

User Manual







For 37 in 1 Sensor Kit(ST1065)

Module List





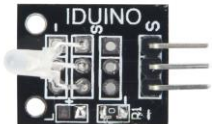

The order of modules is corresponding with the grid's location, the last grid contains two modules.

	Name	Quantity	Picture
1	Joystick Module	1	
2	Relay Module	1	
3	Large Microphone Module	1	
4	Small Microphone Module	1	
5	Line Tracking Module	1	



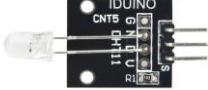



IDUINO for maker's life

6	Obstacle Avoidance Sensor	1	
7	Flame Sensor Module	1	
8	Linear Magnetic Hall Sensor	1	
9	Touch Sensor	1	
10	Digital Temperature Sensor	1	
11	Buzzer Module	1	


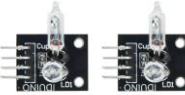

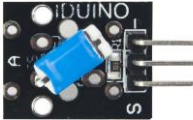

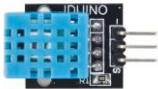
IDUINO for maker's life

12	Passive Buzzer/Sounder	1	
13	RGB LED Module	1	
14	SMD RGB LED Module	1	
15	Two Color LED Module	1	
16	LED Module	1	
17	Reed Switch Module	1	




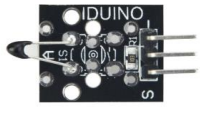


IDUINO for maker's life

18	Mini Reed Switch Module	1	
19	Heartbeat Sensor	1	
20	Seven Color Flashing	1	
21	Laser Module	1	
22	Tactile Switch Module	1	
23	Shock Module	1	



IDUINO for maker's life

24	Rotary Encode Module	1	
25	Magic Light Cup Module	2	
26	Tilt Switch Module	1	
27	Ball Switch Module	1	
28	Light Dependent Resistor Module	1	
29	Temperature and Humidity Module	1	

IDUINO for maker's life

30	Hall Effect Sensor	1	
31	Class Bihor Magnetic Sensor	1	
32	Digital Temperature Module	1	
33	Analog Temperature Sensor	1	
34	IR Transmitter Module	1	
35	IR Receiver Module	1	

IDUINO for maker's life

36	Optical Broken Module	1	
37	Hit Sensor Module	1	

Module 2: Relay Module



1. Introduction

The module is uses SRD relay module to control high-voltage electrical devices. (maximum 250V). It can be used in interactive projects and can also be used to control the lighting, electrical and other equipments. It can be controlled directly by a wide range of microcontrollers and can be controlled through the digital IO port, such as solenoid valves, lamps, motors and other high current or high voltage devices.

specifications:

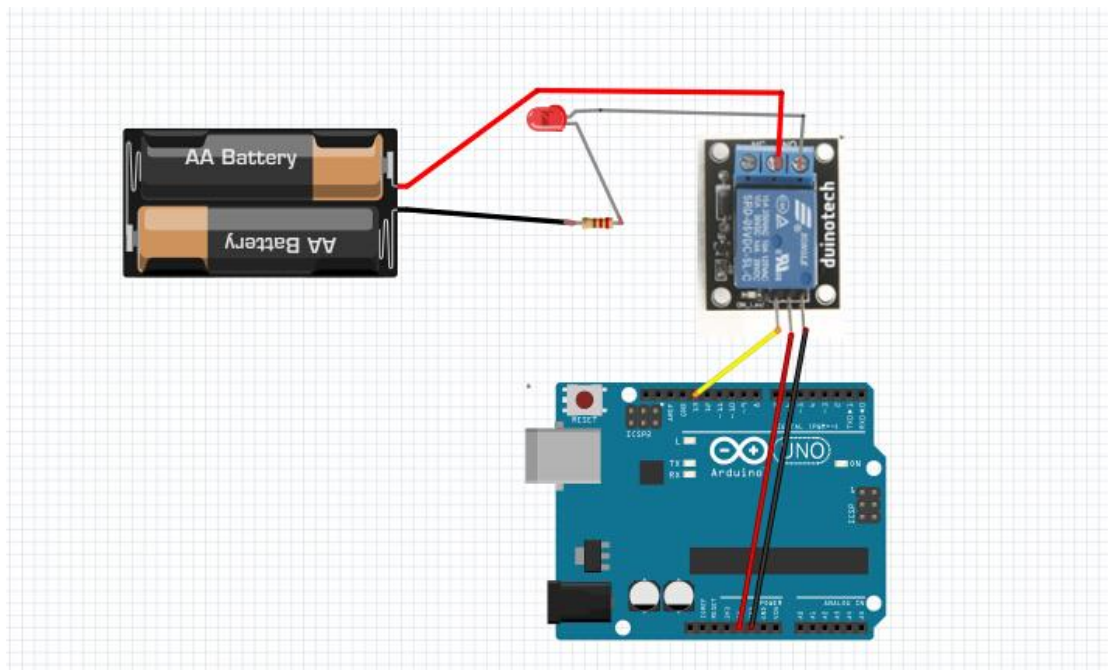
- Number of I/O Channels: 1
- Type: Digital
- Switching capacity available by 10A in spite of small size design for high density P.C. board mounting technique.
- Control signal: TTL level
- Max. Allowable Voltage: 250VAC/110VDC
- Max. Allowable Power Force: From C(800VAC/240W), From A(1200VA/300W)
- UL, CUL, TUV recognized.
- Indication LED for Relay's Status

2. Pinout

Pin Name	Description
"+"	Power(5V DC)
"_"	Gnd
"S"	Singal pin, connected with Arduino
"NO"	Normally Open Connection
"NC"	Normally Closed Connection
"C"(middle pin)	Common Connection, Which connected with the power for the load.

3. Example

This example controls a LED(or other high power load) via the Relay module.
Physical connection as below:



The example code as below:

*****Code begin*****

```
int led = 13;
```

```
// the setup routine runs once when you press reset:
```

```
void setup() {
```

```
    // initialize the digital pin as an output.
```

```
    pinMode(led, OUTPUT);
```

```
}
```

```
// the loop routine runs over and over again forever:
```

```
void loop() {
```

```
    digitalWrite(led, HIGH);    // turn the LED on (HIGH is the voltage level)
```

```
    delay(1000);                // wait for a second
```

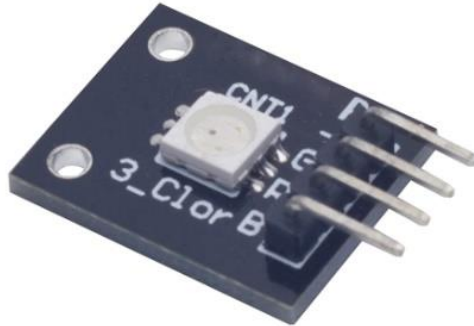
```
    digitalWrite(led, LOW);    // turn the LED off by making the voltage LOW
```

```
    delay(1000);                // wait for a second
```

```
}
```

*****Code End*****

Module14 : SMD RGB LED Module



1. Introduction

RGB LED module consists of a full-color LED made by R, G, B three pin PWM voltage input can be adjusted. Primary colors (red / blue / green) strength in order to achieve full color mixing effect. Control of the module with the Arduino can be achieved Cool lighting effects.

Specification

- Red Vf: 1.8 to 2.1V
- Green Vf: 3.0 to 3.2V
- Blue Vf: 3.0 to 3.2V
- Red color: 620-625 nm
- Green color: 520-525 nm
- Blue color: 465-470 nm
- Red brightness @ ~20mA: 600-800 mcd
- Blue brightness @ ~20mA: 800-1000 mcd
- Green brightness @ ~20mA: 1500-2000mcd

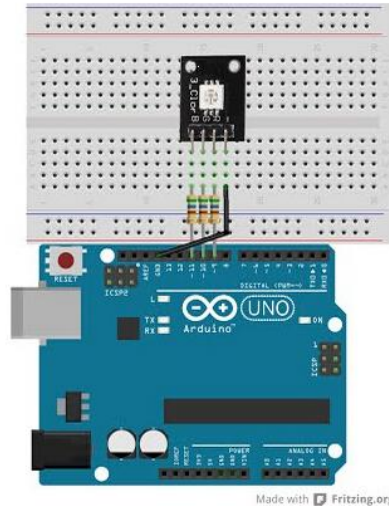
2. Pinout

Pin Name	Description
"R"	Red light
"G"	Green light
"B"	Blue light
"_"	Ground

3. Example

In this example, we blink an LED and using an RGB LED we can generate any color we want.

Here is the physical connection:



Code:

```
*****Code begin*****  
//RGB LED pins  
int ledDigitalOne[] = {10, 11, 9}; //the three digital pins of the digital LED  
                                //10 = redPin, 11 = greenPin, 9 = bluePin  
  
const boolean ON = HIGH;      //Define on as LOW (this is because we use a common  
                                //Anode RGB LED (common pin is connected to +5  
volts)  
const boolean OFF = LOW;     //Define off as HIGH  
  
//Predefined Colors  
const boolean RED[] = {ON, OFF, OFF};  
const boolean GREEN[] = {OFF, ON, OFF};  
const boolean BLUE[] = {OFF, OFF, ON};  
const boolean YELLOW[] = {ON, ON, OFF};  
const boolean CYAN[] = {OFF, ON, ON};  
const boolean MAGENTA[] = {ON, OFF, ON};  
const boolean WHITE[] = {ON, ON, ON};  
const boolean BLACK[] = {OFF, OFF, OFF};  
  
//An Array that stores the predefined colors (allows us to later randomly display a color)  
const boolean* COLORS[] = {RED, GREEN, BLUE, YELLOW, CYAN, MAGENTA, WHITE,  
BLACK};
```

```
void setup(){
  for(int i = 0; i < 3; i++){
    pinMode(ledDigitalOne[i], OUTPUT);    //Set the three LED pins as outputs
  }
}

void loop(){

  /* Example - 1 Set a color
   Set the three LEDs to any predefined color
  */
  setColor(ledDigitalOne, YELLOW);    //Set the color of LED one

  /* Example - 2 Go through Random Colors
   Set the LEDs to a random color
  */
  //randomColor();

}

void randomColor(){
  int rand = random(0, sizeof(COLORS) / 2);    //get a random number within the range
  of colors
  setColor(ledDigitalOne, COLORS[rand]);    //Set the color of led one to a random
  color
  delay(1000);
}

/* Sets an led to any color
   led - a three element array defining the three color pins (led[0] = redPin, led[1] =
  greenPin, led[2] = bluePin)
   color - a three element boolean array (color[0] = red value (LOW = on, HIGH = off),
  color[1] = green value, color[2] =blue value)
  */
void setColor(int* led, boolean* color){
  for(int i = 0; i < 3; i++){
    digitalWrite(led[i], color[i]);
  }
}

/* A version of setColor that allows for using const boolean colors
  */
void setColor(int* led, const boolean* color){
```

```
boolean tempColor[] = {color[0], color[1], color[2]};  
setColor(led, tempColor);  
}  
*****Code End*****
```

Module 6: Obstacle Avoidance Sensor



1. Introduction

Infrared obstacle avoidance sensor is designed for the design of a wheeled robot obstacle avoidance sensor distance adjustable. This ambient light sensor adaptable, high precision, having a pair of infrared transmitter and receiver, transmitter tubes emit a certain frequency of infrared, When detecting the direction of an obstacle (reflector), the infrared receiver tube receiver is reflected back, when the indicator is lit, Through the circuit, the signal output interface output digital signal that can be detected by means of potentiometer knob to adjust the distance, the effective distance From 2 ~ 40cm, working voltage of 3.3V-5V, operating voltage range as broad, relatively large fluctuations in the power supply voltage of the situation Stable condition and still work for a variety of microcontrollers, Arduino controller, BS2 controller, attached to the robot that Can sense changes in their surroundings.

Specification:

- Working voltage: DC 3.3V-5V
- Working current: $\geq 20\text{mA}$
- Operating temperature: $-10\text{ }^{\circ}\text{C} - +50\text{ }^{\circ}\text{C}$
- detection distance :2-40cm
- IO Interface: 4-wire interfaces (- / + / S / EN)
- Output signal: TTL level (low level there is an obstacle, no obstacle high)
- Adjustment: adjust multi-turn resistance
- Effective angle: 35 °
- Size: $28\text{mm} \times 23\text{mm}$
- Weight Size: 9g

2. Pinout

Pin	Description
“+”	Power(3.3V~5V DC)
Gnd	ground
out	Signal pin
EN	Enable pin that Low level works, usually useless

3.Example code

Here is a small example to test the sensor. By default, the sensor returns 1 on the Serial Monitor. When detecting something, the sensor return 0.

The connection as below:

```
“+”=====5V
GND=====Gnd
Out=====9
```

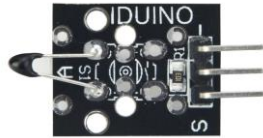
```
*****Code begin*****
```

```
int count;
void setup() {

  Serial.begin (9600);
  pinMode (9, INPUT); //Sensor output
}
void loop() {
  Serial.print ("Sensor: ");
  Serial.println (digitalRead(9)); //print the sensor output
  delay (500); //wait half a second
}
```

```
*****Code End*****
```

Module30: Analog Temperature Sensor



1. Introduction

A thermistor is a type of resistor whose resistance is dependent on temperature, more so than in standard resistors. The word is a portmanteau of thermal and resistor. Thermistors are widely used as inrush current limiter, temperature sensors (NTC type typically), self-resetting overcurrent protectors, and self-regulating heating elements.

The Module's feature as below:

Feature	Value
Model No.	NTC-MF52 3950
Temperature Range	-55°C~+125°C
Accuracy	+/- 0.5°C
Pull-up resistor	10KΩ

2.Pinout

Pin	Description
"S"	Singal pin
"_"	Gnd
"+"	Vcc(reference voltage:5V DC)

Temperature convert Formula

Here we use Steinhart–Hart equation to calculate the corresponding temperature. The equation is

$$\frac{1}{T} = A + B \ln(R) + C [\ln(R)]^3,$$

where:

T is the temperature (in Kelvins)

R is the resistance at T (in ohms)

A , B , and C are the Steinhart–Hart coefficients which vary depending on the type and model of thermistor and the temperature range of interest. (The most general form of the applied equation contains a $[\ln(R)]^2$ term, but this is frequently neglected because it is typically much smaller than the other coefficients).

Note: For this module, the recommended coefficients of A, B, C are

A equals 0.001129148;

B equals 0.000234125;

C equals 0.0000000876741;

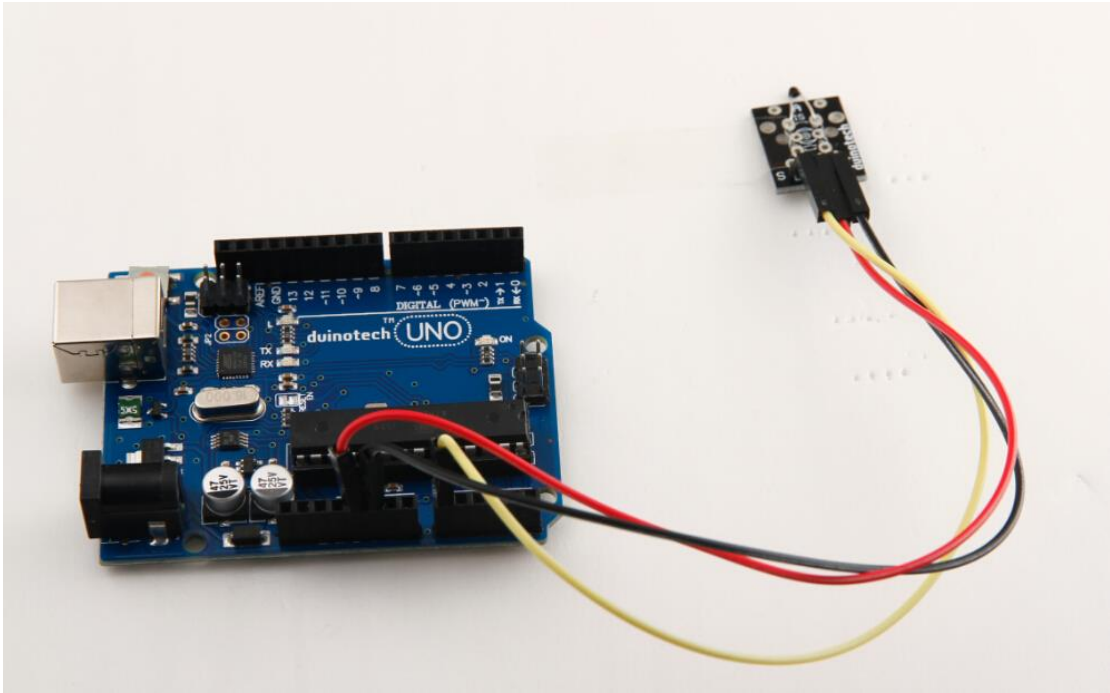
More, the same item products has a little bit different A, B, C coefficients, which depends your environmental temperature. If the recommended coefficients are not accurate enough, you'd better amend the A, B, C coefficients by Thermistor Calculator tool.

3 Example

This is a simple code for the NTC thermistor module, Wire as below:

"S" pin----- A0;
"+" pin-----5V;
"-" pin-----Gnd;

The physical picture is



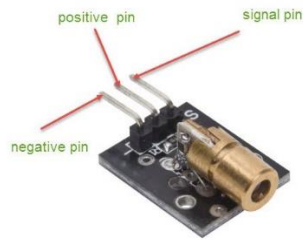
Example code :

*****Code begin*****

```
#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;
return Temp;
}
void setup() {
Serial.begin(9600);
}
void loop()
{ Serial.print(Thermister(analogRead(0)));
  Serial.println("c");
  delay(1000); }
```

*****Code End*****

Module 21: Laser Module



1. Introduction

Laser transmitter module, 650 nm (red), gives a small intense beam. Take care of your eyes, do not look direct into the beam.

2. Pinout

Pin	Name	Description
"_"		Gnd
"S"		Singal pin(input)
"+"		Power(5V DC)

3. Example

This module can be used simply, example code as below, which control the laser diode to turn on and turn off alternately.

```
"_"===== Gnd  
"+"===== 5V  
"S"=====13
```

*****Code begin*****

```
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
```

IDUINO for maker's life

```
    delay (1000); // delay one second  
}
```

```
*****Code begin*****
```