

Keyestudio New sensor kit

36 in 1 box for Arduino starters



Sensor kit for Arduino

Based on open-source hardware

36 various sensors in one box

For you to make interesting projects

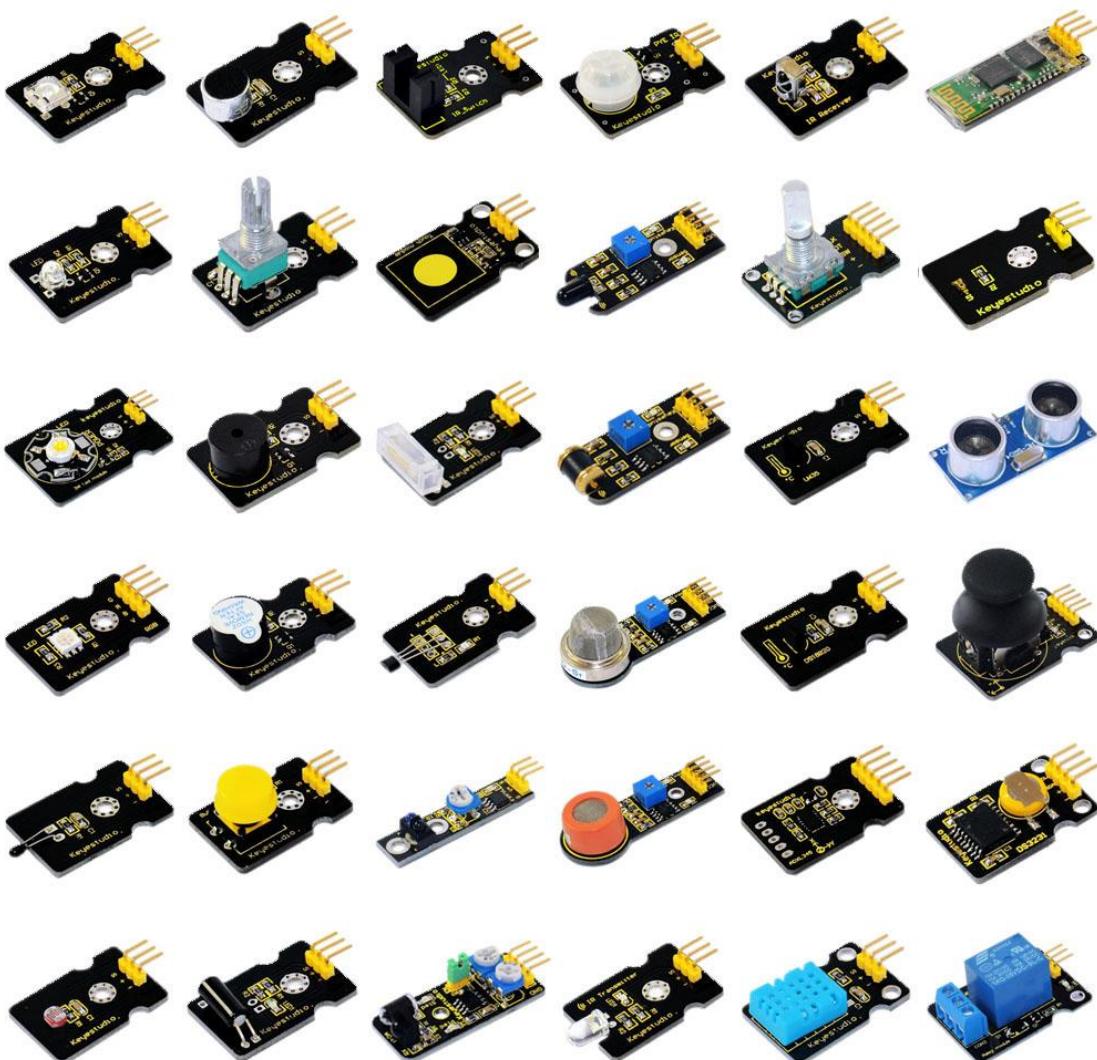
keystudio

1. Summary:

This is an Arduino sensor learning kit developed by Keyestudio. We bring together 36 basic sensors and modules, aiming for the convenience of its learning for starters. Inside this box, there are digital and analog sensors and also some special modules such as ultrasonic, bluetooth,, acceleration modules etc. For each module, there is clear connection diagram and sample code. So even if you are totally new at this, you can get started easily.

The sample codes for this sensor kit are based on ARDUINO because it's open source and easy. And if you are good at this, you can also apply this kit to other MCU development platform, such as 51, STM32, Raspberries Pi. The working principle is pretty much the same.

Now, let us embrace this fascinating world of ARDUINO and learn together!



2. Kit list

- 1: Digital White LED Light Module project
- 2: Piranha LED Light Module
- 3: 3W LED Module
- 4: RGB LED module
- 5: Analog temperature sensor
- 6: Photocell sensor
- 7: Analog Sound Sensor
- 8: Analog Rotation Sensor
- 9: Passive Buzzer module
- 10: Digital Buzzer Module
- 11: Digital Push Button
- 12: Digital Tilt Sensor
- 13: photo interrupter module
14. Capacitive Touch Sensor
- 15: Knock Sensor Module
- 16: Hall Magnetic Sensor
- 17: Line Tracking Sensor
- 18: Infrared Obstacle Avoidance Sensor
- 19: PIR Motion Sensor
- 20: Flame Sensor
- 21: Vibration Sensor
- 22: Analog Gas Sensor
- 23: Analog Alcohol Sensor
- 24: Digital IR Transmitter Module
- 25: Digital IR Receiver Module
- 26: Rotary Encoder module
- 27: LM35 Temperature Sensor
- 28: 18B20 Temperature Sensor
- 29: ADXL345 Three Axis Acceleration Module
- 30: DHT11 Temperature and Humidity Sensor
- 31: Bluetooth Module
- 32: TEMT6000 ambient light sensor
- 33: HC-SR04 Ultrasonic Sensor
- 34: Joystick Module
- 35: DS3231 Clock Module
- 36: 5V Relay Module

3. Project details:

Project 1: LED Light Module



Introduction:

This LED light module has a shiny color, ideal for Arduino starters. It can be easily connected to IO/Sensor shield.

Specification:

Type: Digital

PH2.54 socket

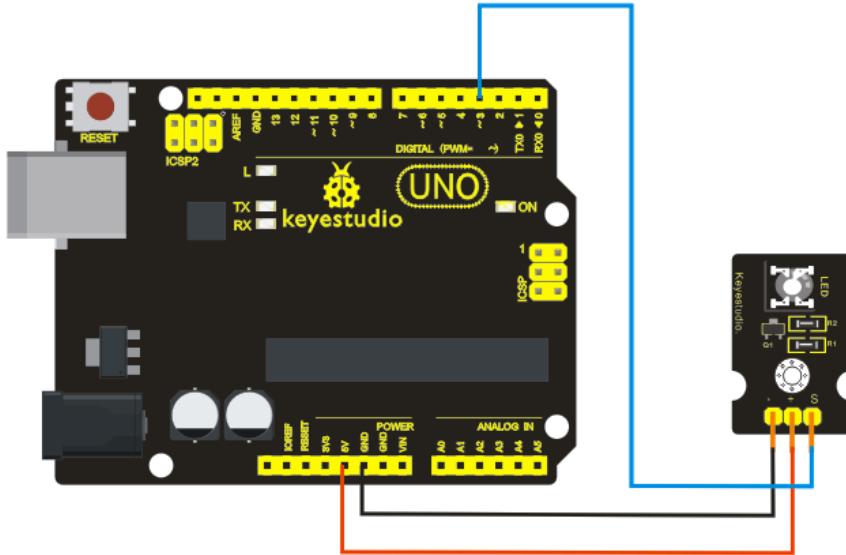
White LED light module

Enables interaction with light-related works

Size: 30*20mm

Weight: 3g

Connection Diagram:



Sample Code:

```
int led = 3;  
void setup()  
{  
    pinMode(led, OUTPUT); //Set Pin3 as output  
}  
void loop()  
{  
    digitalWrite(led, HIGH); //Turn on led  
    delay(2000);  
    digitalWrite(led, LOW); //Turn off led  
    delay(2000);  
}  
*****
```

Project 2: Piranha LED Light Module



Introduction:

This is a special LED module. When you connect it to ARDUINO development board, after program, it can emit beautiful light. Of course, you can also control it using PWM. It will be like fireflies at night. Isn't cool? We can also combine it with other sensors to do various interesting interactive experiments.

Specifications:

Module type: digital

Working voltage: 5v

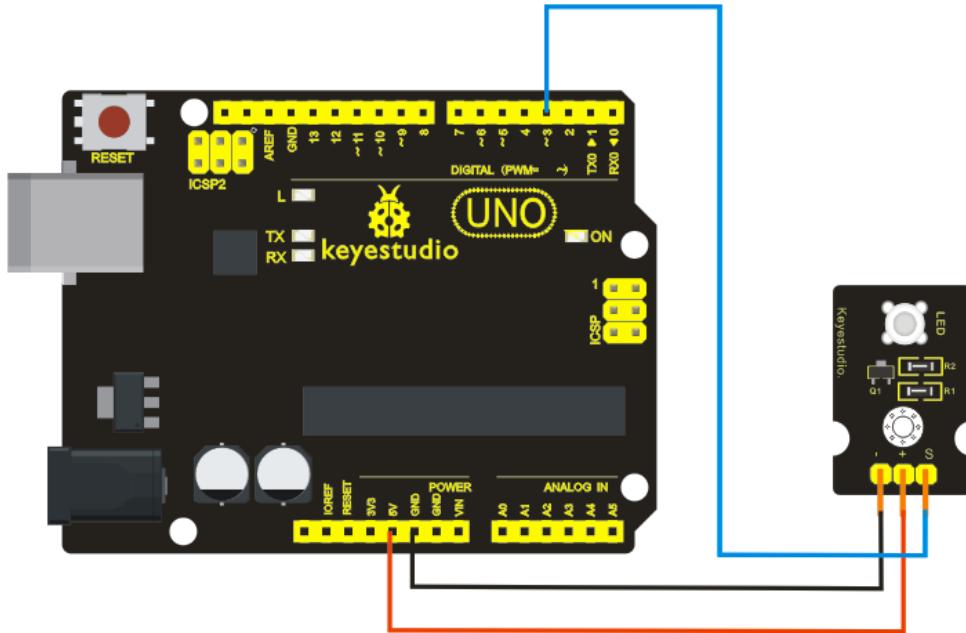
Distance between pins: 2.54mm

Size: 30*20mm

Weight: 3g

keyestudio

Connection Diagram:



Sample Code:

```
int led = 3;  
void setup()  
{  
    pinMode(led, OUTPUT); //Set Pin3 as output  
}  
void loop()  
{  
    digitalWrite(led, HIGH); //Turn off led  
    delay(2000);  
    digitalWrite(led, LOW); //Turn on led  
    delay(2000);  
}  
*****
```

Project 3: 3W LED Module



Introduction:

This LED module is of high brightness because the lamp beads it carries is 3w. We can apply this module to Arduino projects. For example, intelligent robots can use this module for illumination purpose.

Please note that the LED light can't be exposed directly to human eyes for safety concerns.

Specification:

Color temperature: 6000~7000K

Luminous flux: 180~210lm

Current: 700~750mA

Power: 3W

Light angle: 140 degree

Working temperature: -50~80'C

Storage temperature: -50~100'C

High power LED module, controlled by IO port microcontroller

Great for Robot and search & rescue platform application

keystudio

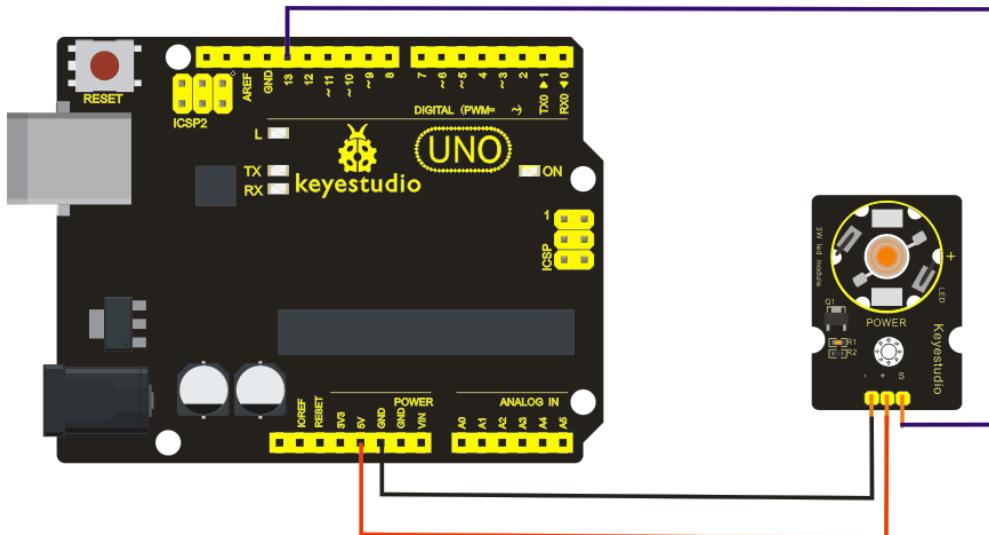
IO Type: Digital

Supply Voltage: 3.3V to 5V

Size: 40x28mm

Weight: 6g

Connection Diagram:

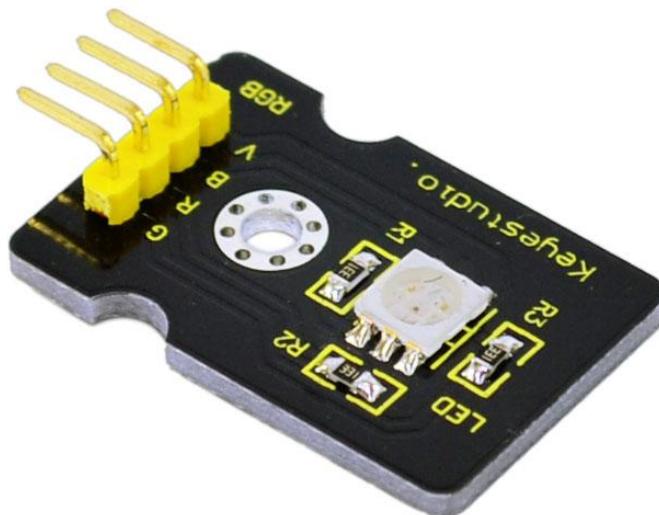


Sample Code:

```
// the setup function runs once when you press reset or power the board
void setup() {
    // initialize digital pin 13 as an output.
    pinMode(13, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
    digitalWrite(13, HIGH);      // turn the LED on (HIGH is the voltage level)
    delay(1000);                // wait for a second
    digitalWrite(13, LOW);       // turn the LED off by making the voltage LOW
    delay(1000);                // wait for a second
}
*****
```

Project 4: RGB LED module



Introduction:

This is a full-color LED module, which contains 3 basic colors—red, green and blue. They can be seen as separate LED lights. After program, we can turn them on and off by sequence. We can also use PWM analog output to mix the three colors to generate different colors.

Specification:

Color: red, green and blue

Brightness: High

Voltage: 5V

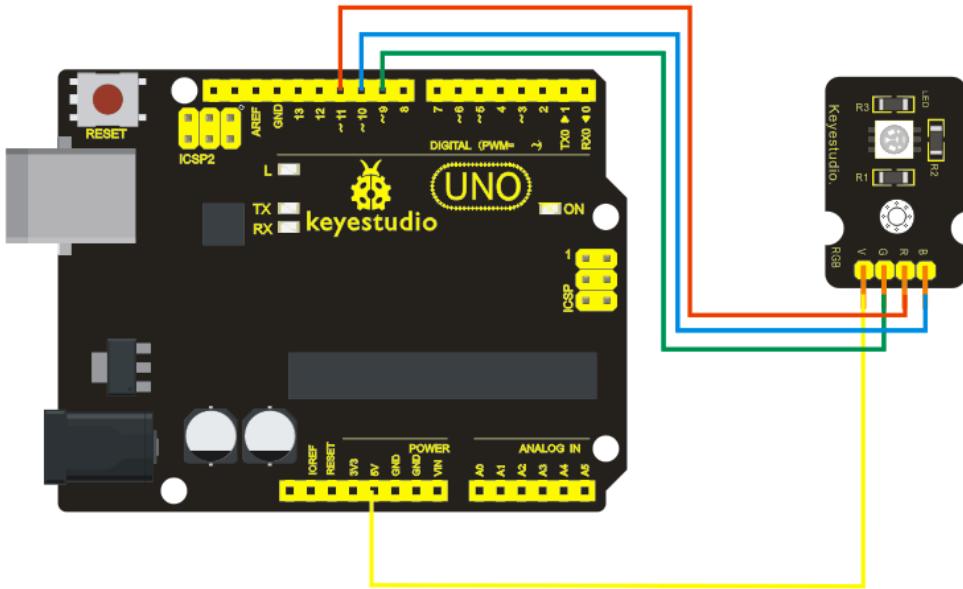
Input: digital level

Size: 30 * 20mm

Weight: 3g

Connection diagram:

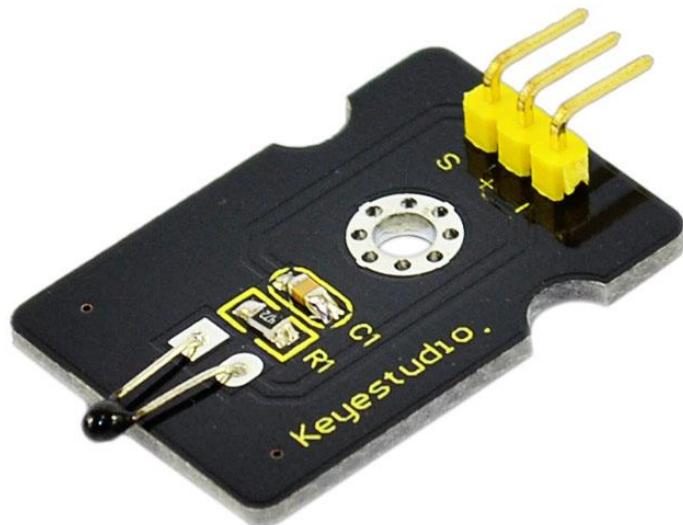
keystudio



Sample code:

```
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);
}
void loop()
{for(val=255; val>0; val--)
    {analogWrite(11, val);
     analogWrite(10, 255-val);
     analogWrite(9, 128-val);
     delay(1);
    }
for(val=0; val<255; val++)
    {analogWrite(11, val);
     analogWrite(10, 255-val);
     analogWrite(9, 128-val);
     delay(1); }}
```

Project 5: Analog temperature sensor



Introduction:

This module is based on the working principle of a thermistor (resistance varies with temperature change in the environment). It can sense temperature change in its surrounding and send the data to the analog IO in the Arduino board. All we need to do is to convert the sensor output data to degrees Celsius temperature by simple programming and display it. It's both convenient and effective, and it's widely applied in gardening, home alarm system and other devices.

Specification:

Interface type: analog

Working voltage: 5V

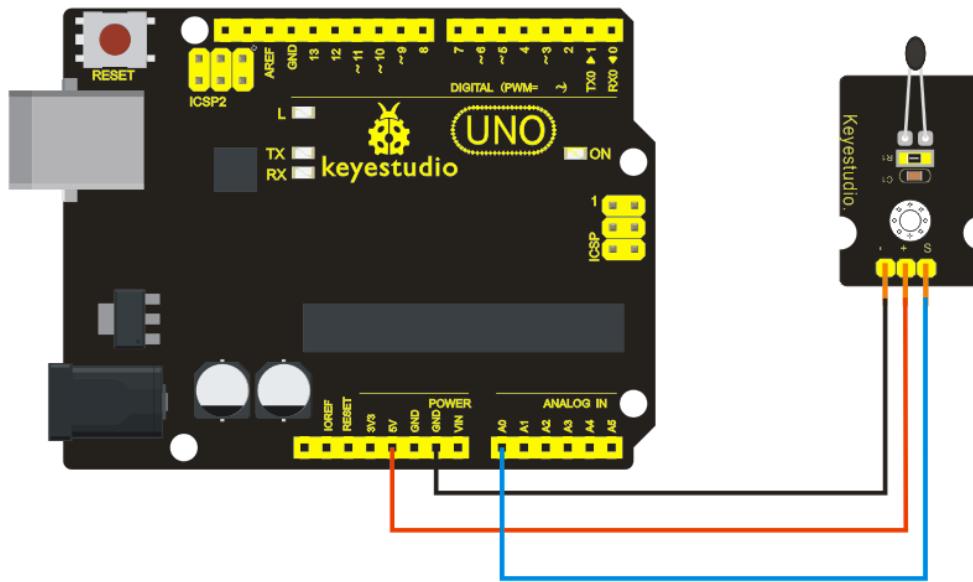
Temperature range: -55 °C ~ 315 °C

Size: 30*20mm

keystudio

Weight: 3g

Connection Diagram:



Sample Code:

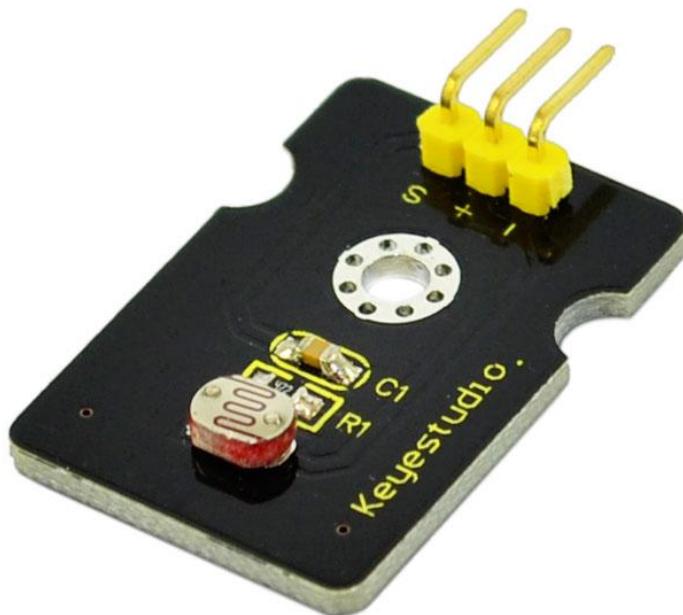
```
void setup()
{Serial.begin(9600);
}
// the loop routine runs over and over again forever:
void loop()
{int sensorValue = analogRead(A0);
Serial.println(sensorValue);
delay(1);}
```

We can see that the analog value is changing according to the temperature change in the environment. But it's not very obvious. Let's solve this by using the following equation. The value read from the serial port is similar to normal temperature, eg. The temperature right now is 30C.

```
#include <math.h>
double Thermister(int RawADC) {
double Temp;
Temp = log(((10240000/RawADC) - 10000));
Temp = 1 / (0.001129148 + (0.000234125 + (0.0000000876741 * Temp * Temp ))* Temp );
Temp = Temp - 273.15; // Convert Kelvin to Celcius
return Temp;
}
void setup()
{Serial.begin(9600);
```

```
} void loop() { Serial.print(Thermister(analogRead(0))); // display Fahrenheit Serial.println("c");
delay(500); }
*****
```

Project 6: Photocell sensor



Introduction:

Photocell is commonly seen in our daily life and is mainly used in intelligent switch, also in common electronic design. To make it easier and more effective, we supply corresponding modules.

Photocell is a semiconductor. It has features of high sensitivity, quick response, spectral characteristic, and R-value consistence, maintaining high stability and reliability in environment extremes such as high temperature, high humidity. It's widely used in automatic control switch fields like cameras, garden solar lights, lawn lamps, money detectors, quartz

keystudio

clocks, music cups, gift boxes, mini night lights, sound and light control switches, etc.

Specification:

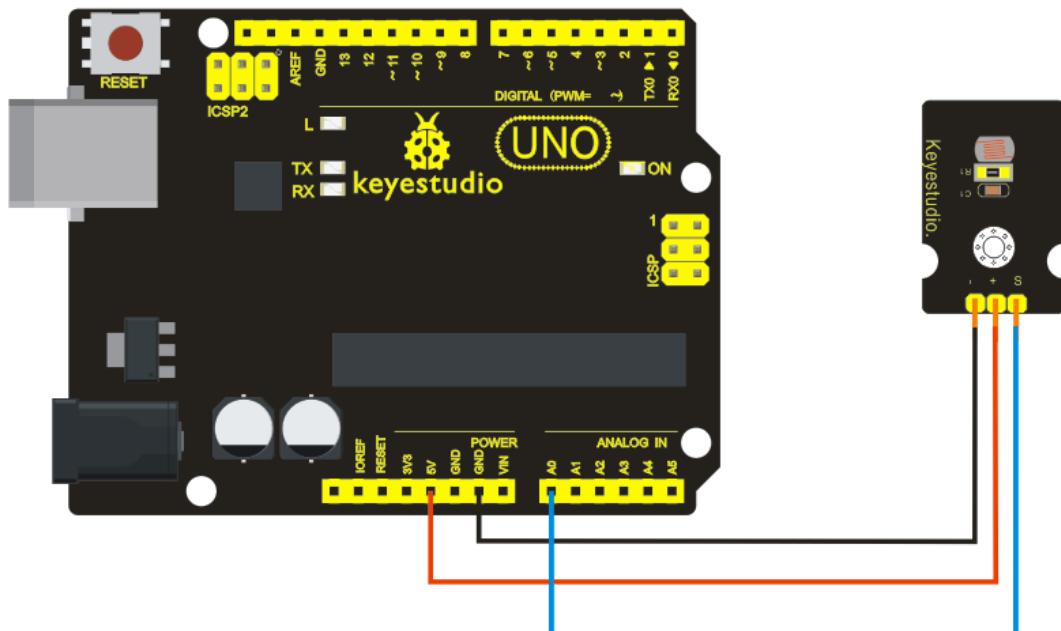
Interface type: analog

Working voltage: 5V

Size: 30*20mm

Weight: 3g

Connection Diagram:



Sample Code:

```
int sensorPin =A0 ;
int value = 0;
void setup()
{
    Serial.begin(9600);
}
void loop()
{
    value = analogRead(sensorPin);

    Serial.println(value, DEC);
```

```
delay(50); }
```

```
*****
```

Project 7: Analog Sound Sensor



Introduction:

Analog Sound Sensor is typically used in detecting the loudness in ambient environment. The Arduino can collect its output signal by imitating the input interface. You can use it to make some interesting interactive works such as a voice operated switch.

Specification:

Supply Voltage: 3.3V to 5V

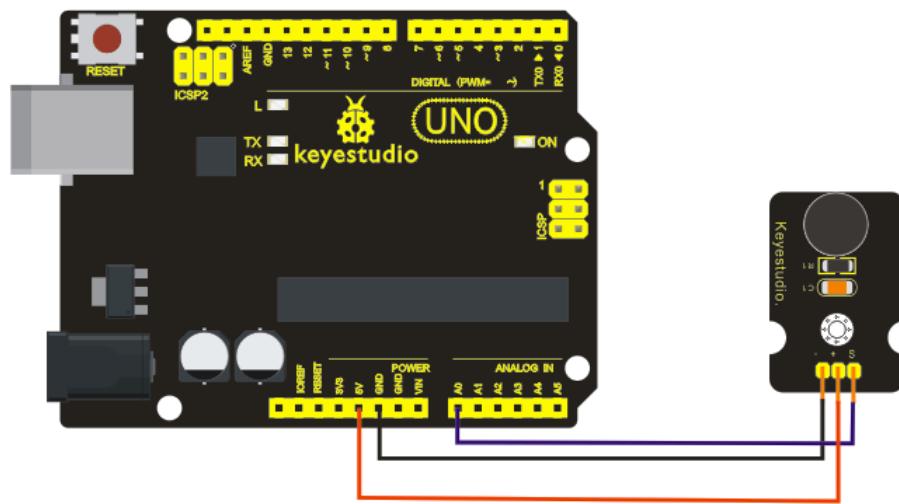
Detecting sound intensity

Interface: Analog

Size: 30*20mm

Weight: 4g

Connection Diagram:

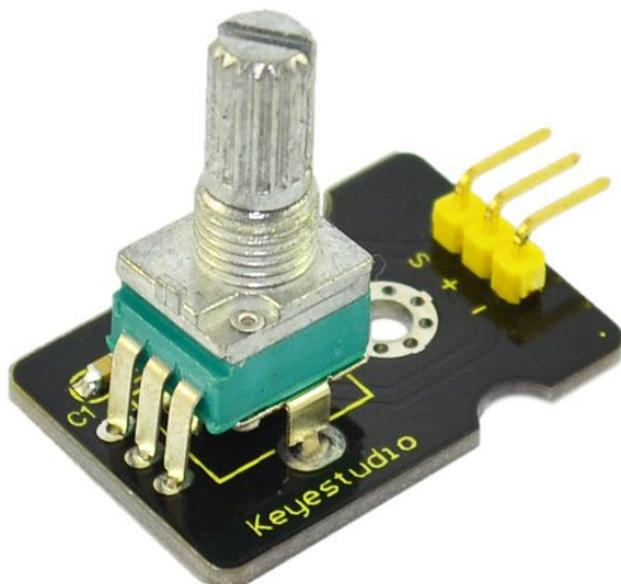


Sample Code:

```
void setup()
{
    Serial.begin(9600); // open serial port, set the baud rate to 9600 bps
}

void loop()
{
    int val;
    val=analogRead(0); //connect mic sensor to Analog 0
    Serial.println(val,DEC);//print the sound value to serial
    delay(100);
}
*****
```

Project 8: Analog Rotation Sensor



Introduction:

This analog Rotation Sensor is arduino compatible. It is based on a potentiometer. Its voltage can be subdivided into 1024, easy to be connected to Arduino with our sensor shield. Combined with other sensors, we can make interesting projects by reading the analog value from the IO port.

Specification:

Supply Voltage: 3.3V to 5V

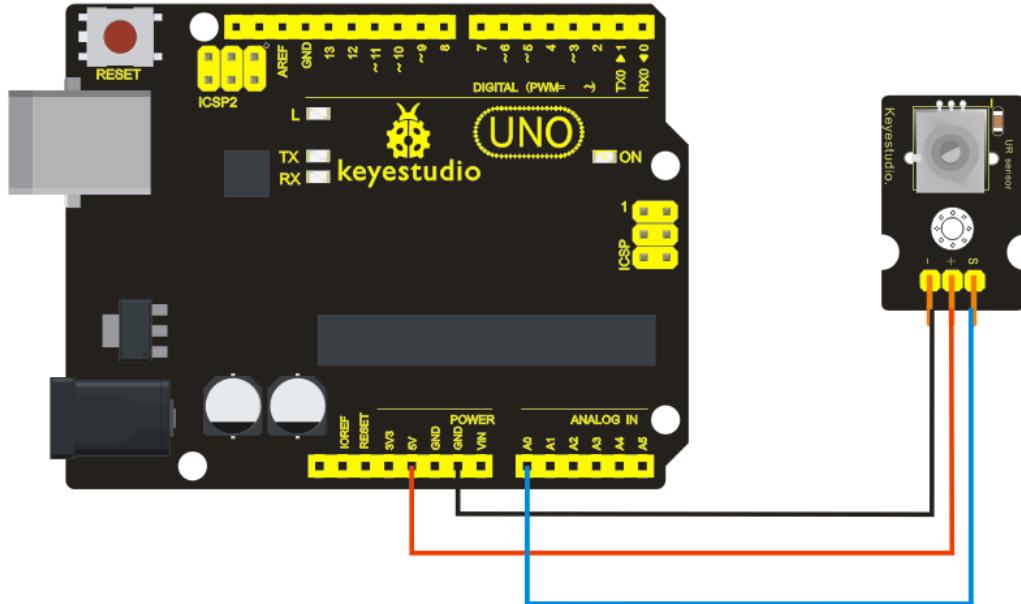
Interface: Analog

Size: 30*20mm

Weight: 8g

keystudio

Connection Diagram:



Sample Code:

```
///Arduino Sample Code
void setup()
{
    Serial.begin(9600); //Set serial baud rate to 9600 bps
}
void loop()
{
    int val;
    val=analogRead(0);//Read rotation sensor value from analog 0
    Serial.println(val,DEC);//Print the value to serial port
    delay(100);
}
*****
```

Project 9: Passive Buzzer module



ntroduction:

We can use Arduino to make many interactive works of which the most commonly used is acoustic-optic display. All the previous experiment has something to do with LED. However, the circuit in this experiment can produce sound. Normally, the experiment is done with a buzzer or a speaker while buzzer is simpler and easier to use. The buzzer we introduced here is a passive buzzer. It cannot be actuated by itself, but by external pulse frequencies. Different frequencies produce different sounds. We can use Arduino to code the melody of a song, which is actually quite fun and simple.

Specification:

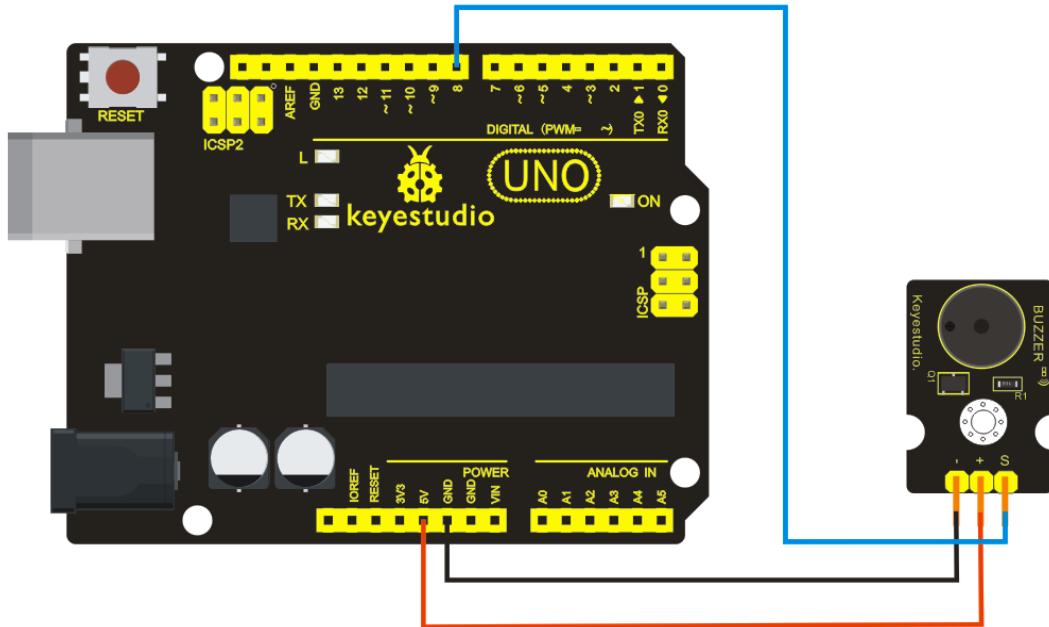
Working voltage: 3.3-5v

Interface type: digital

Size: 30*20mm

Weight: 4g

Connection Diagram:



Sample Code:

```
int buzzer=8;//set digital IO pin of the buzzer
void setup()
{
pinMode(buzzer,OUTPUT);// set digital IO pin pattern, OUTPUT to be output
}
void loop()
{ unsigned char i,j;//define variable
while(1)
{ for(i=0;i<80;i++)// output a frequency sound
{ digitalWrite(buzzer,HIGH);// sound
delay(1);//delay1ms
digitalWrite(buzzer,LOW);//not sound
delay(1);//ms delay
}
for(i=0;i<100;i++)// output a frequency sound
{
digitalWrite(buzzer,HIGH);// sound
digitalWrite(buzzer,LOW);//not sound
delay(2);//2ms delay
}}}
```

After downloading the program, buzzer experiment will been finished.

Project 10: Digital Buzzer Module



Introduction:

Here is the simplest sound making module. You can use high/low level to drive it. Changing the frequency it buzzes can produce different sound. This module is widely used on your daily appliance, like PC, refrigerator, phones etc. And you can also create many interesting interactive project with this small but useful module.

Just try it!! You will find the electronic sound it creates so fascinating.

Specification:

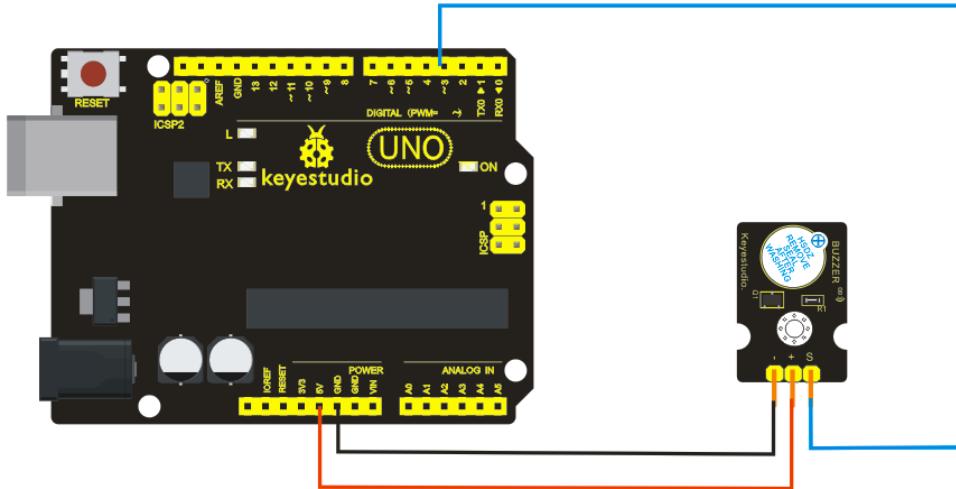
Working voltage: 3.3-5v

Interface type: digital

Size: 30*20mm

Weight: 4g

Connection Diagram:



Sample Code:

```
int buzzPin = 3; //Connect Buzzer on Digital Pin3
void setup()
{
    pinMode(buzzPin, OUTPUT);
}
void loop()
{
    digitalWrite(buzzPin, HIGH);
    delay(1);
    digitalWrite(buzzPin, LOW);
    delay(1);
}
*****
```

Project 11: Digital Push Button



Introduction:

This is a basic application module. You can simply plug it into an IO shield to have your first taste of Arduino.

Advantages:

Wide voltage range from 3.3V to 5V

Standard assembling structure (two 3mm diameter holes with multiple of 5mm as distance from center)

Easily recognizable interfaces of sensors ("A" for analog and "D" for digital)

Icons illustrate sensor function clearly

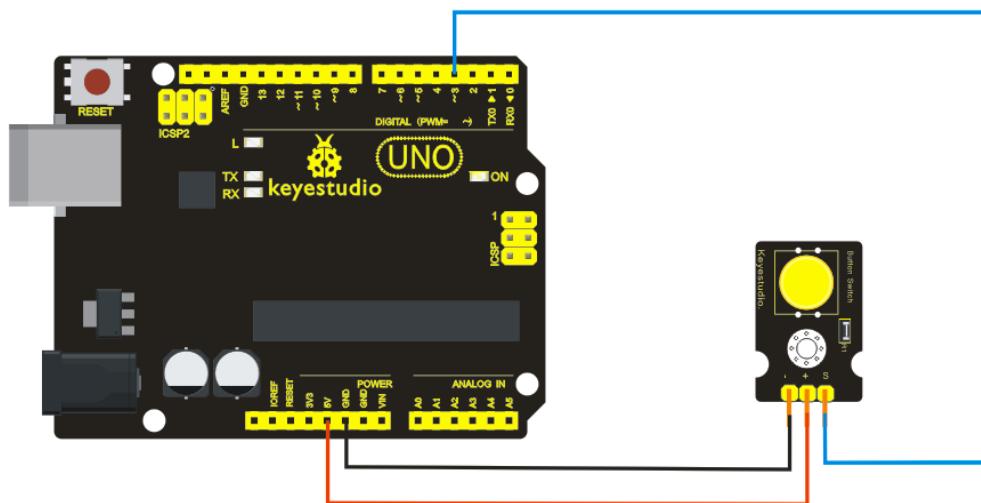
High quality connector

Immersion gold surface

Specification:

Supply Voltage: 3.3V to 5V
Easy to 'plug and operate'
Large button keypad and high-quality first-class cap
Achieve interesting and interactive work
Interface: Digital
Size: 30*20mm
Weight: 4g

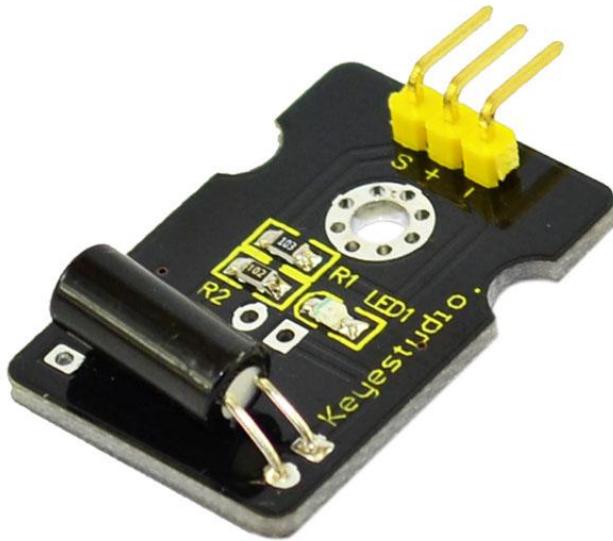
Connection diagram:



Sample Code

```
/* # When you push the digital button, the Led 13 on the board will turn on. Otherwise,the led turns off.  
*/  
int ledPin = 13; // choose the pin for the LED  
int inputPin = 3; // Connect sensor to input pin 3  
void setup() {  
    pinMode(ledPin, OUTPUT); // declare LED as output  
    pinMode(inputPin, INPUT); // declare pushbutton as input  
}  
void loop(){  
    int val = digitalRead(inputPin); // read input value  
    if (val == HIGH) { // check if the input is HIGH  
        digitalWrite(ledPin, LOW); // turn LED OFF  
    } else {  
        digitalWrite(ledPin, HIGH); // turn LED ON  
    }  
}
```

Project 12: Digital Tilt Sensor



Introduction:

Tilt Sensor is a digital tilt switch. It can be used as a simple tilt sensor. Simply plug it to our IO/Sensor shield; you can make amazing interactive projects. With dedicated sensor shield and Arduino, you can achieve interesting and interactive work.

Specification:

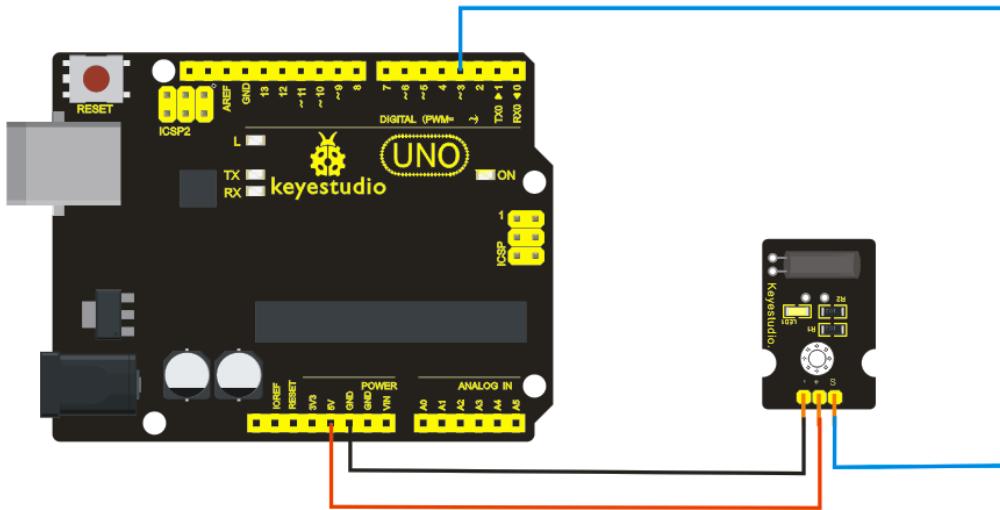
Supply Voltage: 3.3V to 5V

Interface: Digital

Size: 30*20mm

Weight: 3g

Connection Diagram:



Sample Code:

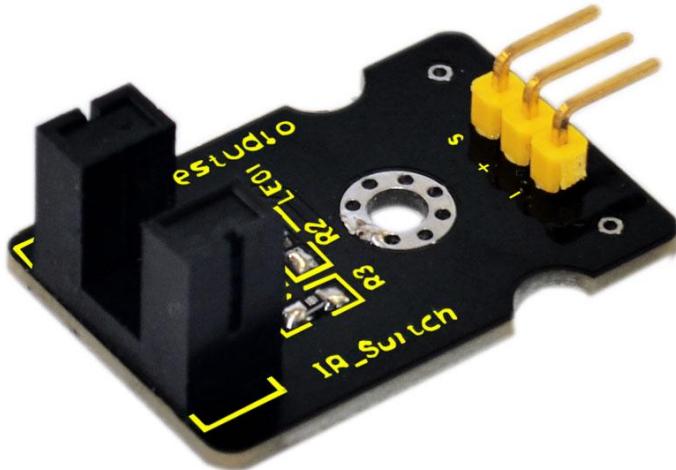
```
int ledPin = 13;          // Connect LED to pin 13
int switcher = 3;         // Connect Tilt sensor to Pin3

void setup()
{
    pinMode(ledPin, OUTPUT); // Set digital pin 13 to output mode
    pinMode(switcher, INPUT); // Set digital pin 3 to input mode
}

void loop()
{

    if(digitalRead(switcher)==HIGH) //Read sensor value
    {
        digitalWrite(ledPin, HIGH); // Turn on LED when the sensor is tilted
    }
    else
    {
        digitalWrite(ledPin, LOW); // Turn off LED when the sensor is not triggered
    }
}
```

Project 13: Photo interrupter module



Introduction:

Upright part of this sensor is an infrared emitter and on the other side, it's a shielded infrared detector. By emitting a beam of infrared light from one end to other end, the sensor can detect an object when it passes through the beam. It is used for many applications including optical limit switches, pellet dispensing, general object detection, etc.

Specification:

Supply Voltage: 3.3V to 5V

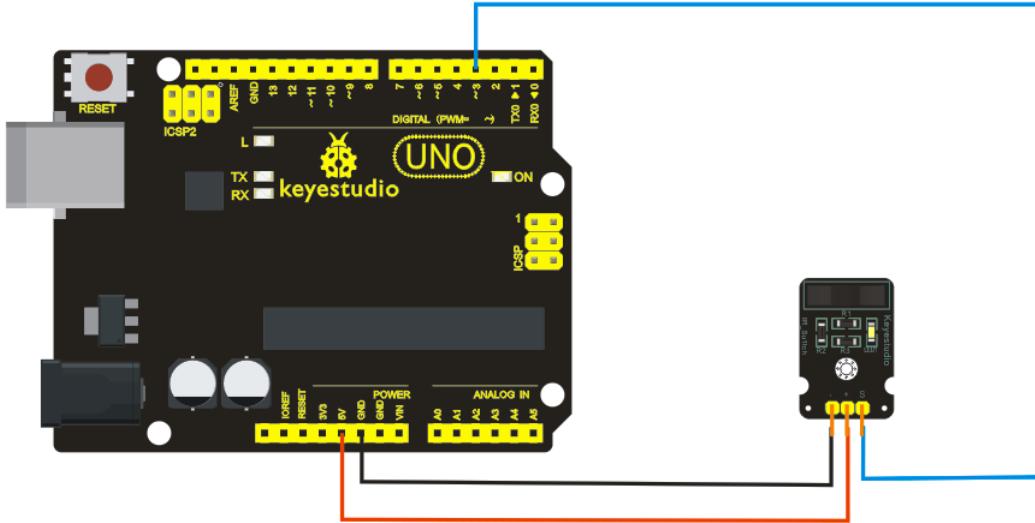
Interface: Digital

Size: 30*20mm

Weight: 3g

Connection Diagram:

keystudio



Sample code

```
// photo interrupter module
```

```
int Led = 13 ;// define LED Interface  
int buttonpin = 3; // define the photo interrupter sensor interface  
int val ;// define numeric variables val  
void setup ()  
{  
    pinMode (Led, OUTPUT) ;// define LED as output interface  
    pinMode (buttonpin, INPUT) ;// define the photo interrupter 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 light sensor detects a signal is interrupted, LED flashes  
    {  
        digitalWrite (Led, HIGH);  
    }  
    else  
    {  
        digitalWrite (Led, LOW);  
    }  
}
```

Project 14: Capacitive Touch Sensor



Introduction:

Are you tired of clicking mechanic button? Well, try our capacitive touch sensor. We can find touch sensors mostly on electronic device. So upgrade your Arduino project with our new version touch sensor and make it cool!!

This little sensor can "feel" people and metal touch and feedback a high/low voltage level. Even isolated by some cloth and paper, it can still feel the touch. Its sensitivity decrease as isolation layer gets thicker. For detail of usage, please check our wiki. To perfect user's experience of our sensor module, we made following improvements.

Specification:

Supply Voltage: 3.3V to 5V

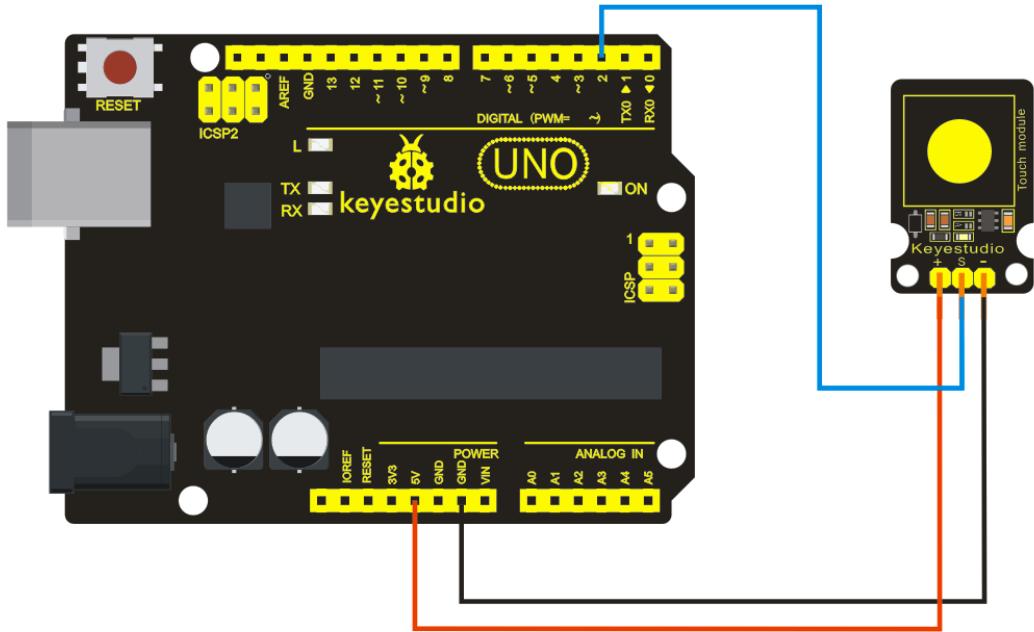
Interface: Digital

Size: 30*20mm

Weight: 3g

Connection Diagram:

keystudio



Sample Code:

```
int ledPin = 13;          // Connect LED on pin 13, or use the onboard one
int KEY = 2;              // Connect Touch sensor on Digital Pin 2

void setup(){
    pinMode(ledPin, OUTPUT); // Set ledPin to output mode
    pinMode(KEY, INPUT);   //Set touch sensor pin to input mode
}

void loop(){
    if(digitalRead(KEY)==HIGH) { //Read Touch sensor signal
        digitalWrite(ledPin, HIGH); // if Touch sensor is HIGH, then turn on
    }
    else{
        digitalWrite(ledPin, LOW); // if Touch sensor is LOW, then turn off the led
    }
}
*****
```

Project 15: Knock Sensor Module



Introduction:

This module is a knock sensor. When you knock it, it can send a momentary signal. We can combine it with Arduino to make some interesting experiment, e.g. electronic drum

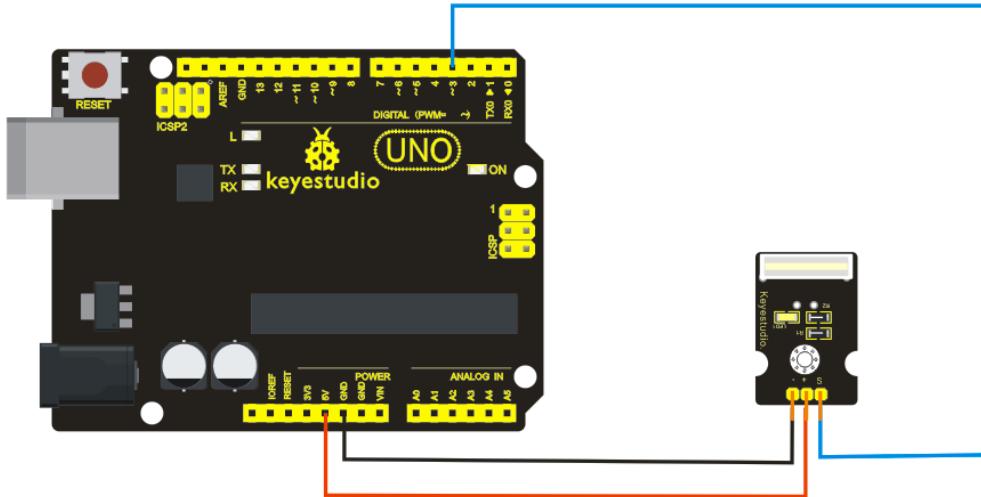
Specification:

Working voltage: 5V

Size: 30*20mm

Weight: 3g

Connection Diagram:



Sample Code

```
int Led=13;//define LED interface  
int Shock=3//define knock sensor interface  
;int val;//define digital variable val  
void setup()  
{  
pinMode(Led,OUTPUT);//define LED to be output interface  
pinMode(Shock,INPUT);//define knock sensor to be output interface  
}  
void loop()  
{  
val=digitalRead(Shock);//read the value of interface3 and evaluate it to val  
if(val==HIGH)//when the knock sensor detect a signal, LED will be flashing  
{  
digitalWrite(Led,LOW);  
}  
else  
{  
digitalWrite(Led,HIGH);  
}  
}
```

Project 16: Hall Magnetic Sensor



Introduction:

This is a Magnetic Induction Sensor. It senses the magnetic materials within a detection range up to 3cm. The detection range and the strength of the magnetic field are proportional. The output is digital on/off. This sensor uses the SFE Reed Switch - Magnetic Field Sensor.

Specification:

Sensing magnetic materials

Detection range: up to 3cm

Output: digital on/off

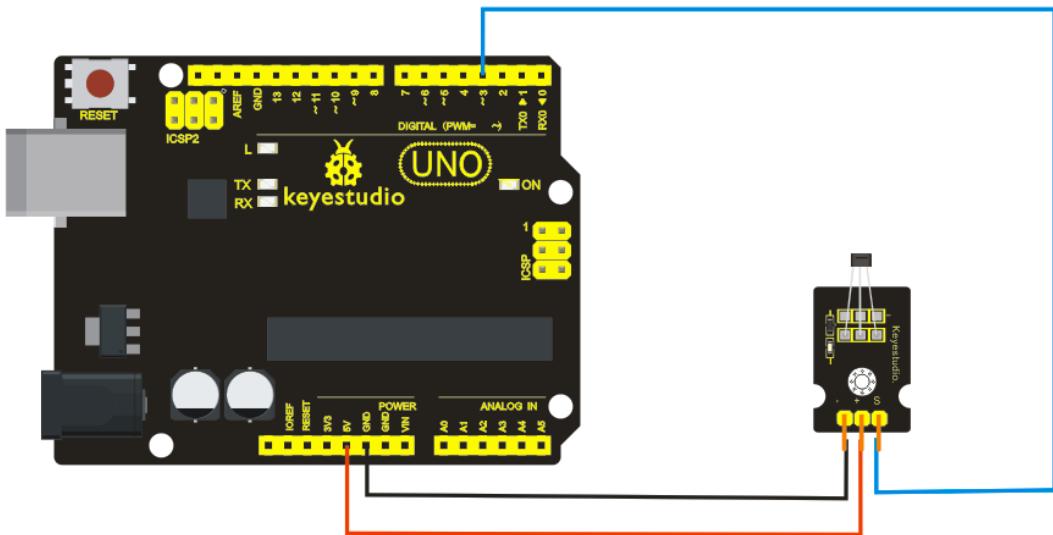
Detection range and magnetic field strength are proportional

Size: 30*20mm

Weight: 3g

Connection Diagram:

keystudio



Sample Code:

```
int ledPin = 13;          // choose the pin for the LED
int inputPin = 3;          // Connect sensor to input pin 3
int val = 0;              // variable for reading the pin status

void setup() {
    pinMode(ledPin, OUTPUT); // declare LED as output
    pinMode(inputPin, INPUT); // declare pushbutton as input
}

void loop(){
    val = digitalRead(inputPin); // read input value
    if (val == HIGH) {          // check if the input is HIGH
        digitalWrite(ledPin, LOW); // turn LED OFF
    } else {
        digitalWrite(ledPin, HIGH); // turn LED ON
    }
}
*****
```

Project 17: Line Tracking Sensor



Introduction:

This Line Tracking Sensor can detect white lines in black and black lines in white. The single line-tracking signal provides a stable output signal TTL for a more accurate and more stable line. Multi-channel option can be easily achieved by installing required line-tracking robot sensors.

Specification:

Power supply: +5V

Operating current: <10mA

Operating temperature range: 0°C ~ + 50°C

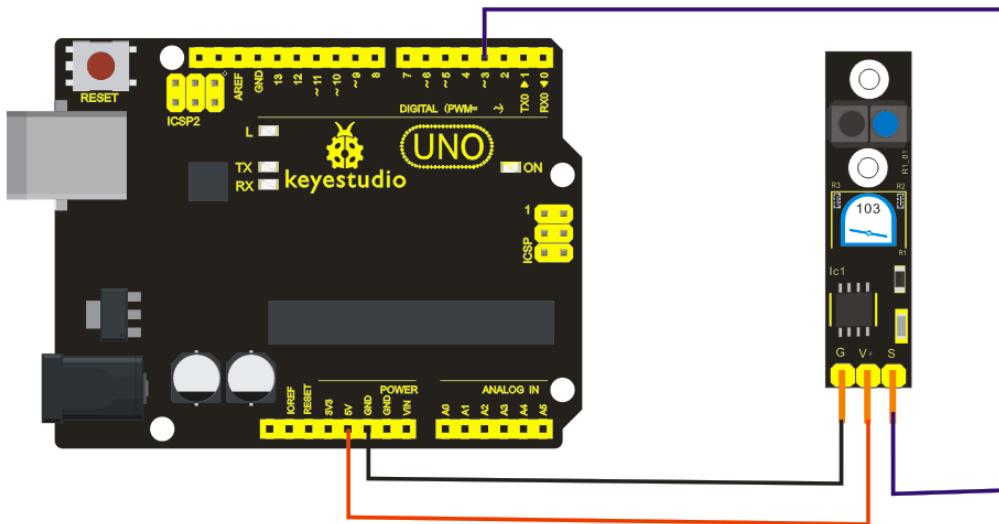
Output interface: 3-wire interface (1 - signal, 2 - power, 3 - power supply negative)

Output Level: TTL level

Size: 41.7*10.7mm

Weight: 3g

Connection Diagram:

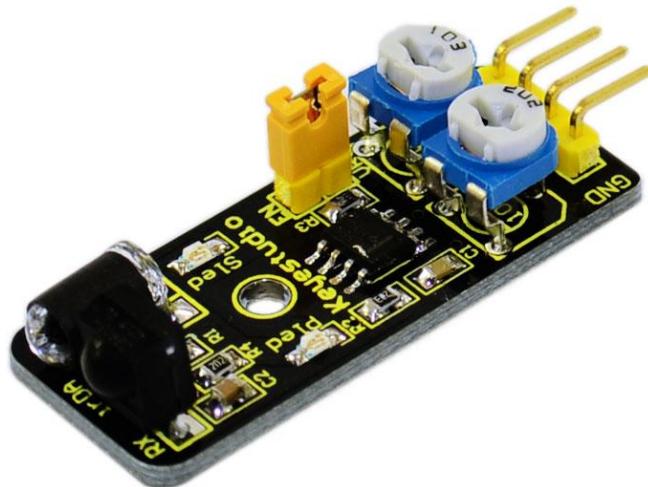


Sample Code:

```
///Arduino Sample Code
void setup()
{
    Serial.begin(9600);
}
void loop()
{
    Serial.println(digitalRead(3)); // print the data from the sensor
    delay(500);
}
```

```
*****
```

Project 18: Infrared Obstacle Avoidance Sensor



Introduction:

Infrared obstacle avoidance sensor is equipped with distance adjustment function and is especially designed for wheeled robots. This sensor has strong adaptability to ambient light and is of high precision. It has a pair of infrared transmitting and receiving tube. When infrared ray launched by the transmitting tube encounters an obstacle (its reflector), the infrared ray is reflected to the receiving tube, and the indicator will light up; the signal output interface outputs digital signal. We can adjust the detection distance through the potentiometer knob (effective distance: 2~40cm, working Voltage: 3.3V-5V). Thanks to a wide voltage range, this sensor can work steadily even under fluctuating power supply voltage and is suitable for the use of various micro-controllers, Arduino controllers and BS2 controllers. A robot mounted with the sensor can sense changes in the environment.

Specification:

Working voltage: DC 3.3V-5V

Working current: $\geq 20\text{mA}$

Working temperature: $-10^{\circ}\text{C} - +50^{\circ}\text{C}$

Detection distance: 2-40cm

IO Interface: 4 wire interface (-/+S/EN)

Output signal: TTL voltage

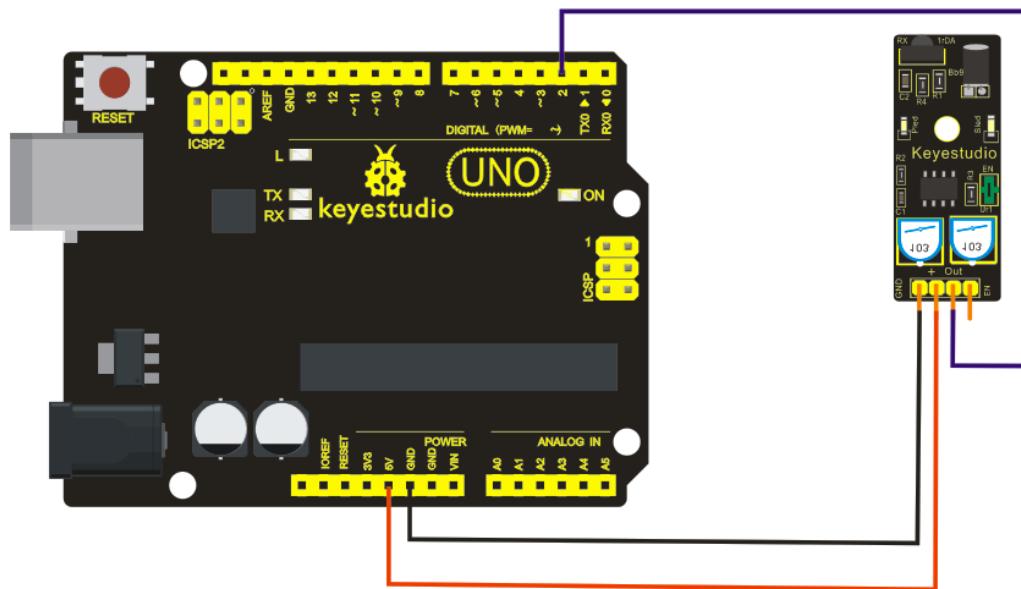
Accommodation mode: Multi-circle resistance regulation

Effective Angle: 35°

Size: 41.7*16.7mm

Weight: 5g

Connection Diagram:



Sample Code:

```
const int sensorPin = 2;      // the number of the sensor pin
const int ledPin = 13;        // the number of the LED pin
int sensorState = 0;         // variable for reading the sensor status
void setup() {
    pinMode(ledPin, OUTPUT);
    pinMode(sensorPin, INPUT); }
void loop(){
    // read the state of the sensor value:
    sensorState = digitalRead(sensorPin);
    // if it is, the sensorState is HIGH:
    if (sensorState == HIGH) {
        digitalWrite(ledPin, HIGH);
    }
    else {
        digitalWrite(ledPin, LOW);
    }
}*****
```

Project 19: PIR Motion Sensor



Introduction:

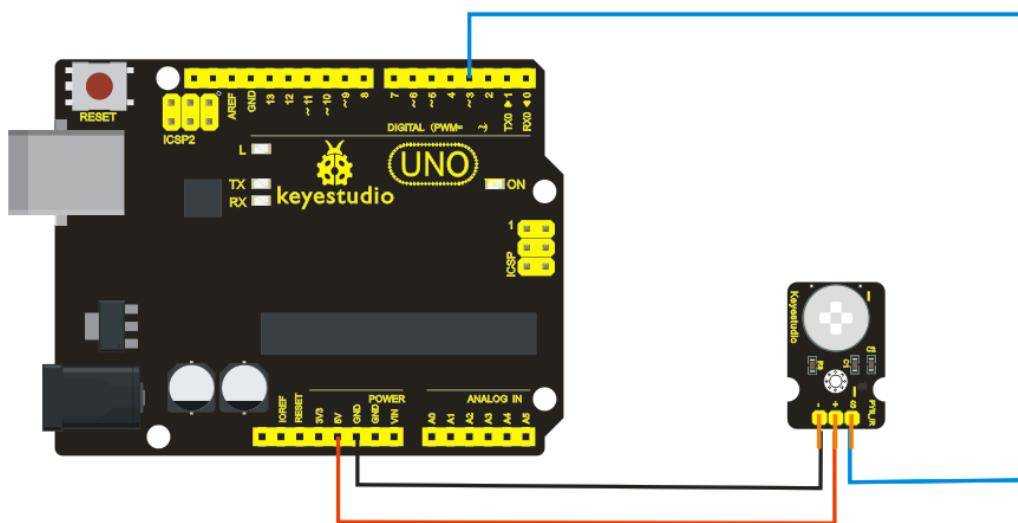
Pyroelectric infrared motion sensor can detect infrared signals from a moving person or moving animal, and output switching signals. It can be applied to a variety of occasions to detect the movement of human body. Conventional pyroelectric infrared sensors require body pyroelectric infrared detector, professional chip, complex peripheral circuit, so the size is bigger, with complex circuit, and lower reliability. Now we launch this new pyroelectric infrared motion sensor, specially designed for Arduino. It uses an integrated digital body pyroelectric infrared sensor, has smaller size, higher reliability, lower power consumption and simpler peripheral circuit.

Specification:

keystudio

Input Voltage: 3.3 ~ 5V, 6V Maximum
Working Current: 15uA
Working Temperature: -20 ~ 85 °C
Output Voltage: High 3V, low 0V
Output Delay Time (High Level): About 2.3 to 3 Seconds
Detection angle: 100 °
Detection distance: 7 meters
Output Indicator LED (When output HIGH, it will be ON)
Pin limit current: 100mA
Size: 30*20mm
Weight: 4g

Connection Diagram:



Sample Code:

```
byte sensorPin = 3;  
byte indicator = 13;  
void setup()  
{  
    pinMode(sensorPin,INPUT);  
    pinMode(indicator,OUTPUT);  
    Serial.begin(9600);  
}  
  
void loop()  
{  
    byte state = digitalRead(sensorPin);  
    digitalWrite(indicator,state);  
}
```

```
if(state == 1)Serial.println("Somebody is in this area!");
else if(state == 0)Serial.println("No one!");
delay(500);
}
```

Project 20: Flame Sensor



Introduction:

This flame sensor can be used to detect fire or other lights whose wavelength stands at 760 nm ~ 1100 nm. In the fire-fighting robot game, the flame plays an important role in the probe, which can be used as the robot's eyes to find fire source.

Specification:

Supply Voltage: 3.3V to 5V

Detection range: 20cm (4.8V) ~ 100cm (1V)

Rang of Spectral Bandwidth: 760nm to 1100nm

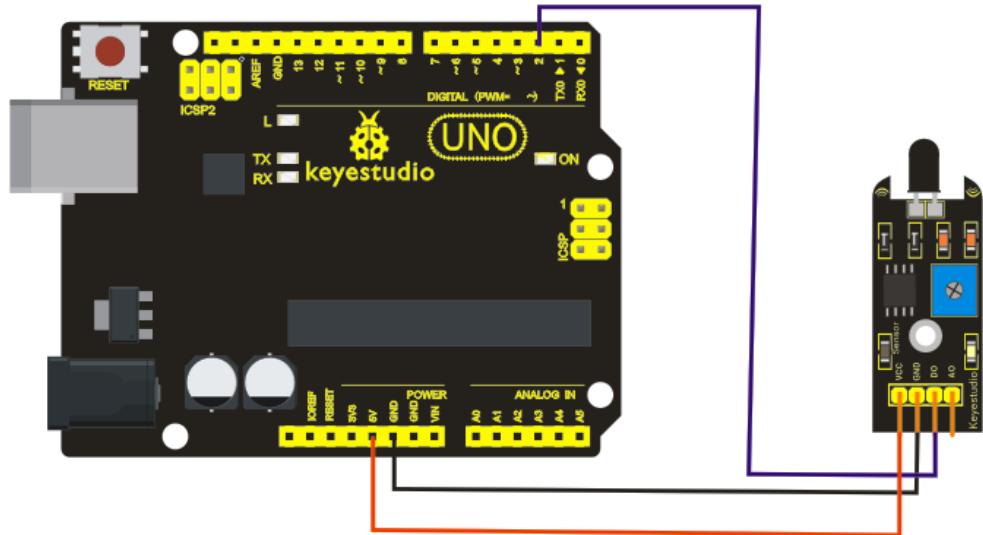
Operating temperature: -25°C to 85°C

Interface: digital

Size: 44*16.7mm

Weight: 4g

Connection Diagram:



Sample Code:

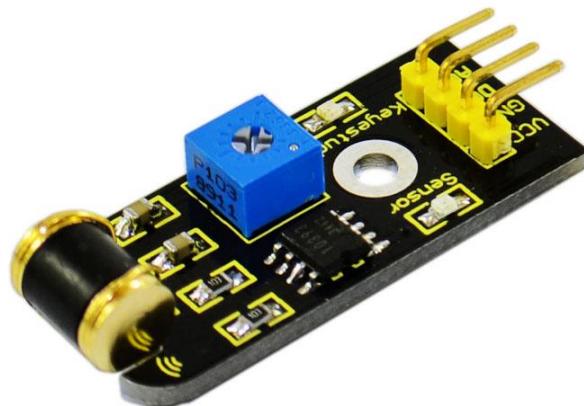
```
const int flamePin = 2;      // the number of the flame pin
const int ledPin = 13;       // the number of the LED pin
// variables will change:
int State = 0;             // variable for reading status
void setup() {
    // initialize the LED pin as an output:
    pinMode(ledPin, OUTPUT);
    // initialize the pushbutton pin as an input:
    pinMode(flamePin, INPUT);
}
void loop(){
    // read the state of the value:
    State = digitalRead(flamePin);
    if (State == HIGH) {
        // turn LED on:
        digitalWrite(ledPin, HIGH);
    }
    else {
        // turn LED off:
        digitalWrite(ledPin, LOW);
    }
}
```

```
}
```

```
}
```

```
*****
```

Project 21: Vibration Sensor



Introduction:

What's the simplest way to check vibration with Arduino? Well, use a vibration sensor from keyestudio. You can directly plug it on our sensor Shield V5, vibrate this sensor, and Arduino can receive a digital signal, making it easy to make computation and programs in Arduino. Despite its simplicity, you can make full use of it with creative thinking, step counting, and crash warning light etc.

Specification:

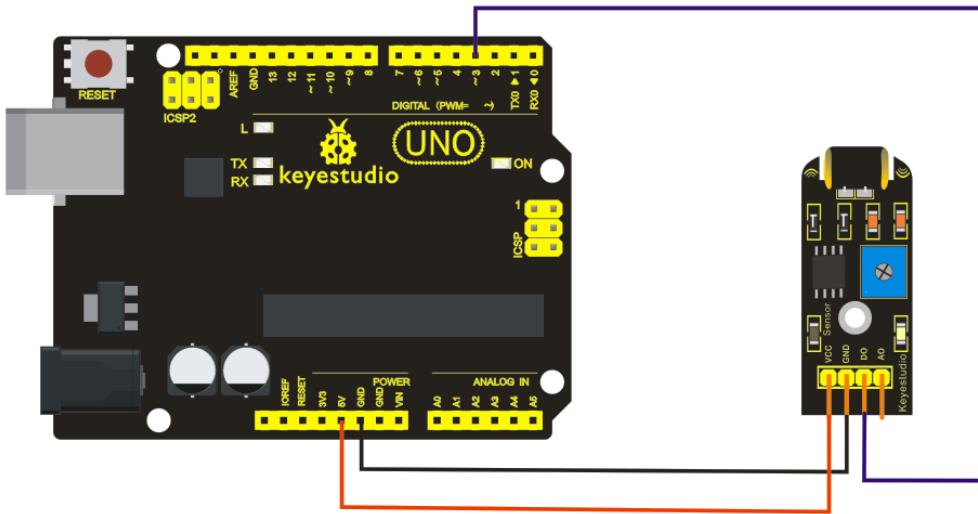
IO Type: Digital

Supply Voltage: 3.3V to 5V

Size: 40.7*16.7mm

Weight: 5g

Connection Diagram:

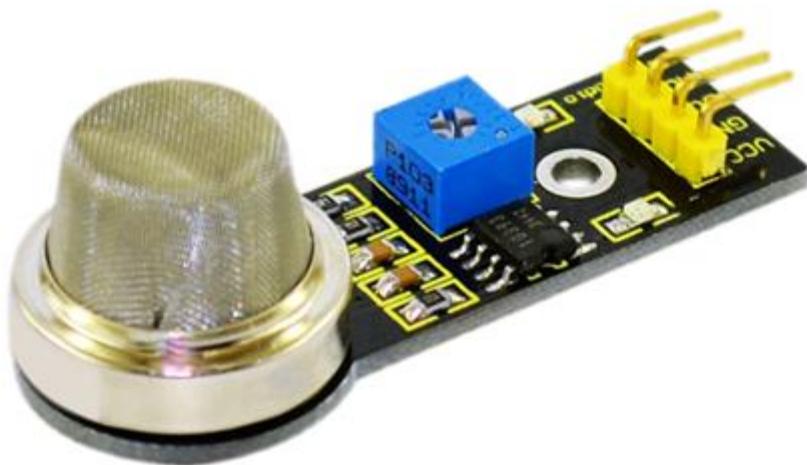


Sample Code:

```
#define SensorLED 13
#define SensorINPUT 3 //Connect the sensor to digital Pin 3 which is Interrupts 1.
unsigned char state = 0;
void setup()
{
    pinMode(SensorLED, OUTPUT);
    pinMode(SensorINPUT, INPUT);
    attachInterrupt(1, blink, FALLING); // Trigger the blink function when the falling edge is detected
}
void loop()
{ if(state!=0)
    {
        state = 0;
        digitalWrite(SensorLED,HIGH);
        delay(500);
    }
    else
        digitalWrite(SensorLED,LOW);
}
void blink()//Interruptions function
{ state++;
```

}

Project 22: Analog Gas Sensor



Introduction:

This analog gas sensor - MQ2 is used in gas leakage detecting equipment in consumer electronics and industrial markets. This sensor is suitable for detecting LPG, I-butane, propane, methane, alcohol, Hydrogen and smoke. It has high sensitivity and quick response. In addition, the sensitivity can be adjusted by the potentiometer.

Specification:

Power supply: 5V

Interface type: Analog

Wide detecting scope

Quick response and High sensitivity

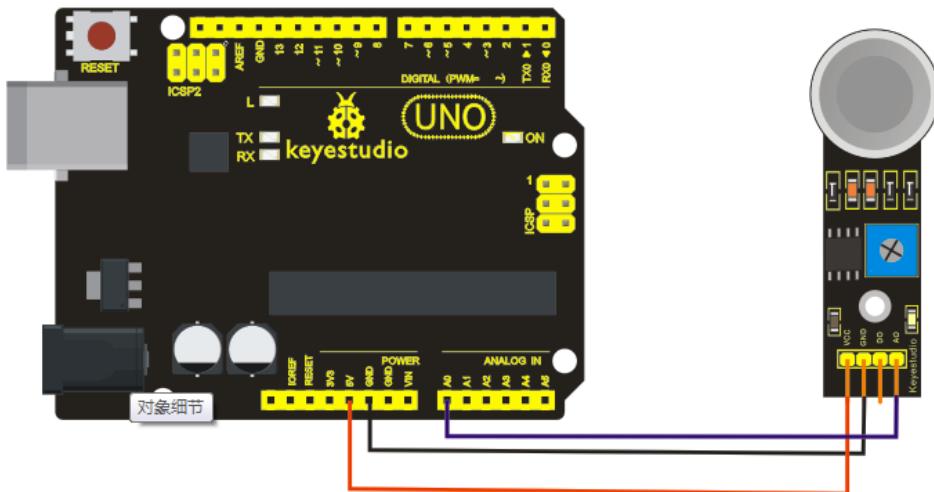
Simple drive circuit

Stable and long lifespan

Size: 49.7*20mm

Weight: 8g

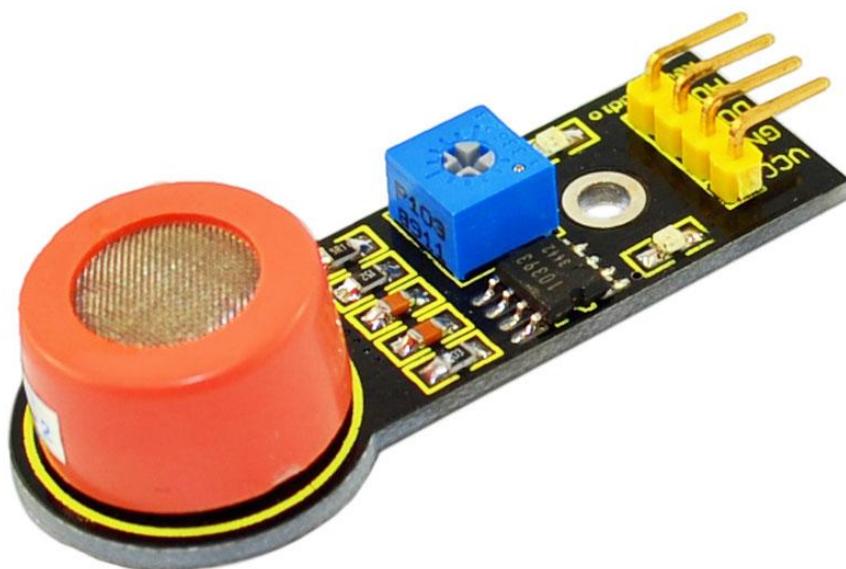
Connection Diagram:



Sample Code:

```
///Arduino Sample Code
void setup()
{
    Serial.begin(9600); //Set serial baud rate to 9600 bps
}
void loop()
{
int val;
val=analogRead(0);//Read Gas value from analog 0
Serial.println(val,DEC);//Print the value to serial port
delay(100);
}
*****
```

Project 23: Analog Alcohol Sensor



Introduction:

This analog gas sensor - MQ3 is suitable for detecting alcohol. It can be used in a Breath analyzer. Also it has high sensitivity to alcohol and low sensitivity to Benzine. The sensitivity can be adjusted by the potentiometer.

Specification:

Power supply: 5V

Interface type: Analog

Quick response and High sensitivity

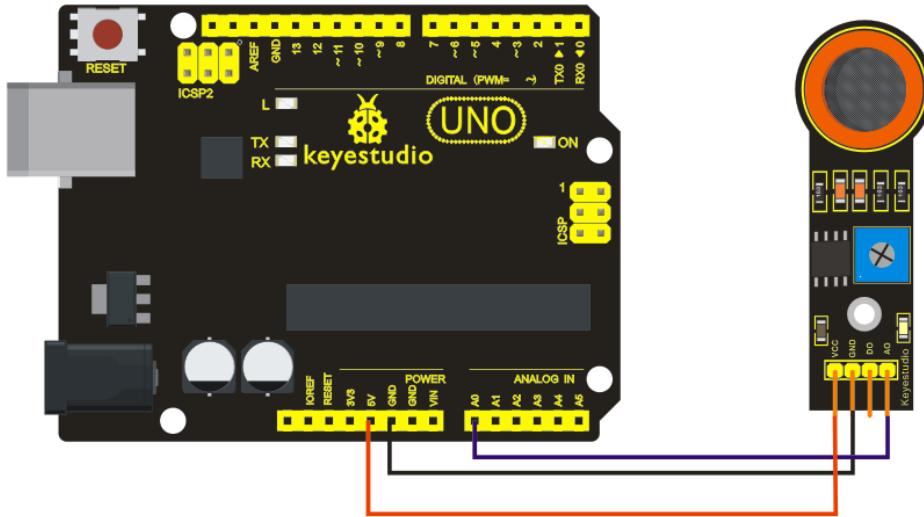
Simple drive circuit

Stable and long service life

Size: 49.7*20mm

Weight: 6g

Connection Diagram:

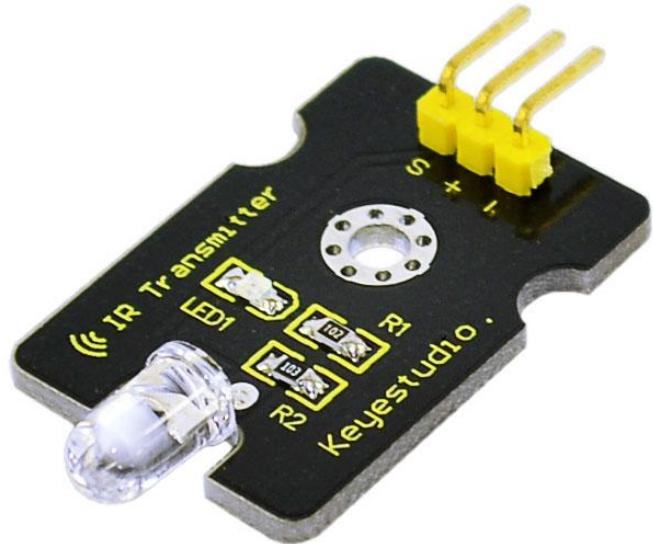


Sample Code:

```
///Arduino Sample Code
void setup()
{
    Serial.begin(9600); //Set serial baud rate to 9600 bps
}
void loop()
{
int val;
val=analogRead(0);//Read Gas value from analog 0
Serial.println(val,DEC);//Print the value to serial port
delay(100);
}
```

```
*****
```

Project 24: Digital IR Transmitter Module



Introduction:

IR Transmitter Module is designed for IR communication which is widely used for operating the television device from a short line-of-sight distance. The remote control is usually contracted to remote.

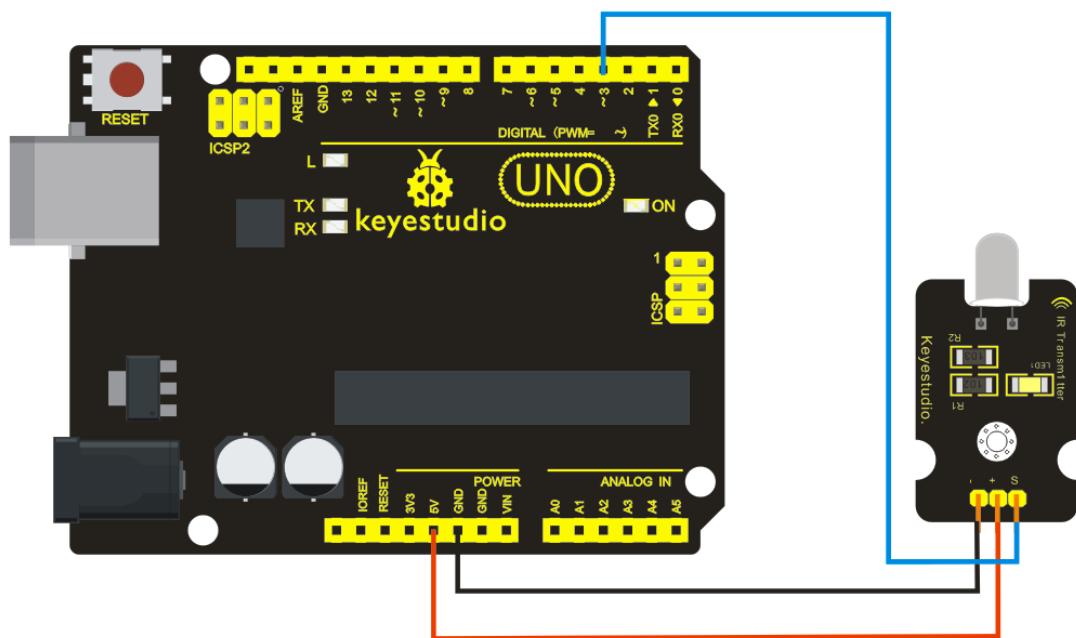
Since infrared (IR) remote controls use light, they require line of sight to operate the destination device. The signal can, however, be reflected by mirrors, just like any other light source.

If operation is required where no line of sight is possible, for instance when controlling equipment in another room or installed in a cabinet, many brands of IR extenders are available for this on the market. Most of these have an IR receiver, picking up the IR signal and relaying it via radio waves to the remote part, which has an IR transmitter mimicking the original IR control. Infrared receivers also tend to have a more or less limited operating angle, which mainly depends on the optical characteristics of the phototransistor. However, it's easy to increase the operating angle using a matte transparent object in front of the receiver.

Specification:

Power Supply: 3-5V
Infrared center frequency: 850nm-940nm
Infrared emission angle: about 20degree
Infrared emission distance: about 1.3m (5V 38Khz)
Interface socket: JST PH2.0
Mounting hole: inner diameter is 3.2mm, spacing is 15mm
Size: 35*20mm
Weight: 3g

Connection Diagram:



Sample code:

```
int led = 3;  
void setup() {  
    pinMode(led, OUTPUT);  
}  
void loop() {  
    digitalWrite(led, HIGH);  
    delay(1000);  
    digitalWrite(led, LOW);  
    delay(1000); }
```

In the darkness of the environment, you are going to see blinking blue light on phone's screen

when using camera to shoot the infrared LED.

Infrared remote/communication:

Hardware List

UNO R3 x2

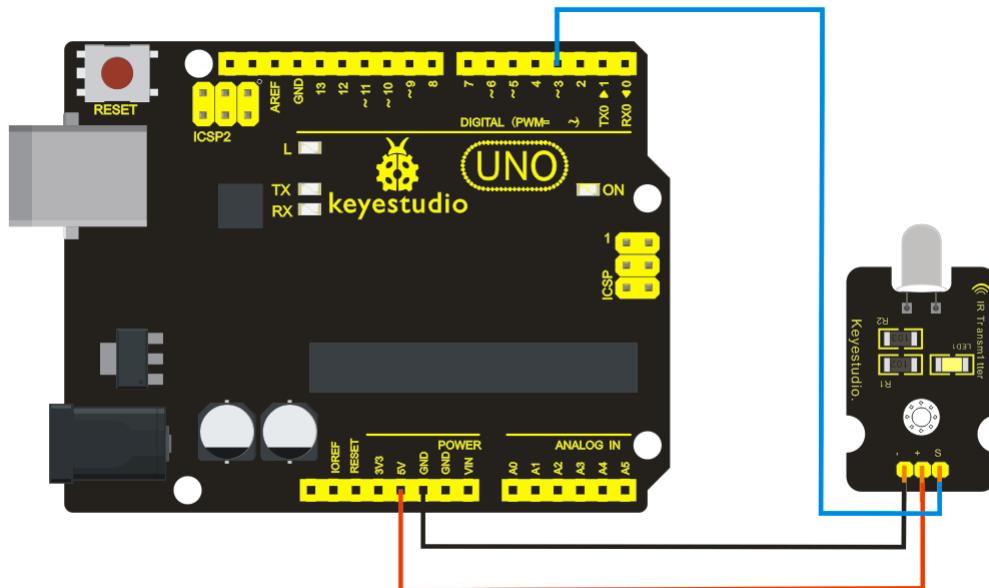
Digital IR Receiver x1

IR Transmitter Module x1

Get Arduino library [Arduino-IRremote](#) and install it

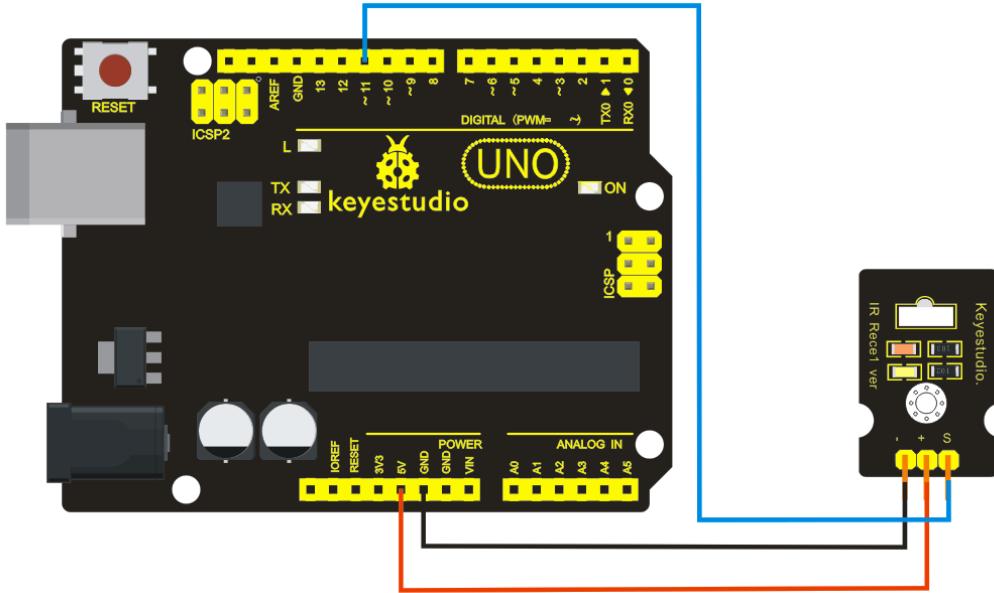
Connection Diagram:

IR Transmitter: same as above, Notice: Arduino-IRremote only supports D3 as transmitter.



IR Receiver: connect it to D11 port.

keystudio



Upload code to the UNO connected with IR Transmitter:

```
#include <IRremote.h>
IRsend irsend;
void setup()
(){}
void loop() {
    irsend.sendRC5(0x0, 8); //send 0x0 code (8 bits)
    delay(200);
    irsend.sendRC5(0x1, 8);
    delay(200); }
```

Upload code to the UNO connected with IR Receiver:

```
#include <IRremote.h>
const int RECV_PIN = 11;
const int LED_PIN = 13;
IRrecv irrecv(RECV_PIN);
decode_results results;
void setup()
{Serial.begin(9600);
irrecv.enableIRIn(); // Start the receiver
}
void loop()
{if (irrecv.decode(&results))
 { if ( results.bits > 0 )
 {
```

```
int state;
if ( 0x1 == results.value )
{
    state = HIGH;
}
else
{
    state = LOW;
}
digitalWrite( LED_PIN, state );
}

irrecv.resume();      // prepare to receive the next value
}}f)
```

Result The "L" LED of the shield connected with IR Receiver will blink when IR Receiver faces to IR Transmitter.

Project 25: Digital IR Receiver Module



Introduction:

IR is widely used in remote control. With this IR receiver, Arduino project is able to receive command from any IR remoter controller if you have the right decoder. Well, it will be also easy

keystudio

to make your own IR controller using IR transmitter.

Specification:

Power Supply: 5V

Interface:Digital

Modulate Frequency: 38Khz

Module interface socket:JST PH2.0

Size: 30*20mm

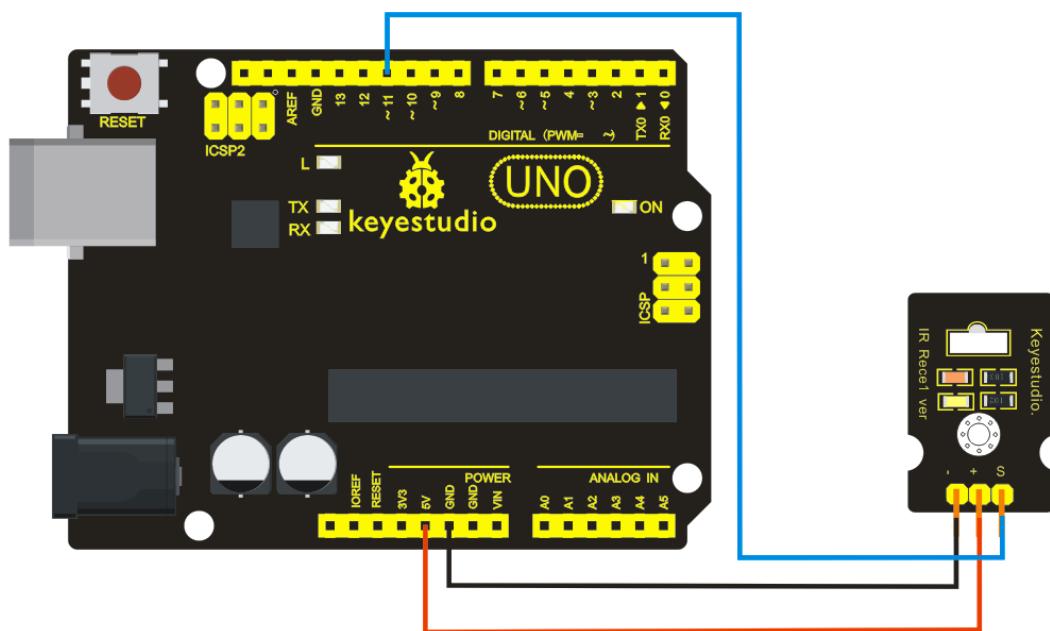
Weight: 4g

Wiring Diagram

The following image shows a suggested connection method. You may use any Digital I/O pin that is not in use by another device.

NOTE: In the sample code below Digital pin 11 is in use, you may either change your wiring or change the sample code to match.

Connection diagram



Sample Code:

```
#include <IRremote.h>
int RECV_PIN = 11;
IRrecv irrecv(RECV_PIN);
decode_results results;
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
    }
}
```

IR Remote Library Includes some sample codes for sending and receiving.

<https://github.com/shirriff/Arduino-IRremote>

Project 26: Rotary Encoder module



Introduction:

The rotary encoder can count the pulse outputting times during the process of its rotation in positive and reverse direction by rotating. This rotating counting is unlimited, not like potential counting. It can be restored to initial state to count from 0 with the button on rotary encoder.

Specification:

Power Supply: 5V

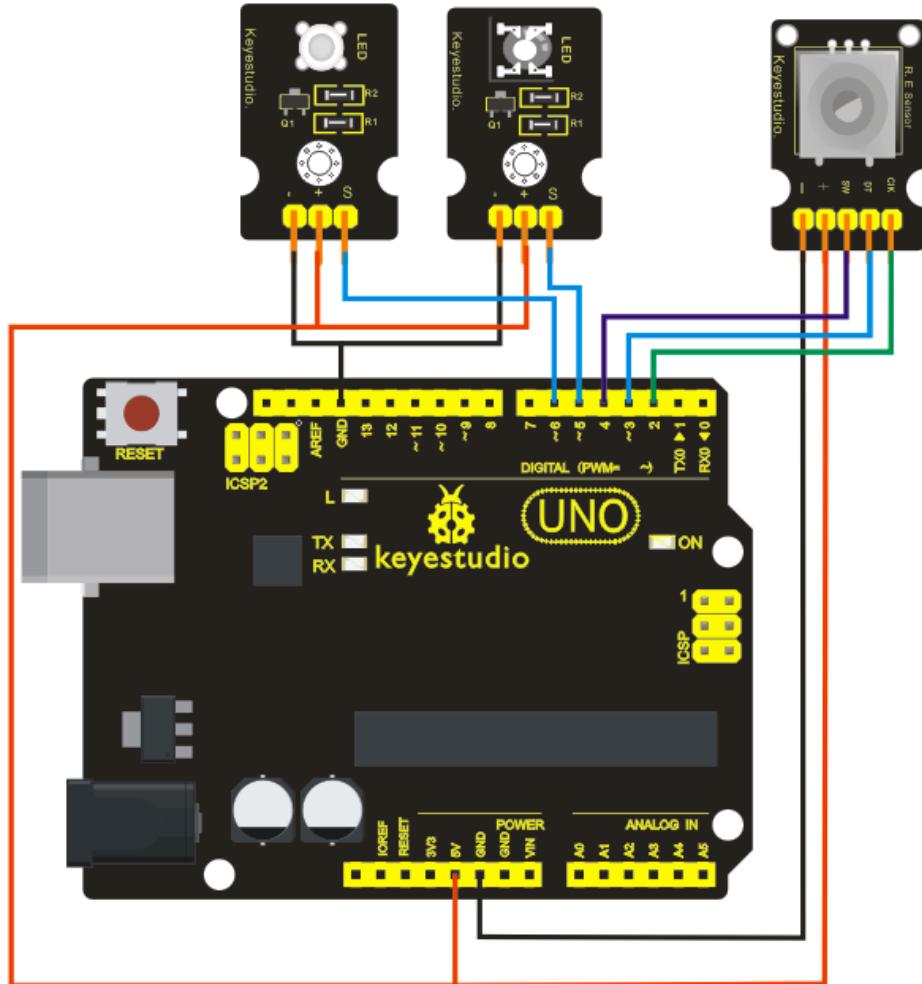
Interface: Digital

Size: 30*20mm

Weight: 7g

Connection Diagram:

keyestudio



Sample Code:

```
const int interruptA = 0;
const int interruptB = 1;
int CLK = 2;      // PIN2
int DAT = 3;      // PIN3
int BUTTON = 4;   // PIN4
int LED1 = 5;     // PIN5
int LED2 = 6;     // PIN6
int COUNT = 0;

void setup()
{
    attachInterrupt(interruptA, RoteStateChanged, FALLING);
    // attachInterrupt(interruptB, buttonState, FALLING);
    pinMode(CLK, INPUT);
    digitalWrite(2, HIGH); // Pull High Restance
```

keystudio

```
pinMode(DAT, INPUT);
digitalWrite(3, HIGH); // Pull High Restance

pinMode(BUTTON, INPUT);
digitalWrite(4, HIGH); // Pull High Restance
pinMode(LED1, OUTPUT);
pinMode(LED2, OUTPUT);
Serial.begin(9600);
}

void loop()
{
if (!(digitalRead(BUTTON)))
{
COUNT = 0;
Serial.println("STOP COUNT = 0");
digitalWrite(LED1, LOW);
digitalWrite(LED2, LOW);
delay (2000);
}
Serial.println(COUNT);
}

//-----
void RoteStateChanged() //When CLK  FALLING READ DAT
{
if (digitalRead(DAT)) // When DAT = HIGH IS FORWARD
{
COUNT++;
digitalWrite(LED1, HIGH);
digitalWrite(LED2, LOW);
delay(20);
}
else // When DAT = LOW IS BackRote
{
COUNT--;
digitalWrite(LED2, HIGH);
digitalWrite(LED1, LOW);
delay(20);
}
```

```
    }  
}  
*****
```

Project 27: LM35 Linear Temperature Sensor



Introduction:

LM35 Linear Temperature Sensor is based on semiconductor LM35 temperature sensor. It can be used to detect ambient air temperature. This sensor offers a functional range among 0 degree Celsius to 100 degree Celsius. Sensitivity is 10mV per degree Celsius. The output voltage is proportional to the temperature.

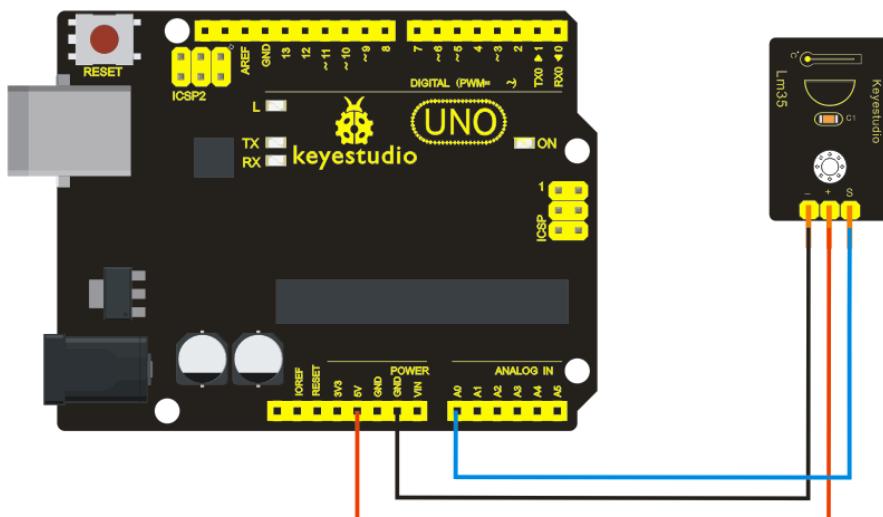
This sensor is commonly used as a temperature measurement sensor. It includes thermocouples, platinum resistance, and thermal resistance and temperature semiconductor chips. The chip is

commonly used in high temperature measurement thermocouples. Platinum resistance temperature sensor is used in the measurement of 800 degrees Celsius, while the thermal resistance and semiconductor temperature sensor is suitable for measuring the temperature of 100-200 degrees or below, in which the application of a simple semiconductor temperature sensor is good in linearity and high in sensitivity. The LM35 linear temperature sensor and sensor-specific Arduino shield can be easily combined.

Specification:

Based on the semiconductor LM35 temperature sensor
Can be used to detect ambient air temperature
Sensitivity: 10mV per degree Celcius
Functional range: 0 degree Celsius to 100 degree Celsius
Size: 30*20mm
Weight: 3g

Connection Diagram:

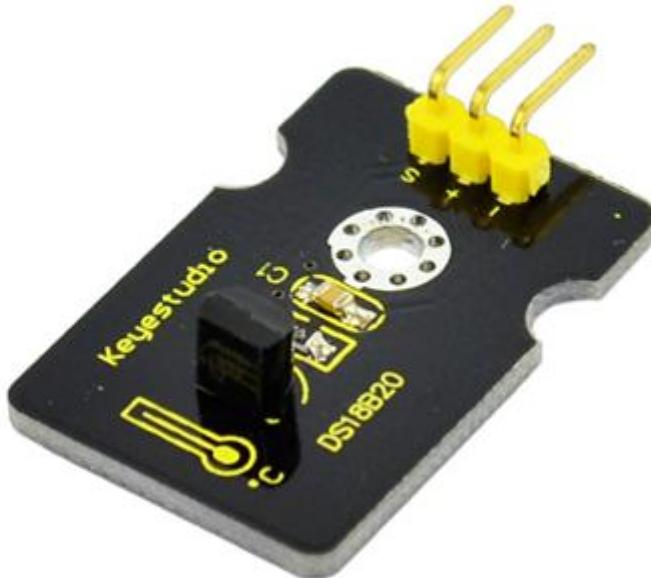


Sample Code :

```
void setup()
{
    Serial.begin(9600); //Set Baud Rate to 9600 bps
}
void loop()
{ int val;
  int dat;
```

```
val=analogRead(0);//Connect LM35 on Analog 0  
dat=(500 * val) /1024;;  
Serial.print("Temp:"); //Display the temperature on Serial monitor  
Serial.print(dat);  
Serial.println("C");  
delay(500);  
}  
*****
```

Project 28: 18B20 Temperature Sensor



Introduction:

DS18B20 is a digital temperature sensor from DALLAS U.S. It can be used to quantify environmental temperature testing.

The temperature range is -55 ~ +125 °C, inherent temperature resolution 0.5 °C. It also support multi-point mesh networking. Three DS18B20 can be deployed on three lines to achieve multi-point temperature measurement. It has a 9-12 bit serial output.

Specification:

keystudio

Supply Voltage: 3.3V to 5V

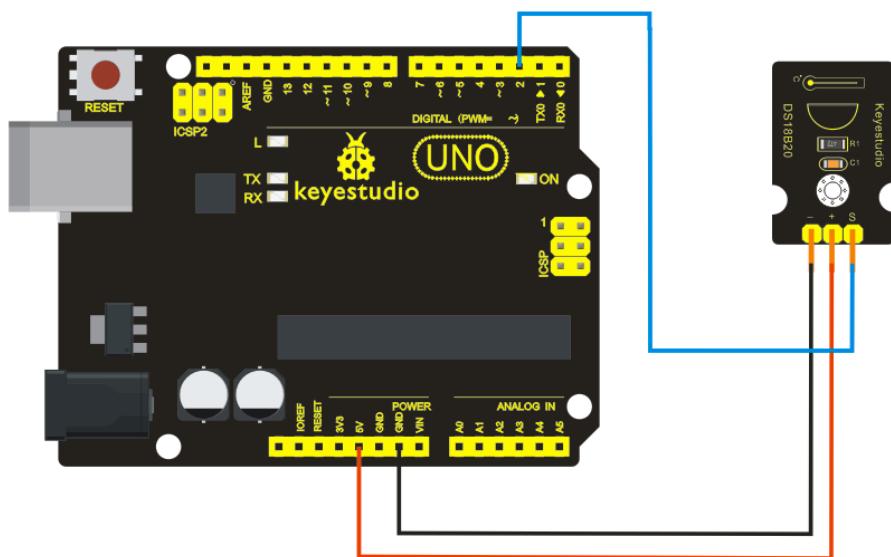
Temperature range: -55 °C ~ +125 °C

Interface: Digital

Size: 30*20mm

Weight: 3g

Connection Diagram:



Sample Code:

```
// http://www.pjrc.com/teensy/arduino\_libraries/OneWire.zip
#include <OneWire.h>
int DS18S20_Pin = 2; //DS18S20 Signal pin on digital 2
//Temperature chip i/o
OneWire ds(DS18S20_Pin); // on digital pin 2
void setup(void) {
  Serial.begin(9600);
}
void loop(void) {
  float temperature = getTemp();
  Serial.println(temperature);
```

keyestudio

```
delay(100); //just here to slow down the output so it is easier to read

}

float getTemp(){
    //returns the temperature from one DS18S20 in DEG Celsius

    byte data[12];
    byte addr[8];

    if ( !ds.search(addr) {
        //no more sensors on chain, reset search
        ds.reset_search();
        return -1000;
    }

    if ( OneWire::crc8( addr, 7 ) != addr[7] ) {
        Serial.println("CRC is not valid!");
        return -1000;
    }

    if ( addr[0] != 0x10 && addr[0] != 0x28 ) {
        Serial.print("Device is not recognized");
        return -1000;
    }

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

    byte present = ds.reset();
    ds.select(addr);
    ds.write(0xBE); // Read Scratchpad

    for (int i = 0; i < 9; i++) { // we need 9 bytes
        data[i] = ds.read();
    }
    ds.reset_search();

    byte MSB = data[1];
    byte LSB = data[0];
```

```
float tempRead = ((MSB << 8) | LSB); //using two's compliment  
float TemperatureSum = tempRead / 16;  
  
return TemperatureSum;  
  
}  
  
*****
```

Project29: ADXL345 Three Axis Acceleration Module



Introduction:

The ADXL345 is a small, thin, low power, 3-axis MEMS accelerometer with high resolution (13-bit) measurement at up to +/−16 g. Digital output data is formatted as 16-bit twos complement and is accessible through either a SPI (3- or 4-wire) or I2C digital interface.

The ADXL345 is well suited to measures the static acceleration of gravity in tilt-sensing applications, as well as dynamic acceleration resulting from motion or shock. Its high resolution (4 mg/LSB) enables measurement of inclination changes less than 1.0 degrees;.

Specification:

2.0-3.6VDC Supply Voltage

Ultra Low Power: 40uA in measurement mode, 0.1uA in standby@ 2.5V

Tap/Double Tap Detection

keystudio

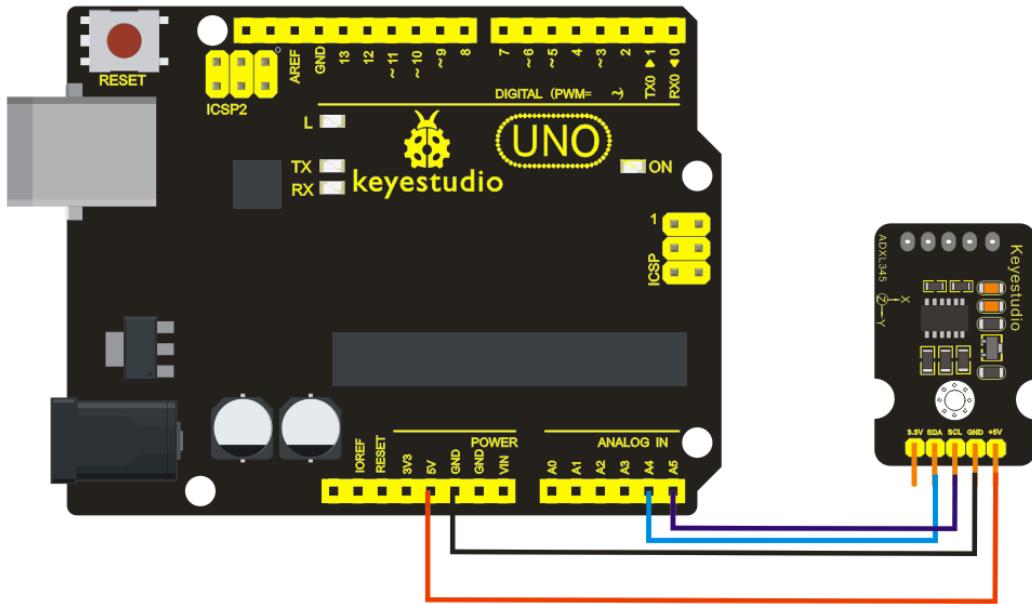
Free-Fall Detection

SPI and I2C interfaces

Size: 30*20mm

Weight: 3g

Connection Diagram:



Sample Code:

```
/*
The circuit:
VCC: 5V
GND: ground
SCL: UNO SCL
SDA: UNO SDA
```

This example code is in the public domain.

```
*/
#include <Wire.h>
// Registers for ADXL345
#define ADXL345_ADDRESS (0xA6 >> 1) // address for device is 8 bit but shift to the
// right by 1 bit to make it 7 bit because the
// wire library only takes in 7 bit addresses
#define ADXL345_REGISTER_XLSB (0x32)
```

keystudio

```
int accelerometer_data[3];
// void because this only tells the cip to send data to its output register
// writes data to the slave's buffer
void i2c_write(int address, byte reg, byte data) {
    // Send output register address
    Wire.beginTransmission(address);
    // Connect to device
    Wire.write(reg);
    // Send data
    Wire.write(data); //low byte
    Wire.endTransmission();
}

// void because using pointers
// microcontroller reads data from the sensor's input register
void i2c_read(int address, byte reg, int count, byte* data) {
    // Used to read the number of data received
    int i = 0;
    // Send input register address
    Wire.beginTransmission(address);
    // Connect to device
    Wire.write(reg);
    Wire.endTransmission();

    // Connect to device
    Wire.beginTransmission(address);
    // Request data from slave
    // Count stands for number of bytes to request
    Wire.requestFrom(address, count);
    while(Wire.available()) // slave may send less than requested
    {
        char c = Wire.read(); // receive a byte as character
        data[i] = c;
        i++;
    }
    Wire.endTransmission();
}

void init_adxl345() {
    byte data = 0;

    i2c_write(ADXL345_ADDRESS, 0x31, 0x0B);    // 13-bit mode +_ 16g
```

keystudio

```
i2c_write(ADXL345_ADDRESS, 0x2D, 0x08); // Power register

i2c_write(ADXL345_ADDRESS, 0x1E, 0x00); // x
i2c_write(ADXL345_ADDRESS, 0x1F, 0x00); // Y
i2c_write(ADXL345_ADDRESS, 0x20, 0x05); // Z

// Check to see if it worked!
i2c_read(ADXL345_ADDRESS, 0x00, 1, &data);
if(data==0xE5)
    Serial.println("it work Success");
else
    Serial.println("it work Fail");
}

void read_adxl345() {
byte bytes[6];
memset(bytes,0,6);

// Read 6 bytes from the ADXL345
i2c_read(ADXL345_ADDRESS, ADXL345_REGISTER_XLSB, 6, bytes);
// Unpack data
for (int i=0;i<3;++i) {
    accelerometer_data[i] = (int)bytes[2*i] + (((int)bytes[2*i + 1]) << 8);
}
}

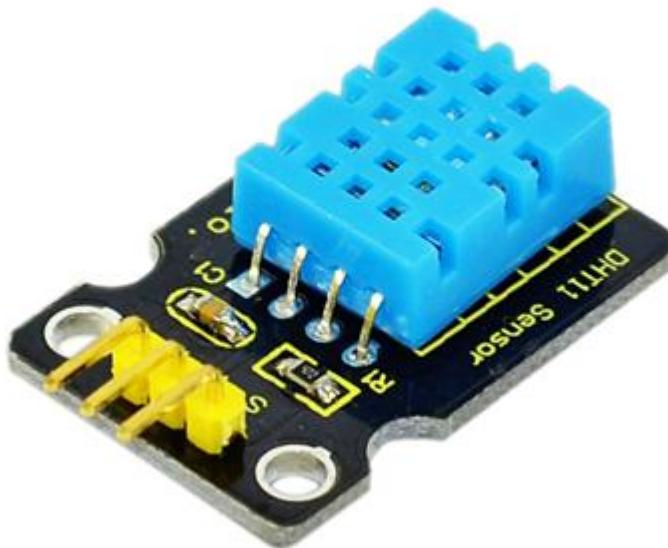
// initialise and start everything
void setup() {
Wire.begin();
Serial.begin(9600);
for(int i=0; i<3; ++i) {
    accelerometer_data[i] = 0;
}
init_adxl345();
}

void loop() {
read_adxl345();
Serial.print("ACCEL: ");
Serial.print(float(accelerometer_data[0])*3.9/1000);//3.9mg/LSB scale factor in 13-bit mode
Serial.print("\t");
Serial.print(float(accelerometer_data[1])*3.9/1000);
```

```
Serial.print("\t");
Serial.print(float(accelerometer_data[2])*3.9/1000);
Serial.print("\n");
delay(100);
}

*****
*****
```

Project 30: DHT11 Temperature and Humidity Sensor



Introduction:

This DHT11 Temperature and Humidity Sensor features calibrated digital signal output with the temperature and humidity sensor complex. Its technology ensures high reliability and excellent long-term stability. A high-performance 8-bit microcontroller is connected. This sensor includes a resistive element and a sense of wet NTC temperature measuring devices. It has excellent quality, fast response, anti-interference ability and high cost performance advantages.

Each DHT11 sensor features extremely accurate calibration data of humidity calibration chamber. The calibration coefficients stored in the OTP program memory, internal sensors detect signals in the process, and we should call these calibration coefficients. The single-wire serial interface system is integrated to make it quick and easy. Qualities of small size, low power, and 20-meter signal transmission distance make it a wide applied application and even the most demanding one. Convenient connection, special packages can be provided according to users

need.

Specification:

Supply Voltage: +5 V

Temperature range: 0-50 °C error of ± 2 °C

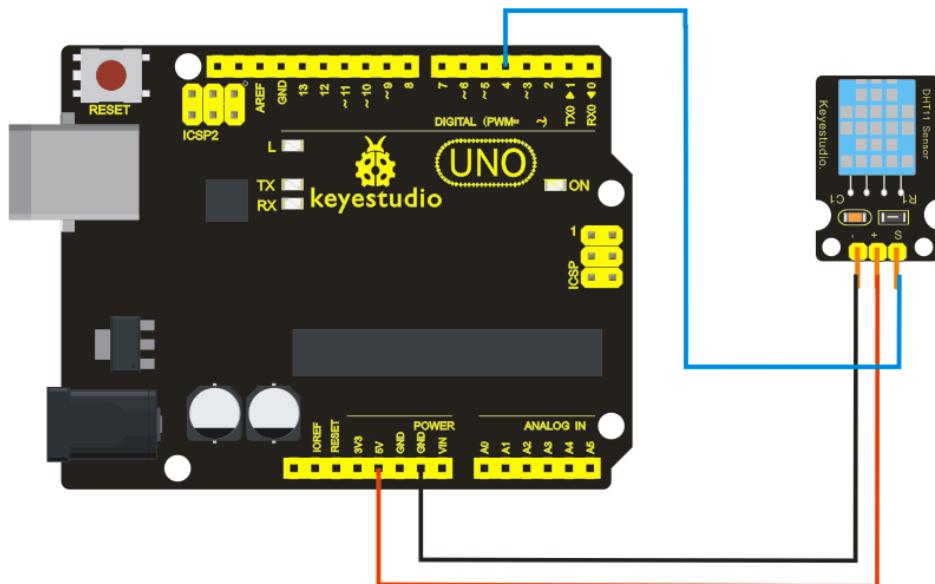
Humidity: 20-90% RH ± 5 % RH error

Interface: Digital

Size: 30*20mm

Weight: 4g

Connection diagram:



Sample Code:

Please download the [DHT11Lib](#) firstly. Or, see the website

```
#include <dht11.h>
dht11 DHT;
#define DHT11_PIN 4

void setup(){
  Serial.begin(9600);
  Serial.println("DHT TEST PROGRAM ");
  Serial.print("LIBRARY VERSION: ");
  Serial.println(DHT11LIB_VERSION);
  Serial.println();
}
```

```
Serial.println("Type,\tstatus,\tHumidity (%),\tTemperature (C)");
}

void loop(){
    int chk;
    Serial.print("DHT11, \t");
    chk = DHT.read(DHT11_PIN); // READ DATA
    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(DHT.humidity,1);
    Serial.print(",\t");
    Serial.println(DHT.temperature,1);

    delay(1000);
}
*****
```

Project 31: Bluetooth Module



Introduction:

This Bluetooth module can easily achieve serial wireless data transmission. Its operating frequency is among the most popular 2.4GHz ISM frequency band (i.e. Industrial, scientific and medical). It adopts Bluetooth 2.1+EDR standard. In Bluetooth 2.1, signal transmit time of different devices stands at a 0.5 seconds interval so that the workload of bluetooth chip can be reduced substantially and more sleeping time can be saved for bluetooth. This module is set with serial interface, which is easy-to-use and simplifying overall design/development cycle.

Specification:

Bluetooth protocol: Bluetooth 2.1+ EDR standard

USB protocol: USB v1.1/2.0

Operating frequency: 2.4GHz ISM frequency band

Modulation mode: Gauss frequency Shift Keying

Transmit power: $\leq 4\text{dBm}$, second stage

Sensitivity: $\leq -84\text{dBm}$ at 0.1% Bit Error Rate

Transmission speed: 2.1Mbps(Max)/160 kbps(Asynchronous); 1Mbps/1Mbps(Synchronous)

Safety feature: Authentication and encryption

Supported configuration: Bluetooth serial port (major and minor)

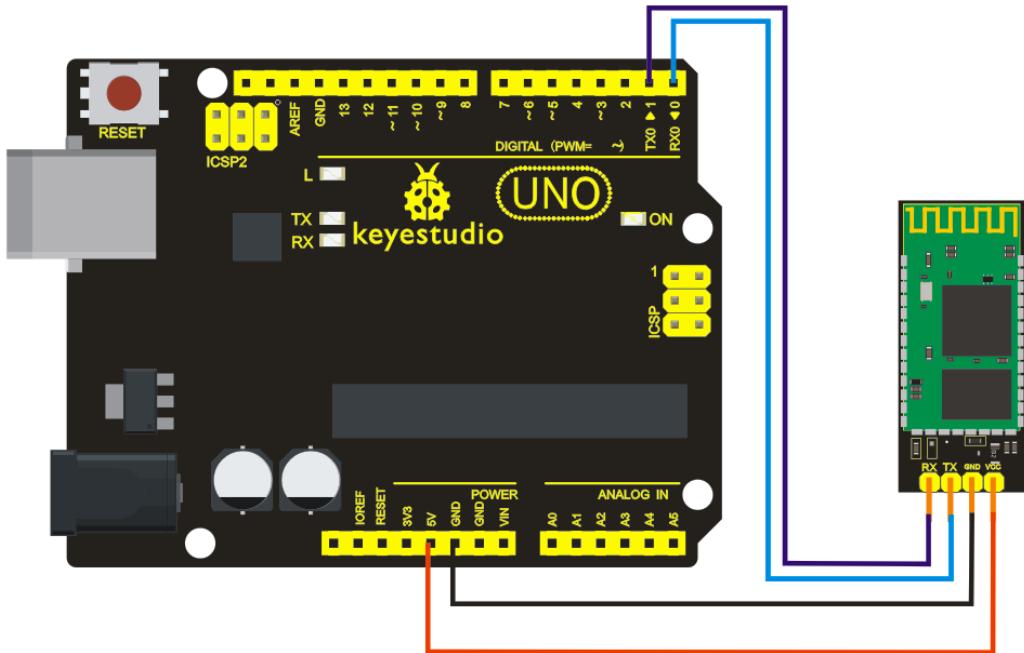
Supply Voltage: +3.3 VDC 50mA

Operating temperature: -20 to 55°C

Size: 36.5*16mm

Weight: 4g

Connection Diagram:



Sample Code:

```
int val;
int ledpin=13;
void setup()
{
Serial.begin(9600);
pinMode(ledpin,OUTPUT);
}
void loop()
{ val=Serial.read();
if(val=='a')
{
digitalWrite(ledpin,HIGH);
delay(250);
digitalWrite(ledpin,LOW);
delay(250);
Serial.println("keyestudio");
}
*****
```

Project 32: TEMT6000 ambient light sensor



Introduction:

At some point you are going to want to sense ambient brightness with better precision than your trusty photoresistor without adding complexity to your project. When that day comes, go get yourself a TEMT6000 ambient light sensor.

The TEMT6000 is supposed to be adapted to the sensitivity of the human eye, but I found it preformed sub-par in low light conditions. It does however work very well reacting to very small changes in a large range of brightnesses. Because it is meant to mimic the human eye, it does not react well to IR or UV light, so just make sure to note that when considering using it in your project.

Hooking It Up

This is an incredibly simple part, just connect power and ground, and the signal pin to your favorite analog input and you are done, the sensor will output analog voltage, that ramps up when it gets brighter . You can power this off of 3.3v if you would like, the output value will just be lower.

Code

You can not get more simple than this – This just reports the reading from the sensor to the serial terminal: 0-1023 with 1023 being very bright, and 0 being very dark.

Specification:

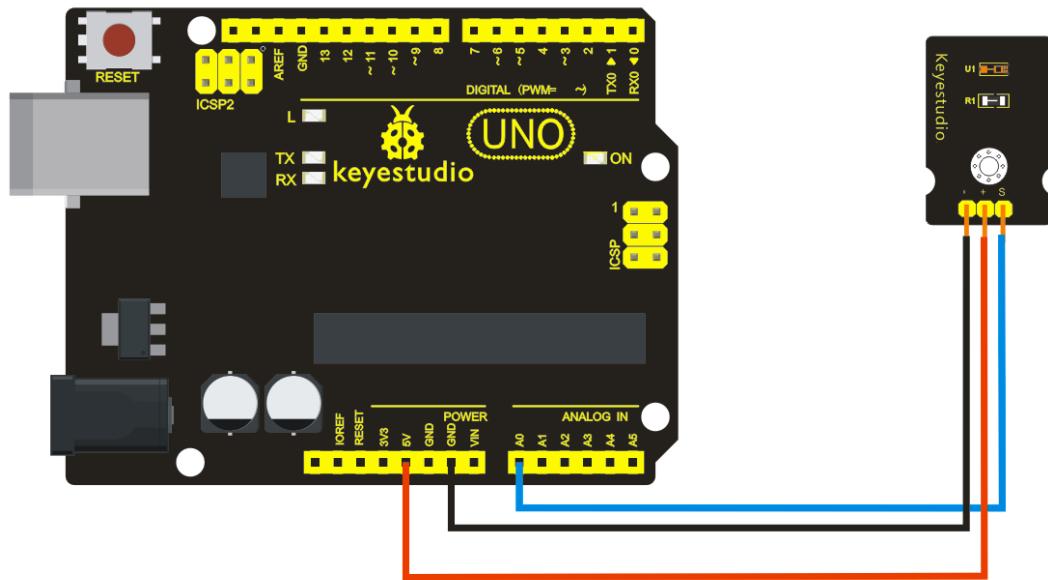
keystudio

Supply Voltage: +5VDC 50mA

Size: 36.5*16mm

Weight: 4g

Connection Diagram:



Sample Code:

```
int temt6000Pin = 0;  
void setup() {  
    Serial.begin(9600);  
}  
  
void loop() {  
    int value = analogRead(temt6000Pin);  
    Serial.println(value);  
  
    delay(100); //only here to slow down the output so it is easier to read  
}
```

Project 33: HC-SR04 Ultrasonic Sensor



Introduction:

The HC-SR04 Ultrasonic Sensor is a very affordable proximity/distance sensor that has been used mainly for object avoidance in various robotics projects. It essentially gives your Arduino eyes / spacial awareness and can prevent your robot from crashing or falling off a table. It has also been used in turret applications, water level sensing, and even as a parking sensor. This simple project will use the HC-SR04 sensor with an Arduino and a Processing sketch to provide a neat little interactive display on your computer screen.

Specification:

Working Voltage: DC 5V

Working Current: 15mA

Working Frequency: 40Hz

Max Range: 4m

Min Range: 2cm

Measuring Angle: 15 degree

keystudio

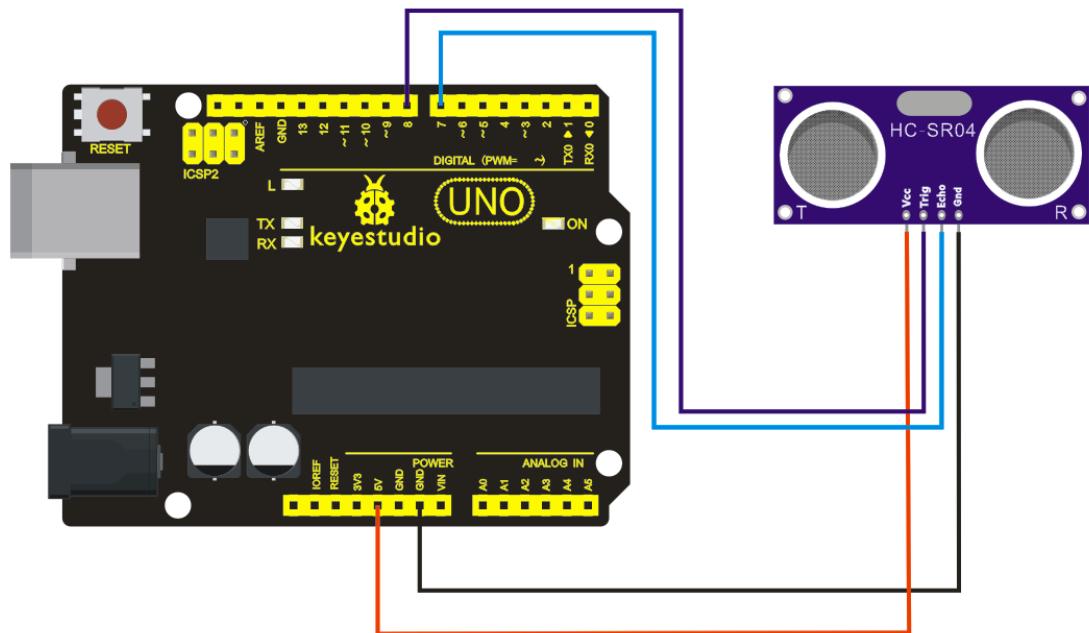
Trigger Input Signal: 10μS TTL pulse

Echo Output Signal Input TTL lever signal and the range in proportion

Size: 46*20.4mm

Weight: 9g

Connection Diagram:



Sample Code:

VCC to arduino 5v

GND to arduino GND

Echo to Arduino pin 7

Trig to Arduino pin 8

```
#define echoPin 7 // Echo Pin  
#define trigPin 8 // Trigger Pin  
#define LEDPin 13 // Onboard LED
```

```
int maximumRange = 200; // Maximum range needed
int minimumRange = 0; // Minimum range needed
long duration, distance; // Duration used to calculate distance

void setup() {
    Serial.begin (9600);
    pinMode(trigPin, OUTPUT);
    pinMode(echoPin, INPUT);
    pinMode(LEDPin, OUTPUT); // Use LED indicator (if required)
}

void loop() {
/* The following trigPin/echoPin cycle is used to determine the
   distance of the nearest object by bouncing soundwaves off of it. */
    digitalWrite(trigPin, LOW);
    delayMicroseconds(2);

    digitalWrite(trigPin, HIGH);
    delayMicroseconds(10);

    digitalWrite(trigPin, LOW);
    duration = pulseIn(echoPin, HIGH);

    //Calculate the distance (in cm) based on the speed of sound.
    distance = duration/58.2;

    if (distance >= maximumRange || distance <= minimumRange){
        /* Send a negative number to computer and Turn LED ON
           to indicate "out of range" */
        Serial.println("-1");
        digitalWrite(LEDPin, HIGH);
    }
    else {
        /* Send the distance to the computer using Serial protocol, and
           turn LED OFF to indicate successful reading. */
        Serial.println(distance);
        digitalWrite(LEDPin, LOW);
    }

    //Delay 50ms before next reading.
    delay(50);
```

}

Project 34: Joystick Module



Introduction:

Lots of robot projects need joystick. This module provides an affordable solution. By simply connecting to two analog inputs, the robot is at your commands with X, Y control. It also has a switch that is connected to a digital pin. This joystick module can be easily connected to Arduino by IO Shield. This module is for Arduino(V5) with cables supplied.

Specification:

keystudio

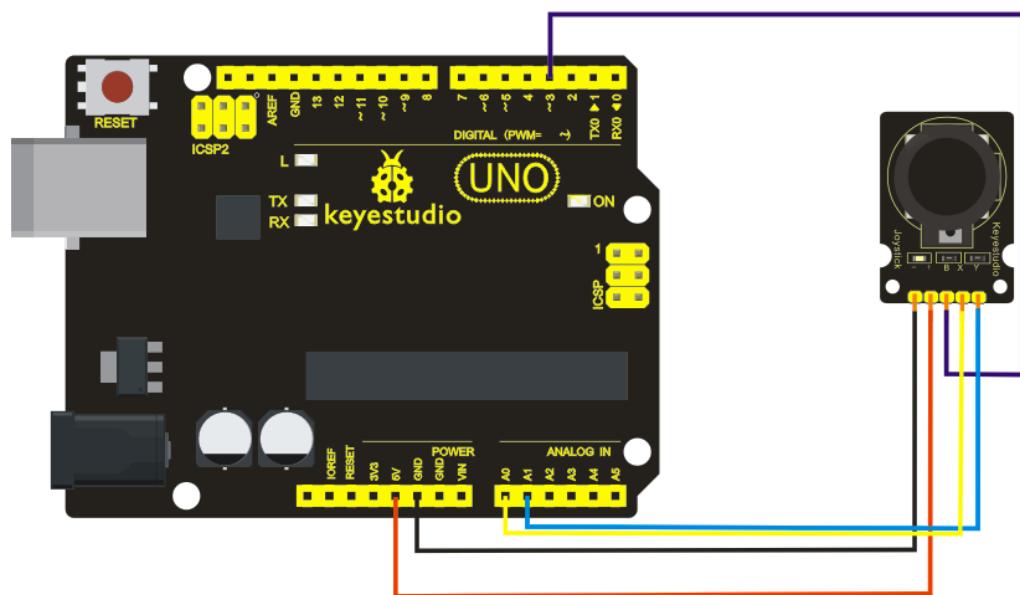
Supply Voltage: 3.3V to 5V

Interface: Analog x2, Digital x1

Size: 40*28mm

Weight: 12g

Connection Diagram:



Sample Code:

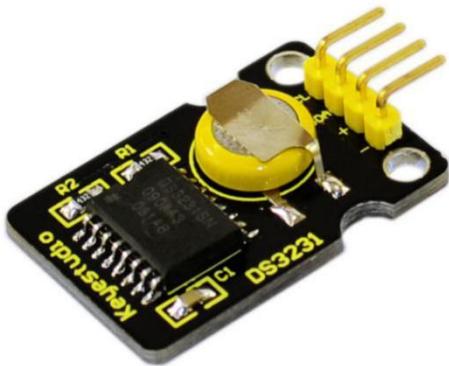
```
int JoyStick_X = 0; //x
int JoyStick_Y = 1; //y
int JoyStick_Z = 3; //key

void setup()
{
    pinMode(JoyStick_Z, INPUT);
    Serial.begin(9600); // 9600 bps
}

void loop()
{
    int x,y,z;
    x=analogRead(JoyStick_X);
    y=analogRead(JoyStick_Y);
    z=digitalRead(JoyStick_Z);
    Serial.print(x ,DEC);
```

```
Serial.print(",");
Serial.print(y ,DEC);
Serial.print(",");
Serial.println(z ,DEC);
delay(100);
}
*****
*****
```

Project 35: DS3231 Clock Module



Introduction:

DS3231 is equipped with integrated TCXO and crystal, which makes it a cost-effective I₂C real time clock with high precision. The device carries a battery input, so if you disconnect the main power supply, it can still maintain accurate timing. The integrated oscillator ensures the long-term accuracy of the device and reduces the number of components. DS3231 provides both commercial and industrial temperature range and supports 16 pins small-outline package (300mil). The module itself can adapt to the system of 3.3V and 5V without level switch, which is quite convenient!

Specification:

Temperature range: -40 to +85; Timing accuracy : ± 5ppm (± 0.432 seconds / day)

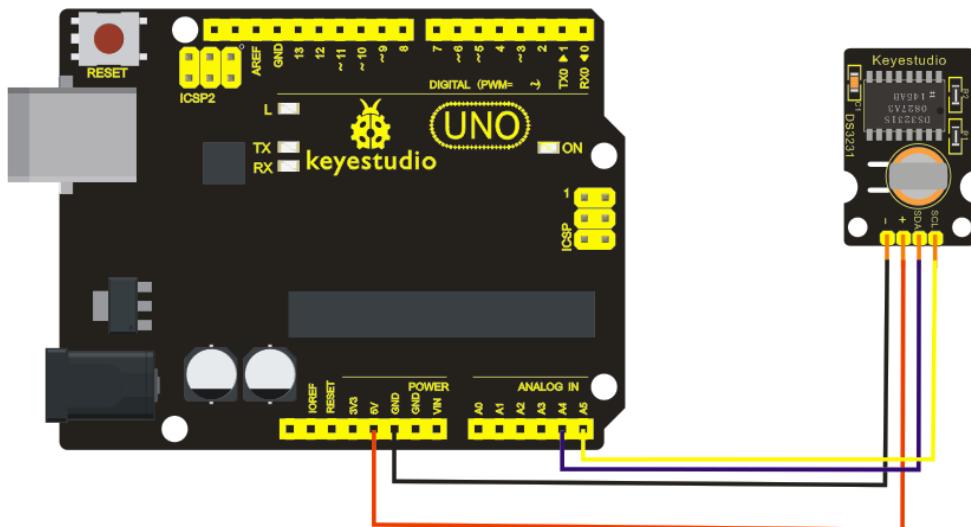
Provide battery backup for continuous timing

keystudio

Low power consumption
Device package and function compatible with DS3231
Complete clock calendar function contains seconds and minutes, hour, week, date, month, and year timing and provides leap year compensation until 2100.
Two calendar clock
Output: 1Hz and 32.768kHz
Reset output and Input Debounce of Pushbutton
High speed (400kHz), I2C serial bus
Supply voltage: +3.3V to +5.5V
Digital temperature sensor with a precision of $\pm 3^{\circ}\text{C}$
Working temperature: -40 ~ C to +85 ~ C
16 pins Small Outline Package (300mil)
Certified by American Association of Underwriters Laboratories (UL)
Size: 30*20mm
Weight: 4g

Connection Diagram:

This module adopts the IIC test method, so we only need to connect 'SDA' to Arduino A4, 'SCL' to A5, '+' to VCC and '-' to GND as follows:



Sample Code:

```
#include <Wire.h>
#include "DS3231.h"
DS3231 RTC; //Create the DS3231 object
char weekDay[][4] = {"Sun", "Mon", "Tue", "Wed", "Thu", "Fri", "Sat" };
//year, month, date, hour, min, sec and week-day(starts from 0 and goes to 6)
//writing any non-existent time-data may interfere with normal operation of the RTC.
```

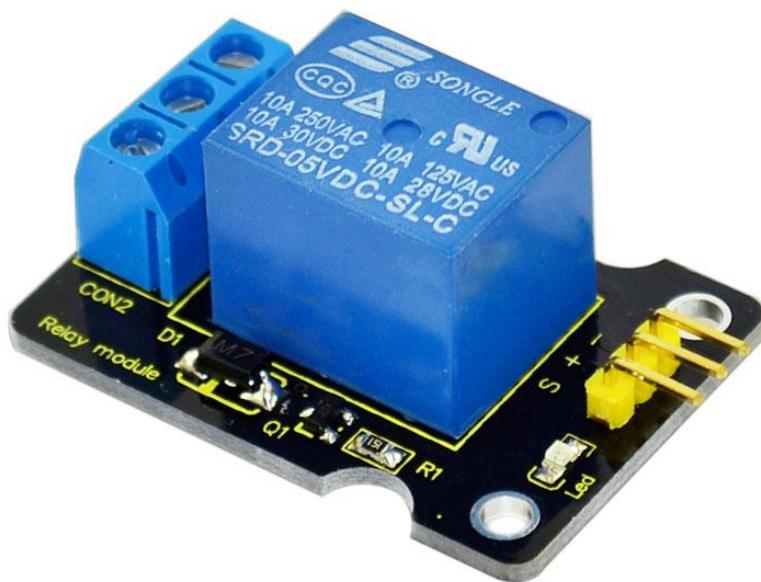
keystudio

```
//Take care of week-day also.  
DateTime dt(2011, 11, 10, 15, 18, 0, 5); //open the series port and you can check time here or  
make a change to the time as needed.  
void setup ()  
{    Serial.begin(57600); //set baud rate to 57600  
    Wire.begin();  
    RTC.begin();  
    RTC.adjust(dt); //Adjust date-time as defined 'dt' above  
}  
void loop ()  
{  
    DateTime now = RTC.now(); //get the current date-time  
    Serial.print(now.year(), DEC);  
    Serial.print('/');  
    Serial.print(now.month(), DEC);  
    Serial.print('/');  
    Serial.print(now.date(), DEC);  
    Serial.print(' ');  
    Serial.print(now.hour(), DEC);  
    Serial.print(':');  
    Serial.print(now.minute(), DEC);  
    Serial.print(':');  
    Serial.print(now.second(), DEC);  
    Serial.println();  
    Serial.print(weekDay[now.dayOfWeek()]);  
    Serial.println();  
    delay(1000);  
}
```

Before compiling the code, you'd better put [DS3231 library](#) under file into Arduino catalogue,. When the above steps are done, you can upload the code to arduino and open the series monitor and get following results:



Project 36: 5V Relay Module



Introduction:

This single relay module can be used in interactive projects. This module uses SONGLE 5v high-quality relay. It can also be used to control lighting, electrical and other equipment. The modular design makes it easy to expand with the Arduino board (not included). The Relay output is by a light-emitting diode. It can be controlled through digital IO port, such as solenoid valves, lamps, motors and other high current or high voltage devices.

Specification:

Type: Digital

Rated current: 10A (NO) 5A (NC)

Maximum switching voltage: 150VAC 24VDC

Digital interface

Control signal: TTL level

Rated load: 8A 150VAC (NO) 10A 24VDC (NO), 5A 250VAC (NO/NC) 5A 24VDC (NO/NC)

keystudio

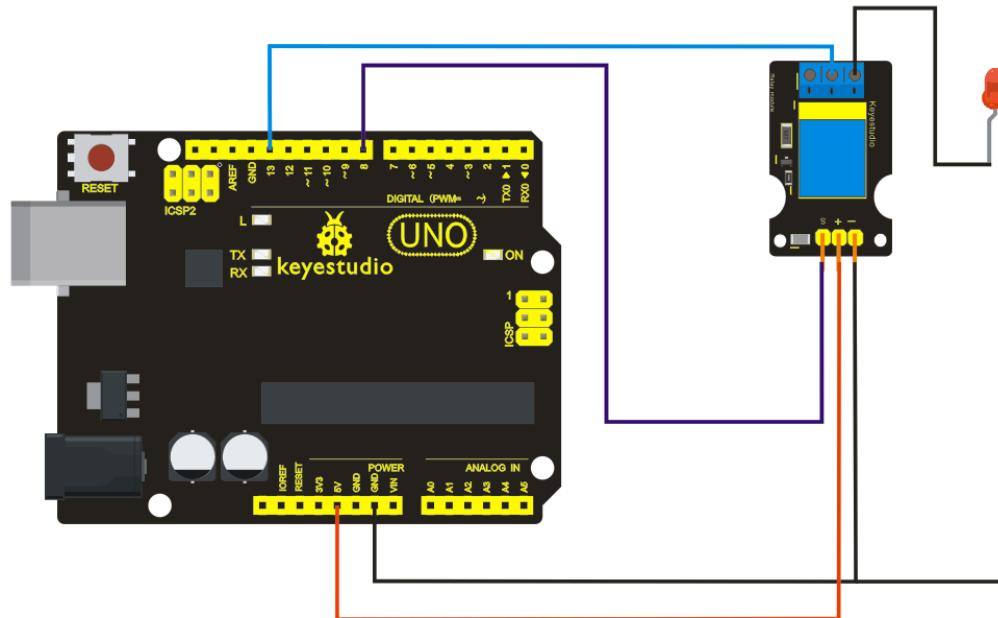
Maximum switching power: AC1200VA DC240W (NO) AC625VA DC120W (NC)

Contact action time: 10ms

Size: 40*28mm

Weight: 15g

Connection Diagram:



Sample Code:

```
int Relay = 8;  
void setup()  
{  
    pinMode(13, OUTPUT);      //Set Pin13 as output  
    digitalWrite(13, HIGH);   //Set Pin13 High  
    pinMode(Relay, OUTPUT);  //Set Pin3 as output  
}  
void loop()  
{  
    digitalWrite(Relay, HIGH); //Turn off relay  
    delay(2000);  
    digitalWrite(Relay, LOW); //Turn on relay  
    delay(2000);  
}  
*****
```