raw data to C
int HighByte, LowByte, TReading, SignBit, Tc_100, Whole, Fract;
byte i;
byte present = 0;
byte data[12];
byte addr[8];

if ( !ds.search(addr)) {

Serial.print("No more addresses.\n");
ds.reset_search();

Electrovolt.ir
```

```
return;
```

```
}
```

```
Serial.print("R=");
```

```
for( i = 0; i < 8; i++) {
```

```
Serial.print(addr[i], HEX);
```

```
Serial.print(" ");
```

```
}
```

```
if ( OneWire::crc8( addr, 7) != addr[7]) {
```

```
Serial.print("CRC is not valid!\n");
```

```
return;
```

```
}
```

```
if ( addr[0] == 0x10) {
```

```
Serial.print("Device is a DS18S20 family device.\n");
```

```
}
```

```
else if ( addr[0] == 0x28) {
```

```
Serial.print("Device is a DS18B20 family device.\n");
```

```
}
```

```
else {
```

```
Serial.print("Device family is not recognized: 0x");
```

```
Serial.println(addr[0],HEX);
```

```
return;
```

```
}
```

```
ds.reset();
```

```
ds.select(addr);
```

```
ds.write(0x44,1);    // start conversion, with parasite power on at the end
```

```
delay(1000);  // maybe 750ms is enough, maybe not
```

```
// we might do a ds.depower() here, but the reset will take care of it.
```

```
present = ds.reset();
```

```
ds.select(addr);
```

```
ds.write(0xBE);    // Read Scratchpad
```

```
Serial.print("P=");
```

```
Serial.print(present,HEX);
```

```
Serial.print(" ");
```

```
for ( i = 0; i < 9; i++) {    // we need 9 bytes
```

```
data[i] = ds.read();
```

Electrovolt.ir

```
Serial.print(data[i], HEX);
```

```
Serial.print(" ");
```

```
}
```

```
Serial.print(" CRC=");
```

```
Serial.print( OneWire::crc8( data, 8), HEX);
```

```
Serial.println();
```

```
//Conversion of raw data to C
```

```
LowByte = data[0];
```

```
HighByte = data[1];
```

```
TReading = (HighByte << 8) + LowByte;
```

```
SignBit = TReading & 0x8000; // test most sig bit
```

```
if (SignBit) // negative
```

```
{
```

```
TReading = (TReading ^ 0xffff) + 1; // 2's comp
```

```
}
```

```
Tc_100 = (6 * TReading) + TReading / 4; // multiply by (100 * 0.0625) or 6.25
```

```
Whole = Tc_100 / 100; // separate off the whole and fractional portions
```

```
Fract = Tc_100 % 100;
```

```
if (SignBit) // If its negative
```

```
{
```

```
Electrovolt.ir
```

```
Serial.print("-");
```

```
}
```

```
Serial.print(Whole);
```

```
Serial.print(".");
```

```
if (Fract < 10)
```

```
{
```

```
Serial.print("0");
```

```
}
```

```
Serial.print(Fract);
```

```
Serial.print("\n");
```

```
//End conversion to C
```

```
}
```

برنامه دوم:

```
#include <OneWire.h>
```

```
#include <DallasTemperature.h>
```

```
#include <LiquidCrystal.h>
```

```
// Data wire is plugged into port 2 on the Arduino
```

```
#define ONE_WIRE_BUS 2
```

Electrovolt.ir

```
// Setup a oneWire instance to communicate with any OneWire devices (not just Maxim/Dallas temperature ICs)
```

```
OneWire oneWire(ONE_WIRE_BUS);
```

```
// Pass our oneWire reference to Dallas Temperature.
```

```
DallasTemperature sensors(&oneWire);
```

```
LiquidCrystal lcd(4, 5, 9, 10, 11, 12);
```

```
void setup(void)
```

```
{
```

```
  lcd.begin(16, 2);
```

```
  // start serial port
```

```
  Serial.begin(9600);
```

```
  Serial.println("Dallas Temperature IC Control Library Demo");
```

```
  // Start up the library
```

```
  sensors.begin();
```

```
}
```

```
void loop(void)
```

```
{
```

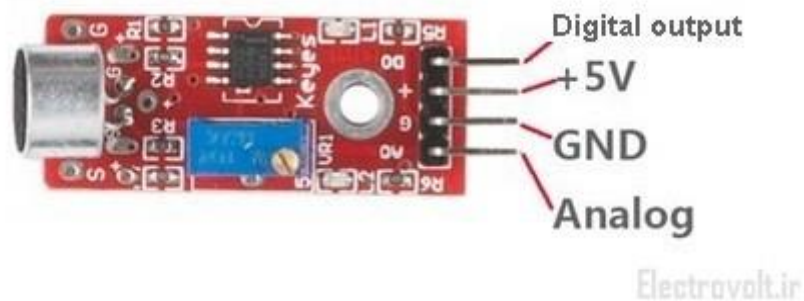
```
  // call sensors.requestTemperatures() to issue a global temperature
```

```
// request to all devices on the bus
Serial.print("Requesting temperatures...");

sensors.requestTemperatures(); // Send the command to get temperatures
Serial.println("DONE");

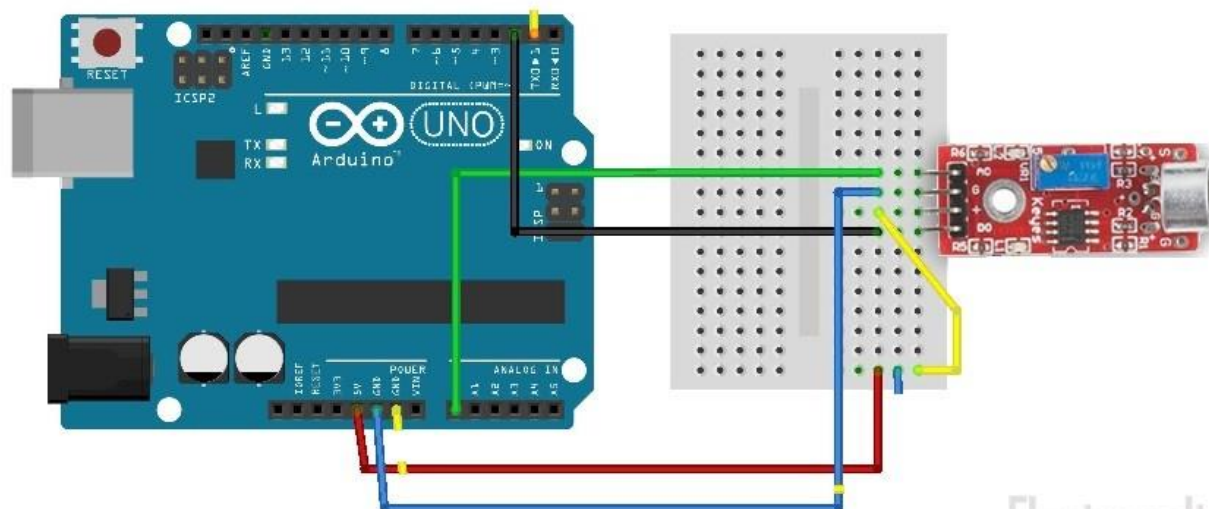
lcd.setCursor(0, 0);
lcd.print("Tem: ");
lcd.print(sensors.getTempCByIndex(0)); //print the temperature on lcd1602
lcd.print(" C");
lcd.setCursor(0, 1);
lcd.print("Tem: ");
lcd.print(sensors.getTempCByIndex(0) + 273.15); //print the temperature on lcd1602
lcd.print(" F");
Serial.print("Temperature for the device 1 (index 0) is: ");
Serial.println(sensors.getTempCByIndex(0)); //print the temperature on serial monitor
}
```

بخش سیزدهم : Big Sound Sensor



این سنسور میتواند صداهای بلند را تشخیص دهد. روی این سنسور یک میکروفن قرار دارد. میکروفن میتواند سیگنال صوتی را به سیگنال الکتریکی تبدیل کند. این سنسور دارای هر دو خروجی آنالوگ و دیجیتال می باشد. میکروفن صداهای محیط را به صورت مستقیم به خروجی آنالوگ می برد. اما زمانی که قدرت صدا از یک سطح مشخصی بالاتر رود ، پایه دیجیتال خروجی High می گردد. یک پتانسیومتر جهت تنظیم حساسیت خروجی دیجیتال نیز تعبیه شده است.

نحوه اتصال به آردوینو:



Electrovolt.ir

برنامه آردوینو اول برای استفاده از پایه دیجیتال:

```
int Led = 13 ;// define LED Interface
int buttonpin = 3; // define DO Sensor Interface
int val = 0; // define numeric variables val

void setup ()
{
  pinMode (Led, OUTPUT) ;// define LED as output interface
  pinMode (buttonpin, INPUT) ;// output interface DO is defined sensor
}
```

```

void loop ()
{
  val = digitalRead(buttonpin); // digital interface will be assigned a value of pin 3 to read val
  if (val == HIGH) // When the sound detection module detects a signal, LED flashes
  {
    digitalWrite (Led, HIGH);
  }
  else
  {
    digitalWrite (Led, LOW);
  }
}

```

توضیح برنامه:

به محض شناسایی صدای بلند ، LED روشن می گردد.

برنامه آردوینو دوم برای استفاده از پایه آنالوگ:

```

int sensorPin = A0; // select the input pin for the potentiometer
int ledPin = 13; // select the pin for the LED
int sensorValue = 0; // variable to store the value coming from the sensor

```

```

void setup ()
{
  pinMode (ledPin, OUTPUT);
  Serial.begin (9600);
}

```

```

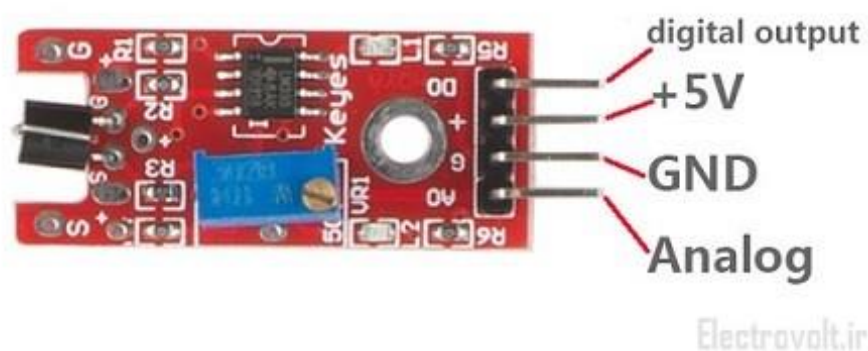
void loop ()
{
  sensorValue = analogRead (sensorPin);
  digitalWrite (ledPin, HIGH);
  delay (sensorValue);
  digitalWrite (ledPin, LOW);
  delay (sensorValue);
  Serial.println (sensorValue, DEC);
}

```

توضیح برنامه:

در این برنامه هر چه صدای دریافت شده قوی تر باشد ، LED موجود روی برد با سرعت بیشتری چشمک می زند. درون نرم افزار Arduino IDE با کلیک بر روی سریال مانیتور (Tools -> Serial Monitor) میتوانید تغییرات مقادیر خوانده شده این سنسور را ببینید.

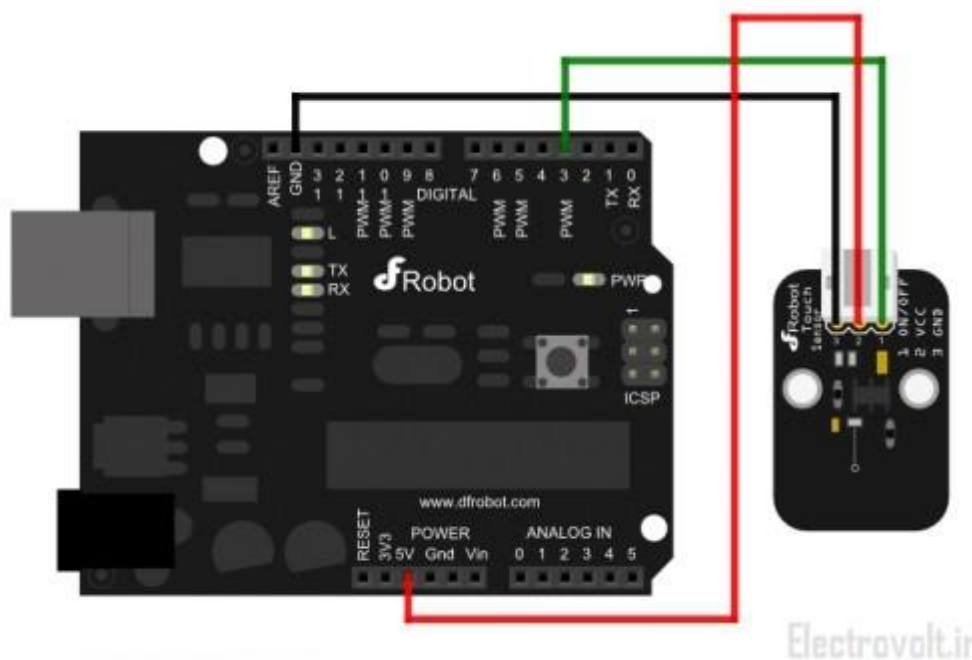
بخش چهاردهم : Metal Touch Sensor



این سنسور میتواند تماس دست یا هر اندام یا وسیله دیگری را تشخیص دهد. درون آن از یک ترانزیستور مدل KSP13 برای تشخیص لمس استفاده می گردد. در صورتی که به پایه وسط این ترانزیستور کوچکترین لمسی وارد شود سنسور تشخیص می دهد. برای همین بهتر است پایه سوم را به سمت بیرون کج کنید تا در دسترس قرار گیرد. علت تشخیص لمس توسط این سنسور به خاطر این مسئله است که بدن

انسان نوعی هادی جریان می باشد که همانند آنتن امواج الکترومغناطیسی را در هوا جذب می کند. این سنسور نیز همانند سنسورهای قبلی دارای دو خروجی دیجیتال و آنالوگ می باشد. حساسیت پایه دیجیتال توسط پتانسیومتر قابل تنظیم است.

نحوه اتصال به آردوینو:



برنامه آردوینو:

```
int Led = 13 ; // define LED Interface
int buttonpin = 3; // define Metal Touch Sensor Interface
int val ; // define numeric variables val

void setup ()
{
  pinMode (Led, OUTPUT) ; // define LED as output interface
  pinMode (buttonpin, INPUT) ; // define metal touch sensor output interface
}

void loop ()
{
  val = digitalRead (buttonpin) ; // digital interface will be assigned a value of 3 to read val
```

```

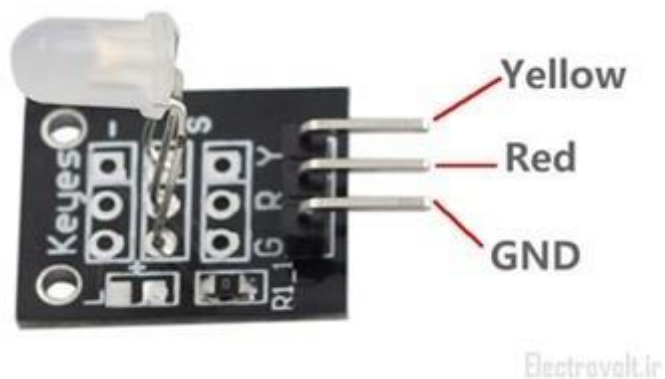
if (val == HIGH) // When the metal touch sensor detects a signal, LED flashes
{
digitalWrite (Led, HIGH);
}
else
{
digitalWrite (Led, LOW);
}
}

```

توضیح برنامه:

به محض تشخیص لمس ، LED روشن می گردد.

بخش پانزدهم : Two Color LED



LED دو رنگ دارای سه پایه (زمین , قرمز , زرد) می باشد. در نتیجه با استفاده از PWM میتوان رنگ های قرمز ، زرد و ترکیبی از این دو را بدست آورد. توصیه می شود برای طول عمر LED از مقاومت 330 اهم برای پایه زمین یا دو پایه دیگر استفاده کنید.

اتصال به آردوینو:

مشابه RGB LED (بخش سوم)

برنامه آردوینو:

```
// Arduino test code for KY011

int redpin = 11; // select the pin for the red LED
int greenpin = 10; // select the pin for the green LED
int val;

void setup () {
  pinMode (redpin, OUTPUT);
  pinMode (greenpin, OUTPUT);
}

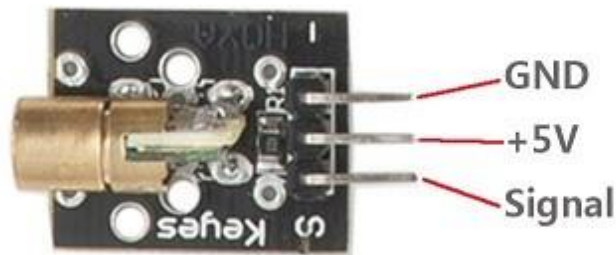
void loop () {
  for (val = 255; val > 0; val--)
  {
    analogWrite (greenpin, val);
    analogWrite (redpin, 255-val);
    delay (15);
  }

  for (val = 0; val < 255; val++)
  {
    analogWrite (greenpin, val);
    analogWrite (redpin, 255-val);
    delay (15);
  }
}
```

توضیح برنامه:

در این برنامه توسط دو حلقه for تمام طیف های رنگی این LED دو رنگ نمایان می شود.

بخش شانزدهم : Laser Emitter



Electrovolt.ir

لیزرها به علت جمع کردن انرژی و شلیک کردن آن در یک جهت باریک ، استفاده گسترده ای در صنایع مختلف نظیر صنایع پزشکی و نظامی دارند. این ماژول یک لیزر است که با ولتاژ 5 ولت کار می کند. یک پایه سیگنال وجود دارد که به محض برقرار شدن لیزر فعال شده و شروع به تابش انرژی می کند. یکی از کاربردهای لیزر تولید پالس های کد شده نوری در انتقال دیتا می باشد (شبیه به فیبر نوری).

اتصال به آردوینو:

پایه S لیزر را به پایه 13 آردوینو متصل می کنیم. پایه های - و + را به ترتیب به GND و 5V متصل می کنیم.

برنامه آردوینو:

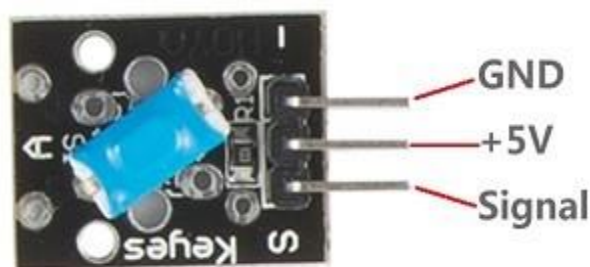
```
void setup ()
{
  pinMode (13, OUTPUT); // define the digital output interface 13 feet
}

void loop () {
  digitalWrite (13, HIGH); // open the laser head
  delay (1000); // delay one second
  digitalWrite (13, LOW); // turn off the laser head
  delay (1000); // delay one second
}
```

توضیح برنامه:

یک ثانیه لیزر روشن و ثانیه بعدی خاموش می شود.

بخش هفدهم : Ball Switch



Electrovolt.ir

این سنسور که نوعی Tilt Switch است ، به محض تغییر زاویه قطع و وصل می شود. درون این سنسور یک توپ کوچک وجود دارد که حرکات کوچک و ریز را تشخیص می دهد. از همین رو به آن Ball Switch نیز گفته می شود.

اتصال به آردوینو:

همانند بخش یازدهم

برنامه آردوینو:

```
int Led = 13 ;// define LED Interface
int buttonpin = 3; // define the tilt switch sensor interfaces
int val ;// define numeric variables val
void setup ()
{
  pinMode (Led, OUTPUT) ;// define LED as output interface
  pinMode (buttonpin, INPUT) ;//define the output interface tilt switch sensor
```

Electrovolt.ir

```

}

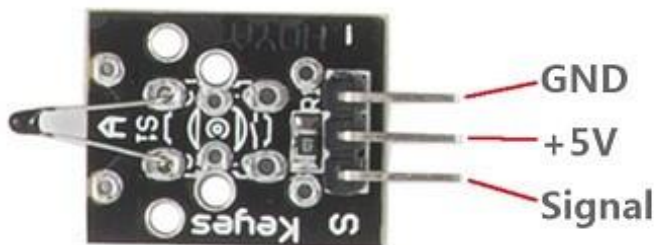
void loop ()
{
  val = digitalRead (buttonpin) ;// digital interface will be assigned a value of 3 to read val
  if (val == HIGH) //When the tilt sensor detects a signal when the switch, LED flashes
  {
    digitalWrite (Led, HIGH);
  }
  else
  {
    digitalWrite (Led, LOW);
  }
}

```

توضیح برنامه:

به محض تغییر زاویه سنسور ، LED روشن می گردد.

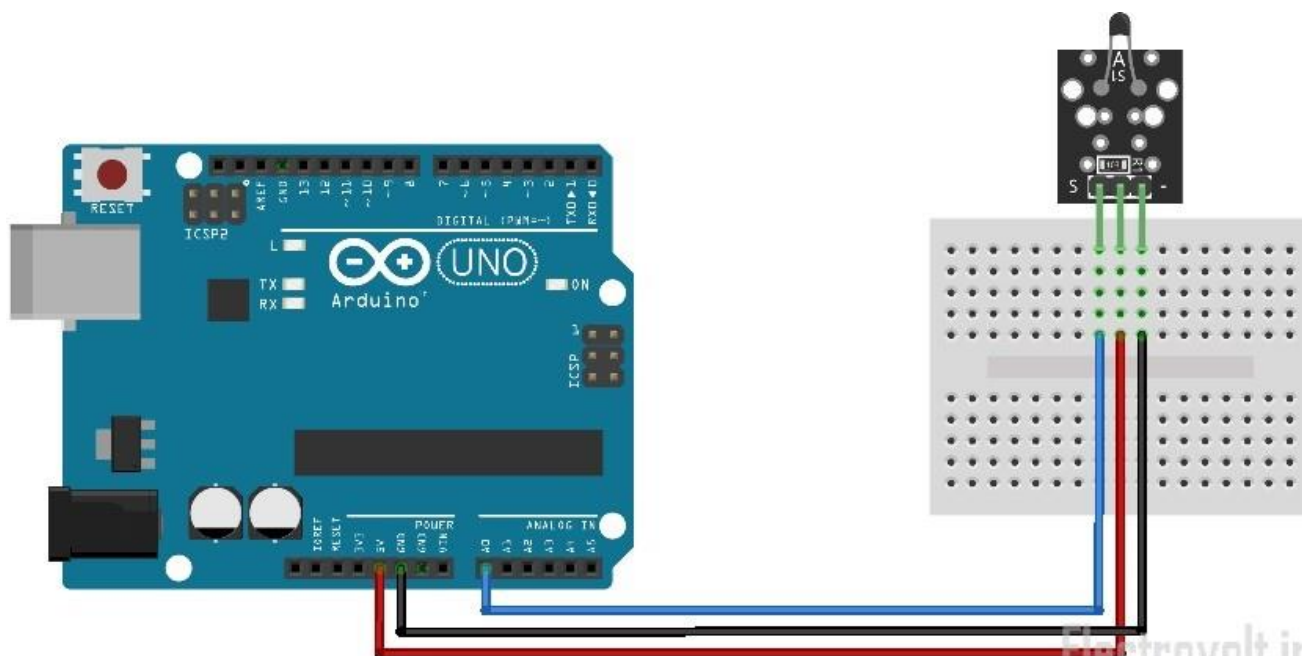
بخش هجدهم : Analog Temperature Sensor



این سنسور دمای آنالوگ نوعی مقاومت متغیر Thermistor (ترمیستور یا ترمال رزیستور) حساس به تغییرات دما می باشد. به طوری که با افزایش دما مقدار ولتاژ خروجی سنسور شروع به کاهش می کند. برای اندازه گیری دقیق دما ابتدا باید سنسور را کالیبره کرد. یعنی ابتدا

سنسور را در دمایی قرار داد که مقدار دقیق آن مشخص باشد، سپس ولتاژ خروجی آنالوگ آن را اندازه گیری کرد و بعد از آن با هر تغییر دمایی میتوان دمای محیط را حدس زد.

اتصال به آردوینو:



برنامه آردوینو:

```
#include <math.h>

int sensorPin = A5; // select the input pin for the potentiometer

double Thermistor(int RawADC) {
    double Temp;

    Temp = log(10000.0*((1024.0/RawADC-1)));

    Temp = 1 / (0.001129148 + (0.000234125 + (0.0000000876741 * Temp * Temp ))* Temp );

    Temp = Temp - 273.15; // Convert Kelvin to Celcius
    //Temp = (Temp * 9.0)/ 5.0 + 32.0; // Convert Celcius to Fahrenheit
    return Temp;
}
```

```

void setup() {
  Serial.begin(9600);
}

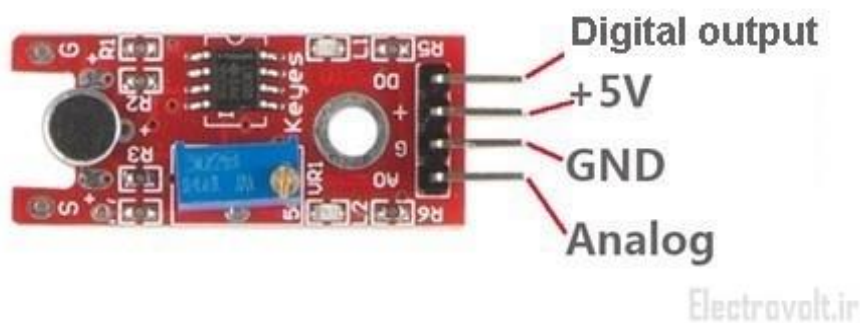
void loop() {
  int readVal=analogRead(sensorPin);
  double temp = Thermistor(readVal);
  Serial.println(temp); // display tempature
  //Serial.println(readVal); // display tempature
  delay(500);
}

```

توضیح برنامه:

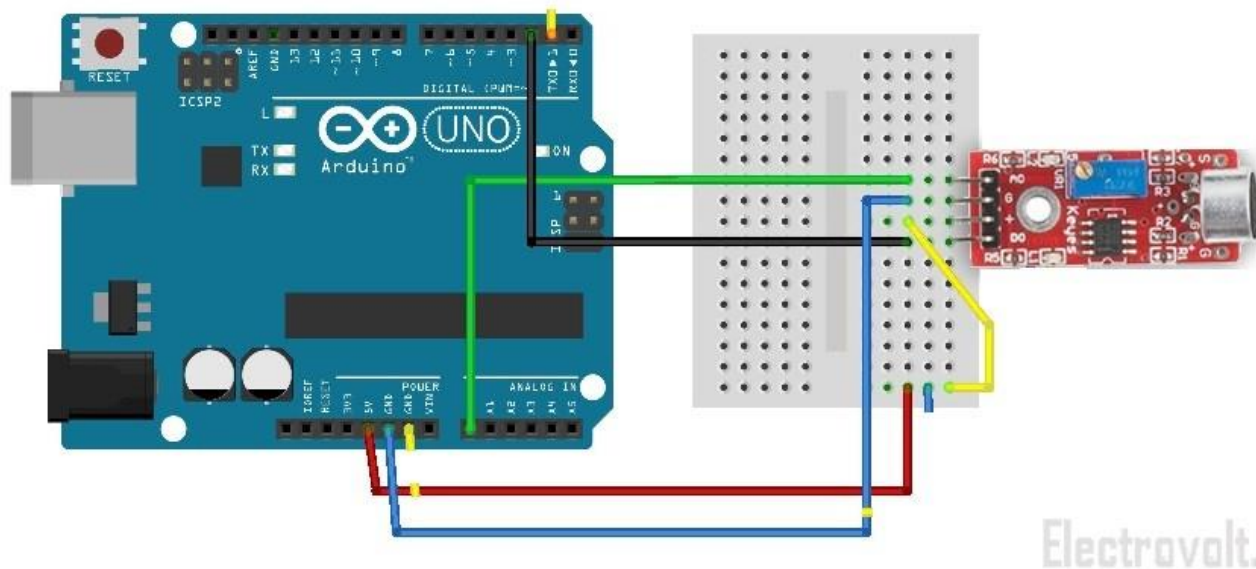
ابتدا مقدار ولتاژ آنالوگ از سنسور خوانده شده و بعد از تبدیل به یک عدد دیجیتال توسط تابع Thermistor به دما تبدیل می شود. درون نرم افزار Arduino IDE با کلیک بر روی سریال مانیتور (Tools -> Serial Monitor) میتوانید تغییرات مقادیر خوانده شده این سنسور را ببینید.

بخش نوزدهم : Small Sound Sensor



این سنسور همانند سنسور بخش سیزدهم می باشد با این تفاوت که این سنسور نسبت به صداهای کوچک حساسیت دارد. روی این سنسور یک میکروفن قرار دارد. میکروفن میتواند سیگنال صوتی را به سیگنال الکتریکی تبدیل کند. این سنسور دارای هر دو خروجی آنالوگ و دیجیتال می باشد. میکروفن صداهای محیط را به صورت مستقیم به خروجی آنالوگ می برد. اما زمانی که قدرت صدا از یک سطح مشخصی بالاتر رود ، پایه دیجیتال خروجی High می گردد. یک پتانسیومتر جهت تنظیم حساسیت خروجی دیجیتال نیز تعبیه شده است.

نحوه اتصال به آردوینو:



برنامه آردوینو اول برای استفاده از پایه دیجیتال:

```
int Led = 13 ;// define LED Interface
int buttonpin = 3; // define D0 Sensor Interface
int val = 0; // define numeric variables val

void setup ()
{
  pinMode (Led, OUTPUT) ;// define LED as output interface
  pinMode (buttonpin, INPUT) ;// output interface D0 is defined sensor
}

void loop ()
{
  val = digitalRead(buttonpin); // digital interface will be assigned a value of pin 3 to read val
```

```
if (val == HIGH) // When the sound detection module detects a signal, LED flashes
{
digitalWrite (Led, HIGH);
}
else
{
digitalWrite (Led, LOW);
}
}
```

توضیح برنامه:

به محض شناسایی صدای کوچک ، LED روشن می گردد.

برنامه آردوینو دوم برای استفاده از پایه آنالوگ:

```
int sensorPin = A0; // select the input pin for the potentiometer
int ledPin = 13; // select the pin for the LED
int sensorValue = 0; // variable to store the value coming from the sensor

void setup ()
{
pinMode (ledPin, OUTPUT);
Serial.begin (9600);
}

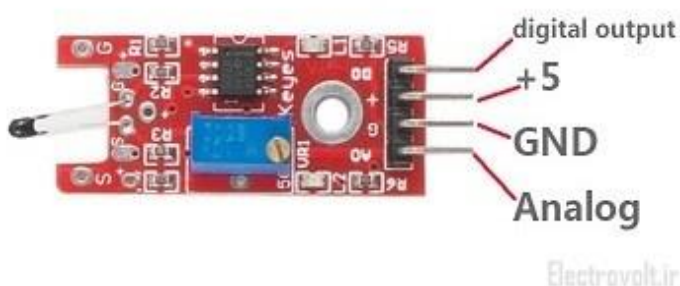
void loop ()
{
sensorValue = analogRead (sensorPin);
```

```
digitalWrite (ledPin, HIGH);  
delay (sensorValue);  
digitalWrite (ledPin, LOW);  
delay (sensorValue);  
Serial.println (sensorValue, DEC);  
}
```

توضیح برنامه:

در این برنامه هر چه صدای دریافت شده قوی تر باشد ، LED موجود روی برد با سرعت بیشتری چشمک می زند. درون نرم افزار Arduino IDE با کلیک بر روی سریال مانیتور (Tools -> Serial Monitor) میتوانید تغییرات مقادیر خوانده شده این سنسور را ببینید.

بخش بیستم : Digital Temperature Sensor



این سنسور همانند سنسور آنالوگ بخش هجدهم می باشد. با این تفاوت که یک مقایسه کننده (آپ امپ) و یک پتانسیومتر به آن اضافه شده است. در نتیجه این سنسور علاوه بر خروجی آنالوگ که برای دما استفاده می شود ، یک خروجی دیجیتال نیز دارد که میتواند یک آستانه دمایی ایجاد کند. به طوری که اگر دما بالاتر از آستانه باشد خروجی دیجیتال سنسور High و در غیر این صورت Low می باشد. توسط تغییر پتانسیومتر این آستانه دمایی برای خروجی دیجیتال تغییر می کند.

اتصال به آردوینو:

پایه خروجی دیجیتال سنسور را به پایه 7 آردوینو و پایه های + و G سنسور را به ترتیب به 5V و GND آردوینو متصل کنید.

برنامه آردوینو برای استفاده از پایه دیجیتال:

```
const int buttonPin = 7; // the number of the pushbutton pin
const int ledPin = 13; // the number of the LED pin

// variables will change:
int buttonState = 0; // variable for reading the pushbutton status

void setup() {
  // initialize the LED pin as an output:
  pinMode(ledPin, OUTPUT);
  // initialize the pushbutton pin as an input:
  pinMode(buttonPin, INPUT);
  Serial.begin(9600);
}

void loop(){
  // read the state of the pushbutton value:
  buttonState = digitalRead(buttonPin);
  int val = analogRead(0);
  Serial.println(val);
  // check if the pushbutton is pressed.
  // if it is, the buttonState is HIGH:
  if (buttonState == HIGH) {
    // turn LED on:
    digitalWrite(ledPin, HIGH);
  }
  else {

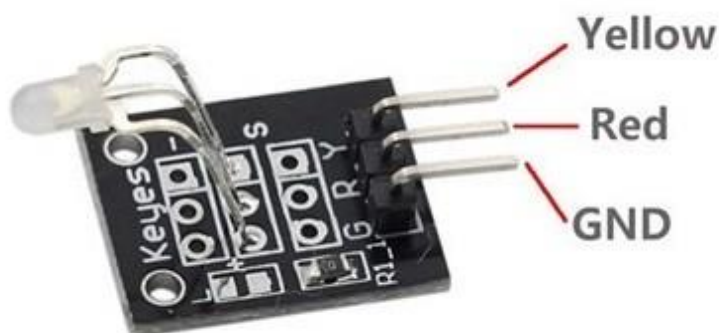
Electrovolt.ir
```

```
// turn LED off:  
digitalWrite(ledPin, LOW);  
  
}  
  
}
```

توضیح برنامه:

به محض افزایش دما از آستانه تنظیم شده سنسور ، LED روشن می گردد.

بخش بیست و یکم : Small Two Color LED



Electrovolt.ir

در این بسته آموزشی دو عدد ماژول LED دو رنگ وجود دارد که در شکل فوق نوع کوچکتر آن را مشاهده می کنید. نوع بزرگ آن را در بخش پانزدهم بررسی کردیم. این دو ماژول هیچ تفاوت عملکردی با هم ندارند و تنها از نظر بسته بندی (Package) با هم تفاوت دارند.

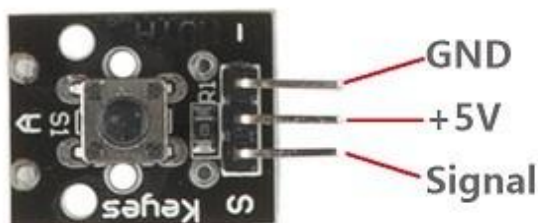
اتصال به آردوینو:

همانند بخش پانزدهم

برنامه آردوینو:

همانند بخش پانزدهم

بخش بیست و دوم : Button



Electrovolt.ir

این ماژول همان کلید فشاری یا پوش باتن (Push Button) می باشد. این ماژول دارای یک خروجی سیگنال می باشد که در حالت عادی High می باشد و به محض فشردن شدن Low می شود.

اتصال به آردوینو:

پایه Signal را به پایه 2 آردوینو و پایه های + و G سنسور را به ترتیب به 5V و GND آردوینو متصل کنید.

برنامه آردوینو:

```
// constants won't change. They're used here to
// set pin numbers:
const int buttonPin = 2; // the number of the pushbutton pin
const int ledPin = 13; // the number of the LED pin
// variables will change:
int buttonState = 0; // variable for reading the pushbutton status
void setup() {
  // initialize the LED pin as an output:
  pinMode(ledPin, OUTPUT);
  // initialize the pushbutton pin as an input:
  pinMode(buttonPin, INPUT);
}
```

```

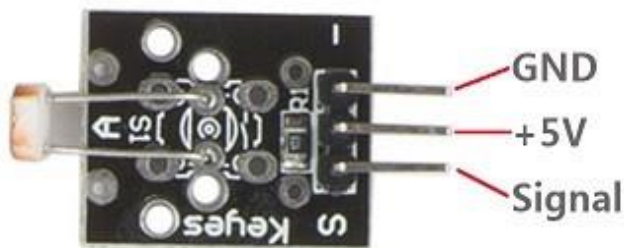
void loop() {
  // read the state of the pushbutton value:
  buttonState = digitalRead(buttonPin);
  // check if the pushbutton is pressed.
  // if it is, the buttonState is HIGH:
  if (buttonState == HIGH) {
    // turn LED on:
    digitalWrite(ledPin, HIGH);
  } else {
    // turn LED off:
    digitalWrite(ledPin, LOW);
  }
}

```

توضیح برنامه:

برنامه دائما منطق پایه سیگنال را می خواند و در صورت زده شدن کلید ، LED خاموش می گردد.

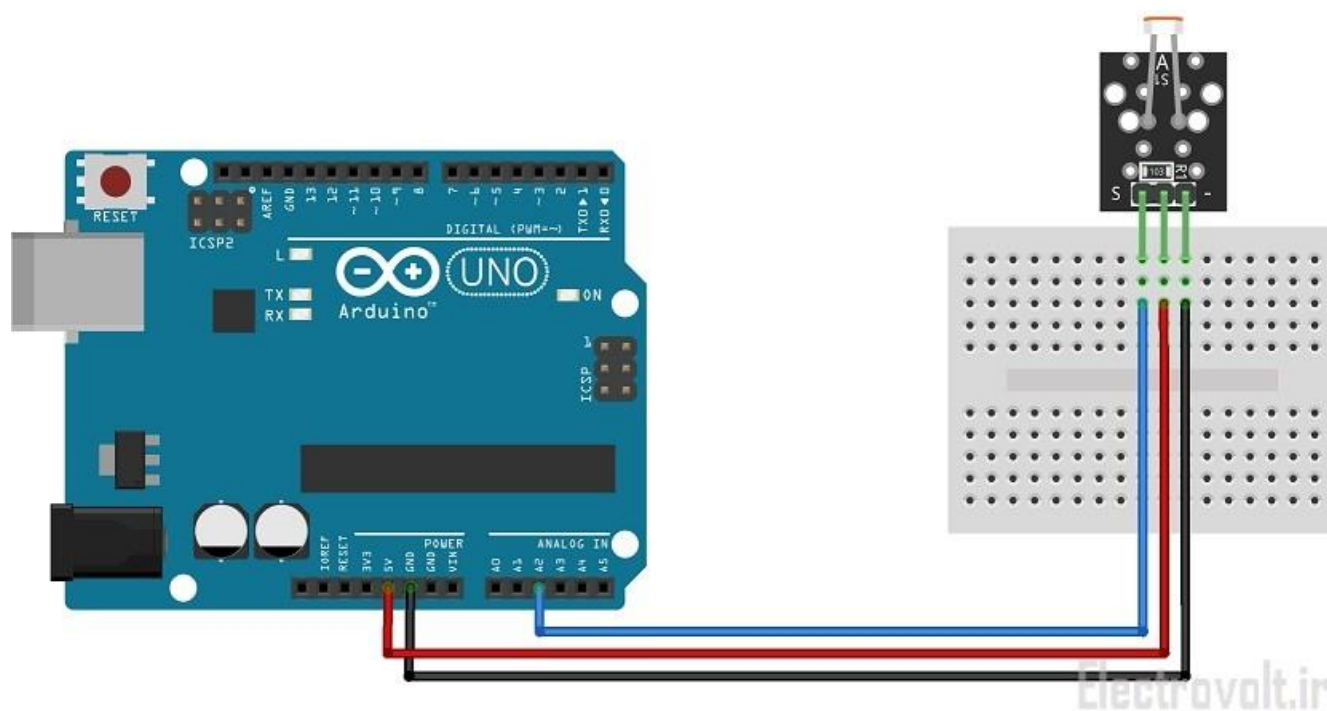
بخش بیست و سوم : Photoresistor Sensor



Electrovolt.ir

سنسور فوتورزیستور نوعی مقاومت وابسته به نور (LDR) می باشد. این مقاومت با شدت نور تابیده شده به سطح آن رابطه عکس دارد یعنی هر چه محیط سنسور شدت نور بیشتری داشته باشد ، LDR مقاومت کمتری دارد و در نتیجه ولتاژ خروجی LDR بیشتر می گردد. از این سنسور میتوان برای روشن کردن اتوماتیک چراغ در زمان کمبود نور استفاده کرد.

اتصال به آردوینو:



برنامه آردوینو:

```
//KY018 Photo resistor module
```

```
int sensorPin = A5; // select the input pin for the potentiometer
```

```
int ledPin = 13; // select the pin for the LED
```

```
int sensorValue = 0; // variable to store the value coming from the sensor
```

```
void setup() {
```

```
  pinMode(ledPin, OUTPUT);
```

```
  Serial.begin(9600);
```

```
}
```

```

void loop()
{
  sensorValue = analogRead(sensorPin);
  digitalWrite(ledPin, HIGH);
  delay(sensorValue);
  digitalWrite(ledPin, LOW);
  delay(sensorValue);
  Serial.println(sensorValue, DEC);
}

```

توضیح برنامه:

هر چه نور محیط بیشتر باشد ، LED با سرعت کمتری چشمک می زند. درون نرم افزار Arduino IDE با کلیک بر روی سریال مانیتور (Serial Monitor -> Tools) می‌توانید تغییرات مقادیر خوانده شده این سنسور را ببینید.

بخش بیست و چهارم : IR Emitter



Electrovolt.ir

این ماژول یک فرستنده مادون قرمز (Infrared Transmitter) می باشد. این ماژول از نظر ظاهری کاملاً شبیه به LED است اما از مدولاسیون 38 کیلوهرتز برای ارسال امواج مادون قرمز بوسیله میکروکنترلر استفاده می کند. از امواج IR در کنترل تلویزیون ها و در دوربین های مادون قرمز استفاده می گردد. توجه نمایید که اشعه مادون قرمز در جهت و محدوده خاصی می‌تواند منتشر شود.

اتصال به آردوینو:

پایه 5 را به پایه 3 و پایه دیگر را به GND آردوینو متصل می کنیم.

برنامه آردوینو:

```
#include <IRremote.h>

IRsend irsend;

const int ledPin = 13;

void setup()
{
  pinMode(ledPin, OUTPUT);
}

void loop()
{
  digitalWrite(ledPin, HIGH);
  irsend.sendNEC(0xFFA25D, 32);
  delay(1000);
  digitalWrite(ledPin, LOW);
  delay(1000);
}
```

توضیح برنامه:

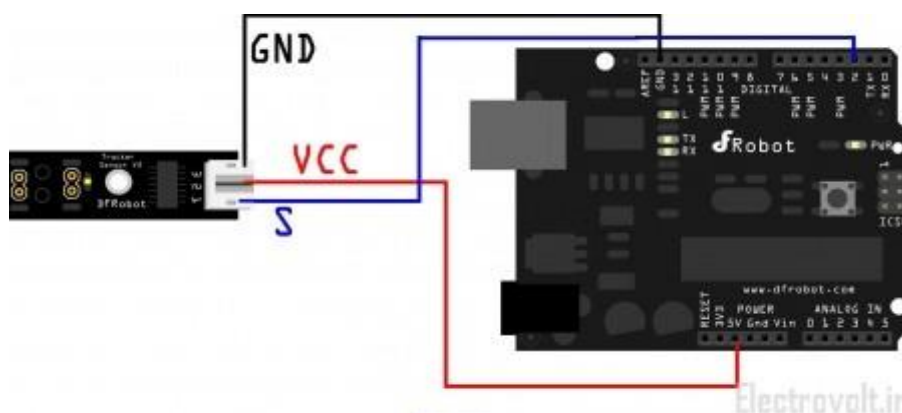
در این برنامه از کتابخانه استاندارد IRremote استفاده شده است که ابتدا باید این کتابخانه را از قسمت Manage Libraries با سرچ کردن نام آن نصب کنید. در این برنامه کد 0xFFA25D هر ثانیه ارسال می شود و در هنگام ارسال LED روشن می گردد. برای تشخیص درستی ارسال و یا برای دریافت دیتا در طرف دیگر نیاز به گیرنده IR می باشد که در بخش سی ام بررسی شده است.

بخش بیست و پنجم : Tracking Sensor



این ماژول دارای سنسور فرستنده و گیرنده مادون قرمز در یک بسته بندی می باشد و میتواند خطوط سفید و سیاه را تشخیص دهد. از این قابلیت در ربات های مسیریاب استفاده می شود. فرستنده مادون قرمز همواره روشن است و سیگنال ارسال می کند. اگر این سیگنال به سطح سفید برخورد کند ، منعکس شده و گیرنده مادون قرمز آن را جذب می کند و پایه S در این حالت Low می گردد. اما اگر سیگنال به سطح مشکی برخورد کند ، بازتابش صورت نمی گیرد و سطح پایه S در این حالت High می گردد.

اتصال به آردوینو:



برنامه آردوینو:

```
///Arduino Sample Code

void setup()
{
  Serial.begin(9600);
}

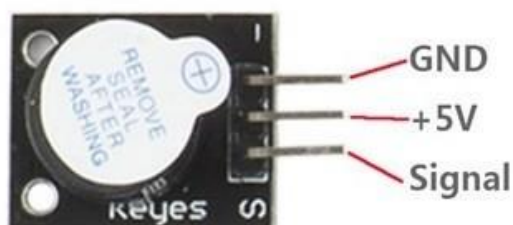
void loop()
{
  Serial.println(digitalRead(2)); // print the data from the sensor
}
```

```
delay(500);  
}
```

توضیح برنامه:

در این برنامه هر 500 میلی ثانیه ، پایه سیگنال سنسور خوانده شده و به کامپیوتر ارسال می گردد. درون نرم افزار Arduino IDE با کلیک بر روی سریال مانیتور (Tools -> Serial Monitor) میتوانید تغییرات مقادیر خوانده شده این سنسور را ببینید.

بخش بیست و ششم : Active Buzzer

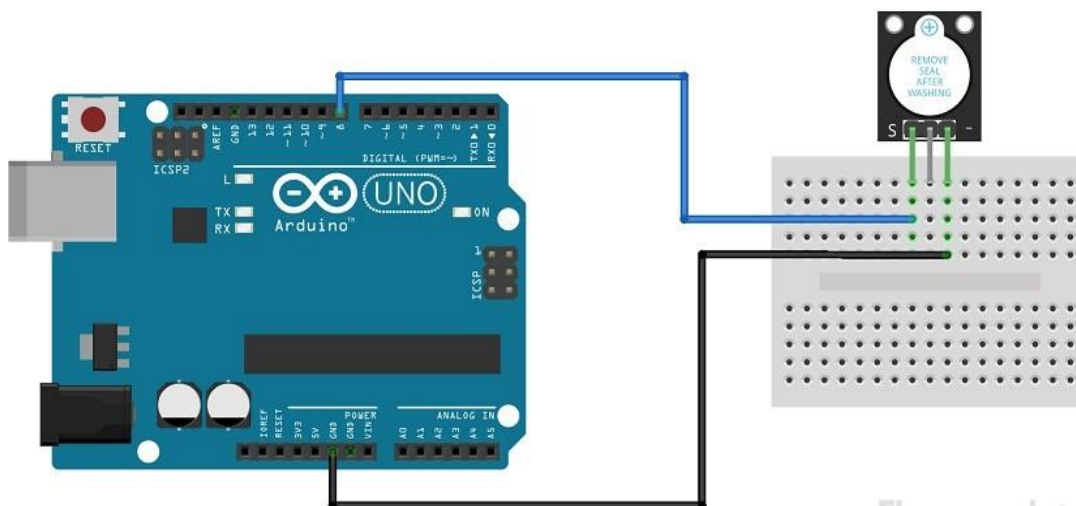


Active

Electrovolt.ir

این ماژول یک بازر اکتیو (فعال) می باشد. بازر نوعی بوق است که بوسیله آن میتوان صداهای کوچک و متنوعی ساخت. به وسیله بازر حتی میتوان آهنگ برنامه ریزی و پخش کرد.

اتصال به آردوینو:



Electrovolt.ir

```
//Example code KY012 active buzzer  
  
int speakerPin = 8;  
  
void setup () {  
  pinMode (speakerPin, OUTPUT);  
}  
  
void loop () {  
  analogWrite (speakerPin, 255);  
  
  delay (50);  
  
  analogWrite (speakerPin, 0);  
  
  delay (10);  
}
```

توضیح برنامه:

روش کار با بازر همانند LED می باشد و برای تولید صداهای مختلف باید به آن موج PWM با پهنای مختلف اعمال کرد. توسط تابع analogWrite میتوان موج PWM با پهنای بین 0 تا 255 ساخت.

بخش بیست و هفتم : Reed Switch

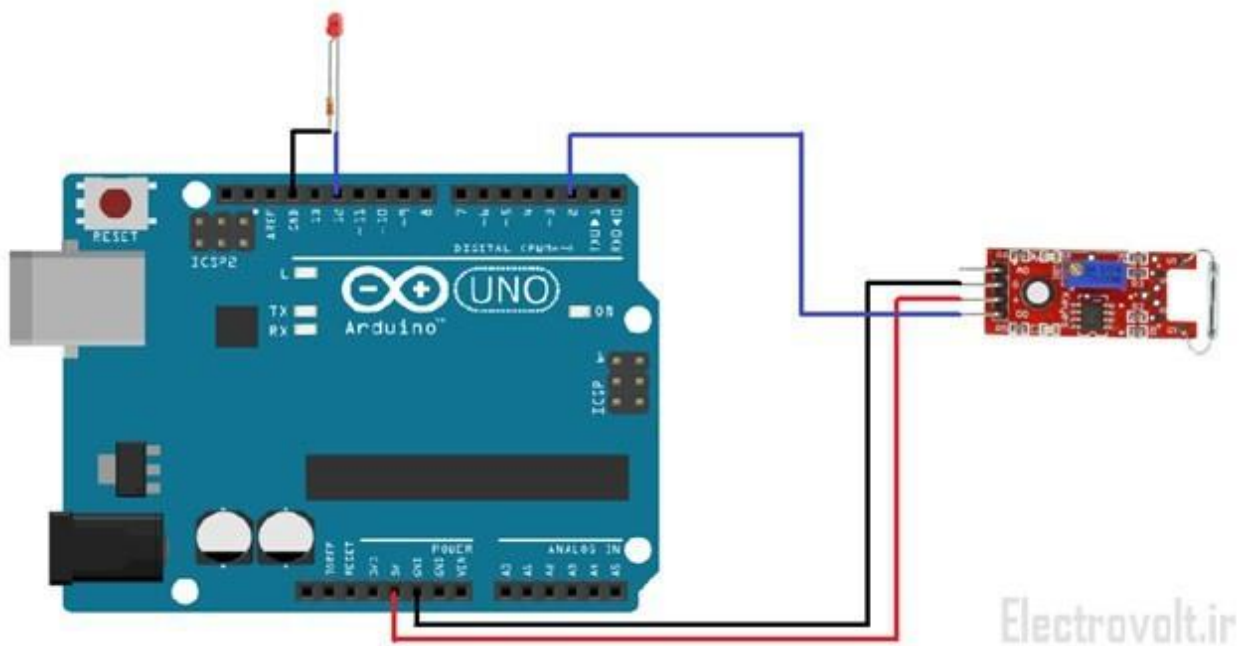


Reed Switch

Electrovolt.ir

رید سوئیچ ، کلیدی است که در صورت قرار گرفتن آهنربا روی آن ، روشن می شود. در واقع این سنسور همانند سنسور اثر هال ، میدان مغناطیسی قوی را تشخیص می دهد. از سنسورهای اثر هال به علت داشتن خروجی دقیق اغلب در اندازه گیری سرعت ماشین ها و شمارش محصولات در خطوط تولید استفاده می شود در حالی که از رید سوئیچ برای شناسایی وجود/عدم وجود میدان مغناطیسی استفاده می گردد. رید سوئیچ دارای یک پوسته شیشه ای می باشد که درون آن دو کنتاکت از جنس آهن وجود دارد. در حالت عادی این دو کنتاکت به هم وصل نیستند اما در صورت وجود میدان مغناطیسی قوی باردار شده و به هم متصل می گردند. توسط پتانسیومتری که روی برد وجود دارد میتوان حساسیت سنسور به میدان های اطراف آن را کنترل کرد.

اتصال به آردوینو:



برنامه آردوینو:

```
int Led = 13 ; // define LED Interface
int buttonpin = 3; // define the Reed sensor interfaces
int val ; // define numeric variables val
void setup ()
{
  pinMode (Led, OUTPUT) ; // define LED as output interface
  pinMode (buttonpin, INPUT) ; // output interface as defined Reed sensor
}
```

```

void loop (){
  val = digitalRead (buttonpin) ; // digital interface will be assigned a value of 3 to read val
  if (val == HIGH) // When the Reed sensor detects a signal, LED flashes
  {
    digitalWrite (Led, HIGH);
  }
  else
  {
    digitalWrite (Led, LOW);
  }
}

```

توضیح برنامه:

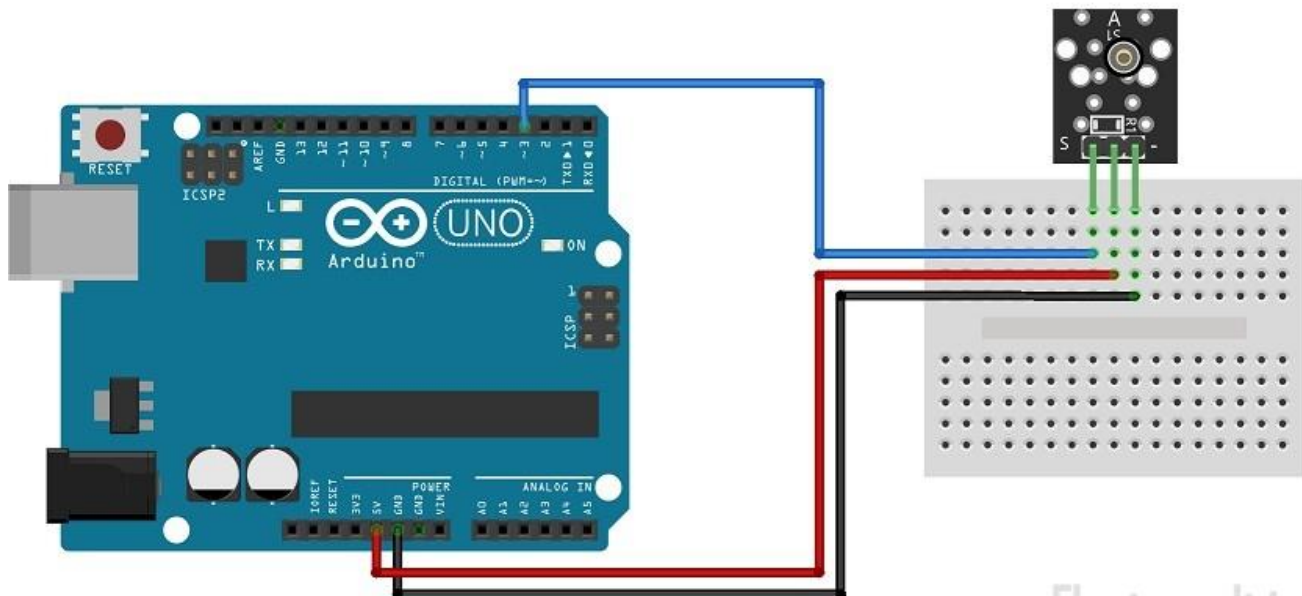
به محض روشن شدن کلید و تشخیص میدان مغناطیسی قوی ، LED روشن می شود.

بخش بیست و هشتم : Shock Sensor



در این ماژول ، سنسور تشخیص شوک (لرزش) قرار دارد. این سنسور در حالت عادی خاموش است اما زمانی که لرزش را احساس کند وضعیتش تغییر می کند. ظاهر این سنسور به خازن نیز شباهت دارد اما در درون سنسور فنر و توپ های فلزی وجود دارد که در حالت عادی از یکدیگر جدا هستند و به محض لرزش ، با برخورد به یکدیگر سیگنال تریگر (Trigger) را ایجاد می کنند.

اتصال به آردوینو:



Electrovolt.ir

برنامه آردوینو:

```
int Led = 13 ;// define LED Interface
int Shock = 10; // define the vibration sensor interface
int val; // define numeric variables val
void setup ()
{
  pinMode (Led, OUTPUT) ; // define LED as output interface
  pinMode (Shock, INPUT) ; // output interface defines vibration sensor
}
void loop ()
{
  val = digitalRead (Shock) ; // read digital interface is assigned a value of 3 val
  if (val == HIGH) // When the shock sensor detects a signal, LED flashes
  {
```

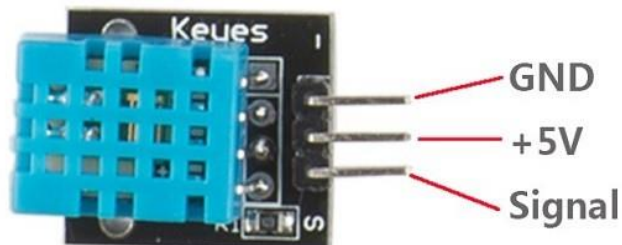
Electrovolt.ir

```
digitalWrite (Led, LOW);
}
else
{
digitalWrite (Led, HIGH);
}
}
```

توضیح برنامه:

به محض لرزش ، LED روشن می شود.

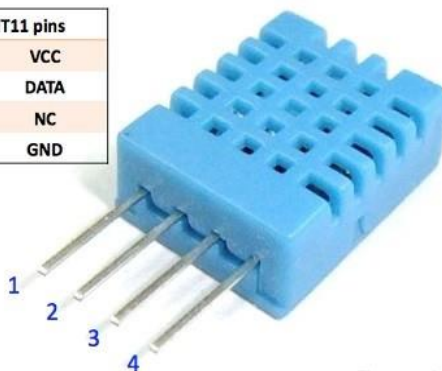
بخش بیست و نهم : Temperature and Humidity Sensor



Electrovolt.ir

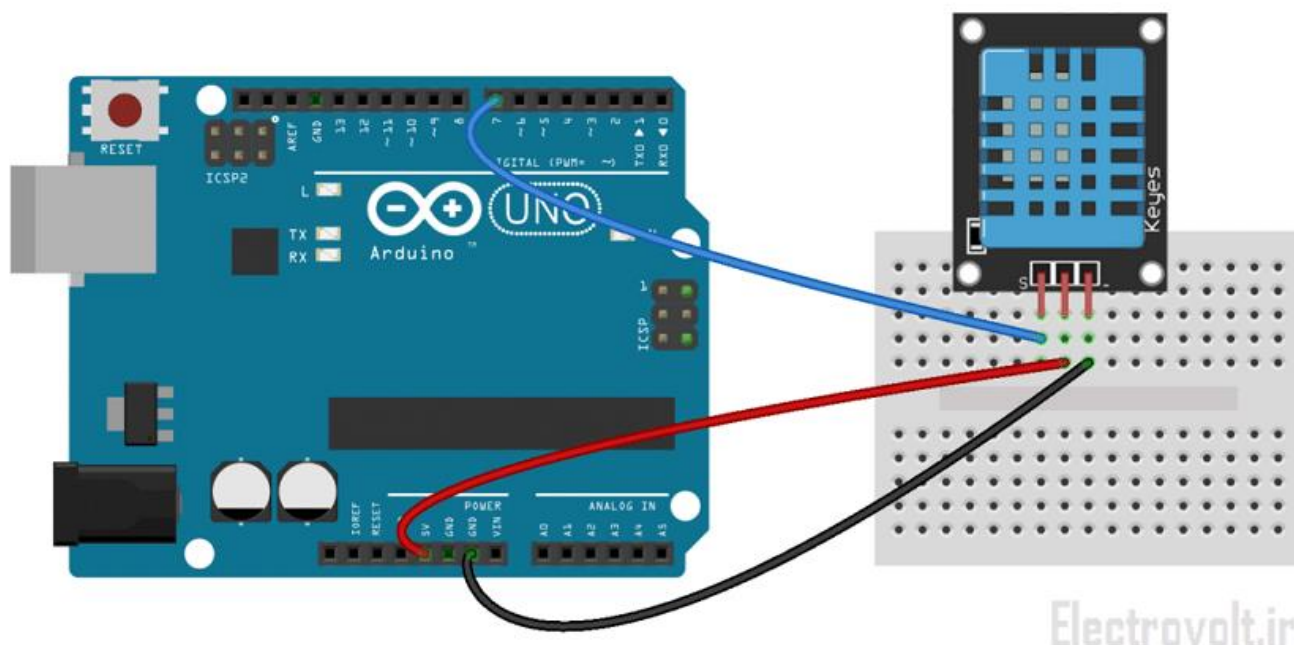
این ماژول شامل یک سنسور دما و رطوبت مدل DHT11 می باشد. DHT11 یک سنسور دیجیتال یکپارچه و کالیبره شده است که میتواند همزمان رطوبت و دمای محیط را با پروتکل تک سیمه (Wire 1) ارسال نماید. پایه های این سنسور را در شکل زیر مشاهده می کنید.

DHT11 pins	
1	VCC
2	DATA
3	NC
4	GND



Electrovolt.ir

اتصال به آردوینو:



برنامه آردوینو:

همانند سنسور دمای DS18B20 برای این سنسور هم دو نوع برنامه میتوان نوشت. اولی با استفاده از کتابخانه dht.h و دومی بدون استفاده از کتابخانه می باشد. توجه داشته باشید که برای استفاده از کتابخانه باید ابتدا آن را با سرچ کردن نام آن در منوی Manage Library نصب نمایید.

برنامه اول:

```
#include <DHT.h>

dht DHT;

#define DHT11_PIN 4

void setup()
{
  Serial.begin(9600);
}

void loop()
{
  Electrovolt.ir
```

```
// READ DATA

Serial.print("DHT11, \t");

//read the value returned from sensor
int chk = DHT.read11(DHT11_PIN);

switch (chk)
{
case DHTLIB_OK:

Serial.print("OK,\t");

break;

case DHTLIB_ERROR_CHECKSUM:

Serial.print("Checksum error,\t");

break;

case DHTLIB_ERROR_TIMEOUT:

Serial.print("Time out error,\t");

break;

default:

Serial.print("Unknown error,\t");

break;

}

// DISPLAY DATA

Serial.print("Tem:");

Serial.print(DHT.temperature,1); //print the temperature on lcd

Serial.print(" C");

Serial.print("Hum:");

Serial.print(DHT.humidity,1); //print the humidity on lcd

Serial.print(" %");

delay(200); //wait a while
```

```
}
```

برنامه دوم:

```
//KY015 DHT11 Temperature and humidity sensor
```

```
int DHpin = 8;
```

```
byte dat [5];
```

```
byte read_data () {
```

```
    byte data;
```

```
    for (int i = 0; i < 8; i++) {
```

```
        if (digitalRead (DHpin) == LOW) {
```

```
            while (digitalRead (DHpin) == LOW); // wait for 50us
```

```
            delayMicroseconds (30); // determine the duration of the high level to determine the data is '0 'or '1'
```

```
            if (digitalRead (DHpin) == HIGH)
```

```
                data |= (1 << (7-i)); // high front and low in the post
```

```
            while (digitalRead (DHpin) == HIGH); // data '1 ', wait for the next one receiver
```

```
        }
```

```
    }
```

```
    return data;
```

```
}
```

```
void start_test () {
```

```
    digitalWrite (DHpin, LOW); // bus down, send start signal
```

```
    delay (30); // delay greater than 18ms, so DHT11 start signal can be detected
```

```
    digitalWrite (DHpin, HIGH);
```

```
    delayMicroseconds (40); // Wait for DHT11 response
```

```
    pinMode (DHpin, INPUT);
```

```
    while (digitalRead (DHpin) == HIGH);
```

```
    delayMicroseconds (80); // DHT11 response, pulled the bus 80us
```

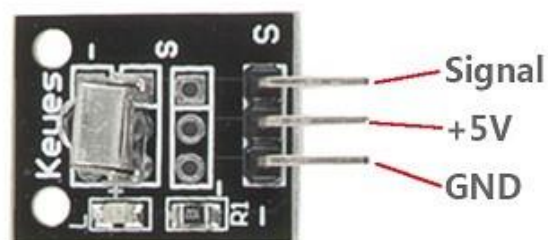
Electrovolt.ir

```

if (digitalRead (DHpin) == LOW);
delayMicroseconds (80); // DHT11 80us after the bus pulled to start sending data
for (int i = 0; i < 4; i ++) // receive temperature and humidity data, the parity bit is not considered
dat[i] = read_data ();
pinMode (DHpin, OUTPUT);
digitalWrite (DHpin, HIGH); // send data once after releasing the bus, wait for the host to open the
next Start signal
}
void setup () {
  Serial.begin (9600);
  pinMode (DHpin, OUTPUT);
}
void loop () {
  start_test ();
  Serial.print ("Current humidity =");
  Serial.print (dat [0], DEC); // display the humidity-bit integer;
  Serial.print ('.');
  Serial.print (dat [1], DEC); // display the humidity decimal places;
  Serial.println ("%");
  Serial.print ("Current temperature =");
  Serial.print (dat [2], DEC); // display the temperature of integer bits;
  Serial.print ('.');
  Serial.print (dat [3], DEC); // display the temperature of decimal places;
  Serial.println ('C');
  delay (700);
}

```

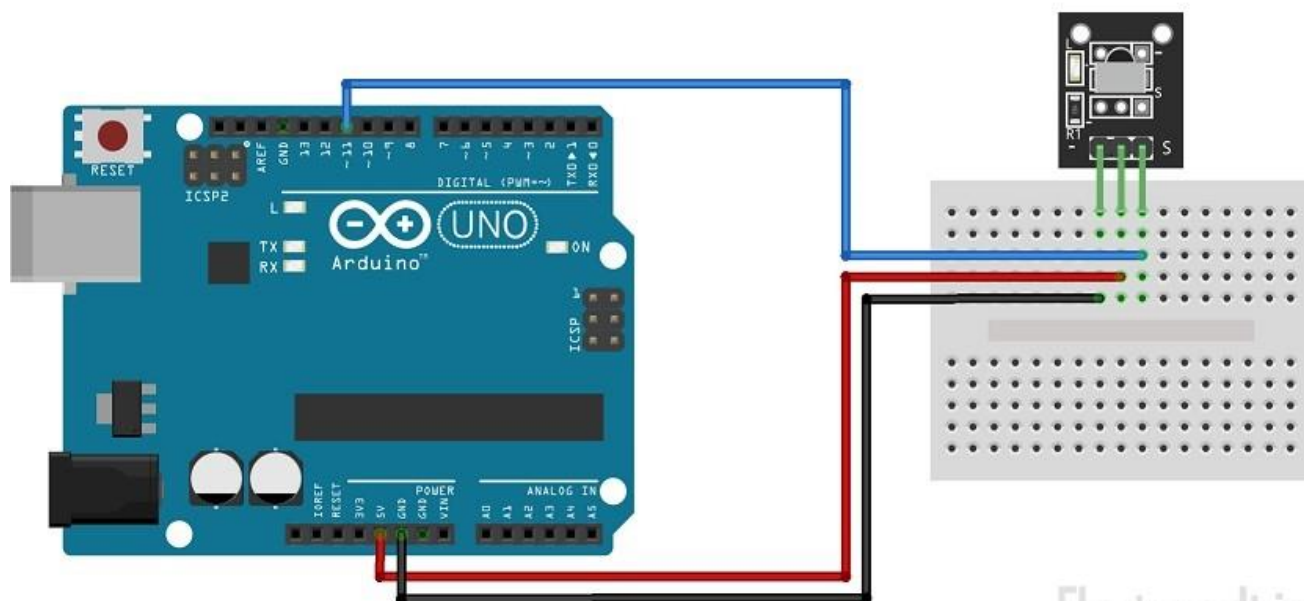
بخش سی ام : IR Reciever



Electrovolt.ir

در بخش بیست و چهارم فرستنده IR را توضیح دادیم که برای ارسال دیتا استفاده می شود همچنین گفتیم که برای دریافت دیتا باید از ماژول گیرنده IR استفاده نماییم. درون این ماژول یک فرستنده مادون قرمز 38 کیلوهرتز مدل vs1838b وجود دارد که سیگنال دریافتی مادون قرمز را به سطح ولتاژ منطقی تبدیل می کند.

اتصال به آردوینو:



Electrovolt.ir

برنامه آردوینو:

```
#include <IRremote.h>

int RECV_PIN = 11;

IRrecv irrecv(RECV_PIN);

decode_results results;
```

Electrovolt.ir

```

void setup()
{
  Serial.begin(9600);
  irrecv.enableIRIn(); // Start the receiver
}

void loop() {
  if (irrecv.decode(&results)) {
    Serial.println(results.value, HEX);
    irrecv.resume(); // Receive the next value
  }
}

```

توضیح برنامه:

در این برنامه از کتابخانه IRremote استفاده شده است که قبلاً آن را از طریق Library Manager نصب کرده بودیم. در این برنامه به محض دریافت دیتای مادون قرمز، نتیجه به کامپیوتر فرستاده می‌شود. درون نرم افزار Arduino IDE با کلیک بر روی سریال مانیتور (Tools -> Serial Monitor) میتوانید مقادیر خوانده شده این سنسور را ببینید.

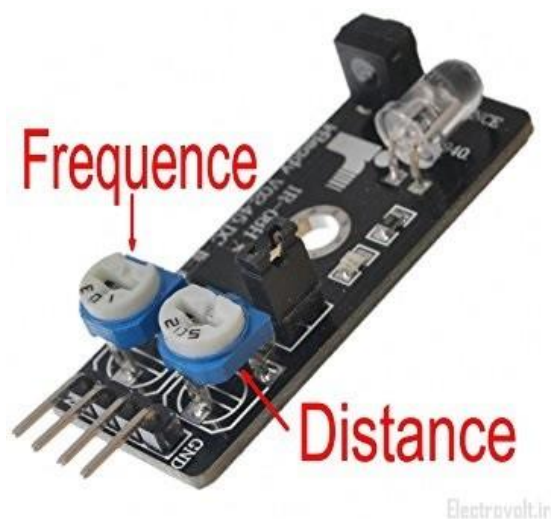
بخش سی و یکم : Avoidance Sensor



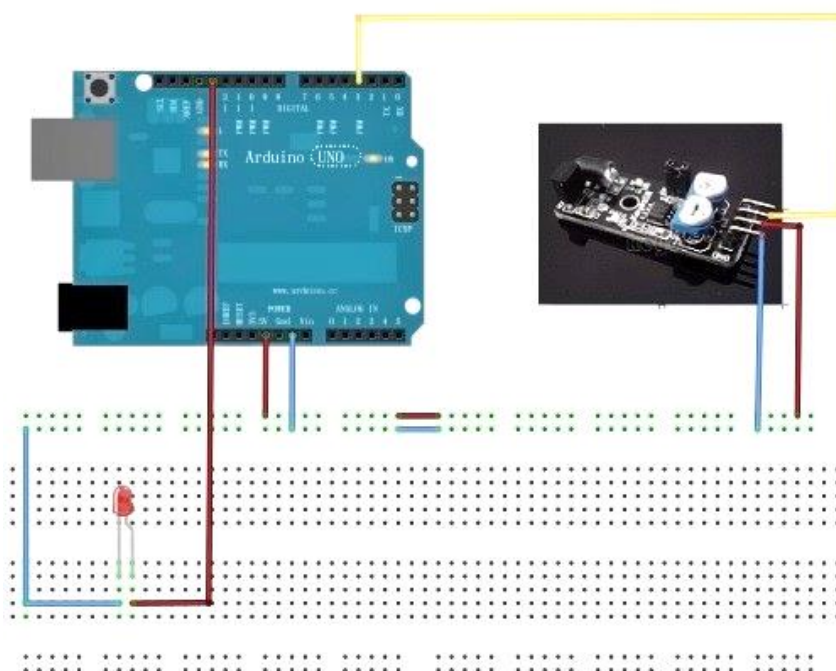
Electrovolt.ir

این ماژول به منظور جلوگیری از برخورد با موانع می‌باشد که در انواع ربات‌ها می‌تواند کاربرد داشته باشد. درون این ماژول یک فرستنده و یک گیرنده مادون قرمز وجود دارد که بررسی کننده موانع می‌باشد. فرستنده مادون قرمز همواره سیگنال ارسال می‌کند. هنگامی که هیچ

جسمی مقابل سنسور قرار نگرفته باشد ، گیرنده مادون قرمز سیگنالی دریافت نمی کند و در نتیجه خروجی ماژول Low می شود. در صورتی که جسمی مقابل سنسور تشخیص داده شود ، سیگنال ارسال شده بازتابش می شود و توسط گیرنده دریافت می گردد و در نتیجه خروجی ماژول High می شود. عملکرد این سنسور مشابه Tracking Sensor بخش بیست و پنجم است با این تفاوت که اولاً در این سنسور ، فرستنده و گیرنده مادون قرمز در بسته بندی جدا از هم بوده و ثانیاً میتواند فاصله بیشتری را پوشش دهند. همچنین دو عدد پتانسیومتر برای تنظیم حساسیت ارسال و دریافت وجود دارد. (شکل زیر)



اتصال به آردوینو:



برنامه آردوینو:

```
int Led = 13 ;// define LED Interface

int buttonpin = 3; // define the obstacle avoidance sensor interface

int val ;// define numeric variables val

void setup ()

{

  pinMode (Led, OUTPUT) ;// define LED as output interface

  pinMode (buttonpin, INPUT) ;// define the obstacle avoidance sensor output interface

}

void loop ()

{

  val = digitalRead (buttonpin) ;// digital interface will be assigned a value of 3 to read val

  if (val == HIGH) // When the obstacle avoidance sensor detects a signal, LED flashes

  {

    digitalWrite (Led, HIGH);

  }

  else

  {

    digitalWrite (Led, LOW);

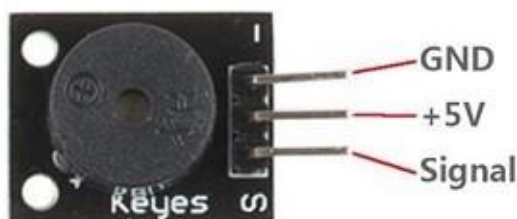
  }

}
```

توضیح برنامه:

به محض شناسایی مانع ، LED روشن می گردد.

بخش سی و دوم : Passive Buzzer



Passive

Electrovolt.ir

این ماژول شامل یک بازر پسیو (غیر فعال) می باشد. بازر پسیو مشابه بازر اکتیو است با این تفاوت که در بازر اکتیو از یک اسیلاتور داخلی برای کنترل صدای بوق استفاده می شود و در نتیجه کنترل آن توسط پایه های بیرونی راحت تر شده است. اما در بازر پسیو هیچ مدار اضافی وجود ندارد و در نتیجه اگر ولتاژ DC قابل توجهی به این بازر اعمال نگردد کار نمی کند. بنابراین برای کار با بازر غیرفعال باید از موج PWM با فرکانس های بین 2 تا 5 کیلوهرتز استفاده کرد. بازرهای فعال به علت استفاده از مدار داخلی کمی گران تر هستند. در شکل زیر تفاوت ظاهری بازرهای پسیو و اکتیو را مشاهده می کنید.

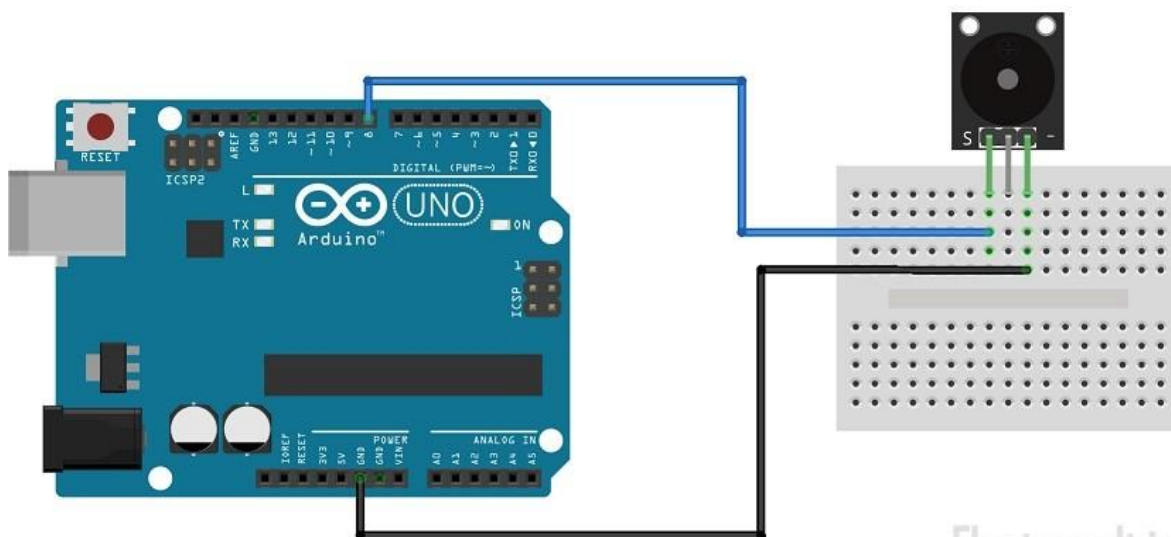


Active

Passive

Electrovolt.ir

اتصال به آردوینو:



Electrovolt.ir

```
//Example Code for KY-006

int buzzer = 8 ;// setting controls the digital IO foot buzzer

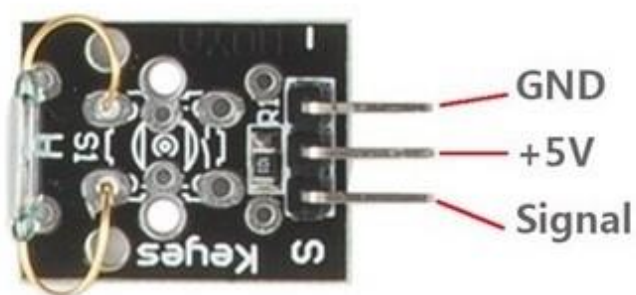
void setup ()
{
  pinMode (buzzer, OUTPUT) ;// set the digital IO pin mode, OUTPUT out of Wen
}

void loop ()
{
  unsigned char i, j ;// define variables
  while (1)
  {
    for (i = 0; i <80; i++) // Wen a frequency sound
    {
      digitalWrite (buzzer, HIGH) ;// send voice
      delay (1) ;// Delay 1ms
      digitalWrite (buzzer, LOW) ;// do not send voice
      delay (1) ;// delay ms
    }
    for (i = 0; i <100; i++) // Wen Qie out another frequency sound
    {
      digitalWrite (buzzer, HIGH) ;// send voice
      delay (2) ;// delay 2ms
      digitalWrite (buzzer, LOW) ;// do not send voice
      delay (2) ;// delay 2ms
    }
  }
}
```

}
توضیح برنامه:

در این برنامه توسط حلقه for و به صورت نرم افزاری دو موج PWM متفاوت ایجاد و اعمال شده است.

بخش سی و سوم : Mini Reed Switch



Mini Reed

Electrovolt.ir

این ماژول دارای یک رید سوئیچ تکی می باشد. نحوه عملکرد آن مشابه توضیحات بخش بیست و هفتم می باشد. تفاوت بین آنها یکی این است که در این سنسور از مدار کوچکتري استفاده شده است و دیگری این است که این سنسور دارای پتانسیومتر جهت تنظیم حساسیت نمی باشد و تنها میدان های مغناطیسی قوی را شناسایی می کند.

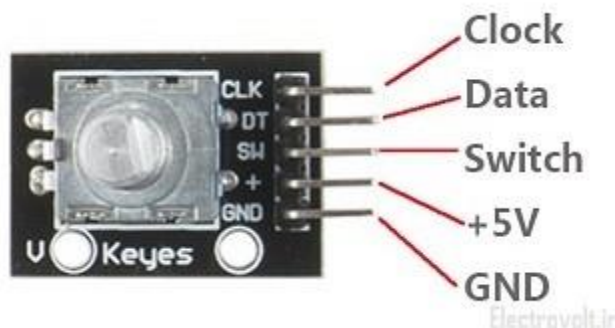
اتصال به آردوینو:

همانند بخش بیست و هفتم

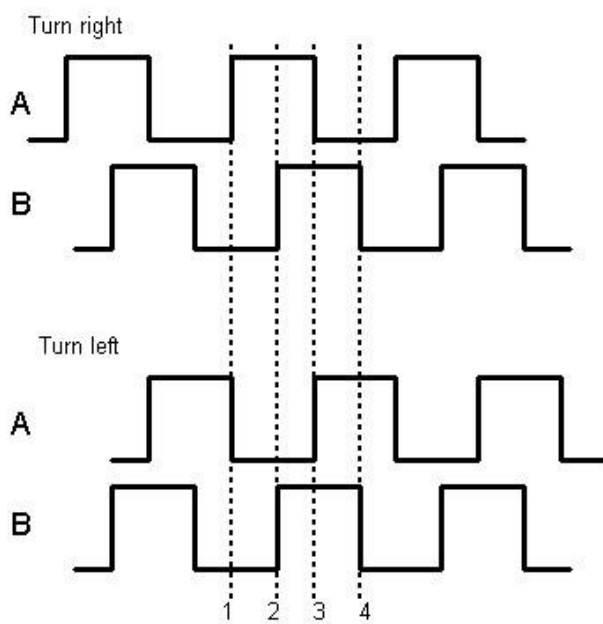
برنامه آردوینو:

همانند بخش بیست و هفتم

بخش سی و چهارم : Rotary Encoder

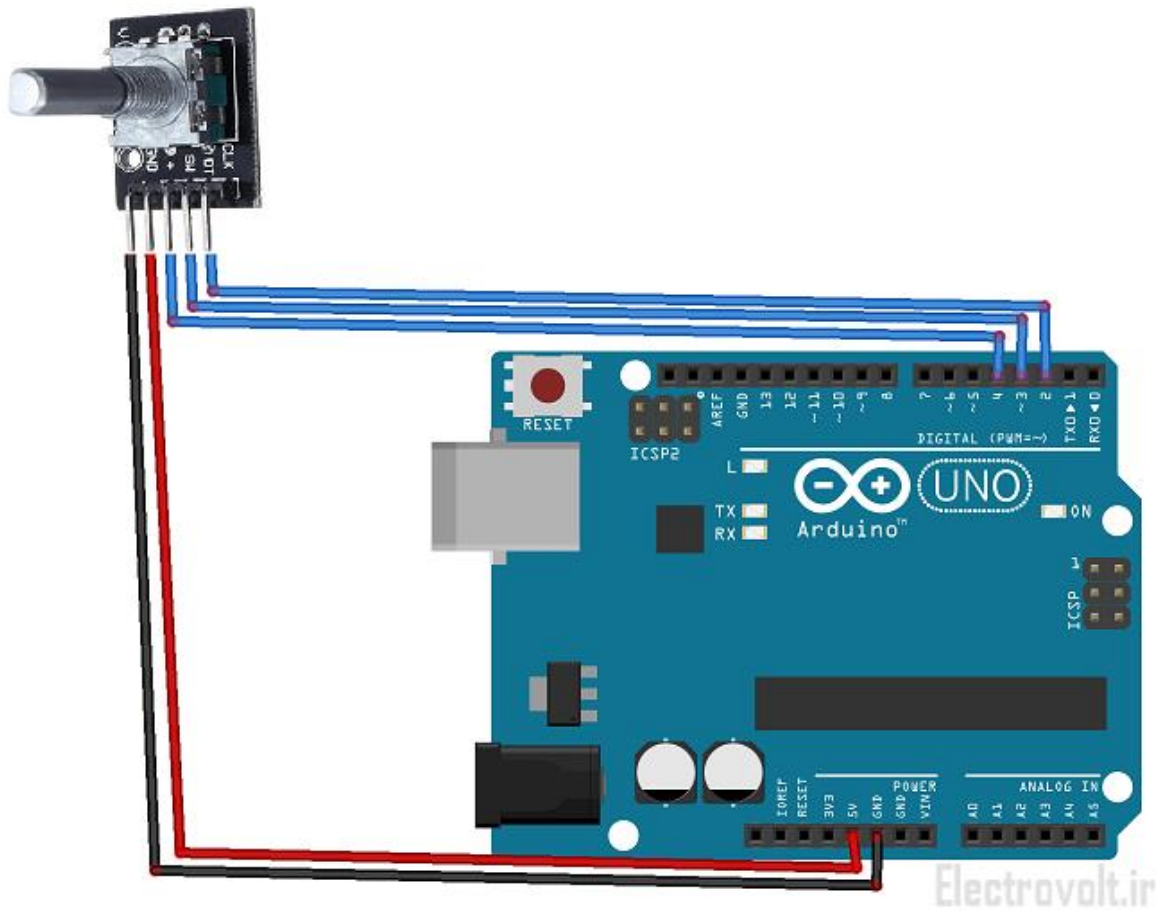


این ماژول دارای یک انکودر چرخشی و یک محور فلزی (شبیه به پتانسیومتر) می باشد. این ماژول در کل یک وسیله الکترومکانیکی می باشد که موقعیت زاویه ای یا سرعت چرخش محور شفت خود را به صورت دیجیتالی کدگذاری می کند. از جمله کاربردهای آن می توان به کاربرد در کنترل صنعتی انواع موتورها ، در روباتیک و برآورده ساختن بعضی از هدف های لنزهای عکاسی و همچنین در دستگاه های ورودی رایانه نام برد. یک انکودر (Encoder) از نوع افزایشی داخل و عمود بر محور چرخش قرار گرفته است که با توجه به مقدار چرخش و سرعت چرخش ، دو سیگنال دیجیتالی A و B را به خروجی ارسال می کند. این دو سیگنال A و B همواره اختلاف فاز 90 درجه با یکدیگر دارند. یک سوئیچ فشاری درون این ماژول قرار دارد که به محض فشار دادن عمل می کند (پایه SW) وظیفه این سوئیچ صفر کردن زاویه است به طوری که در ابتدای روشن شدن جهت شفت را در نقطه صفر دلخواه قرار می دهیم و یک بار آن را فشار می دهیم. سپس بسته به اینکه انکودر در جهت راست بچرخد یا در جهت چپ ، سیگنال های A و B شکل متفاوتی خواهد داشت. همانطور که در شکل زیر هم مشاهده می کنید ، در صورتی که چرخش به سمت راست (ساعتگرد) باشد ، سیگنال A به اندازه 90 درجه جلوتر از سیگنال B است و در صورتی که چرخش به سمت چپ (پاد ساعتگرد) باشد ، سیگنال B به اندازه 90 درجه جلوتر از سیگنال A است.



نکته : در این ماژول A همان پایه CLK (Clock) و B همان پایه DT (Data) است.

اتصال به آردوینو:



برنامه آردوینو:

```
int redPin = 2;
int yellowPin = 3;
int greenPin = 4;
int aPin = 6;
int bPin = 7;
int buttonPin = 5;
int state = 0;
int longPeriod = 5000; // Time at green or red
int shortPeriod = 700; // Time period when changing
```

```
int targetCount = shortPeriod;
int count = 0;
void setup ()
{
  pinMode (aPin, INPUT);
  pinMode (bPin, INPUT);
  pinMode (buttonPin, INPUT);
  pinMode (redPin, OUTPUT);
  pinMode (yellowPin, OUTPUT);
  pinMode (greenPin, OUTPUT);
}
void loop ()
{
  count++;
  if (digitalRead (buttonPin))
  {
    setLights (HIGH, HIGH, HIGH);
  }
  else
  {
    int change = getEncoderTurn ();
    int newPeriod = longPeriod + (change * 1000);
    if (newPeriod >= 1000 && newPeriod <= 10000)
    {
      longPeriod = newPeriod;
    }
    if (count > targetCount)
```

```
{
  setState ();
  count = 0;
}
}

delay (1);
}

int getEncoderTurn ()
{
  // Return -1, 0, or +1
  static int oldA = LOW;
  static int oldB = LOW;

  int result = 0;

  int newA = digitalRead (aPin);
  int newB = digitalRead (bPin);
  if (newA != oldA || newB != oldB)
  {
    //Something has changed
    if (oldA == LOW && newA == HIGH)
    {
      result = - (oldB * 2 - 1);
    }
  }

  oldA = newA;
  oldB = newB;
  return result;
}
```

```
int setState ()
{
    if (state == 0)
    {
        setLights (HIGH, LOW, LOW);
        targetCount = longPeriod;
        state = 1;
    }
    else if (state == 1)
    {
        setLights (HIGH, HIGH, LOW);
        targetCount = shortPeriod;
        state = 2;
    }
    else if (state == 2)
    {
        setLights (LOW, LOW, HIGH);
        targetCount = longPeriod;
        state = 3;
    }
    else if (state == 3)
    {
        setLights (LOW, HIGH, LOW);
        targetCount = shortPeriod;
        state = 0;
    }
}
```

```
void setLights (int red, int yellow, int green)
{
digitalWrite (redPin, red);
digitalWrite (yellowPin, yellow);
digitalWrite (greenPin, green);
}
```

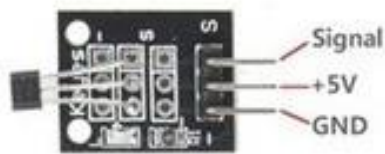
توضیح برنامه:

در این برنامه برای نشان دادن وضعیت انکودر چرخشی سه LED با سه رنگ متفاوت (یا یک LED سه رنگ) استفاده شده است. همچنین 5 وضعیت تعریف شده است. وضعیت اول در نقطه 0 شروع است. با فشردن دکمه برنامه به این وضعیت می رود و در این وضعیت هر سه LED روشن است.

بخش سی و پنجم : Analog Hall Sensor



عملکرد این سنسور همانند بخش ششم می باشد با این تفاوت که علاوه بر خروجی دیجیتال ، دارای خروجی آنالوگ نیز می باشد. در این ماژول از سنسور اثر هال مدل 49E به همراه مدار جانبی استفاده شده است. این سنسور در هنگام قرار گرفتن در میدان مغناطیسی ولتاژ آنالوگ متناسب با قدرت آن میدان تولید می کند. ولتاژ آنالوگ خروجی پس از دریافت توسط آردوینو به دیجیتال تبدیل می گردد. تفاوت این ماژول با ماژول بخش هشتم در این است که اولاً دارای خروجی آنالوگ به تنهایی است (خروجی دیجیتال و LED مربوطه را ندارد) و ثانیاً پتانسیومتر برای تنظیم حساسیت خروجی دیجیتال وجود ندارد. در شکل زیر پایه های این ماژول را مشاهده می کنید.



Linear Hall

Electrovolt.ir

اتصال به آردوینو:

پایه خروجی سیگنال سنسور را به پایه A5 آردوینو و پایه های تغذیه سنسور را به پایه های GND و +5V آردوینو متصل می کنیم.

برنامه آردوینو:

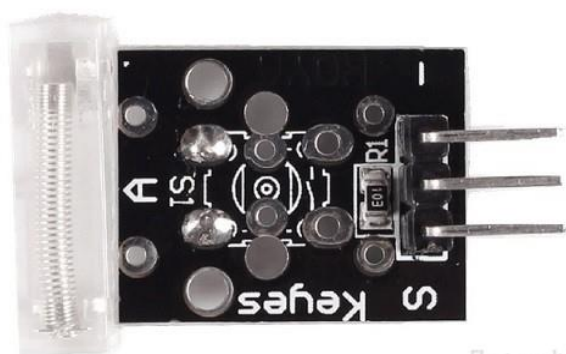
```
/*  
KY-035 Hall analog sensor  
*/  
  
int sensorPin = A5; // select the input pin  
int ledPin = 13; // select the pin for the LED  
int sensorValue = 0; // variable to store the value coming from the sensor  
  
void setup () {  
  pinMode (ledPin, OUTPUT);  
  Serial.begin (9600);  
}  
  
void loop () {  
  sensorValue = analogRead (sensorPin);  
  digitalWrite (ledPin, HIGH);  
  delay (sensorValue);  
}
```

```
digitalWrite (ledPin, LOW);
delay (sensorValue);
Serial.println (sensorValue, DEC);
}
```

توضیح برنامه:

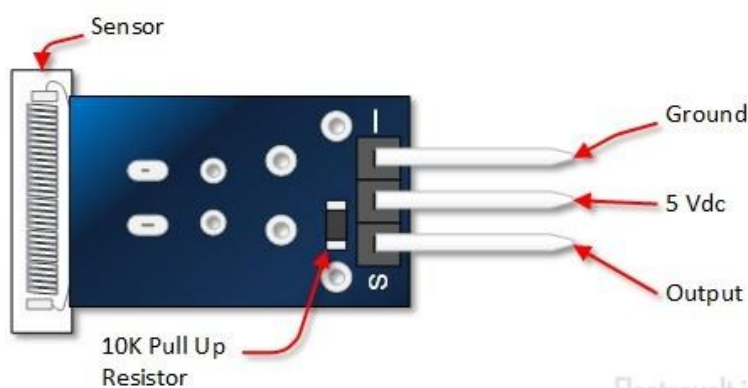
در این برنامه مقدار آنالوگ از پایه سیگنال سنسور خوانده شده و به صورت دسیمال (مبنای 10) به کامپیوتر ارسال می گردد. درون نرم افزار Arduino IDE با کلیک بر روی سریال مانیتور (Tools -> Serial Monitor) میتوانید مقادیر خوانده شده این سنسور را ببینید.

بخش سی و ششم : Knock Sensor



Electrovolt.ir

این سنسور شامل یک فنر است که داخل یک باکس شیشه ای قرار گرفته است. کاربرد این سنسور همانند سنسور شوک است که در بخش بیست و هشتم بررسی شد با این تفاوت که این سنسور میتواند لرزش های بسیار کوچک را نیز شناسایی نماید. در شکل زیر پایه های این سنسور را مشاهده می کنید.



Electrovolt.ir

اتصال به آردوینو:

پایه خروجی سنسور را به پایه 3 آردوینو و پایه های - و + را به ترتیب به GND و 5V آردوینو متصل می کنیم.

برنامه آردوینو:

```
const int buttonPin = 3; // the pin that the pushbutton is attached to
const int ledPin = 13; // the pin that the LED is attached to
// Variables will change:
int buttonPushCounter = 0; // counter for the number of button presses
int buttonState = 0; // current state of the button
int lastButtonState = 0; // previous state of the button
void setup() {
  // initialize the button pin as a input:
  pinMode(buttonPin, INPUT);
  // initialize the LED as an output:
  pinMode(ledPin, OUTPUT);
  // initialize serial communication:
  Serial.begin(9600);
}
void loop() {
  // read the pushbutton input pin:
  buttonState = digitalRead(buttonPin);
  // compare the buttonState to its previous state
  if (buttonState != lastButtonState) {
    // if the state has changed, increment the counter
    if (buttonState == HIGH) {
      // if the current state is HIGH then the button
```

```
// wend from off to on:
buttonPushCounter++;
Serial.println("on");
Serial.print("number of button pushes: ");
Serial.println(buttonPushCounter);
}
else {
// if the current state is LOW then the button
// wend from on to off:
Serial.println("off");
}
}

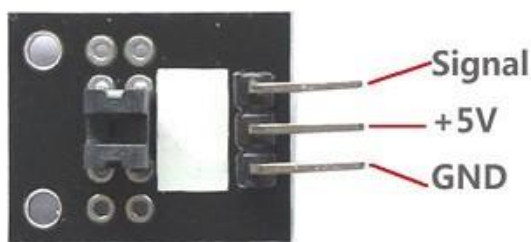
// save the current state as the last state,
//for next time through the loop
lastButtonState = buttonState;

// turns on the LED every four button pushes by
// checking the modulo of the button push counter.
// the modulo function gives you the remainder of
// the division of two numbers:
if (buttonPushCounter % 4 == 0) {
digitalWrite(ledPin, HIGH);
}
else
{
digitalWrite(ledPin, LOW);
}
}
```

توضیح برنامه:

در این برنامه منطق خروجی سنسور خوانده می شود و در صورت تغییر منطق وارد حلقه if می شود. اگر منطق خوانده شده Low باشد ، Off و اگر منطق خوانده شده High باشد on به کامپیوتر ارسال می گردد. همچنین تعداد دفعاتی که منطق High شده است شمرده شده و به کامپیوتر ارسال می گردد. درون نرم افزار Arduino IDE با کلیک بر روی سریال مانیتور (Tools -> Serial Monitor) میتوانید مقادیر خوانده شده این سنسور را ببینید.

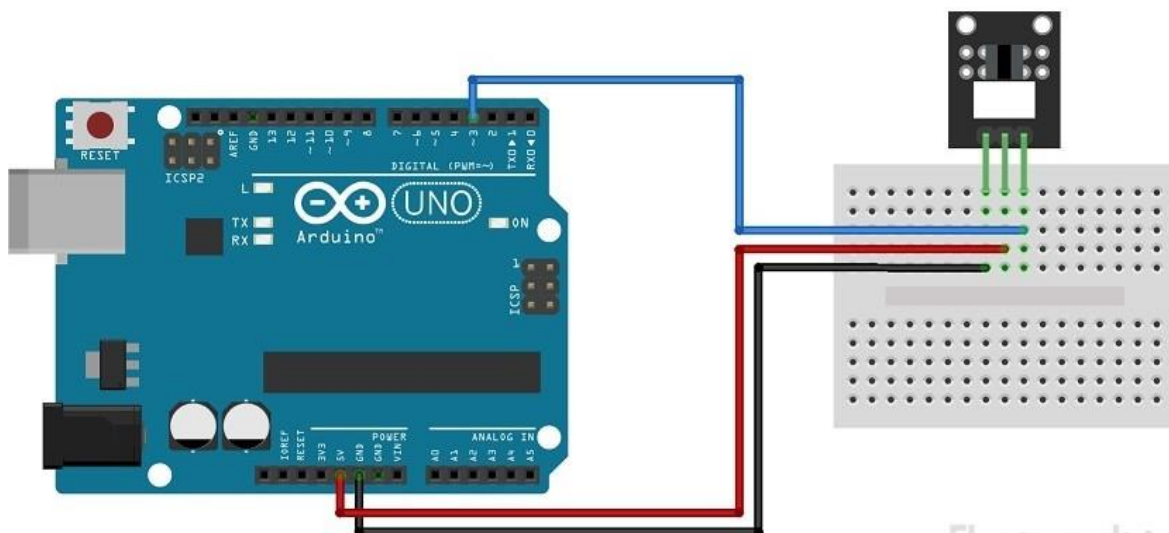
بخش سی و هفتم : Light Blocking Sensor



Electrovolt.ir

این سنسور میتواند وجود مانع بین دو پره خود را تشخیص دهد. در حالت عادی خروجی سنسور High است و زمانی که جسمی در وسط سنسور (در وسط U شکل) قرار گیرد ، سنسور Low می شود.

اتصال به آردوینو:



Electrovolt.ir

```
// set pin numbers:

const int buttonPin = 2; // the number of the pushbutton pin

const int ledPin = 13; // the number of the LED pin

// variables will change:

int buttonState = 0; // variable for reading the pushbutton status

void setup() {

  // initialize the LED pin as an output:

  pinMode(ledPin, OUTPUT);

  // initialize the pushbutton pin as an input:

  pinMode(buttonPin, INPUT);

}

void loop(){

  // read the state of the pushbutton value:

  buttonState = digitalRead(buttonPin);

  // check if the pushbutton is pressed.

  // if it is, the buttonState is HIGH:

  if (buttonState == HIGH) {

    // turn LED on:

    digitalWrite(ledPin, HIGH);

  }

  else

  {

    // turn LED off:

    digitalWrite(ledPin, LOW);

  }

}
```

توضیح برنامه:

در صورت تشخیص مانع ، LED روشن می گردد.

پایان

آموزش الکترونیک برای همه

Electro Volt.ir



Electrovolt_ir



Electrovolt.ir