YOUTH-IIT

Young Organizers Using Technology for Hybrid Innovative IoT Solutions and Collaborative Tools

IoT sensors, examples and applications

Young Organizers Using Technology for Hybrid Innovative IoT Solutions and Collaborative Tools

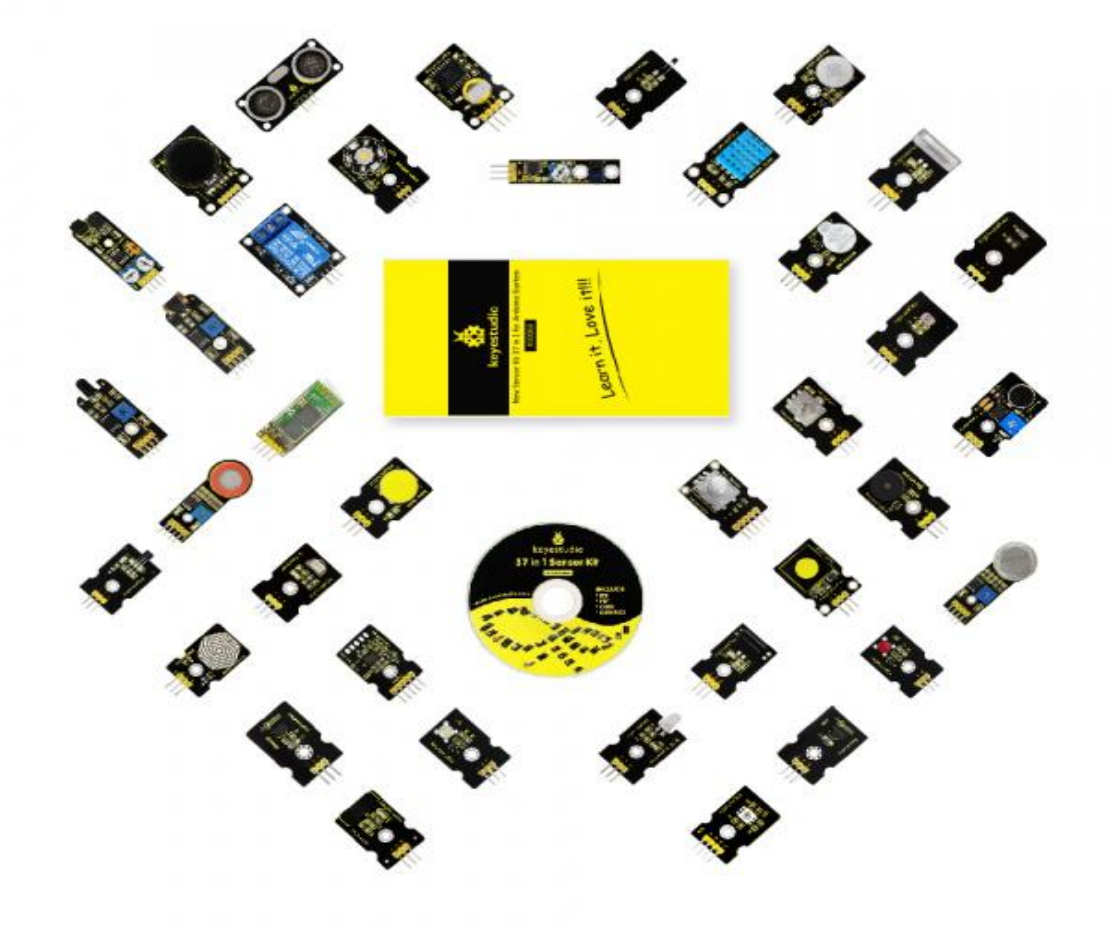
YOUTH-IIT

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the Youth and Lifelong Learning Foundation (INEDIVIM). Neither the European Union nor the granting authority can be held responsible for them.

ARDUINO MODULES

This thesis extends to 37 modules and presents basic sensors which are digital and analogue as well as some special modules such as ultrasound, Bluetooth, acceleration module, etc.

For each module, there is a clear connection diagram and sample code. So, even if the user is completely new to it, they can easily start understanding the process. The sample codes for this sensor kit are based on ARDUINO. Arduino is an open-source, easy-to-use software.



INTRODUCTION

Arduino was a small revolution in the field of technology, allowing a multitude of interested parties the opportunity to easily develop the application they want, quickly and at low cost. Today it is the largest development platform on the entire planet. This "small computer" is a fairly important creative tool that can be used by young and old alike. So the Arduino is rightly given

the title that it holds the ticket for those who start experimenting with the field of modern applications, while being a creative pastime for children, offering them satisfaction and self-confidence and highlighting their skills and talent.

In addition, students who use it implement experimental devices for applications in health, industry, measurements, etc. Still other engineers use this technology for artificial intelligence applications where a swarm of autonomous robotic machines listen to the environment and implement natural processes.

In Greece it is used by pupils, students, engineers, professionals and hobbyists. This is mainly due to the combination of its features which are its low cost, open architecture, available software tools, ease of programming, variety of sensors and other circuits.

https://wiki.keyestudio.com/Ks0068_keyestudio_37_in_1_Sensor_Kit_for_Arduino_Starters

CHAPTER ONE

PROJECT 1: WHITE LED LIGHT

1.1 Introduction

This is a white LED module. Its main function is to turn on and off an LED plugin. When connecting to the ARDUINO, after programming, it will emit a white color of light.

The pin pitch of 3Pin is 2.54mm. This LED module has 3 pins. - The pin is connected to the ground, + the pin is connected to the VCC (3.3-5V), the S pin is for signal control. It can adjust the high or low level to control the on and off of the LED.

It can combine it with other sensors to do various interactive experiments. It can also choose other LED modules to emit different colors of light such as blue, green, yellow, and red.

1.2 Features

Control Interface: Digital

Working Voltage: DC 3.3-5V

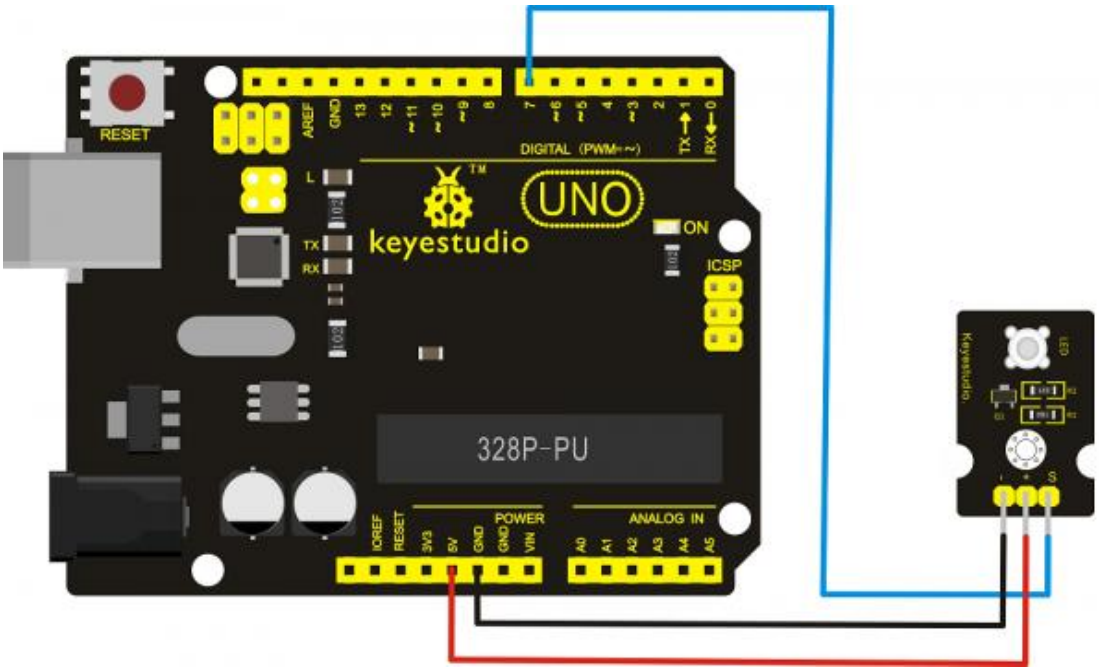
Pin pitch: 2.54 mm

LED color: white

Easy to use

Useful for light projects

It has the ability to connect the LED module to the control panel using three jumper wires. Next, it connects the control board to the computer with a USB cable.



1.3 The Code

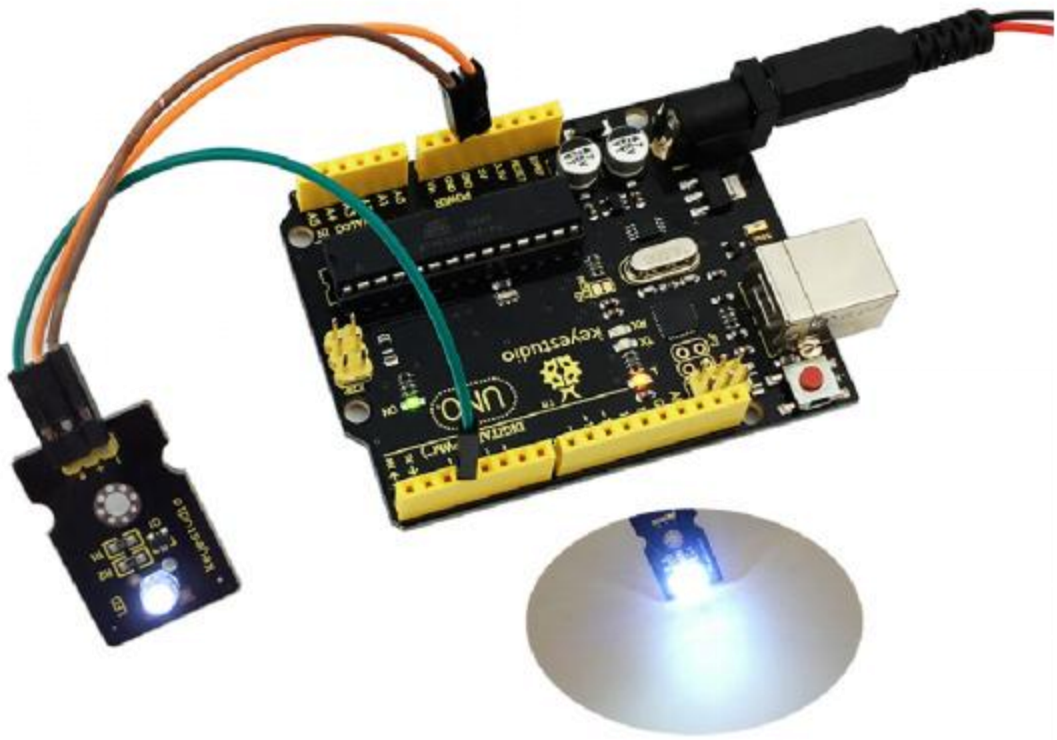
The code is as follows:

```
int led = 7;
void setup()
{
  pinMode(led, OUTPUT);     //Set Pin7 as output
}
void loop()
{    digitalWrite(led, HIGH);   //Turn off led
          delay(1000);
          digitalWrite(led, LOW);    //Turn on led
          delay(1000);
}
```

1.4 Test result

The LED will flash for one second, then turn off for one second, repeatedly and alternately.

If it does not, the user must make sure that they have assembled the circuit correctly and verified and uploaded the code to the board.



.

CHAPTER TWO

PROJECT 2: RED LED LIGHT

2.1 Introduction

This is a red LED module. Its main function is to turn on and off an LED plugin. When connecting to ARDUINO, after programming, it will emit a red light.

The pin pitch of 3Pin is 2.54mm. This LED module has 3 pins. - The pin is connected to the ground, + the pin is connected to the VCC (3.3-5V), the S pin is for signal control. it can adjust the high or low level to control the on and off of the LED.

It can combine it with other sensors to do various interactive experiments.

It can also select other LED modules to emit different color of light such as blue, green, yellow, and white.

2.2 Features

Control Interface: Digital

Working Voltage: DC 3.3-5V

Pin pitch: 2.54 mm

LED color: red

Easy to use

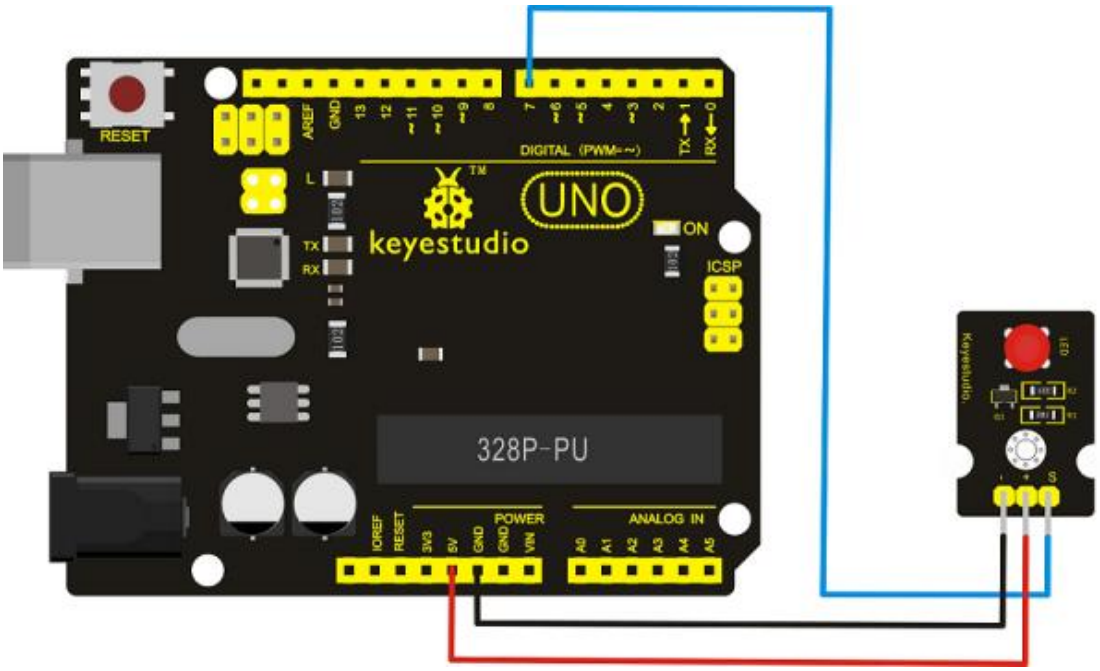
Useful for light projects

Technical Details

Dimensions: 34mm*20mm*11.5mm

Weight: 2.7g

It is necessary to connect the red LED module to the control panel using three jumper wires. The control board is then connected to the computer with a USB cable.



2.3 Posting of the Code

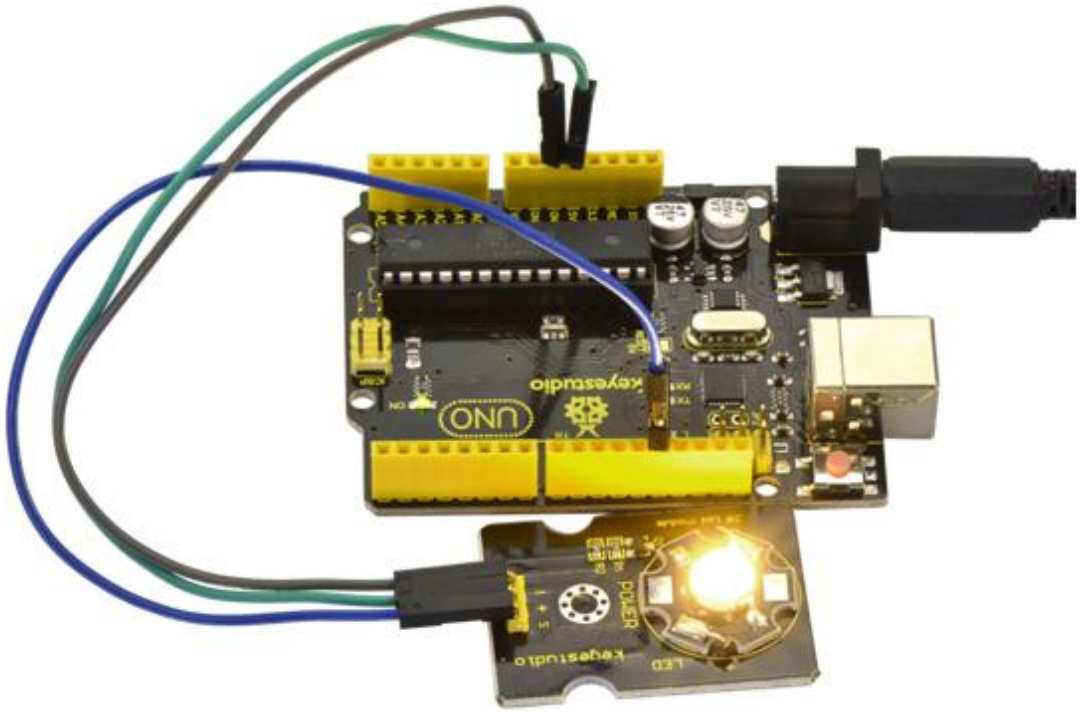
The following code needs to be copied and pasted into the Arduino IDE

```
int led = 7;
void setup()
{
  pinMode(led, OUTPUT);     //Set Pin7 as output
}
void loop()
{    digitalWrite(led, HIGH);   //Turn off led
          delay(1000);
          digitalWrite(led, LOW);    //Turn on led
          delay(1000);
}
```

2.4 Test result

The LED will flash for one second, then turn off for one second, repeatedly and alternately.

If they don't, the user needs to make sure that they have assembled the circuit correctly and verified and uploaded the code to the board.



CHAPTER THREE

PROJECT 3: 3W LED MODULE

3.1 Introduction

This LED module is high brightness because the lamp beads it carries are 3W. User can apply this module to Arduino projects.

For example, smart robots can use this unit for lighting purposes.

Let the user take into account that the LED light cannot be directly exposed to human eyes for safety reasons.

3.2 Specification

Color Temperature: 6000~7000K

Luminous Flux: 180~210lm

Current: 700~750mA

IO Type: Digital

Supply voltage: 3.3V to 5V

Power: 3W

Light Angle: 140 degrees

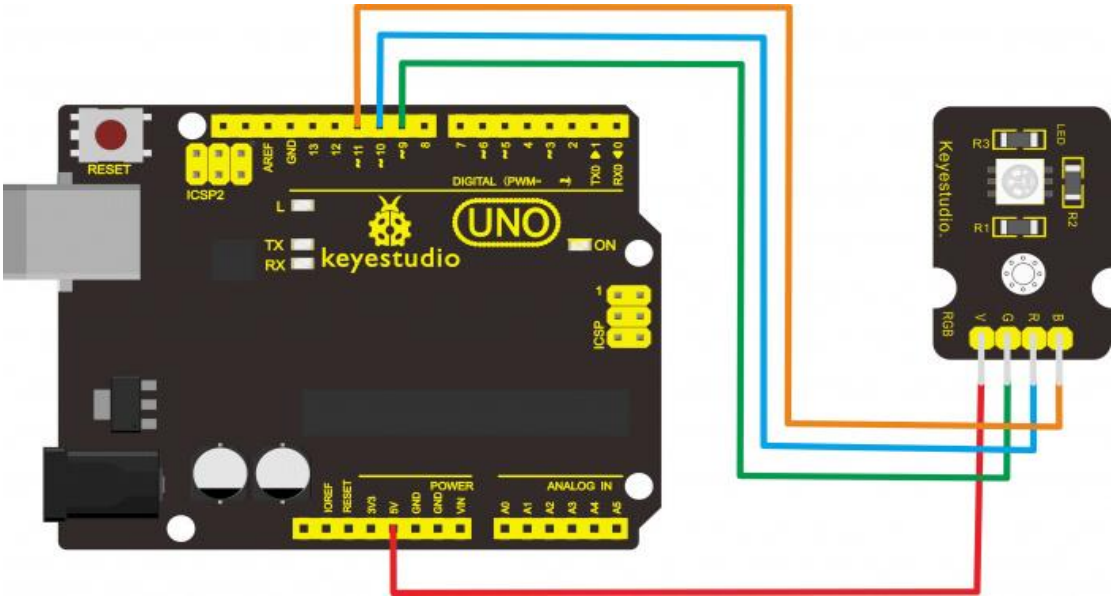
Operating Temperature: -50~80°C

Storage Temperature: -50~100°C

High-power LED module controlled by IO port microcontroller

Ideal for robots and search and rescue platform application

3.3 Connection diagram

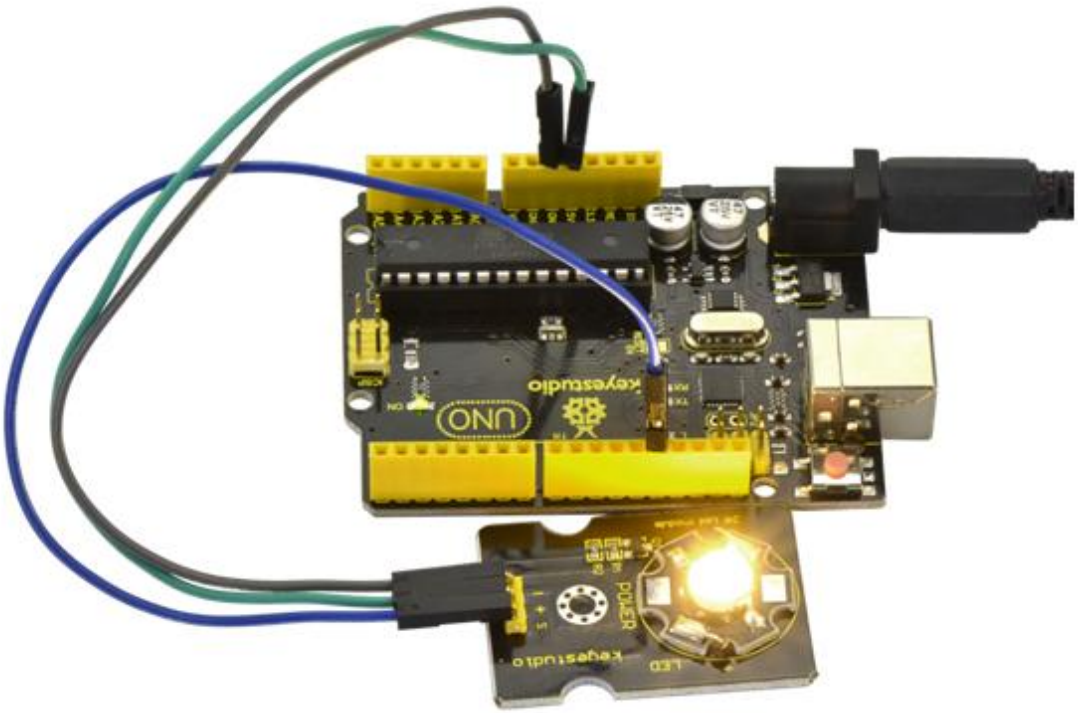


3. 4 The Code

```
// 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
}
```

3.5 Result

When the wiring is finished, it turns on. The code must be correctly raised and then both the D13 led and the module led will flash for one second and then turn off, circularly.

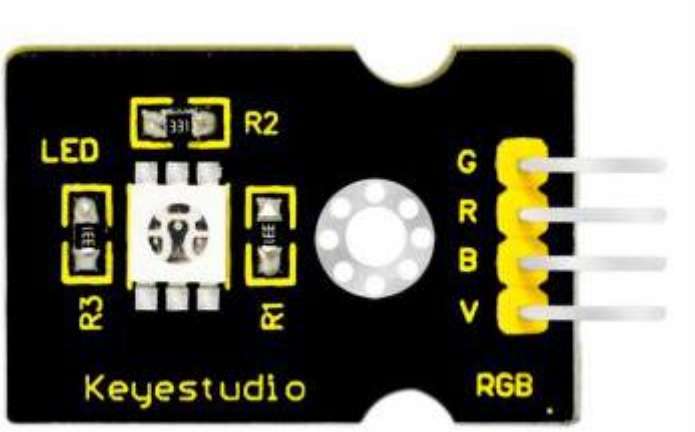

keyestudio
UNO
POWER
LED
keyestudio

CHAPTER FOUR

PROJECT 4: RGB LED

4.1 Introduction

This is a full-color LED module, which contains 3 primary colors - red, green, and blue. They can be considered as separate LED lights. After programming, it can turn them on and off in order, or it can also use analog PWM output to mix three colors to create different colors.



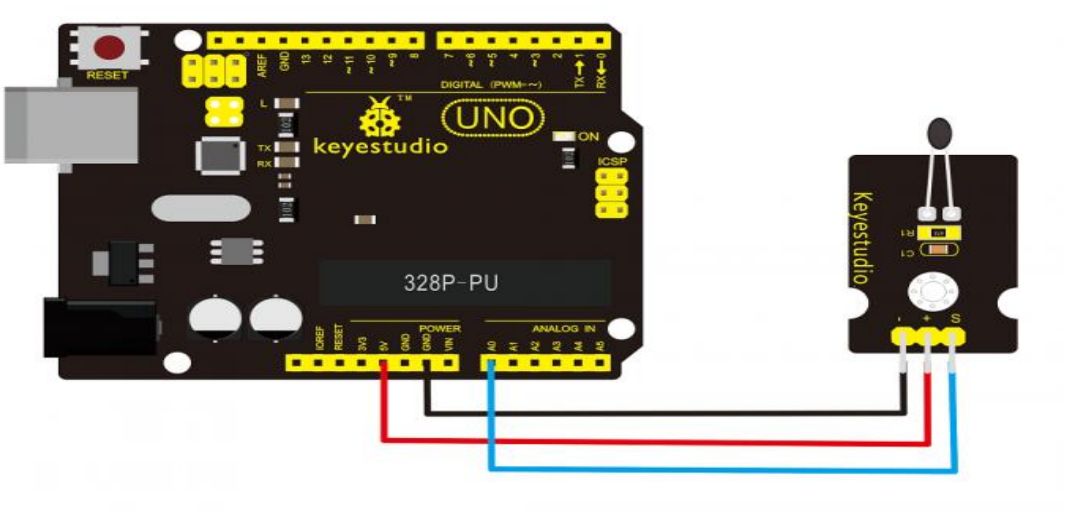
4.2 Specification

Color: red, green and blue

Brightness: High

Voltage: 5V

Input: digital level

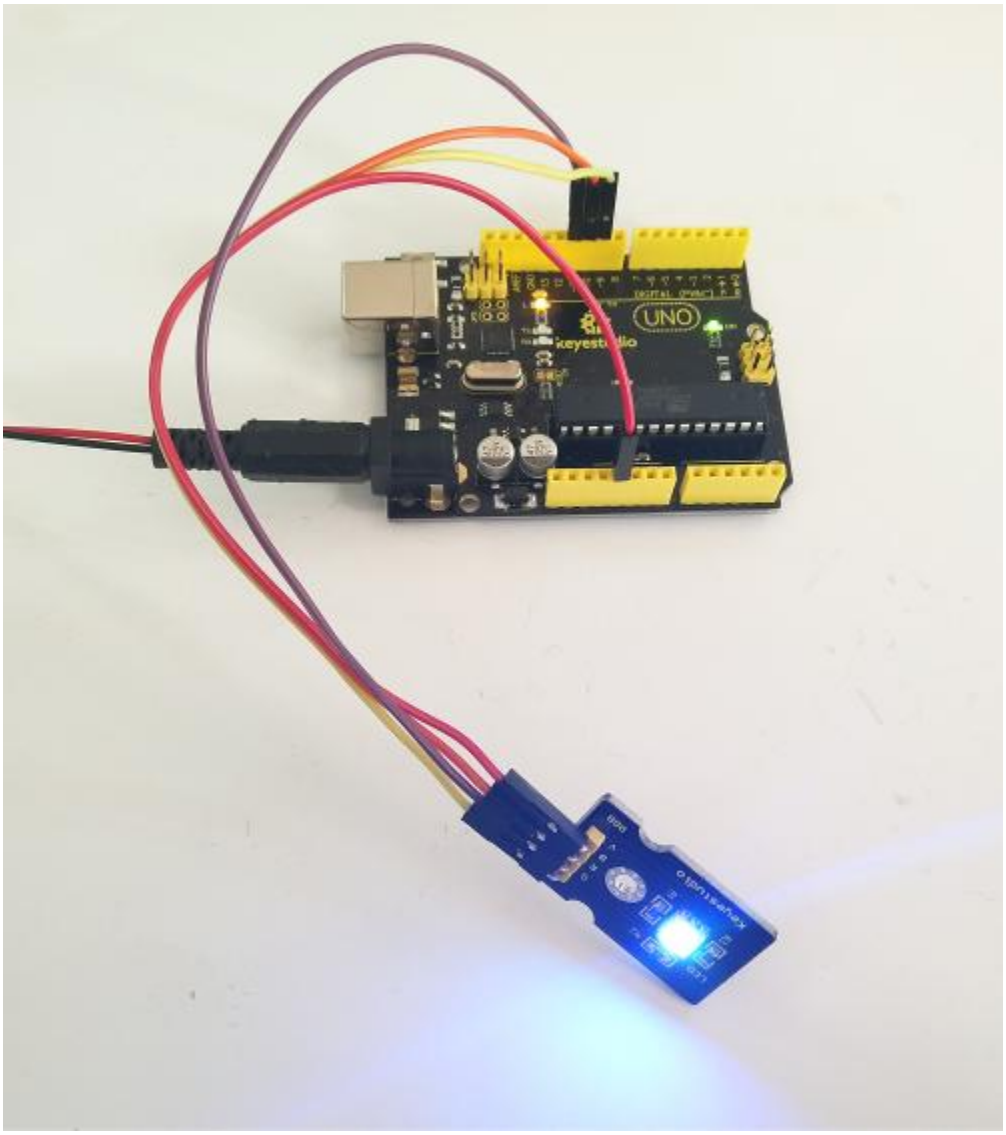


4.3 The Code

```
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);
  }
}
```

4.4 Result

By finishing uploading the code, it should appear that the RGB LED flashes with different colors.



CHAPTER FIVE

PROJECT 5: ANALOG TEMPERATURE

5.1 Introduction

This unit is based on the principle of operation of a thermistor (resistance varies depending on the change in temperature in the environment). It can sense the temperature change in its environment and send the data to the analog IO on the Arduino board. All it needs to do is convert the sensor's output data into degrees Celsius through simple programming and finally display it. It is both convenient and effective, so it is widely applied in gardening, home alarm system and other devices.

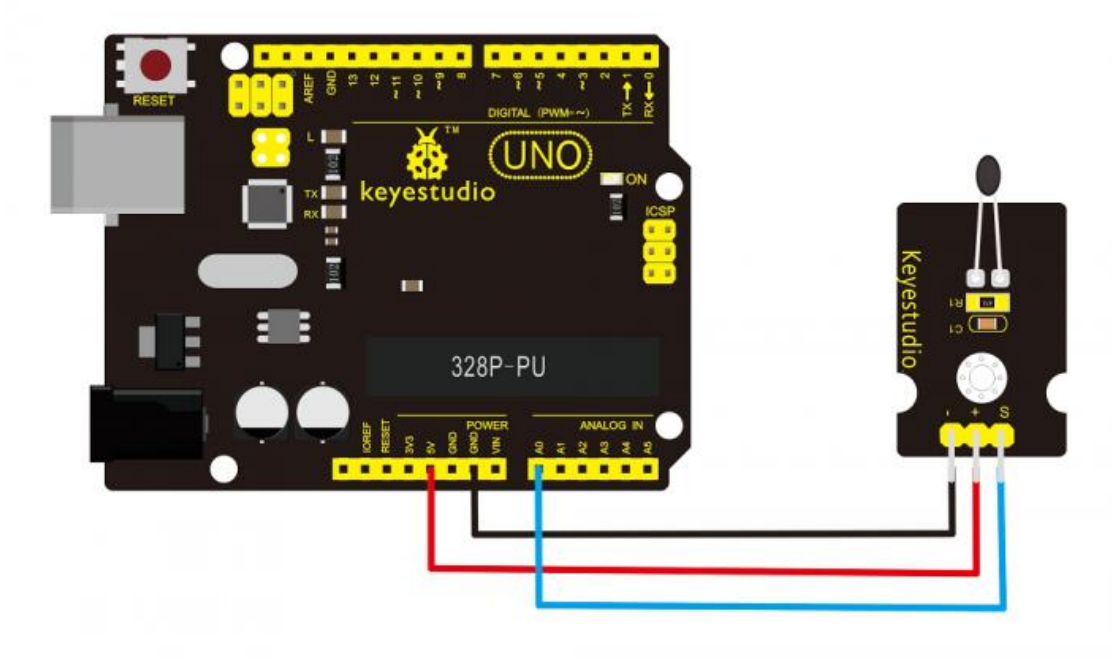
5.2 Specification

Interface Type: Analog

Operating Voltage: 5V

Temperature range: -55°C-315°C

Connection diagram



5.3 Sample Code

The code is as follows:

```
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); }
```

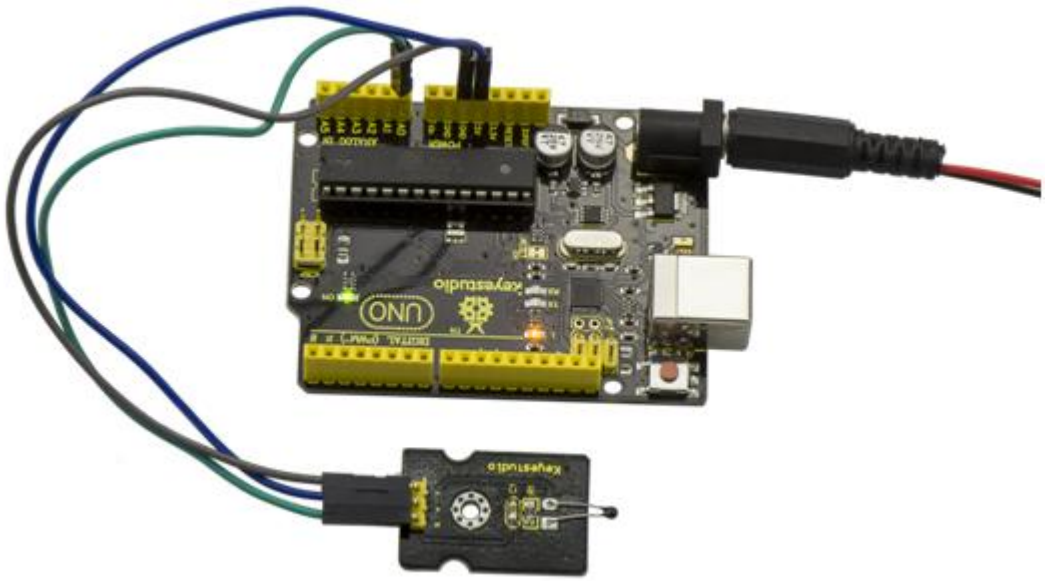
The above code is for proportional price only. It can be seen that the proportional value changes according to the temperature change in the environment. Sometimes this is not very obvious and

is solved using the equation below. It then uploads the following code to the Arduino board. The value read by the serial port is similar to the normal temperature.

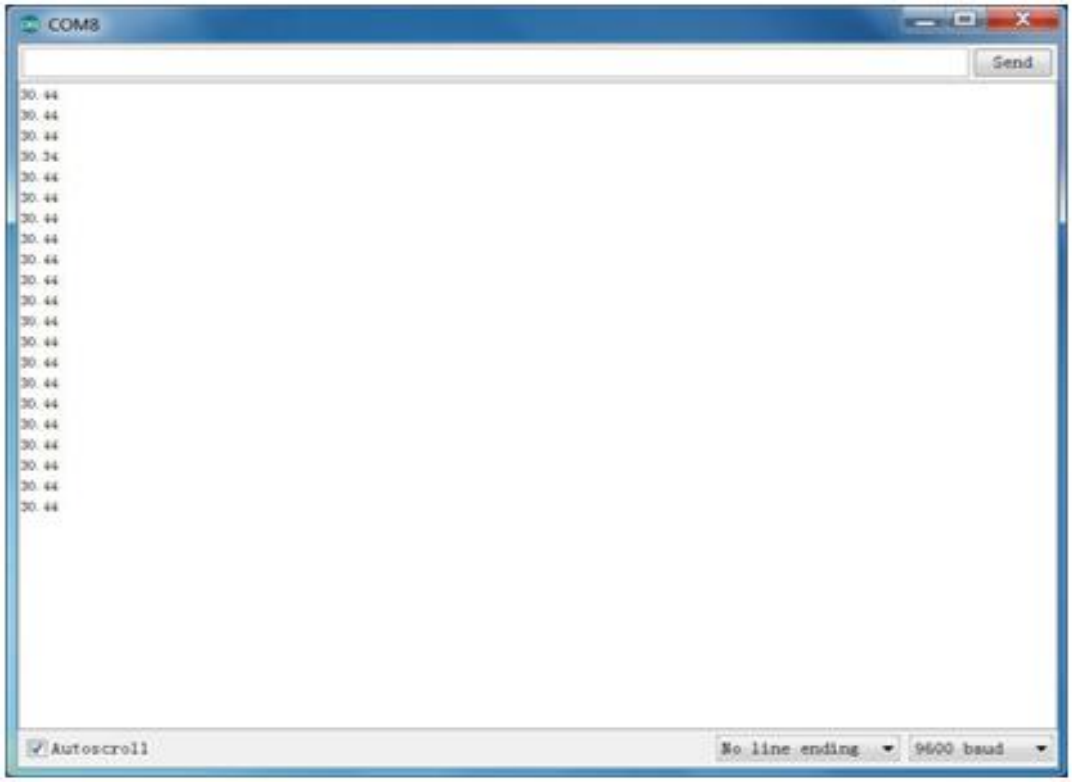
e.g. The temperature at this time is 30°C.

```
#include <math.h>
void setup()
{
   Serial.begin(9600);
}
void loop()
{
  double val=analogRead(0);
  double fenya=(val/1023)*5;
  double r=(5-fenya)/fenya*4700;
  Serial.println( 1/(  log(r/10000) /3950 +
1/(25+273.15))-273.15);
  delay(1000);
}
```

5.4 Results



After the wiring and power is completed, a serial screen needs to be opened and the current temperature value is displayed as shown below.



CHAPTER SIX

PROJECT 6: PHOTOCELL SENSOR

6.1 Introduction

The photocell appears in several uses in everyday life. It is mainly used in smart switches and common electronic design.

To make it easier and more efficient, it provides the corresponding modules. The photocell is a semiconductor. It possesses characteristics of high sensitivity, fast response, spectral characteristics, and R-value consistency, maintaining high stability and reliability in extreme environmental conditions such as high temperature, high humidity.

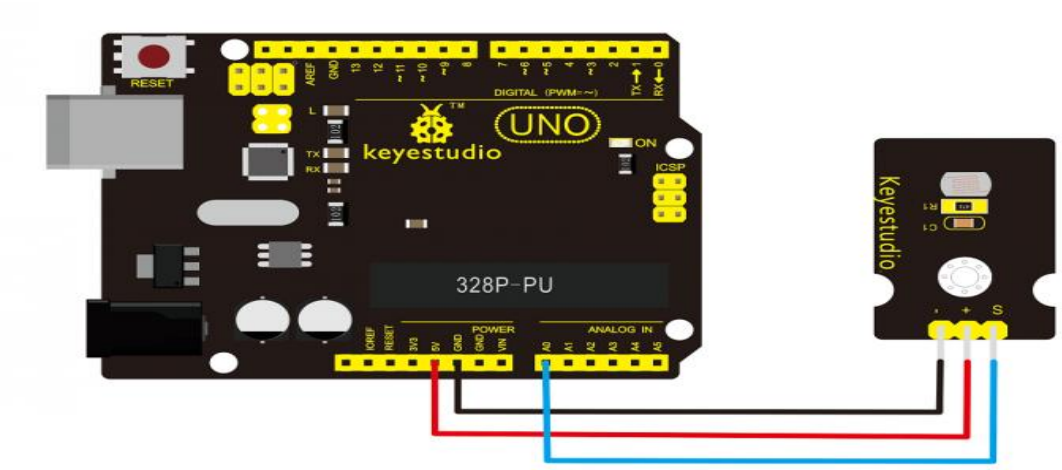
It is widely used in automatic control switch fields such as cameras, solar garden lights, lawn lamps, money detectors, quartz clocks, music cups, gift boxes, mini night lights, sound and light control switches, etc.

6.2 Specification

Interface Type: Analog

Operating Voltage: 5V

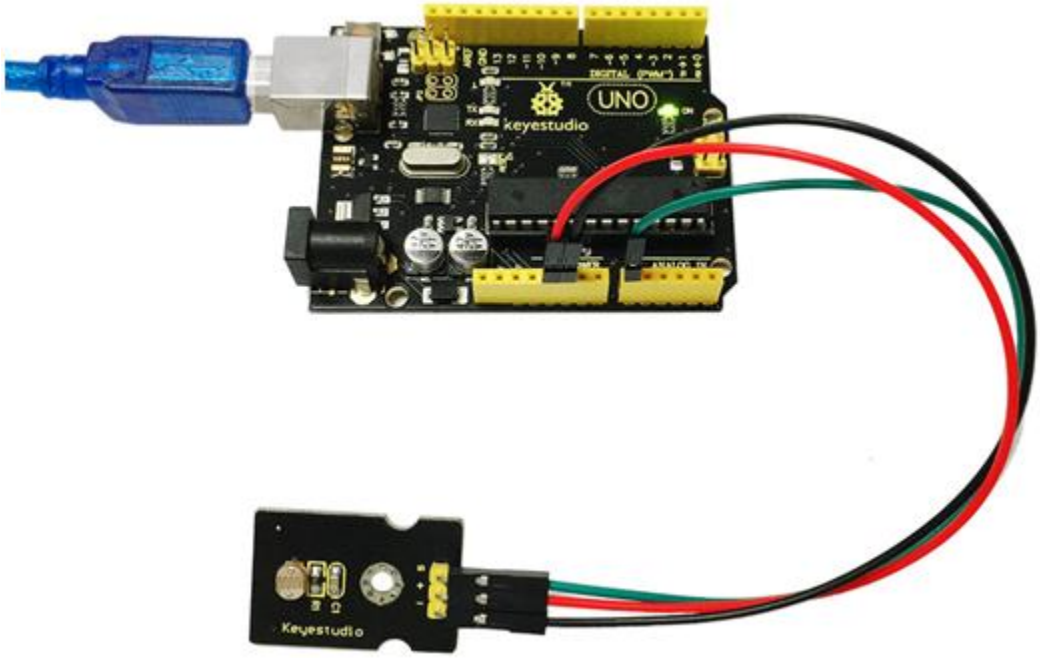
6.3 Connection diagram



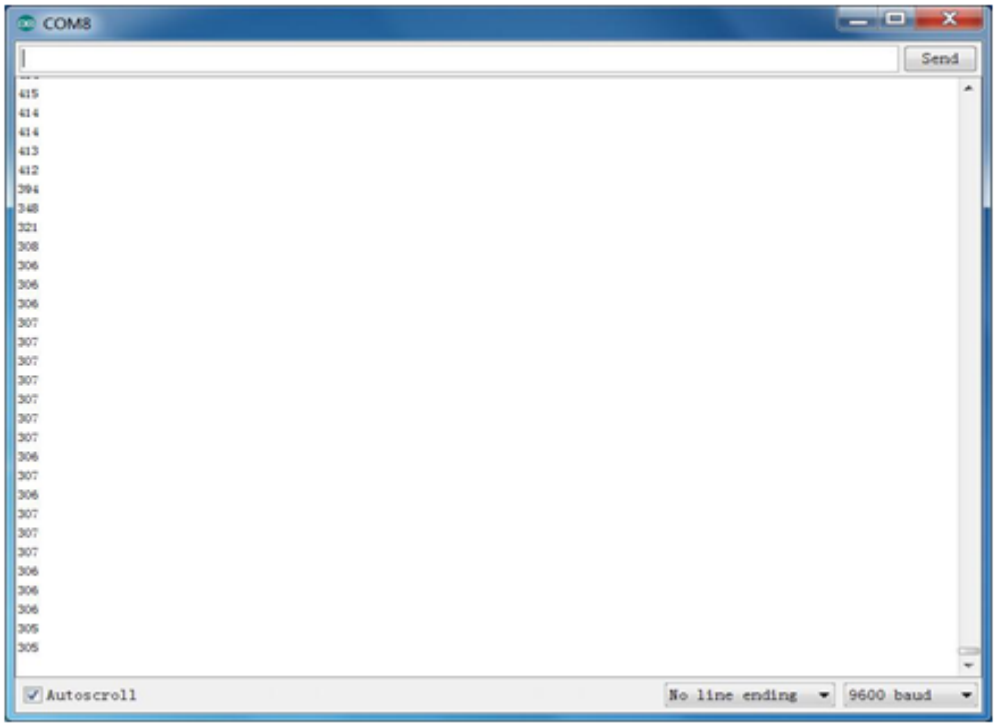
6.4 Sample Code

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

Serial.println(value, DEC);

delay(50);
}
```

6.5 Results



When the wiring and power is complete, the code is uploaded, then the serial display opens. If the photocell is covered in the sensor by hand, it will see the proportion value decrease. It is shown as below.



CHAPTER EIGHT

PROJECT 8: ANALOG ROTATION SENSOR

8.1 Introduction

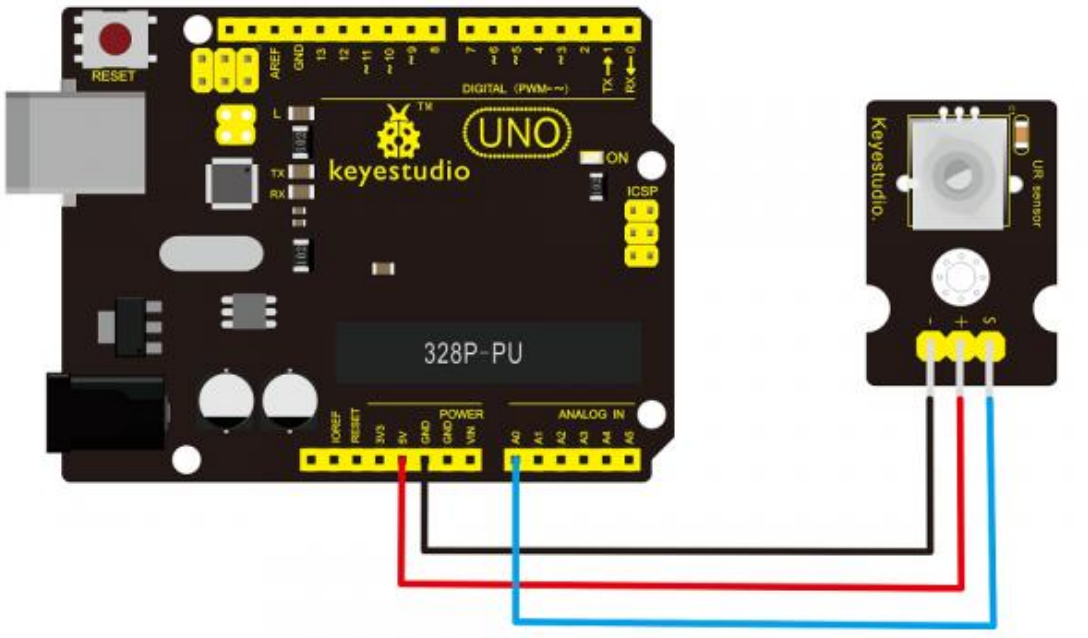
This analog rotation sensor is compatible with arduino. It is based on a potentiometer. Its voltage can be subdivided into 1024. It is easy to connect to the Arduino with the sensor shield. In combination with other sensors, it can do interesting projects by reading the analog value from the IO port.

8.2 Specification

Supply voltage: 3.3V to 5V

Interface: Analog

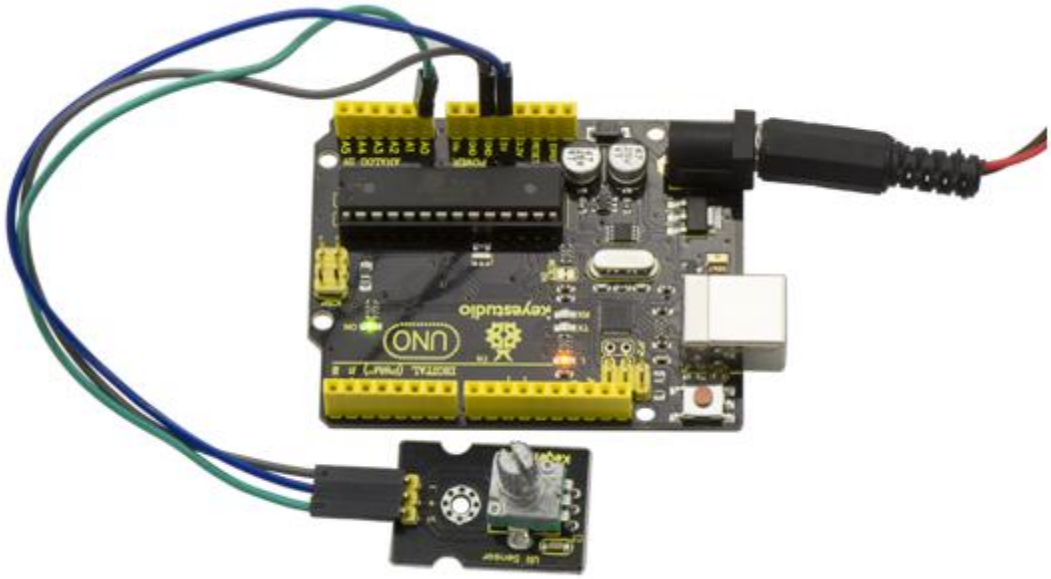
8.3 Connection diagram



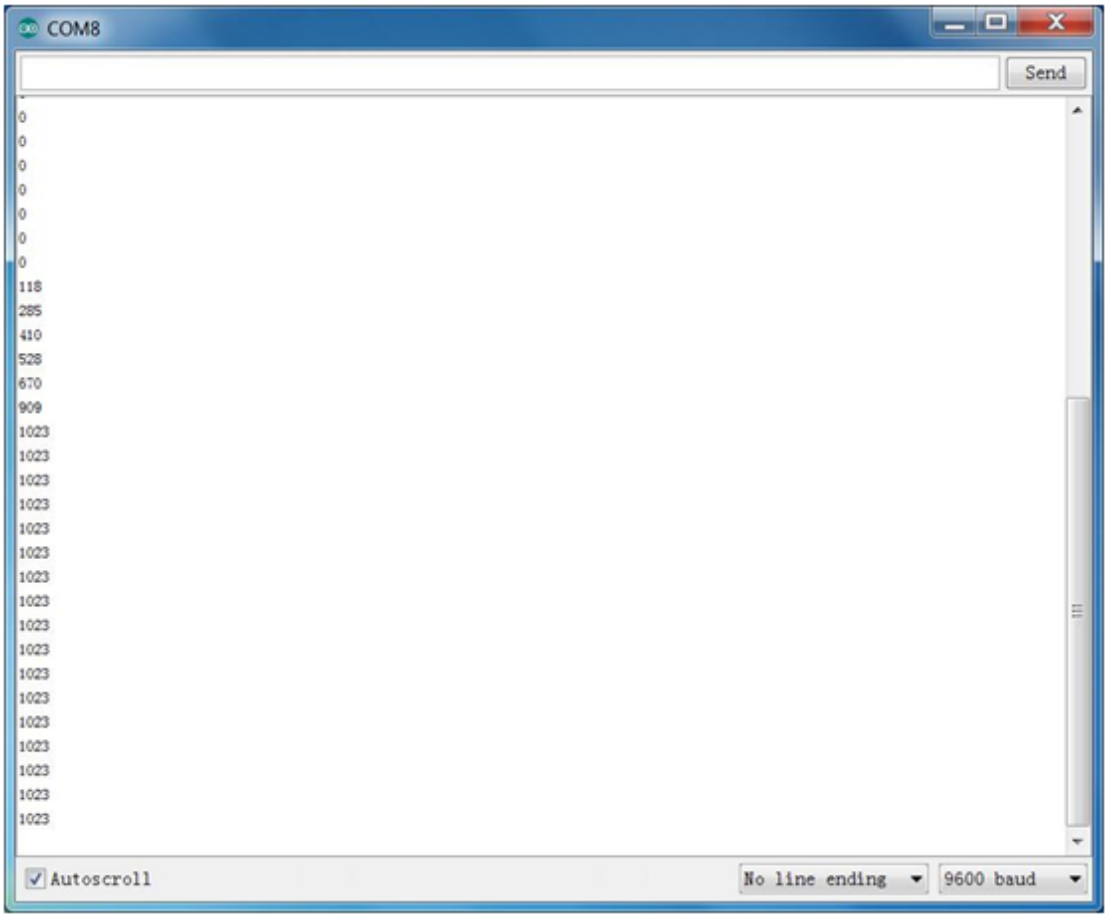
8.4Sample Code

```
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);
}
```

8.5 Result



After the wiring and power is completed, the above code is uploaded. Then the serial screen opens and the baud rate is set to 9600, it will finally see the analog value. If the knob on the rotation sensor is rotated, the value will change between 0-1023 as shown below.



CHAPTER NINE

PROJECT 9: PASSIVE BUZZER

9.1 Introduction

Arduino can be used to make many interactive projects of which the most commonly used is the acoustic-visual display. The circuit in this experiment can produce sound. Normally, the experiment is done with a buzzer or speaker, but the buzzer is simpler and easier to use.

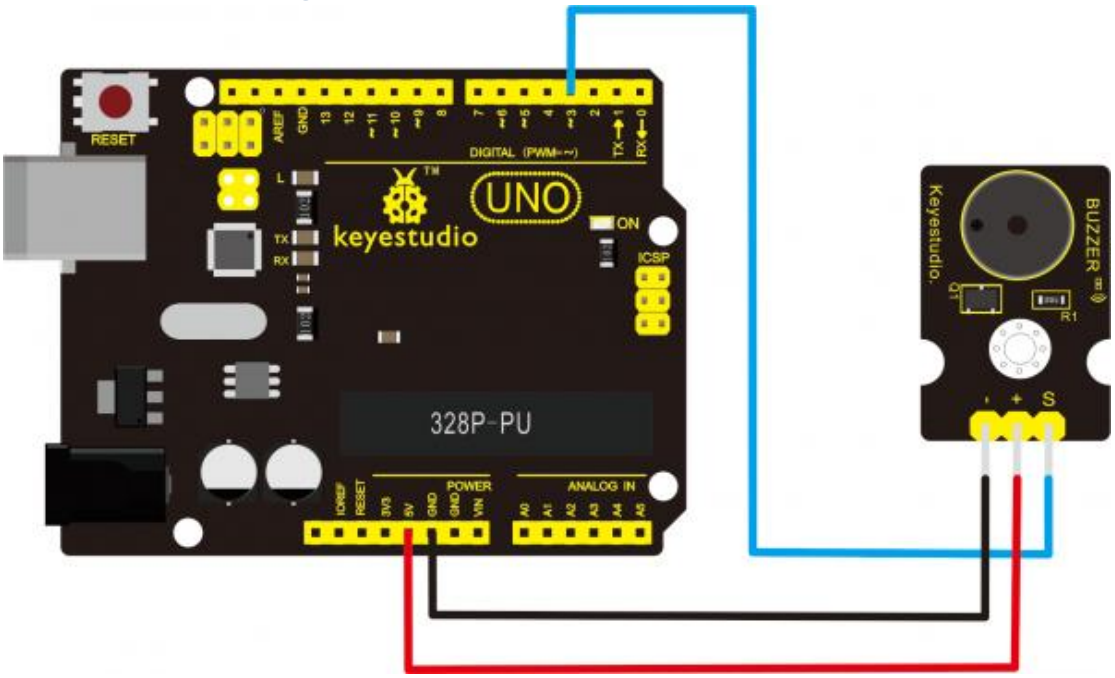
The buzzer shown here is a passive buzzer. It cannot be activated by itself, but by external pulse frequencies. Different frequencies produce different sounds. It can use Arduino to code the melody of a song, which is quite fun and simple.

9.2 Specification

Working Voltage: 3.3-5V

Interface Type: Digital

9.3 Connection diagram



9.4 Sample Code

```
int buzzer=3;//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
}
}
}
```

9.5 Test Result

After receiving the program, the buzzer experiment has been completed. The chime of the buzzer should be heard.

CHAPTER TEN

PROJECT 10: DIGITAL BUZZER

10.1 Introduction

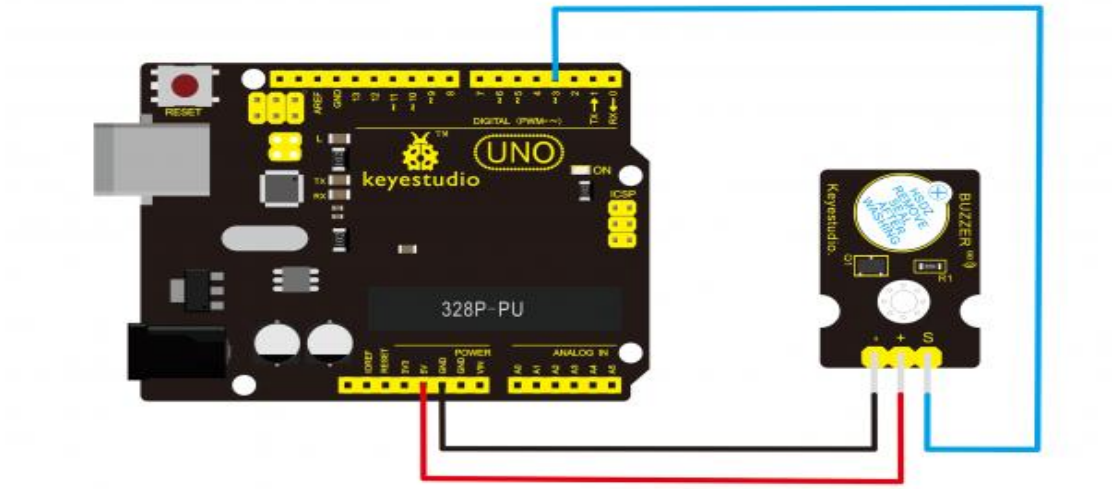
Here is the simplest audio production unit. Can be used high/low By changing the buzzing frequency, it can produce a different sound. This unit is widely used in everyday appliances, such as computer, refrigerator, phones, etc. Also, this small but useful unit can create many interesting interactive projects as long as it is tested. The user will find the electronic sound that creates something exciting.

10.2 Specification

Working Voltage: 3.3-5V

Interface Type: Digital

10.3 Connection diagram



10.4 The 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);
}
```

10.5 Test Result

After uploading the code, the user can hear the buzzer continuously.

CHAPTER ELEVEN

PROJECT 11: DIGITAL BUTTON

11.1 Introduction

This is a basic button app section. The momentary button switch usually stays on. When pressed down, the circuit is connected. When released, it will return to the disconnected state. The unit has three pins for easy connection. The user can simply connect it to an IO shield, with the aim of implementing the first Arduino test.

11.2 Details

Interface: Digital

Supply voltage: 3.3V to 5V

Easy Connection and Operation

High-quality large button keyboard and button cap

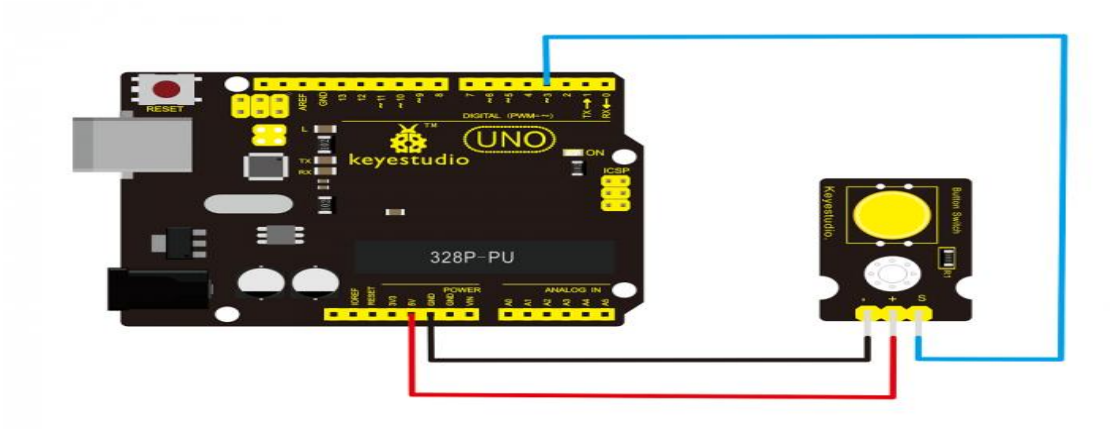
Standard Assembly Structure

Easily recognizable pins

Icons clearly illustrate the operation of the sensor

Achieving Interactive Projects

11.3 Connection diagram

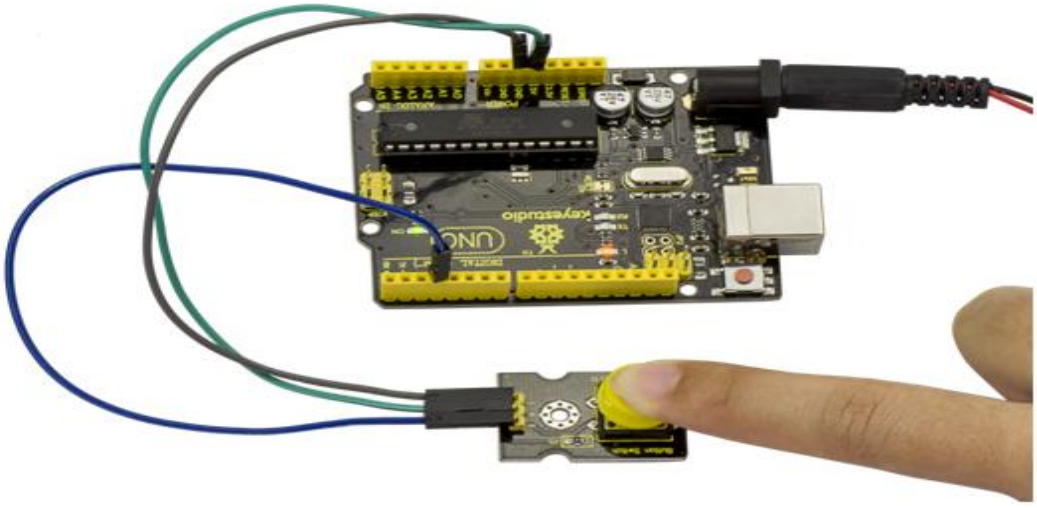


11.4 Sample Code

```
/* # 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
  }
}
```

11.5 Result

When the digital button is pressed, the Led 13 on the UNO board will be on. When he releases the button, the led is off. It is displayed as the image below shows.


keyestudio
UNO

CHAPTER TWELVE

PROJECT 12: DIGITAL TILT SENSOR

12.1 Introduction

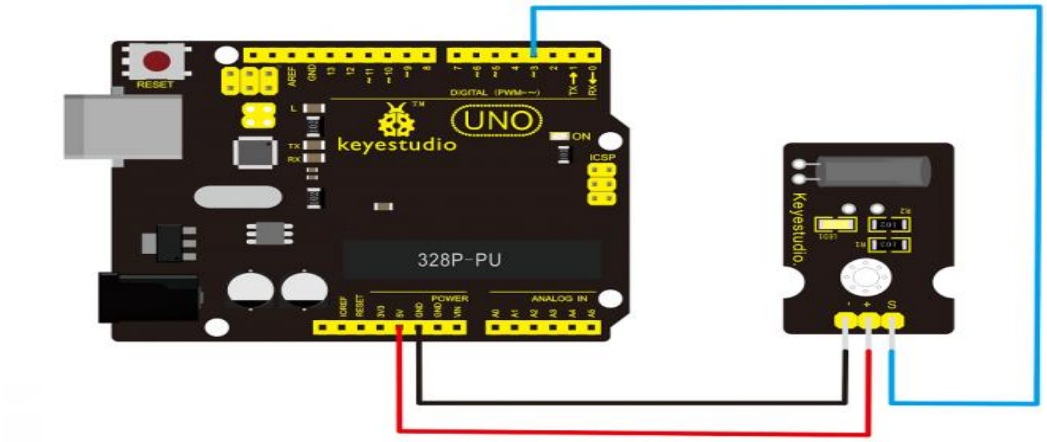
The tilt sensor is a digital tilt switch. It can be used as a simple tilt sensor. Tilt sensors (tilt ball switch) allow orientation or tilt to be detected. They are small, inexpensive, low-consumption and easy to use. It simply connects the sensor to the IO/Sensor shield so it can do amazing interactive projects.

12.2 Specification

Supply voltage: 3.3V to 5V

Interface: Digital

12.3 Connection diagram

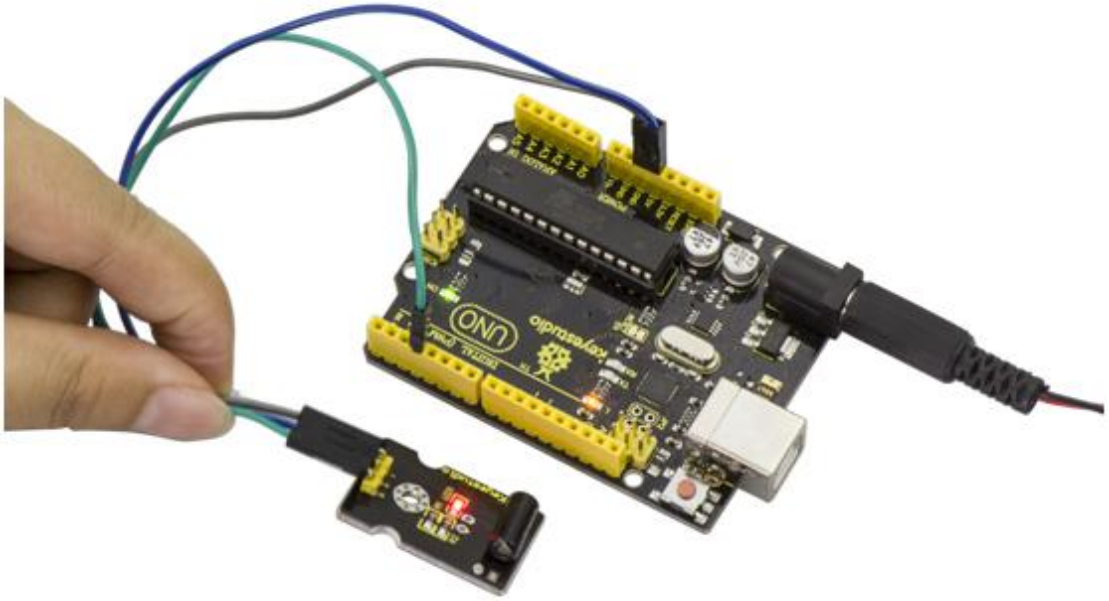


12.4 Sample Code

```
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);
}
}
```

12.5 Result

The user uploads the code to the board. Then, tilt the sensor, and it will see that the led on the sensor is on. It is shown as below.


UNO

CHAPTER THIRTEEN

PROJECT 13: PHOTO SWITCH

13.1 Introduction

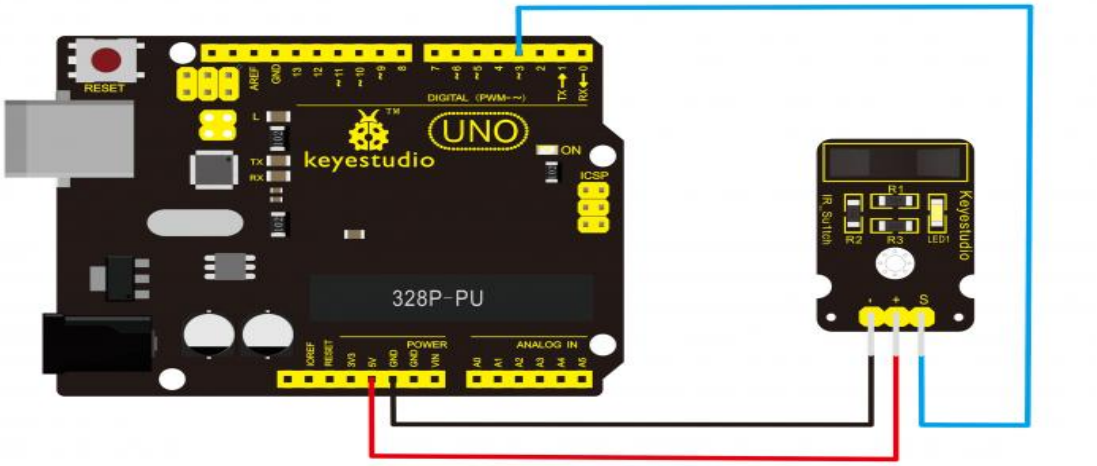
The upright part of this sensor is an infrared transmitter, and on the other hand, it is an armored infrared detector. By emitting a beam of infrared light from one end to the other, the sensor can detect an object when it passes through the beam. It is used for many applications, including optical switch terminals, pellet dispensing, general object detection, etc.

13.2 Specification

Supply voltage: 3.3V to 5V

Interface: Digital

13.3 Connection diagram

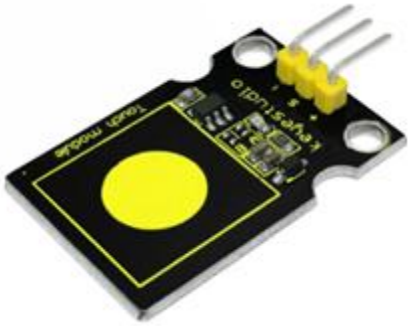


13.4 Sample Code

```
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);
  }
}
```

CHAPTER FOURTEEN

PROJECT 14: CAPACITIVE TOUCH



14.1 Introduction

In case the user has chosen to click on the engineer button, they test the capacitive touch sensor. Touch sensors may be found that are primarily used in electronic devices. The project can be upgraded to a more modern one using the new touch sensor version and makes it more modern.

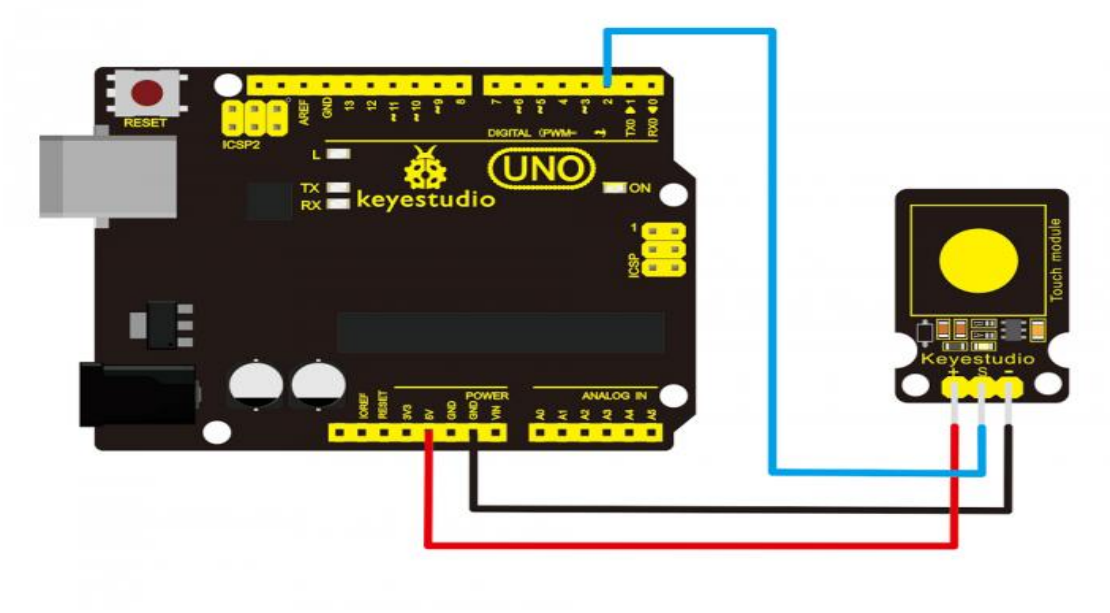
This small sensor can "feel" people and metal touching and feeding back a high/low voltage level. Even isolated with some fabric and paper, it can still feel the touch. Its sensitivity decreases as the isolation layer becomes thicker.

14.2 Specification

Supply voltage: 3.3V to 5V

Interface: Digital

14.3 Connection diagram

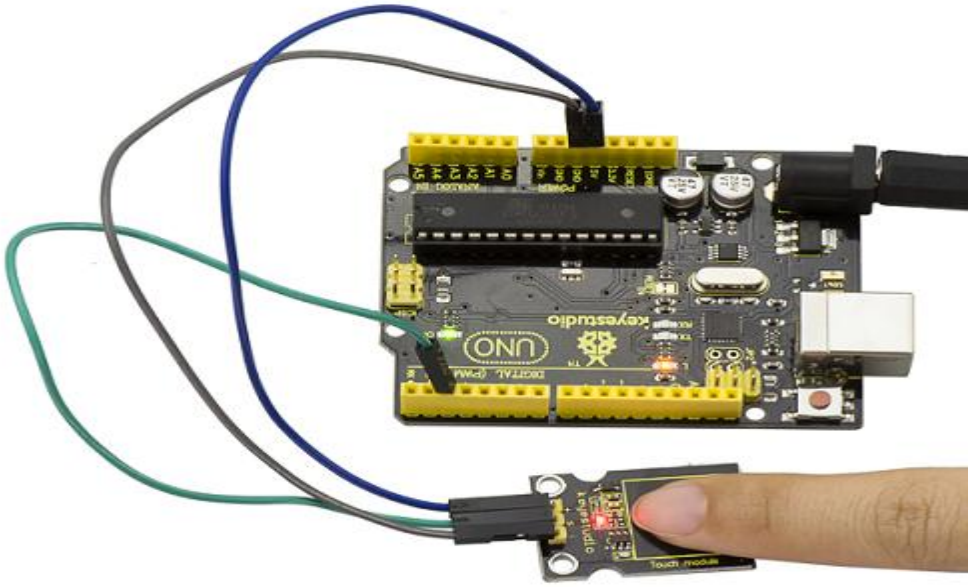


14.4 Sample Code

```
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
     }
}
```

14.5 Result

When the wiring and power is complete, it is necessary for the user to transform the code well, and then, touch the sensor with the finger. The two D2 lights on the sensor and the D13 indicator on the UNO board are on. Otherwise, these two indicators are disabled.


keyestudio
UNO
Touch module

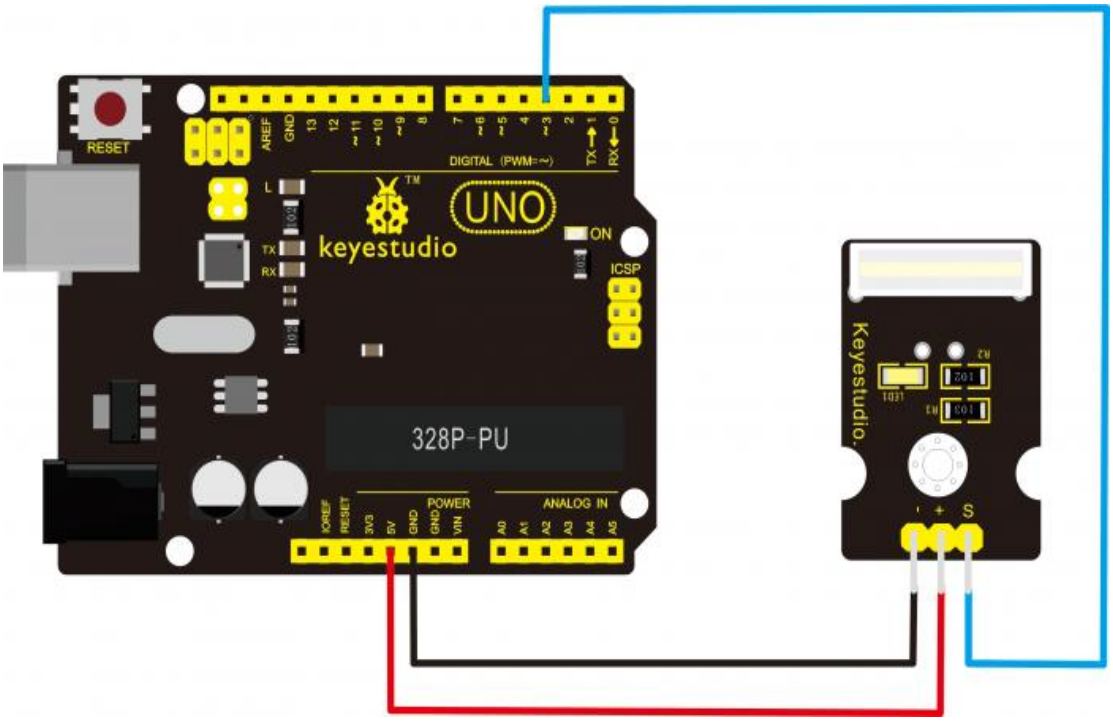
CHAPTER FIFTEEN

PROJECT 15: Impact unit

15.1 Introduction

This is a case sensor unit. When hit, it can send an instant signal. It can be combined with Arduino to do some interesting experiment, e.g. electronic drum.

15.2 Connection diagram


RESET
AREF
GND
DIGITAL (PWM=~)
TX
RX
UNO
keyestudio
ON
ICSP
328P-PU
POWER
ANALOG IN
IOREF
RESET
3V3
5V
GND
GND
VIN
A0
A1
A2
A3
A4
A5
Keyestudio
S

15.3 Sample Code

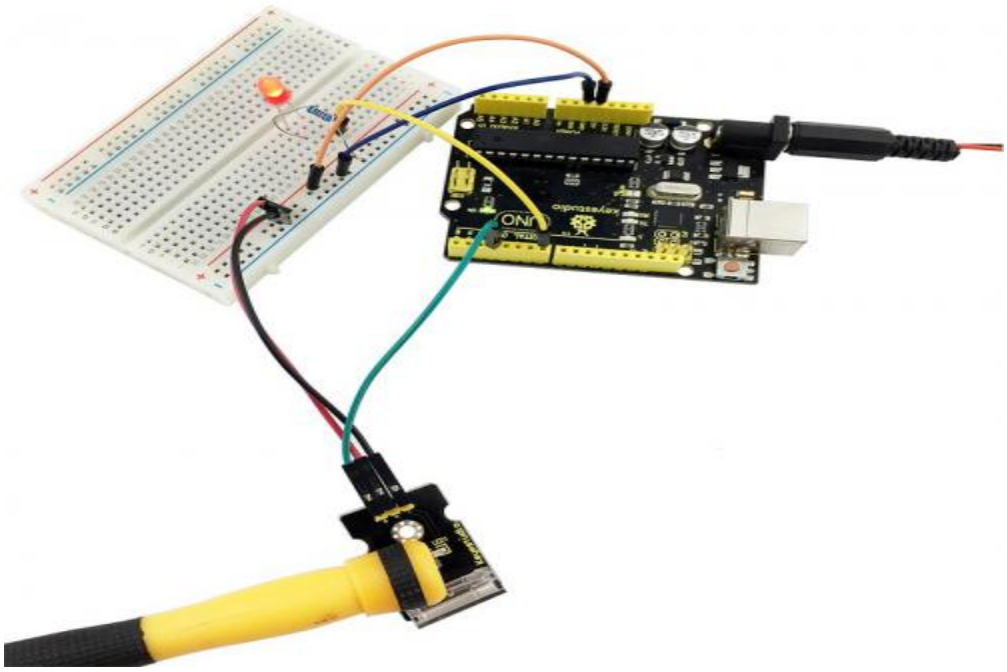
```
int Led=13;//define LED interface
int Shock=3;//define knock sensor interface
int val;//define digital variable val
void setup()
{
pinMode(Led,OUTPUT);//define LED to be output
interface
pinMode(Shock,INPUT);//define knock sensor to be
output interface
}
void loop()
{
val=digitalRead(Shock);//read 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);
}
}
```

15.4 Result

The code is uploaded to the board. When the sensor detects an impact signal, both the led on the sensor and the led 13 on the UNO board light up.

15.5 Extension

It has the ability to expand to connect an external LED. When it hits the sensor, the external LED will light up. For example:



CHAPTER SIXTEEN

PROJECT 16: MAGNETIC HALL SENSOR

16.1 Introduction

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

16.2 Specification

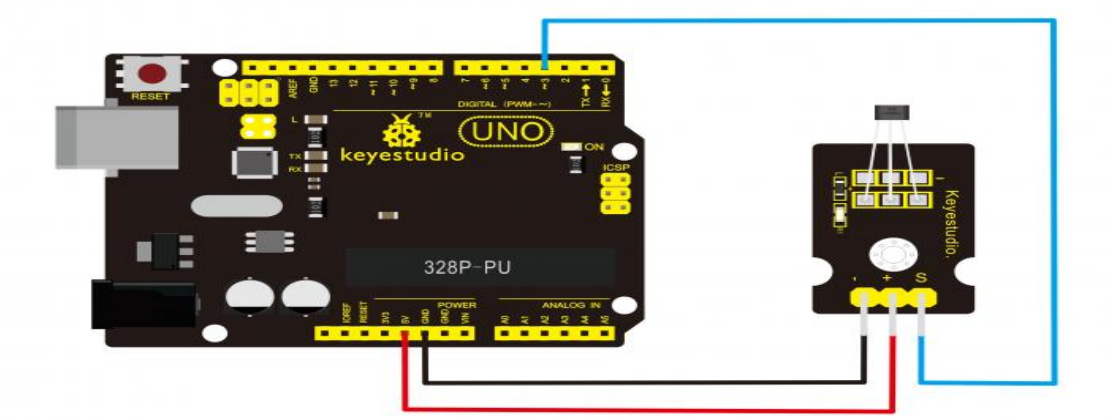
Detection of Magnetic Materials

Detection range: up to 3 cm

Output: digital on/off

The detection range and the strength of the magnetic field are proportional

16.3 Connection diagram

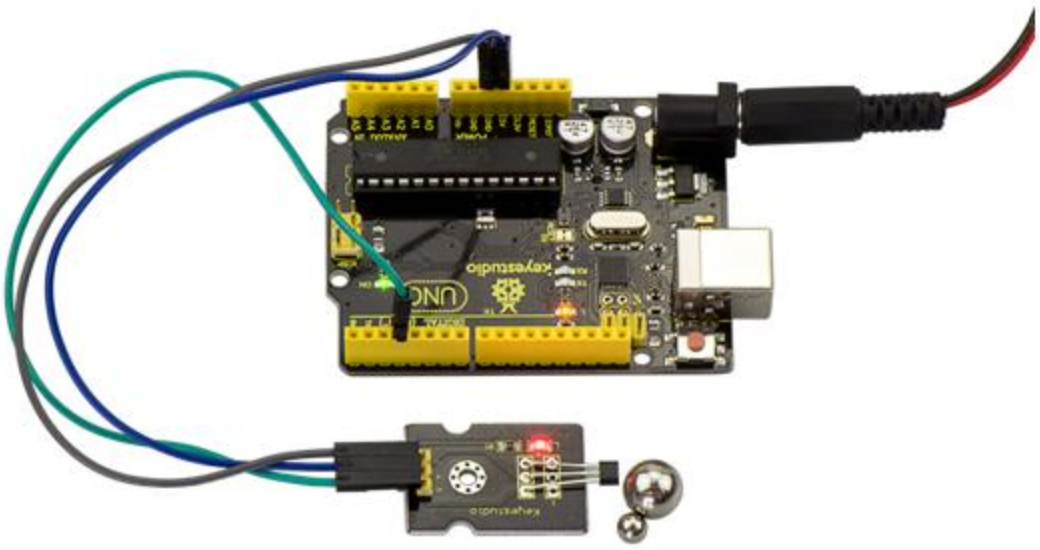


16.4 Sample Code

```
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 push
button 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
  }
}
```

16.5 Result

When the code is connected and uploaded to the boarding then it will see that the D13 indicator on the UNO board is off and the LED on the module is also off. But if the user puts a magnetic ball near the hall module, they will see that the D13 indicator on the UNO board is turned on, and the led on the module is also on.



CHAPTER SEVENTEEN

PROJECT 17: LINE TRACKING SENSOR

17.1 Introduction

This line tracking sensor can detect white lines in black and black lines in white. The single-line tracking signal provides a stable TTL output signal for a more accurate and more stable line.

Multi-channel selection can be easily achieved by installing the required line monitoring robot sensors.

17.2 Specification

Power supply: +5V

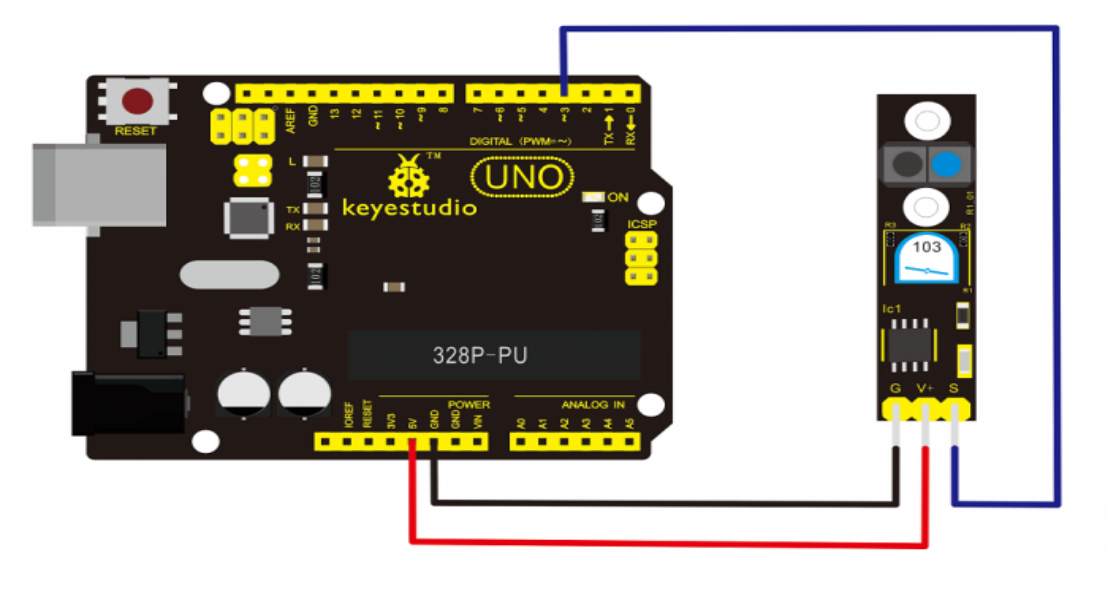
Operating Current: <10mA

Operating Temperature Range: 0°C~+50°C

Output interface: 3-PIN (1 - signal; 2 - power; 3 - power supply negative)

Output Level: TTL Level

17.3 Connection diagram



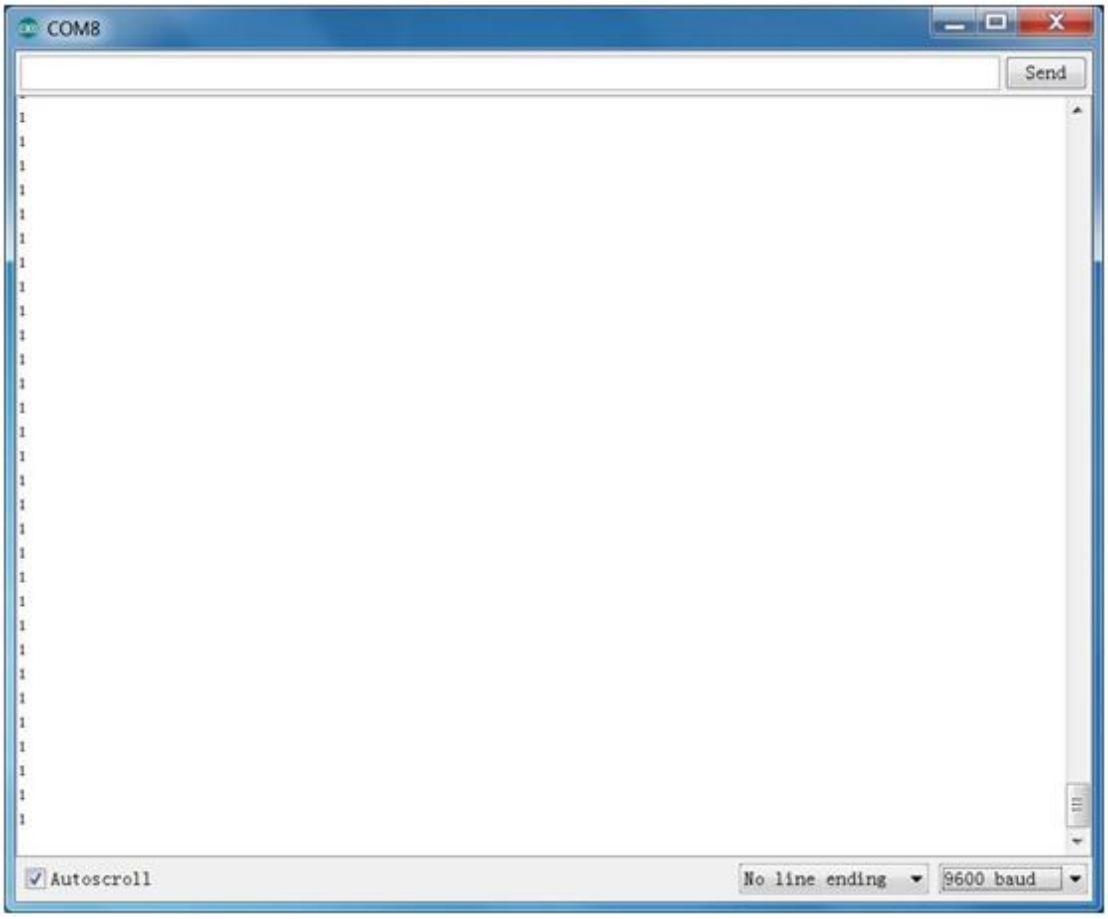
17.4 Sample Code

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

}

17.5 Result

When the code is finished uploading to boarding, the serial screen opens and sets the baud rate to 9600. Then he can see the data from the sensor as shown below.



CHAPTER EIGHTEEN

PROJECT 18: INFRARED OBSTACLE AVOIDANCE

18.1Introduction

The infrared obstacle avoidance sensor is equipped with a distance adjustment function and is specially designed for wheeled robots.

This sensor has strong adaptability to ambient light and is highly accurate. It has a pair of infrared emitting and receiving tubes. When the infrared beam ejected from the emitting tube encounters an obstacle (its reflector), the infrared beam is reflected in the receiving tube, and the indicator will illuminate. The signal output interface outputs digital signal.

It can adjust the detection distance through the potentiometer button (active distance: 2~40cm, Working voltage: 3.3V-5V).

Thanks to a wide voltage range, this sensor can operate stably even under fluctuating supply voltage and is suitable for the use of various microcontrollers, Arduino controllers, and BS2 controllers. A robot mounted with the sensor can sense changes in the environment.

18.2Specification

Working Voltage: DC 3.3V-5V

Operating Current: ≥20mA

Operating temperature: -10°C to -50°C

Detection distance: 2-40cm

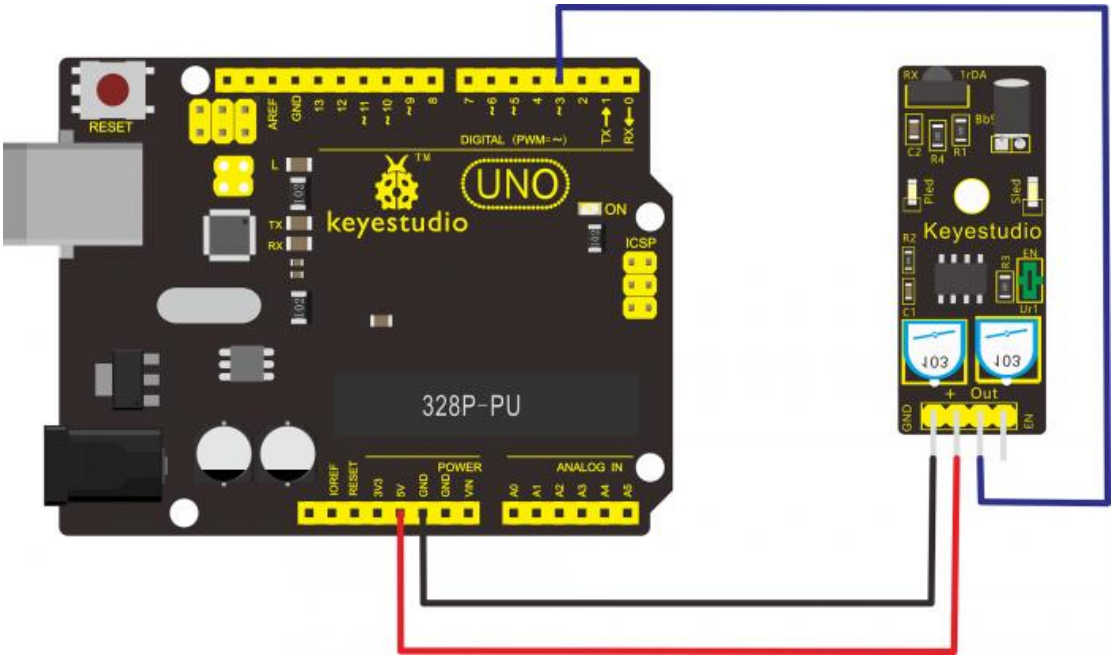
IO Interface: 4 PIN (-/+/S/EN)

Output Signal: TTL Voltage

Accommodation Mode: Multi-Cycle Resistor Adjustment

Effective Angle: 35°

18.3Connection diagram



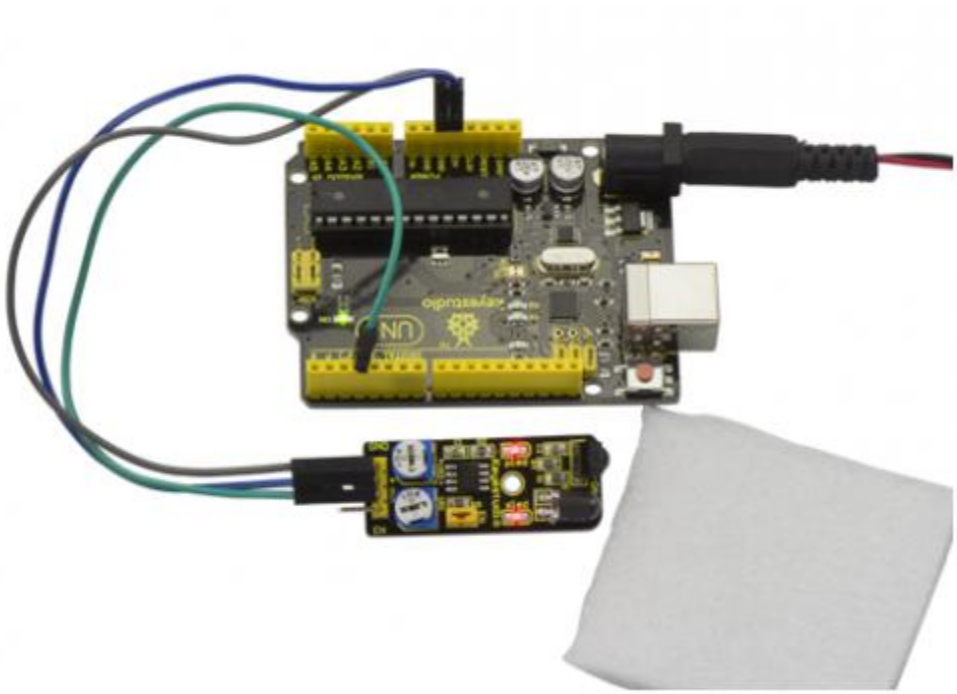
18.4 Sample Code

```
const int sensorPin = 3;     // 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);
  }
}
```

18.5 Result

Finishing uploading the code on boarding, you can see the led on the UNO board. The obstacle detector sensor is enabled.

If a block of foam is put in front of the sensor, this time when the sensor detects the obstacle, the sensor will be activated.



CHAPTER NINETEEN

PROJECT 19: PIR MOTION SENSOR

19.1 Introduction

The pyroelectric infrared motion sensor can detect infrared signals from a moving person or moving animal, and output switching signals. It can be applied in various occasions to detect the movement of the human body. Conventional infrared pyroelectric sensors require a body infrared pyroelectric detector, professional chip, complex peripheral circuit, so the size is larger, with complex circuitry and lower reliability. This new pyroelectric infrared motion sensor is launched specially designed for Arduino. It uses built-in digital infrared pyroelectric sensor, has a smaller size, higher reliability, lower power consumption, and simpler peripheral circuitry.

19.2 Specification

Input Voltage: 3.3~5V, 6V Max

Operating Current: 15uA

Operating Temperature: -20~85°C

Output Voltage: High 3V, Low 0V

Output delay time (high level): Approximately 2.3 to 3 seconds

Detection Angle: 100°

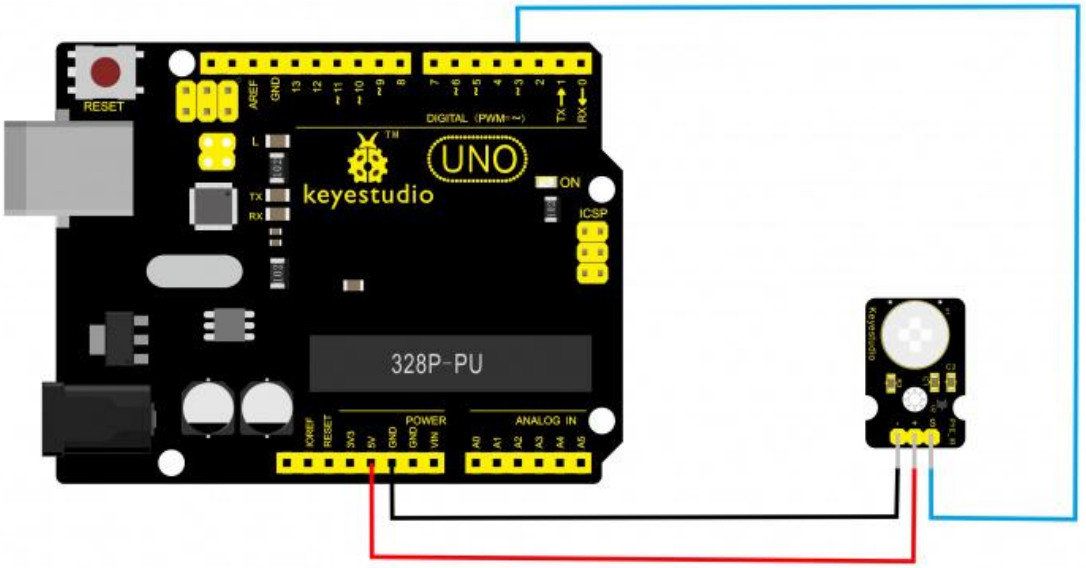
Detection distance: 7 meters

Output indicator light (When the output is HIGH, it will be on)

Pin limit current: 100mA

19.3 Connection diagram

It is important to connect the module's S pin to the UNO board's Digital 3, and connect the negative pin to the GND port and the positive pin to the 5V port.



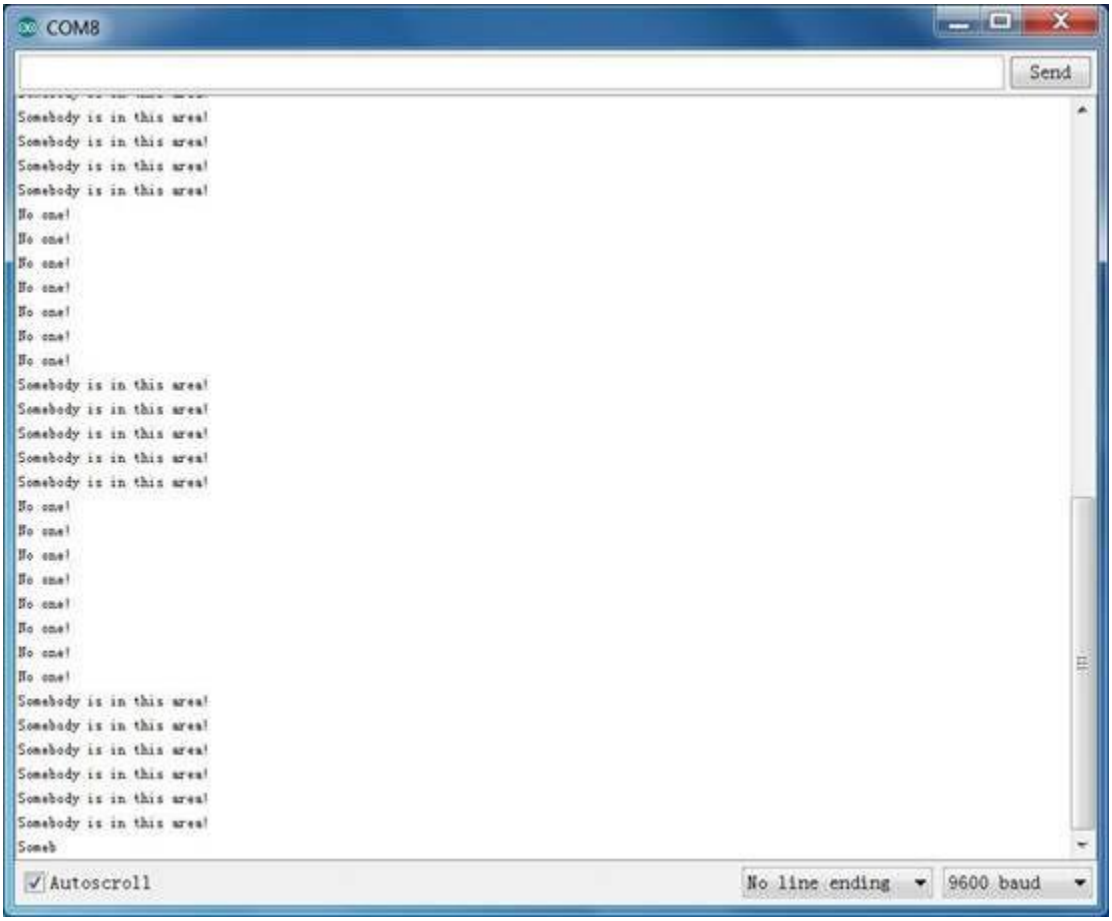
19.4 Sample Code:

```
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);
}
```

19.5 Result

When the wiring is complete, the code is turned on and raised, if the sensor detects someone moving nearby, the D13 indicator on the UNO board will light up and "Someone is in this area!" is displayed on the serial display.

If there is no movement, the D13 indicator on the UNO board does not light up and "None!" appears on the serial display.


COM8
Send
Somebody is in this area!
Somebody is in this area!
Somebody is in this area!
Somebody is in this area!
No one!
No one!
No one!
No one!
No one!
No one!
No one!
Somebody is in this area!
Somebody is in this area!
Somebody is in this area!
Somebody is in this area!
Somebody is in this area!
No one!
No one!
No one!
No one!
No one!
No one!
No one!
No one!
Somebody is in this area!
Somebody is in this area!
Somebody is in this area!
Somebody is in this area!
Somebody is in this area!
Somebody is in this area!
Someb
Autoscroll
No line ending
9600 baud

CHAPTER TWENTY

PROJECT 20: FLAME SENSOR

20.1 Introduction

This flame sensor can be used to detect fire or other lights with wavelength mounts at 760nm~1100nm. In firefighting robot applications, the flame plays an important role in the probe, which can be used as the robot's eyes to find the source of the fire.

20.2 Specification

Supply voltage: 3.3V to 5V

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

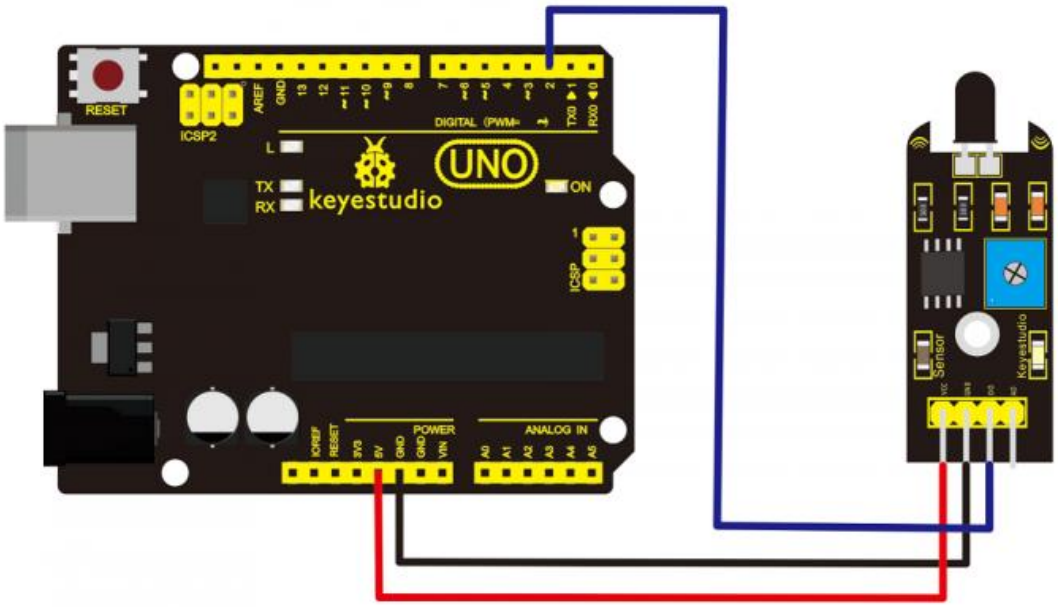
Spectral bandwidth range: 760 nm to 1100 nm

Operating Temperature: -25°C to 85°C

Interface: Digital

20.3 Connection diagram

It connects pin D0 to digital 2, GND pin to GND port, VCC pin to 5V port.

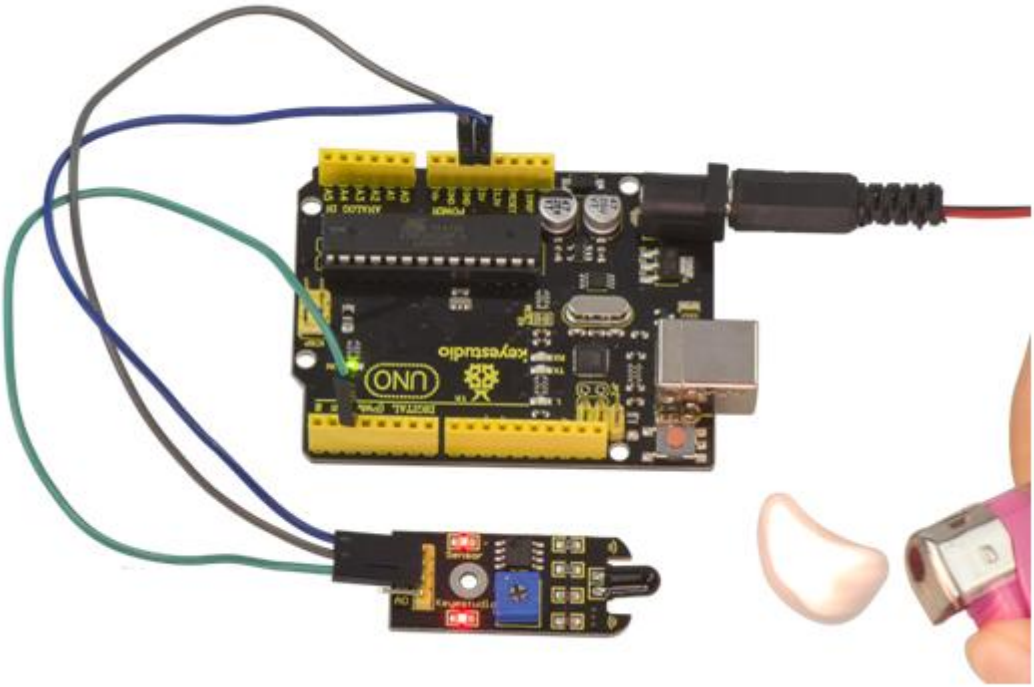


20.4 Sample Code

```
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);
}
```

20.5 Result

The wiring is completed and turned on, it must upload the code well to the board. Then, if it puts a lighter near the sensor, when the sensor detects the flame, it lights up another led on the sensor.


UNO
keyestudio

CHAPTER TWENTY-ONE

PROJECT 21: Vibration sensor

21.1 Introduction

The simplest way to control vibration with Arduino is to use a vibration sensor from keyestudio. It can be connected directly to the sensor's Shield V5, vibrate this sensor, and the Arduino can receive a digital signal, making calculations and programs on the Arduino easier.

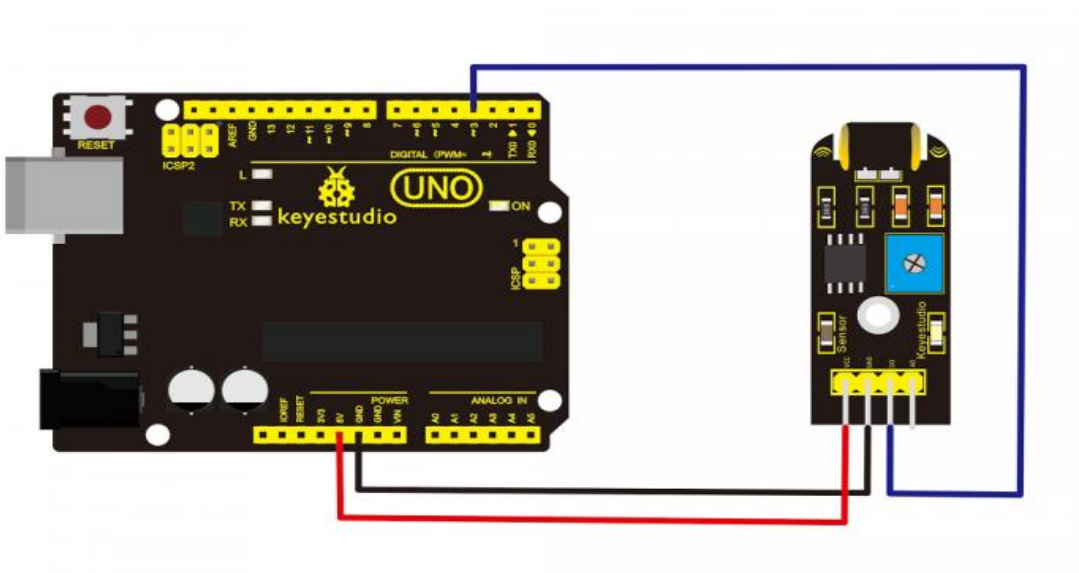
Despite its simplicity, it can be fully utilized with creative thinking, step counting, and collision warning light, etc.

Specification:

IO Type: Digital

Supply voltage: 3.3V to 5V

21.2 Connection diagram



21.3 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()//Interrupts function
{  state++;
}
```

CHAPTER TWENTY-TWO

PROJECT 22: ANALOG GAS SENSOR

22.1 Introduction

This analog gas sensor - MQ2 is used in gas leak detection 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, its sensitivity can be adjusted by the potentiometer.

22.2 Specification

Power Supply: 5V

Interface Type: Analog

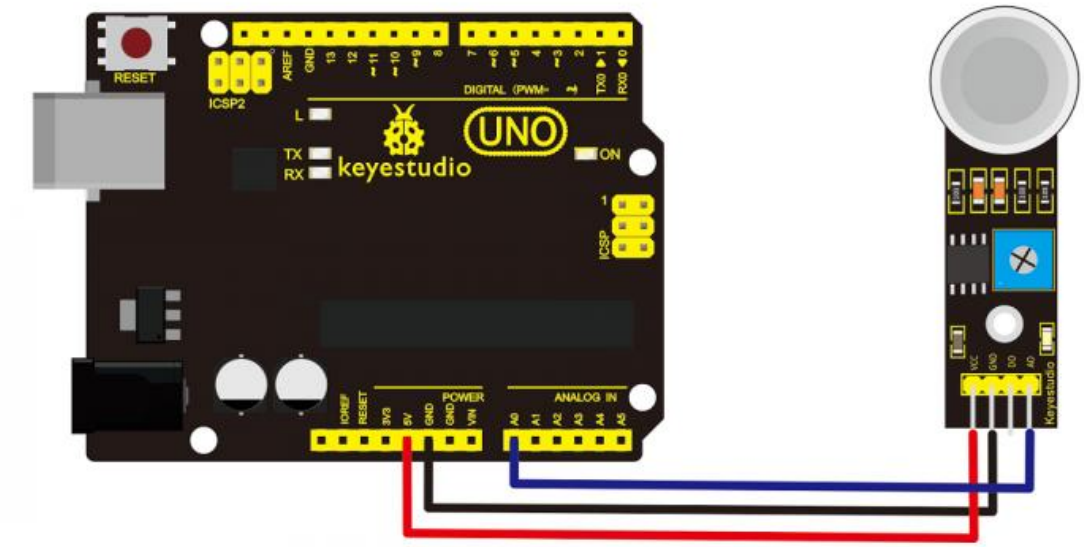
Wide detection range

Fast response and high sensitivity

Simple drive circuit

Stable and long service life

22.3 Connection diagram



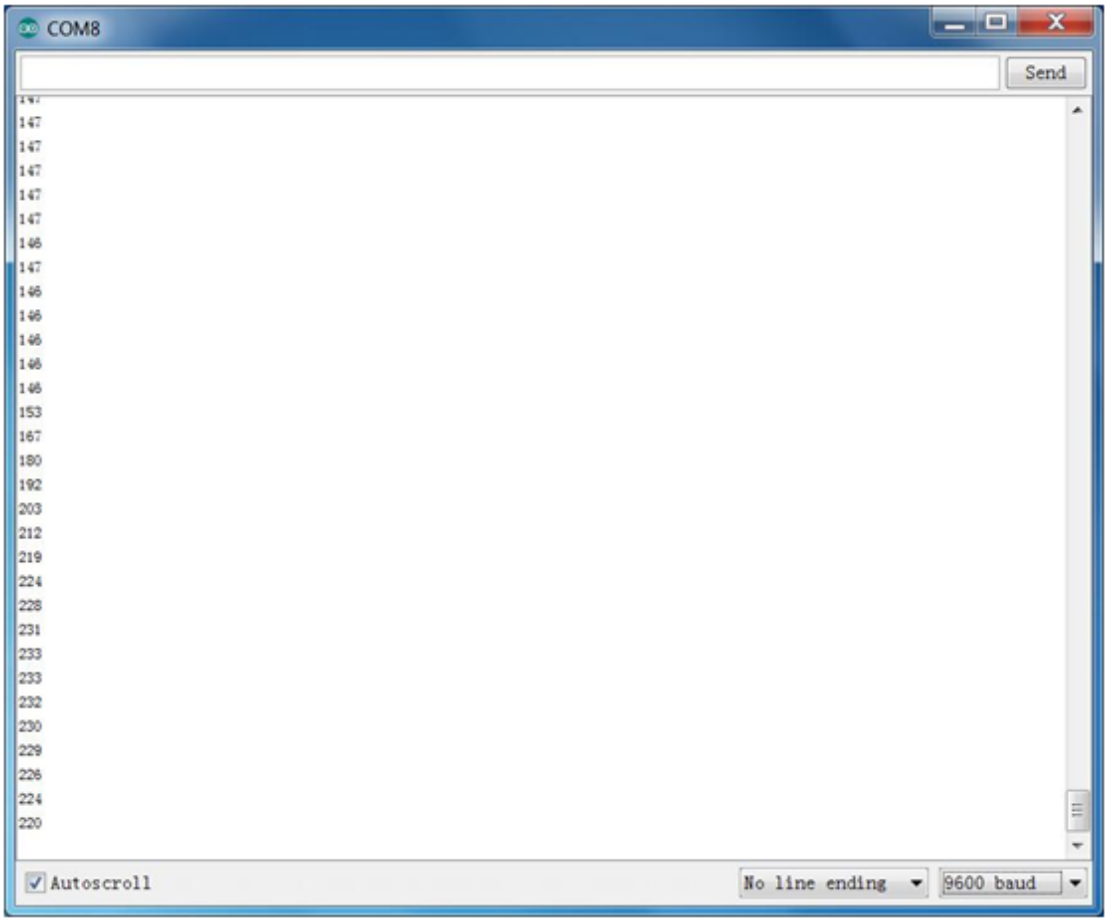
22.4 Sample Code

```
///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);
}
```

22.5 Result

When the wiring and power is complete, the user uploads the code well, then opens the serial display and sets the baud rate as 9600, then he will see the analog value.

When detecting the gas, the value will change.



CHAPTER TWENTY-THREE

PROJECT 23: ANALOGUE ALCOHOL SENSOR

23.1 Introduction

This analog gas sensor - MQ3 is suitable for alcohol detection. It can be used in a breath analyzer. It has good selectivity because it has a higher sensitivity to alcohol and a lower sensitivity to gasoline. The sensitivity can be adjusted by rotating the potentiometer.

23.2 Specification

Power Supply: 5V

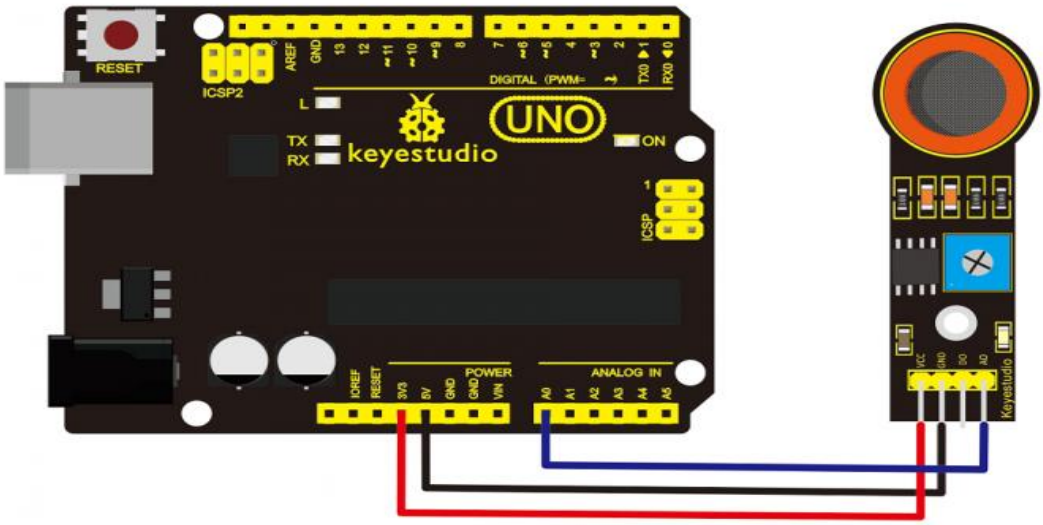
Interface Type: Analog

Fast response and high sensitivity

Simple drive circuit

Stable and long service life

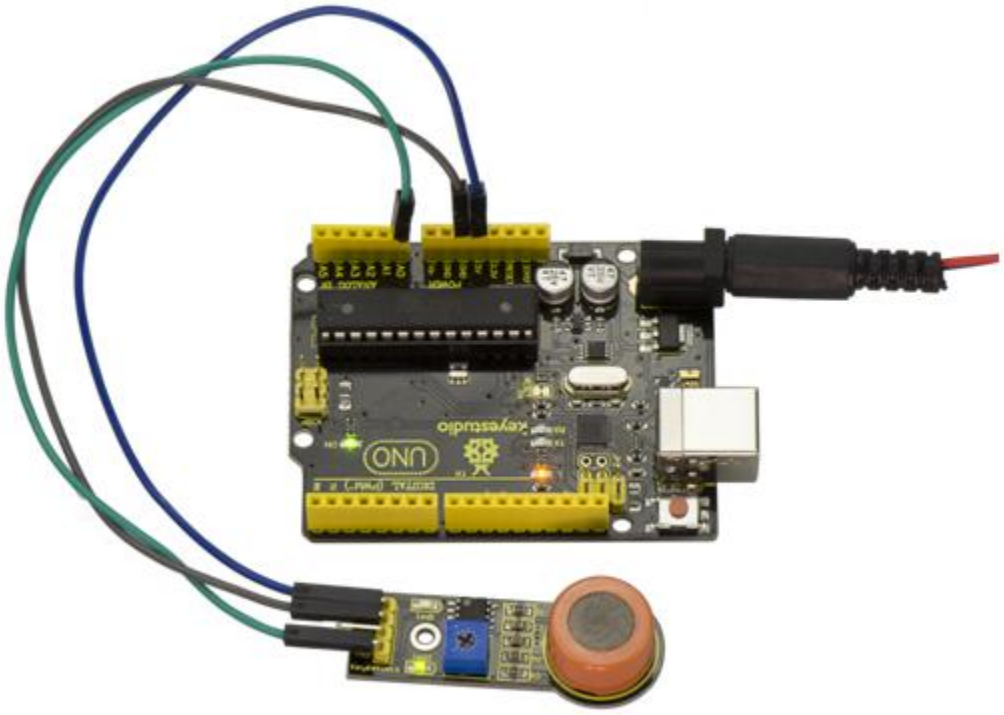
23.3 Connection diagram



23.4 Sample Code

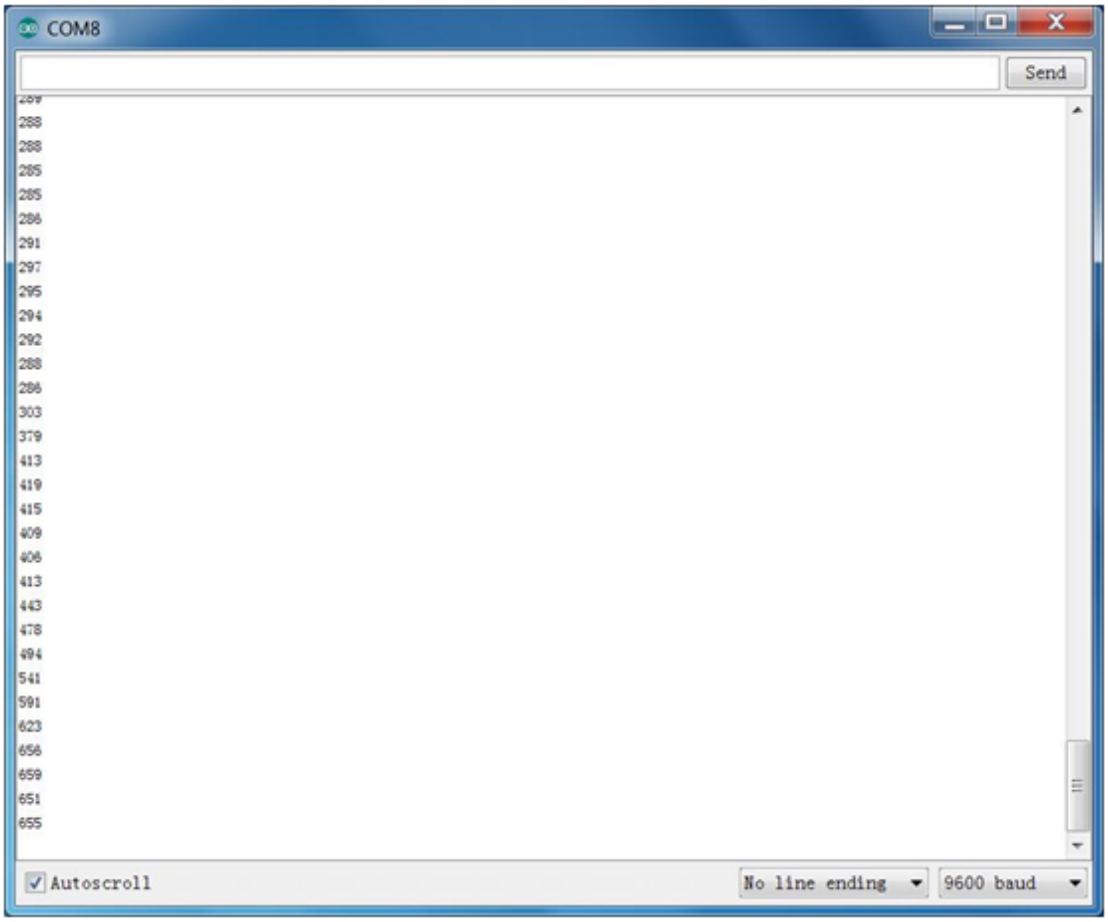
```
///Arduino Sample Code
void setup()
{
  Serial.begin(9600); //Set serial baud rate to
9600 bps
}
void loop()
{
int val;
val=analogRead(0);//Read Gas value from analog 0
Serial.println(val,DEC);//Print the value to serial
port
delay(100);
}
```

23.5 Result



When the wiring and power is complete, the user uploads the code, then opens the serial display and sets the baud rate as 9600 and will see the analog value.

When detecting the alcohol gas, the value will change.


COM8
Send
288
288
285
285
286
291
297
295
294
292
288
286
303
379
413
419
415
409
406
413
443
478
494
541
591
623
656
659
651
655
Autoscroll
No line ending
9600 baud

CHAPTER TWENTY FOURTH

PROJECT 24: Digital infrared transmitter

24.1 Introduction

The IR transmitter module is designed for infrared communication, which is widely used to operate the TV device from a short field of view. Since infrared (IR) remote controls use light, they require eye contact to operate the target device. The signal can, however, be reflected by mirrors, just like any other light source.

If a function is required where eye contact is not possible, for example when controlling equipment in another room or installed in a cabinet, several brands of IR are available for this on the market. Most of them have an IR receiver, which receives the IR signal and transmits it via radio waves to the remote part, which has an infrared transmitter that mimics the original IR control.

Infrared receivers also tend to have a more or less limited operating angle, which depends mainly on the optical characteristics of the phototransistor. However, it is easy to increase the operating angle by using a matte transparent object in front of the receiver.

24.2 Specification

Power Supply: 3-5V

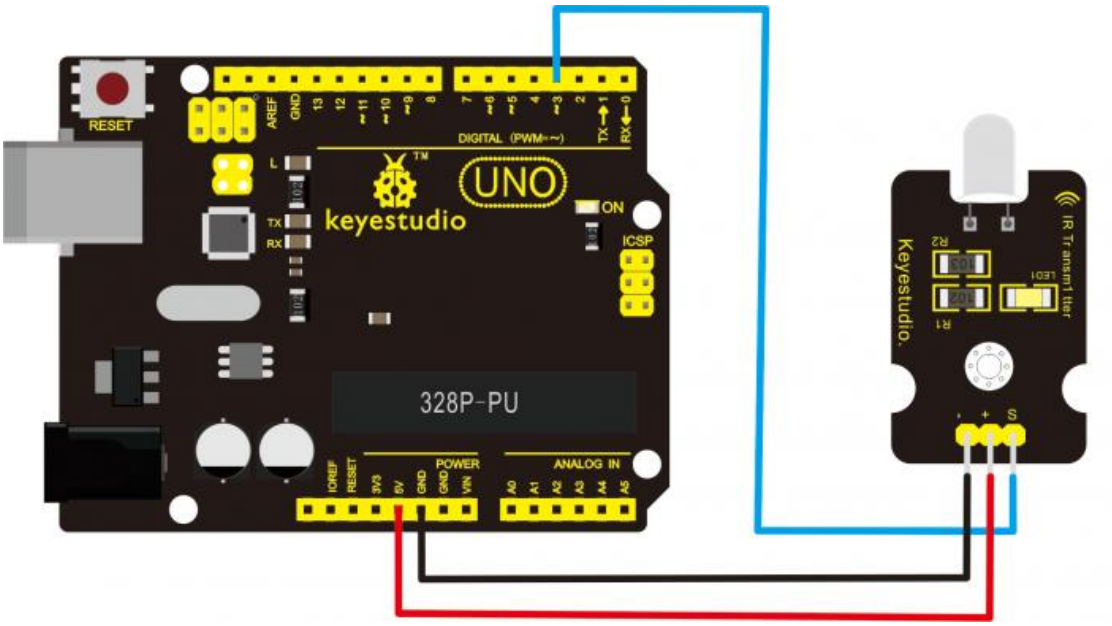
Infrared Center Frequency: 850nm-940nm

Infrared emission angle: about 20 degrees

Infrared emission distance: about 1.3m (5V 38Khz)

Mounting hole: inner diameter is 3.2mm, distance is 15mm

24.3 Connection diagram



24.4 Sample Code 1:

```
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, it will flash blue light on the phone screen when using the camera to capture the infrared LED.

The user needs to upload the above code well to the board, and the led on the sensor will flash a red light.

Next, it will move on to an interactive example between an IR receiver and an IR transmitter module.

24.5 Infrared Remote Control/Communication:

Material List

UNO R3 x2

Digital Infrared Receiver x1

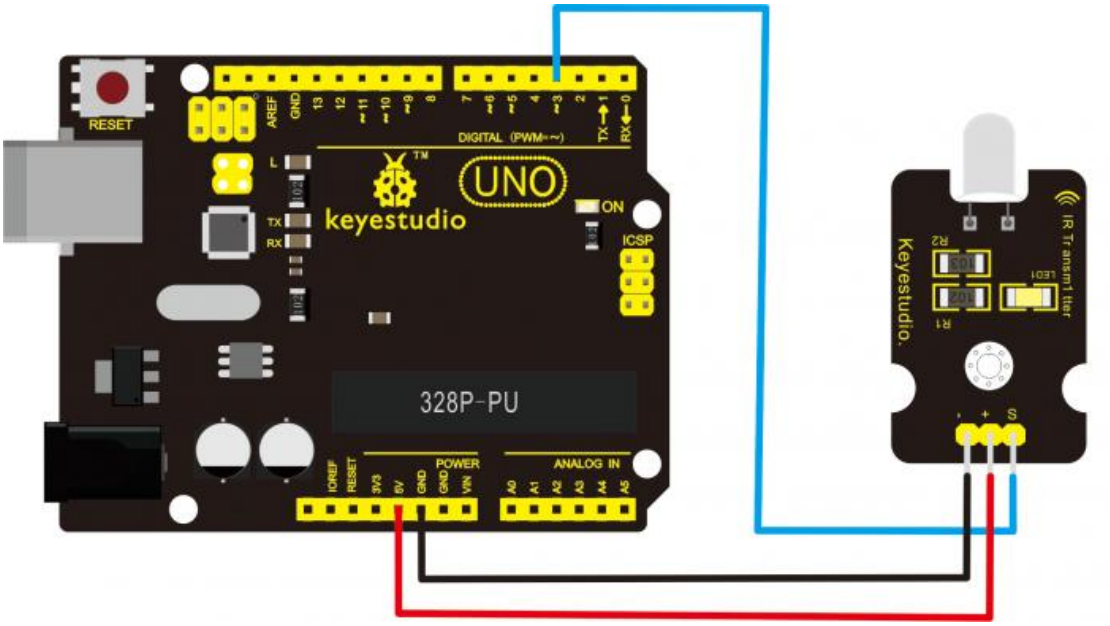
Infrared Transmitter Module x1

It is necessary to acquire an Arduino Arduino-IRremote library and install it.

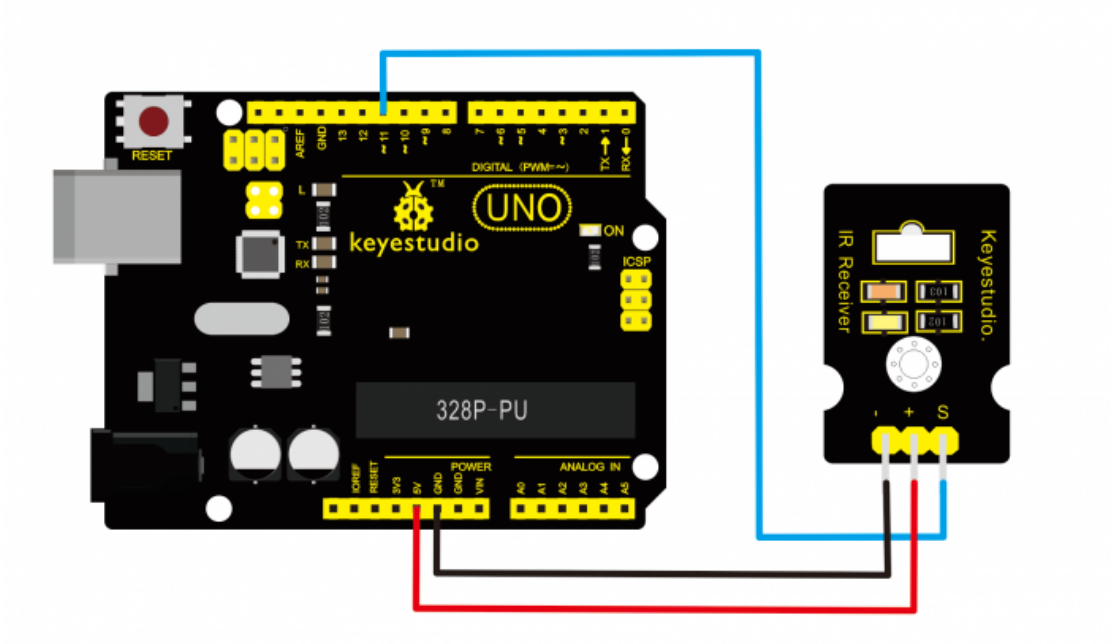
Note: if the user does not have two main boards, he can replace it with the board for connection, which can be easier and more convenient.

24.6 Connection diagram

For IR Transmitter:



For IR receiver: connect to port D11.



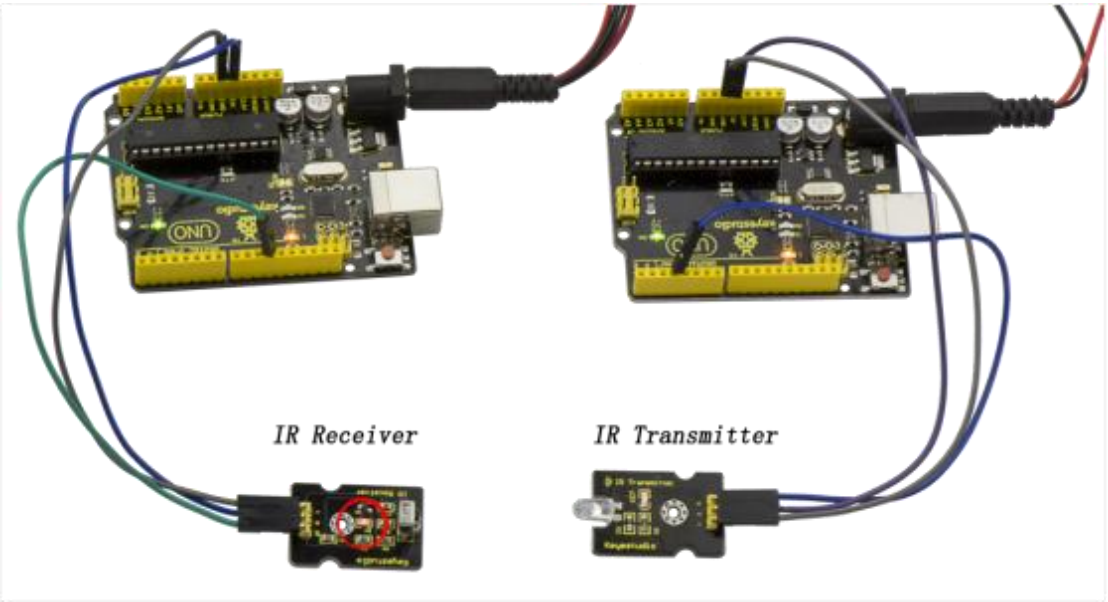
Upload the code 2 to the UNO connected to an infrared transmitter:

```
#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); }
```

```
#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
   }}
```

24.7 Test Result

When the IR Receiver receives the infrared signal from the IR transmitter, the D1 indicator light on the IR Receiver module will flash. It appears as in the image below.



CHAPTER TWENTY-FIVE

PROJECT 25: DIGITAL INFRARED RECEIVER

25.1 Introduction

IR is widely used in remote control. With this IR receiver, the Arduino project can take commands from any IR remote controller if it has the right decoder. It will also be easy for the user to make the IR controller using an IR transmitter.

25.2 Specification

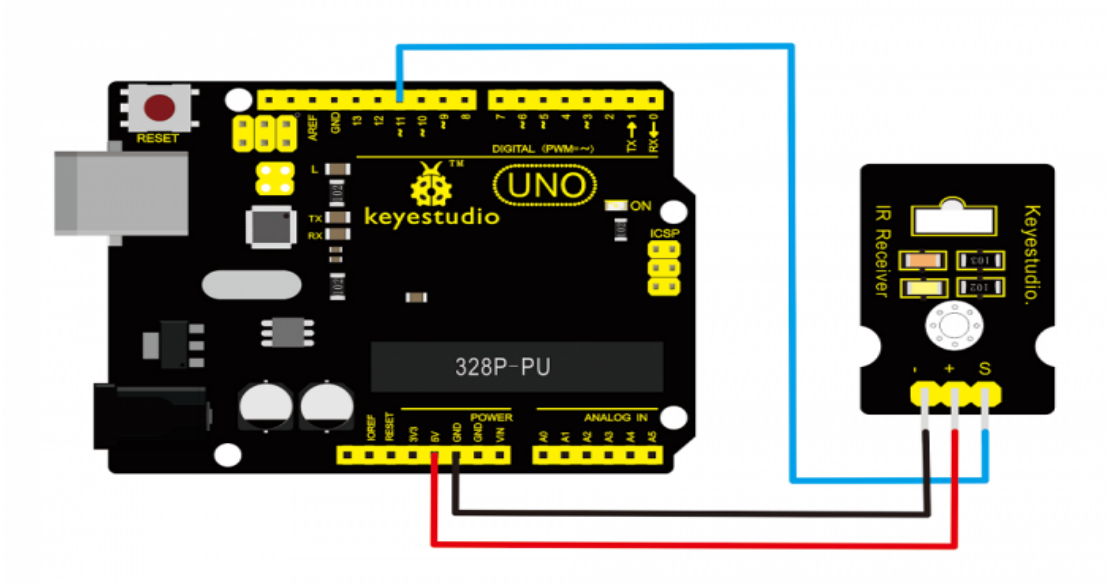
Power Supply: 5V

Interface: Digital

Modulation Frequency: 38Khz

25.3 Connection diagram

The image below shows a recommended connection method. Any digital input/output pin that is not used by another device can be used.



NOTE: In the sample code below digital pin 11 is used, the user can either change the wiring or change the sample code to match.

25.4 Sample Code

Note: before compiling the code, the user needs to remember to place the library in the Arduino IDE library directory. Otherwise, the compilation will fail.

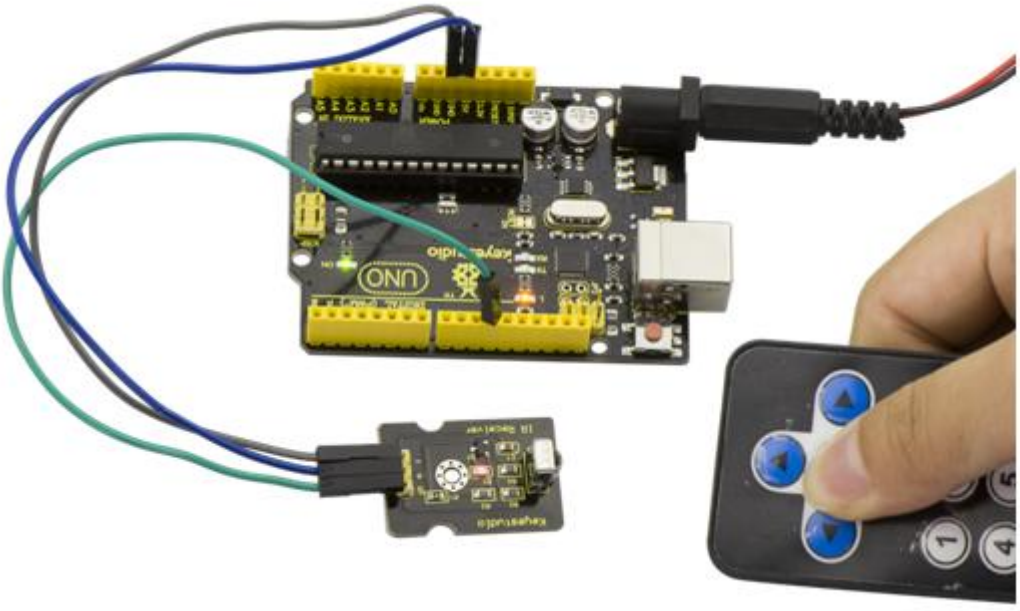
```
#include <IRremote.h>
 int RECV_PIN = 11;
 IRrecv irrecv(RECV_PIN);
 decode_results results;
 void setup()
{
  Serial.begin(9600);
  irrecv.enableIRIn(); // Start the receiver
}
 void loop() {
  if (irrecv.decode(&results)) {
    Serial.println(results.value, HEX);
    irrecv.resume(); // Receive the next value
  }
}
```

The remote IR library includes some sample codes to send and receive:

Remote Infrared Library

25.5 Result

The wiring and sending of the code has been completed, and then the user needs to control the IR receiver module with an IR remote control. D1 will flash and display as below.


UNO

CHAPTER TWENTY-SIX

PROJECT 26: ROTARY ENCODER

26.1 Introduction

The rotary encoder can measure the pulse output times during the rotation process in the positive and reverse direction by rotating.

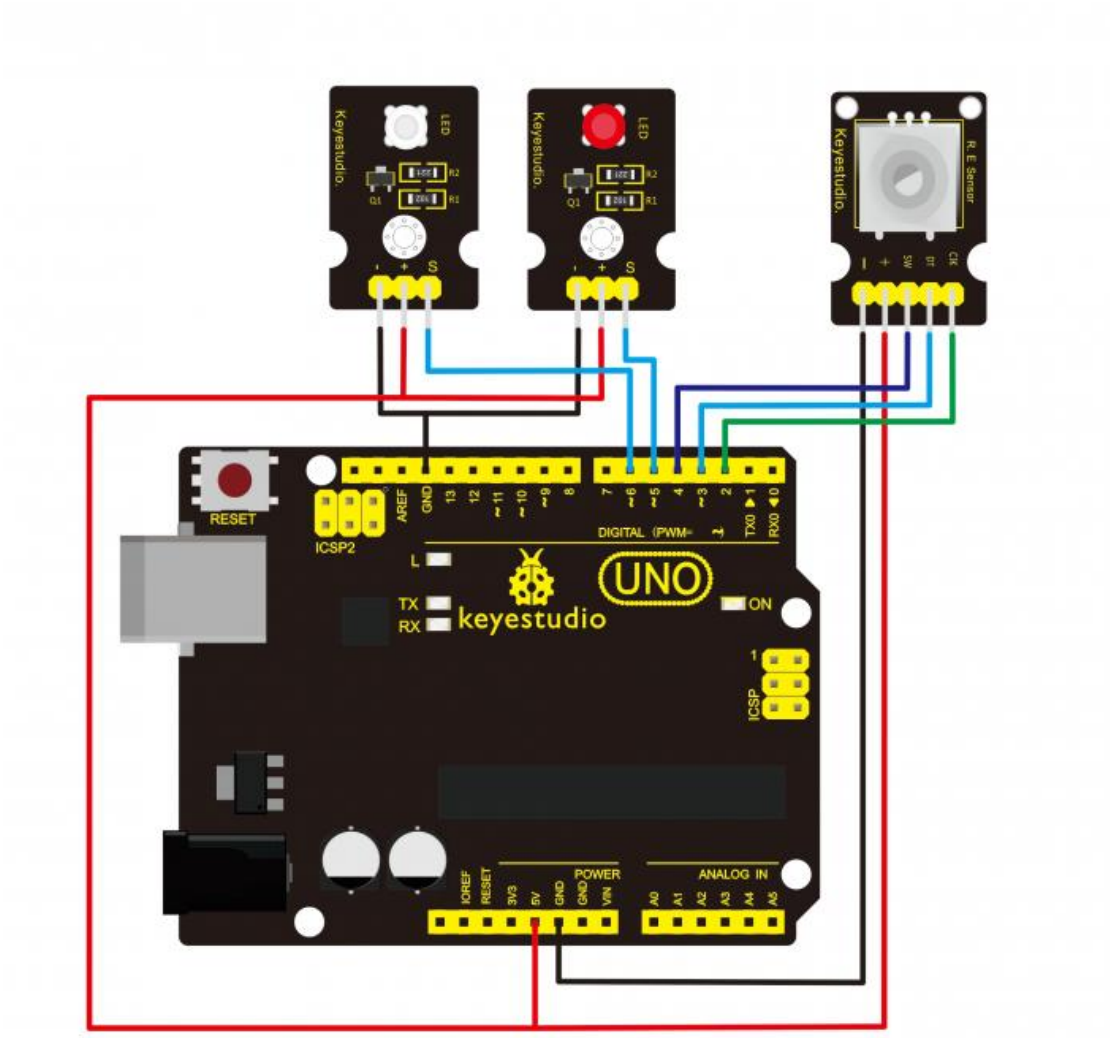
This rotational measurement is unlimited, not like the potential measurement. It can be reset to the original state to count from 0.

26.2 Specification

Power Supply: 5V

Interface: Digital

26.3 Connection diagram



26.4 Sample Code

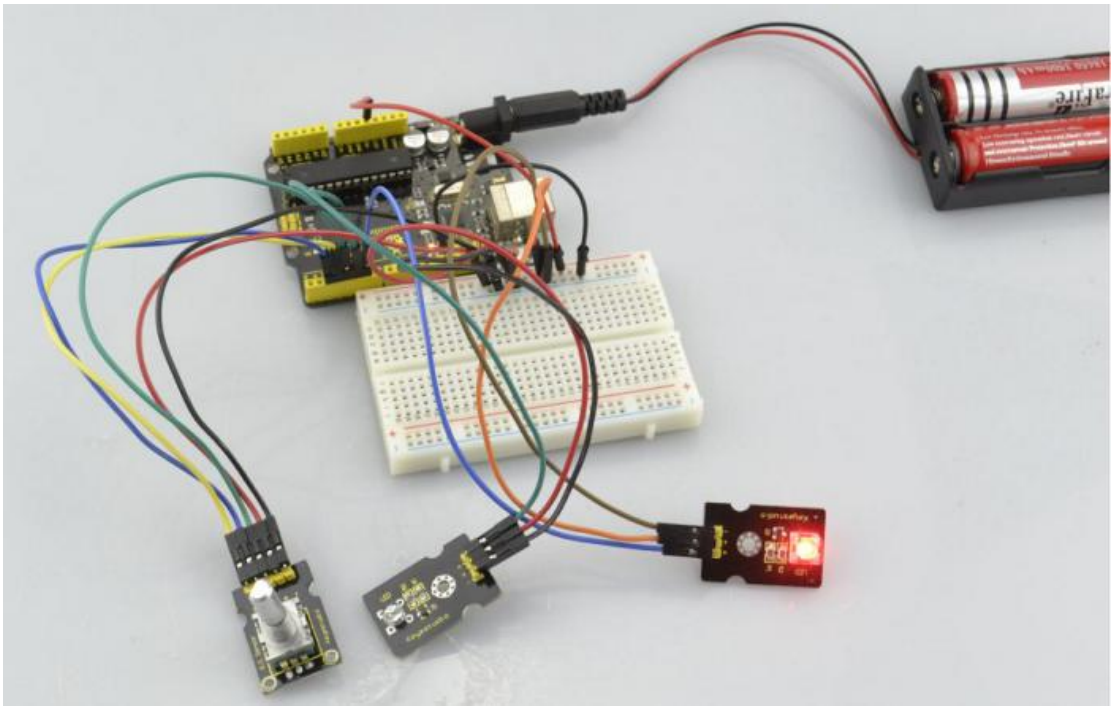
```
#include <OneWire.h>
 int DS18S20_Pin = 2; //DS18S20 Signal pin on digital pin 2
 //Temperature chip i/o
OneWire ds(DS18S20_Pin);  // on digital pin 2
 void setup(void) {
  Serial.begin(9600);
}
 void loop(void) {
  float temperature = getTemp();
  Serial.println(temperature);

  delay(100); //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;
}
```

26.5 Result

By connecting well and uploading the above code, it can rotate the encoder module to randomly turn on and off two LED modules.

When it rotates the encoder module, one LED module is first turned on but another is off. If it continues to rotate the encoder module, one LED module turns off while another is turned on, repeatedly.



CHAPTER TWENTY-SEVEN

PROJECT 27: LINEAR TEMPERATURE LM35

27.1 Introduction

LM35 linear temperature sensor is based on LM35 semiconductor temperature sensor. It can be used to detect ambient air temperature.

This sensor offers a functional range of 0 degrees Celsius to 100 degrees Celsius. The sensitivity is 10 mV 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 resistor, and thermal resistance and temperature semiconductor chips.

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

27.2 Specification

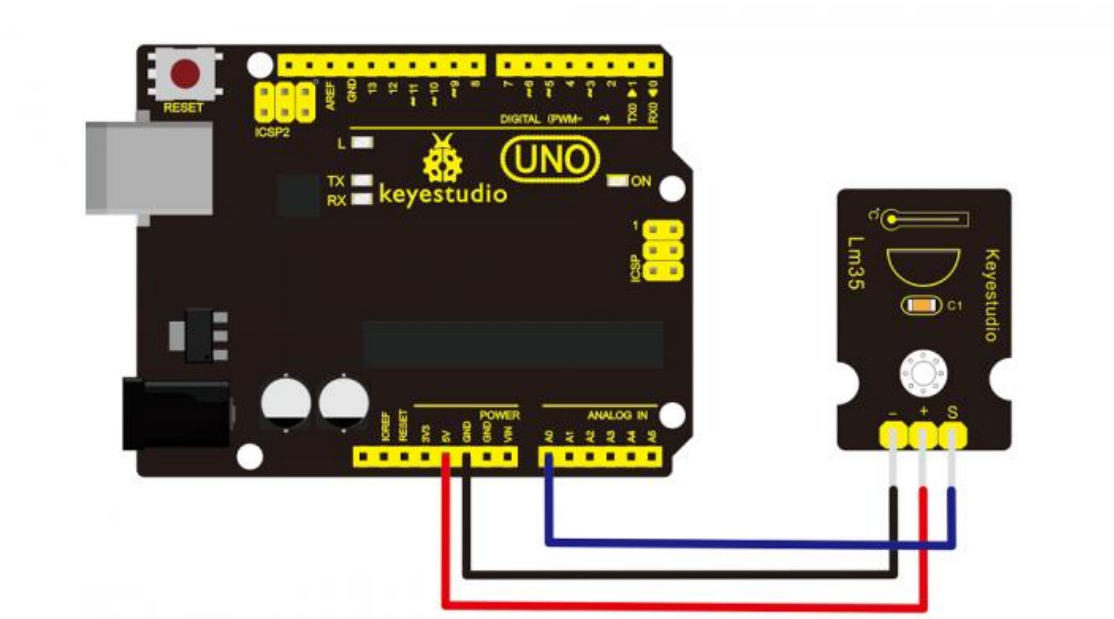
Based on LM35 semiconductor temperature sensor

Can be used to detect ambient air temperature

Sensitivity: 10mV per degree Celsius

Operating Range: 0 degrees Celsius to 100 degrees Celsius

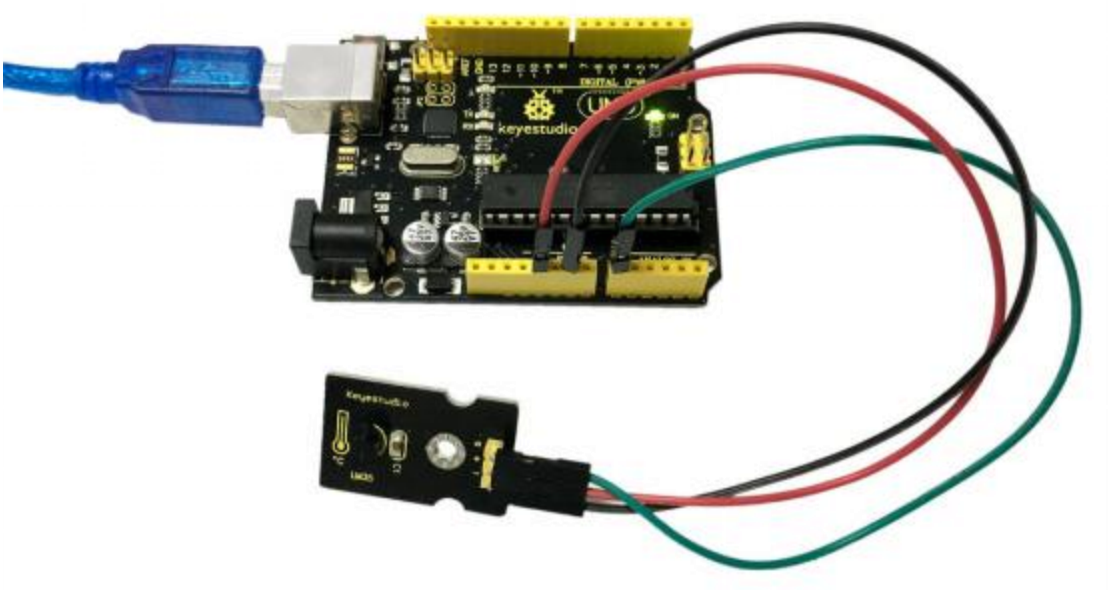
27.3 Connection diagram



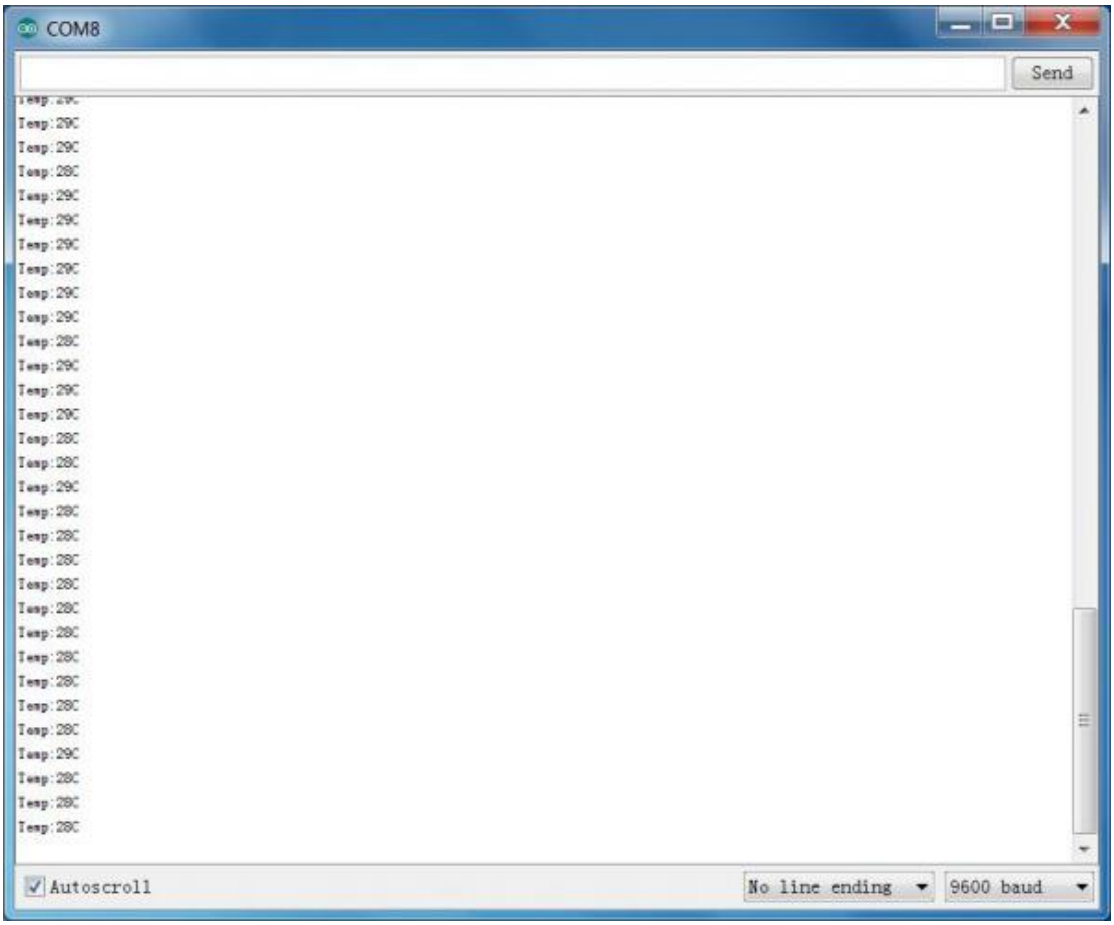
27.4 Sample Code

```
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);
}
```

27.5 Result



The user needs to plug it in like the diagram above and upload the code well to the board, then open the serial display and set the baud rate as 9600. Completing the process will see the current temperature value shown below. The price may have a slight difference due to different place and weather.


COM8
Send
Temp:29C
Temp:29C
Temp:28C
Temp:29C
Temp:29C
Temp:29C
Temp:29C
Temp:29C
Temp:29C
Temp:28C
Temp:29C
Temp:29C
Temp:29C
Temp:28C
Temp:28C
Temp:29C
Temp:28C
Temp:28C
Temp:28C
Temp:28C
Temp:28C
Temp:28C
Temp:28C
Temp:28C
Temp:28C
Temp:29C
Temp:28C
Temp:28C
Autoscroll
No line ending
9600 baud

CHAPTER TWENTY-EIGHT

PROJECT 28: TEMPERATURE SENSOR 18B20

28.1 Introduction

The DS18B20 is a digital temperature sensor. It can be used to quantify the environmental temperature test.

The temperature range is -55 ± 125 °C, native temperature resolution of 0.5 °C. It also supports multi-point grid networking.

The DS18B20 can be developed to achieve multi-point temperature measurement. It has a 9-12 bit serial output.

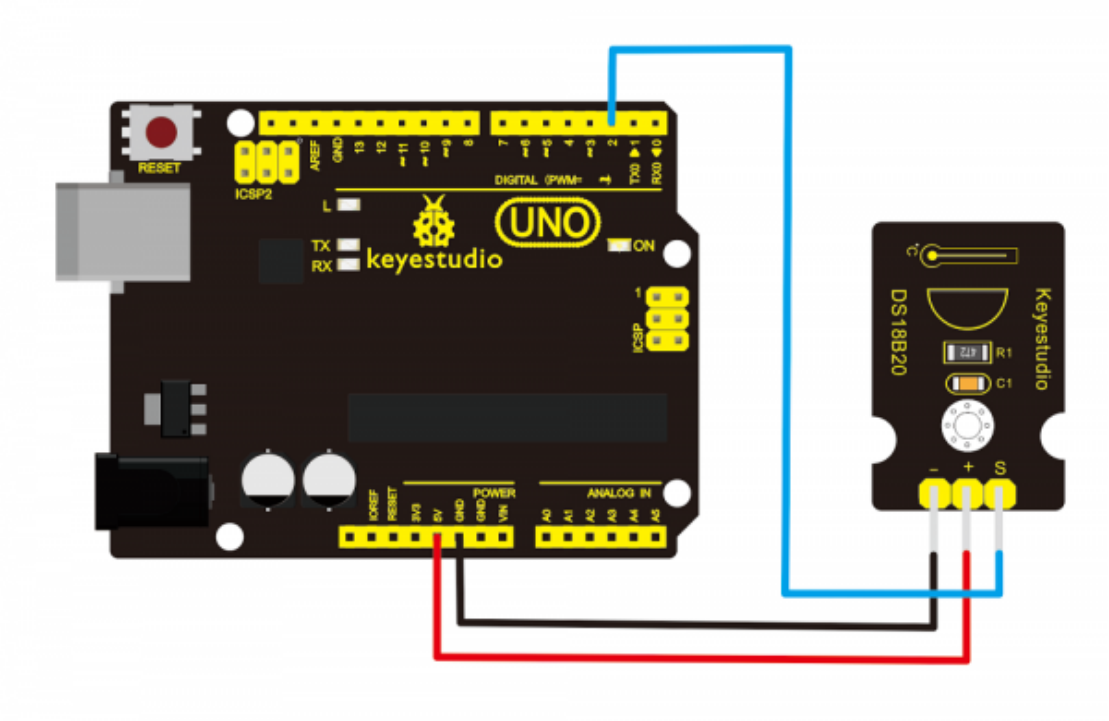
28.2 Specifications:

Supply voltage: 3.3V to 5V

Temperature range: -55°C~+125°C

Interface: Digital

28.3Connection diagram



```
#include <OneWire.h>
 int DS18S20_Pin = 2; //DS18S20 Signal pin on
digital pin 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);

  delay(100); //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;
}
```

After the code is uploaded to the board, the user opens the serial display and can "read" the measured temperature data.

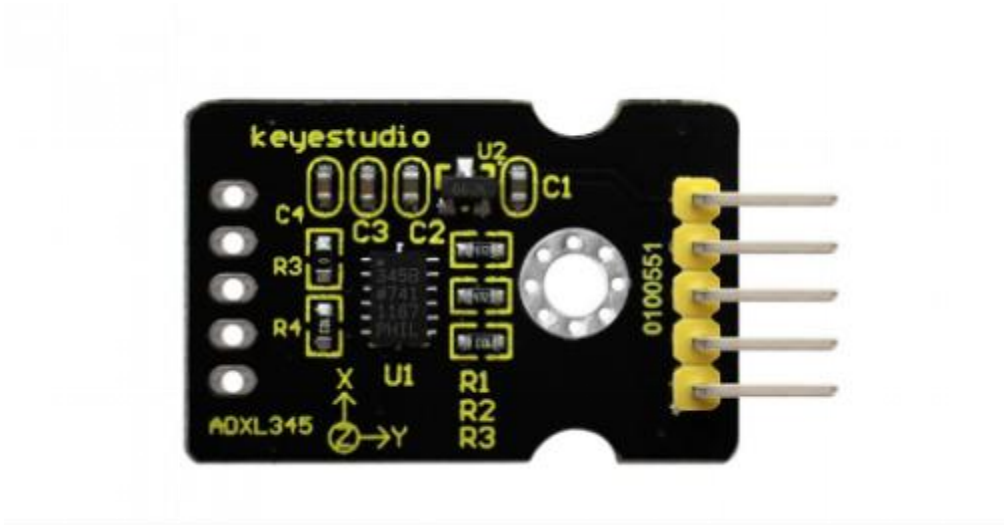
CHAPTER TWENTY-NINE

PROJECT 29: ACCELERATION OF THREE AXES ADXL345

29.1Introduction

The ADXL345 is a small, thin, low-power, 3-axis MEMS accelerometer with a high-resolution (13-bit) measurement at up to ±16 g. The digital output data is configured as a 16-bit twos complement and can be accessed either via an SPI digital interface (3- or 4-wire) or via I2C.

The ADXL345 is suitable for measuring the static acceleration of gravity in tilt detection applications, as well as the dynamic acceleration resulting from motion or impact. Its high resolution (4 mg/LSB) allows for the measurement of tilt changes of less than 1.0 degrees.



29.2Specifications

Supply voltage 2.0-3.6VDC

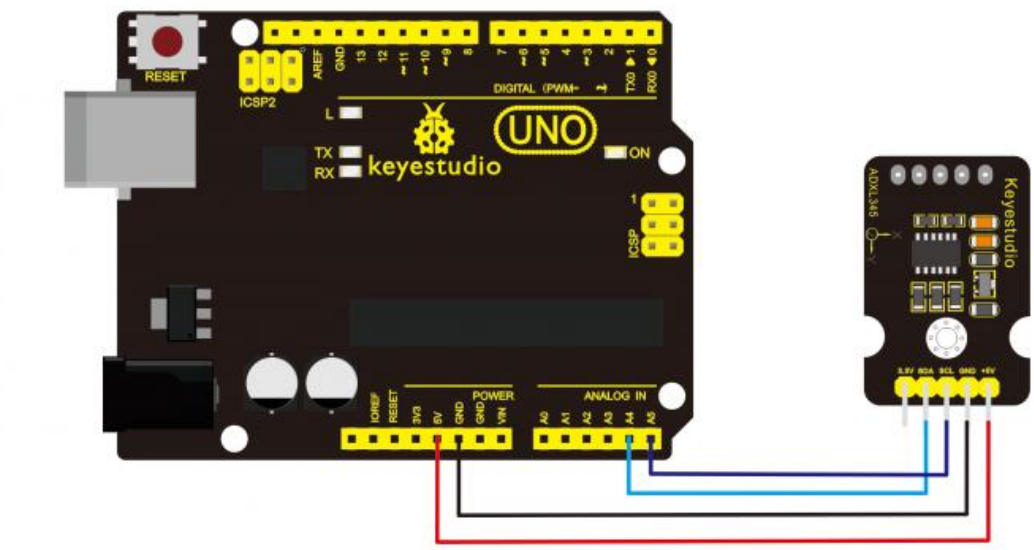
Ultra-low power: 40uA in metering mode, 0.1uA in standby mode at 2.5V

Faucet/Double Faucet Detection

Free Fall Detection

SPI and I2C Interface

29.3Connection diagram



29.4Sample Code

The circuit connection is as follows:

Virtual Credit Cards: 5V

GND: ground

SCL: UNO A5

SDA: UNO A4

```
#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)

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();
```

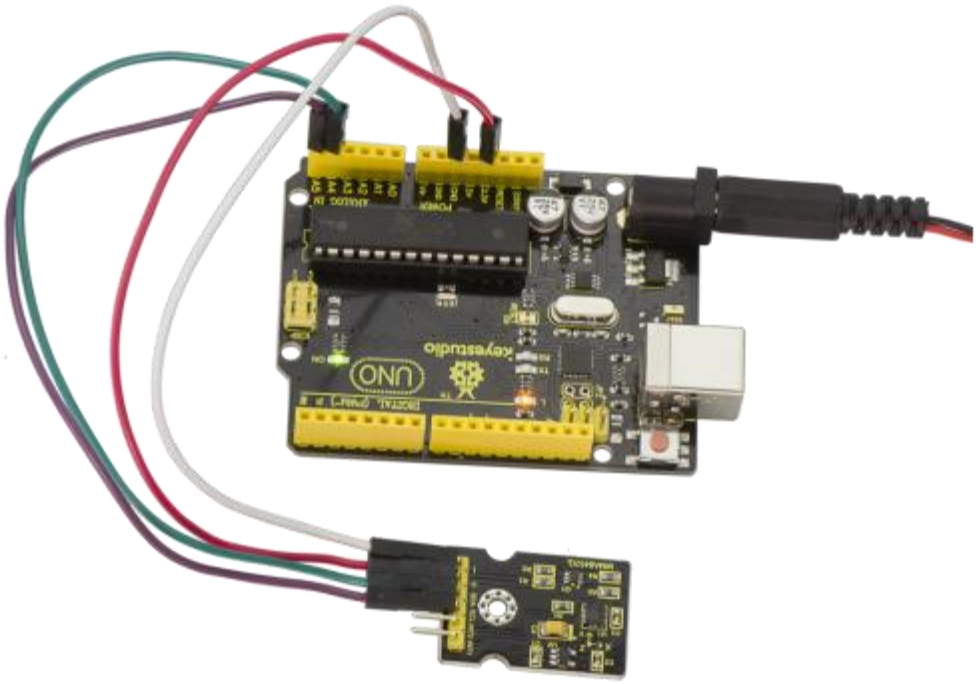
```
}

// 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
  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
```

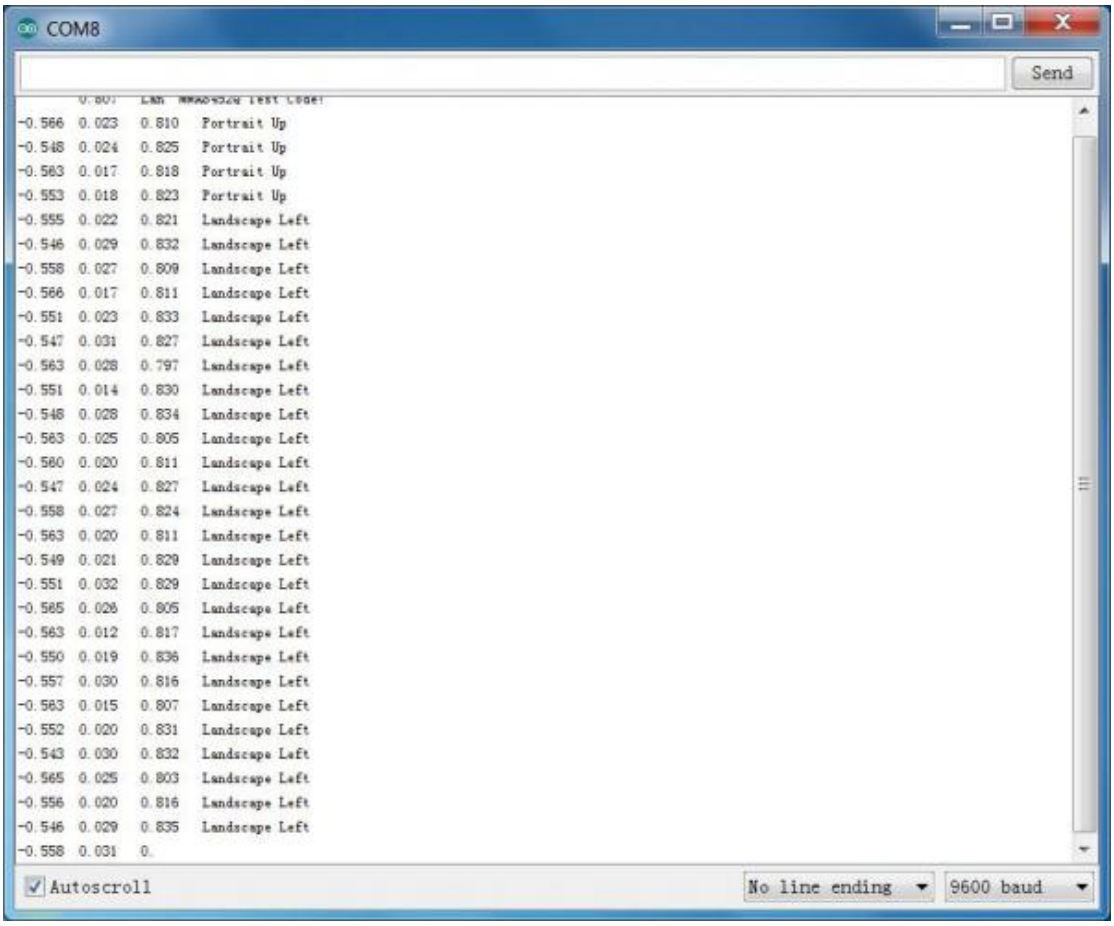
```
    Serial.println("it work Fail");
}

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);

}
```

29.4 Result



The wiring like the above diagram and the power supply, then need to load the code and open the serial display. It will then display the sensor's triaxial acceleration and its status, as shown below.


COM8
Send
-0.566 0.023 0.810 Portrait Up
-0.548 0.024 0.825 Portrait Up
-0.563 0.017 0.818 Portrait Up
-0.553 0.018 0.823 Portrait Up
-0.555 0.022 0.821 Landscape Left
-0.546 0.029 0.832 Landscape Left
-0.558 0.027 0.809 Landscape Left
-0.566 0.017 0.811 Landscape Left
-0.551 0.023 0.833 Landscape Left
-0.547 0.031 0.827 Landscape Left
-0.563 0.028 0.797 Landscape Left
-0.551 0.014 0.830 Landscape Left
-0.548 0.028 0.834 Landscape Left
-0.563 0.025 0.805 Landscape Left
-0.560 0.020 0.811 Landscape Left
-0.547 0.024 0.827 Landscape Left
-0.558 0.027 0.824 Landscape Left
-0.563 0.020 0.811 Landscape Left
-0.549 0.021 0.829 Landscape Left
-0.551 0.032 0.829 Landscape Left
-0.565 0.026 0.805 Landscape Left
-0.563 0.012 0.817 Landscape Left
-0.550 0.019 0.836 Landscape Left
-0.557 0.030 0.816 Landscape Left
-0.563 0.015 0.807 Landscape Left
-0.552 0.020 0.831 Landscape Left
-0.543 0.030 0.832 Landscape Left
-0.565 0.025 0.803 Landscape Left
-0.556 0.020 0.816 Landscape Left
-0.546 0.029 0.835 Landscape Left
-0.558 0.031 0.
Autoscroll
No line ending
9600 baud

CHAPTER THIRTY

PROJECT 30: DHT11 TEMPERATURE AND HUMIDITY SENSOR

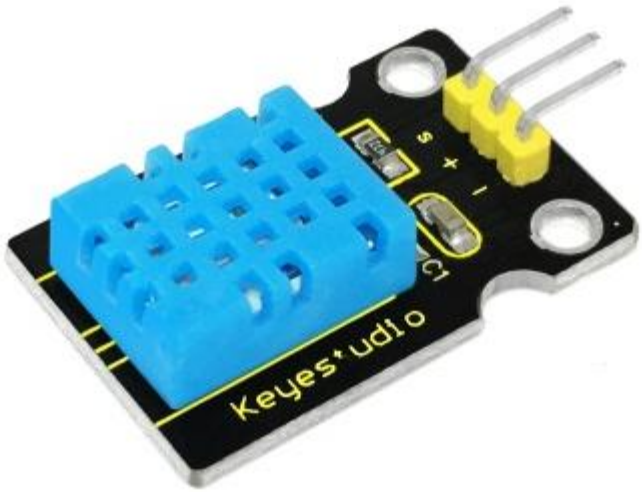
30.1 Introduction

This DHT11 Temperature and Humidity Sensor is a composite sensor that contains a calibrated digital output signal of temperature and humidity. Its technology ensures high reliability and excellent long-term stability.

A high-performance 8-bit microcontroller is connected. This sensor includes a resistance element and a sense of NTC liquid temperature measuring devices. It features excellent quality, fast response, anti-interference capability, and high-cost performance advantages.

Each DHT11 sensor has extremely expensive calibration data of the humidity calibration chamber. The calibration factors stored in the OTP program memory as well as the internal sensors detect signals in the process, and the user should dial these calibration factors.

The single-cable serial interconnect system is integrated to make it quick and easy. The properties of small size, low power and 20 meters signal transmission distance make it a widely applied application even the most demanding. Convenient connection, special packaging can be provided according to user needs.



30.2 Specifications

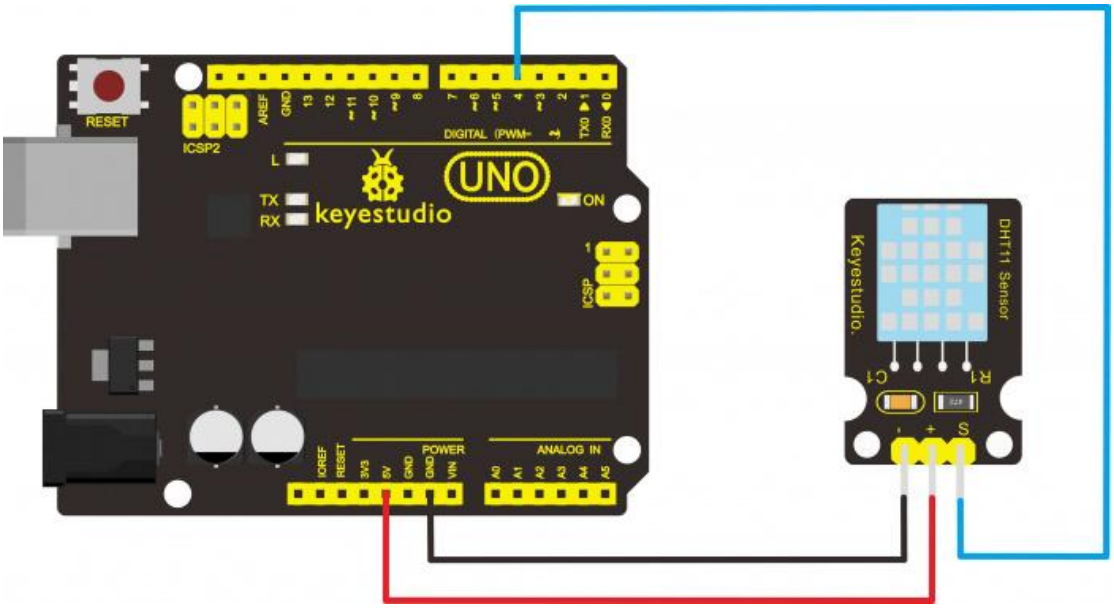
Supply voltage: +5 V

Temperature range: +5.5 V °C error ± 2 °C

Humidity: 20-90% RH ± 5% RH error

Interface: Digital

30.3Connection diagram



30.4 Sample Code

The user needs to download DHT11Lib.Or to view the website

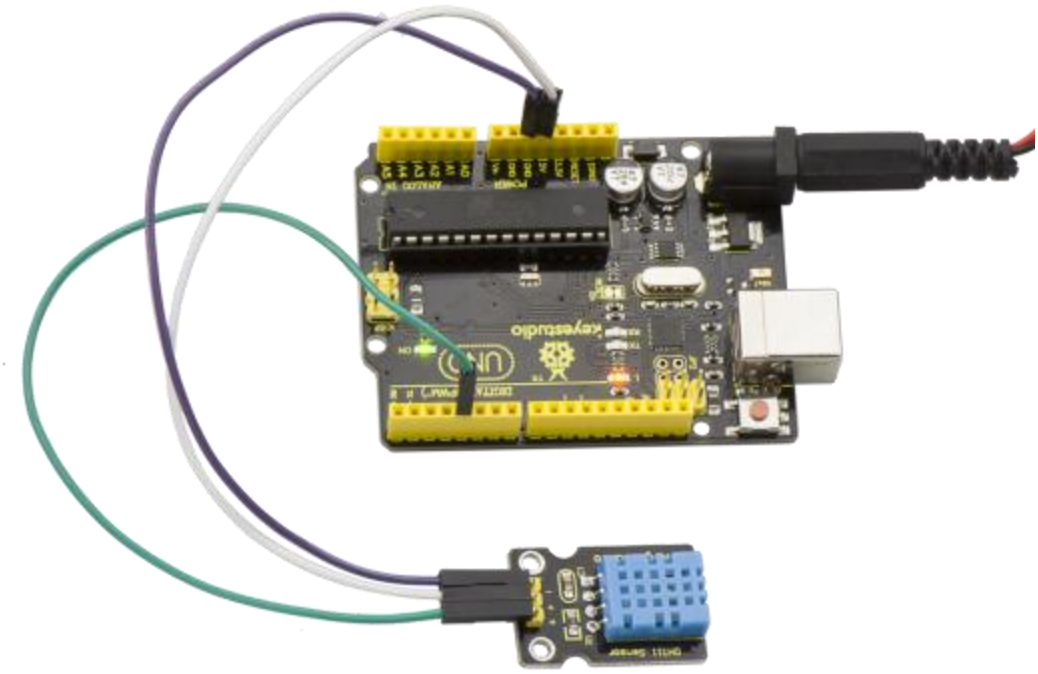
Note: before compiling the code, the user should remember to place the library in the Arduino IDE libraries directory. Otherwise, the compilation will fail.

```
#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:
```

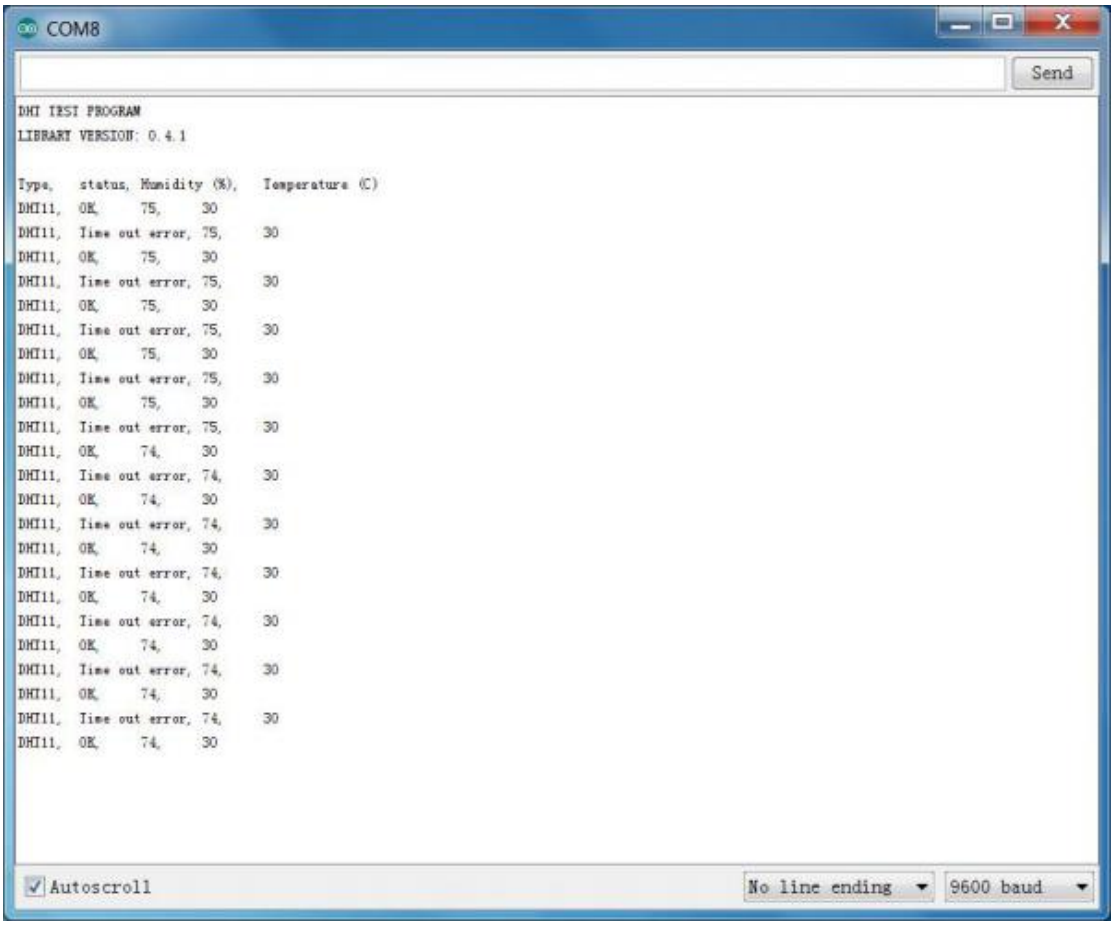
```
                Serial.print("Checksum error,\t");
                break;
    case DHTLIB_ERROR_TIMEOUT:
                Serial.print("Time out error,\t");
                break;
    default:
                Serial.print("Unknown error,\t");
                break;
  }
 // DISPLAT DATA
  Serial.print(DHT.humidity,1);
  Serial.print(",\t");
  Serial.println(DHT.temperature,1);

  delay(1000);
}
```

30.5 Result

Wire it well and upload the above code to the UNO board.



The user will then open the serial screen and set the baud rate as 9600 and finally see the current temperature and humidity value.


COM8
Send
DHT TEST PROGRAM
LIBRARY VERSION: 0.4.1
Type, status, Humidity (%), Temperature (C)
DHT11, OK, 75, 30
DHT11, Time out error, 75, 30
DHT11, OK, 75, 30
DHT11, Time out error, 75, 30
DHT11, OK, 75, 30
DHT11, Time out error, 75, 30
DHT11, OK, 75, 30
DHT11, Time out error, 75, 30
DHT11, OK, 75, 30
DHT11, Time out error, 75, 30
DHT11, OK, 74, 30
DHT11, Time out error, 74, 30
DHT11, OK, 74, 30
DHT11, Time out error, 74, 30
DHT11, OK, 74, 30
DHT11, Time out error, 74, 30
DHT11, OK, 74, 30
DHT11, Time out error, 74, 30
DHT11, OK, 74, 30
DHT11, Time out error, 74, 30
DHT11, OK, 74, 30
DHT11, Time out error, 74, 30
DHT11, OK, 74, 30
Autoscroll
No line ending
9600 baud

CHAPTER THIRTY-ONE

PROJECT 31: Bluetooth Module

31. 1 Introduction

This Bluetooth module can easily achieve serial wireless data transmission. Its operating frequency is among the most popular 2.4GHz ISM frequencies (i.e. industrial, scientific and medical).

It adopts Bluetooth 2.1+EDR standard. In Bluetooth 2.1, the signal transmission time of different devices is within 0.5 seconds, so that the workload of the Bluetooth chip can be greatly reduced and more sleep time can be saved for Bluetooth.

This unit is configured with a serial interface, which is easy to use and simplifies the overall design/development cycle.

31.2 Specifications

Bluetooth Protocol: EDR

USB Protocol: USB v1.1/2.0

Operating frequency: 2.4GHz ISM frequency band

Configuration Mode: Gauss

Transmitting power: ≤ 4dBm, second stage

Sensitivity: ≤-84dBm at 0.1% bit error rate

Transmission speed: 2.1Mbps(Max)/160 kbps (asynchronous); 1Mbps/1Mbps (synchronous)

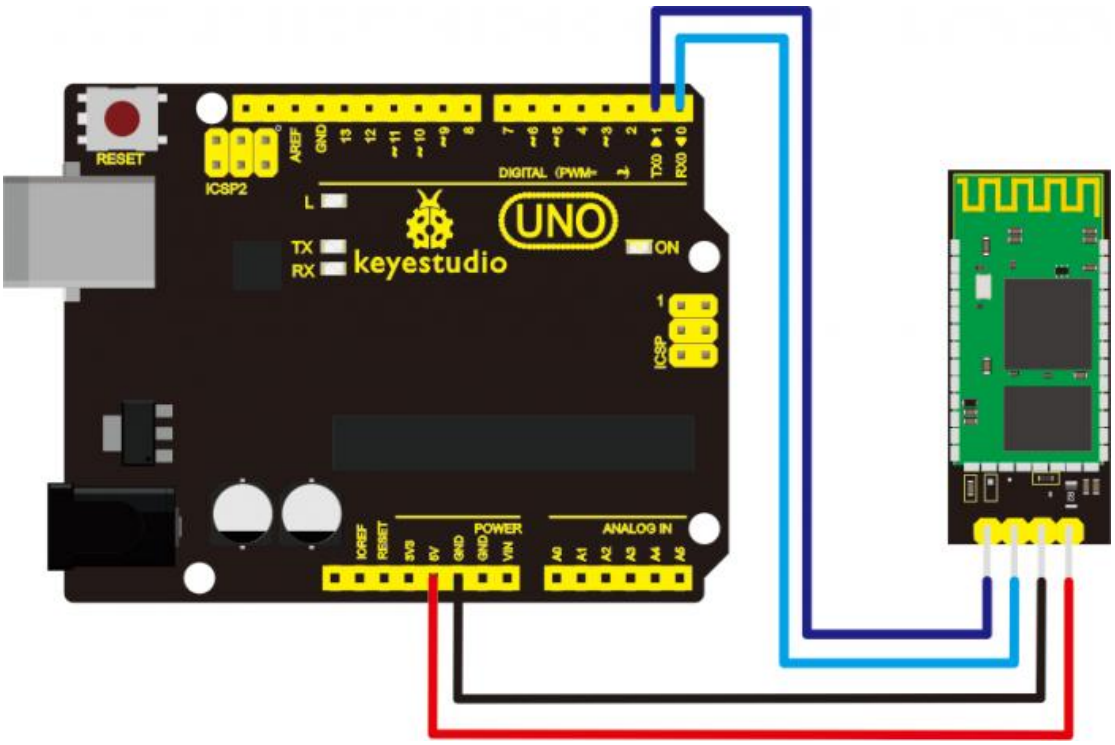
Security Feature: Network Interaction: Identity and Encryption

Supported configuration: Bluetooth serial port (major and minor)

Supply Voltage: 5V DC 50mA

Operating temperature: -20 to 55 °C

31.3 Connection diagram



31.4 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");
}
}
```

CHAPTER THIRTY-TWO

PROJECT 32: AMBIENT LIGHT TEMT6000

32.1 Introduction

It is used by users who want to sense the brightness of the environment more accurately than reliable photoresistor, which is why they get an ambient light sensor TEMT6000.

The TEMT6000 is supposed to be adapted to the sensitivity of the human eye, but found that it pre-performed inferior performance in low light conditions.

However, it works very well by reacting to very small changes in a wide range of brightness. Because it is meant to mimic the human eye, it does not react well to infrared or ultraviolet light, so let the user make sure to note this when using it in the project.

32.2 Specifications

Supply voltage: +50mA

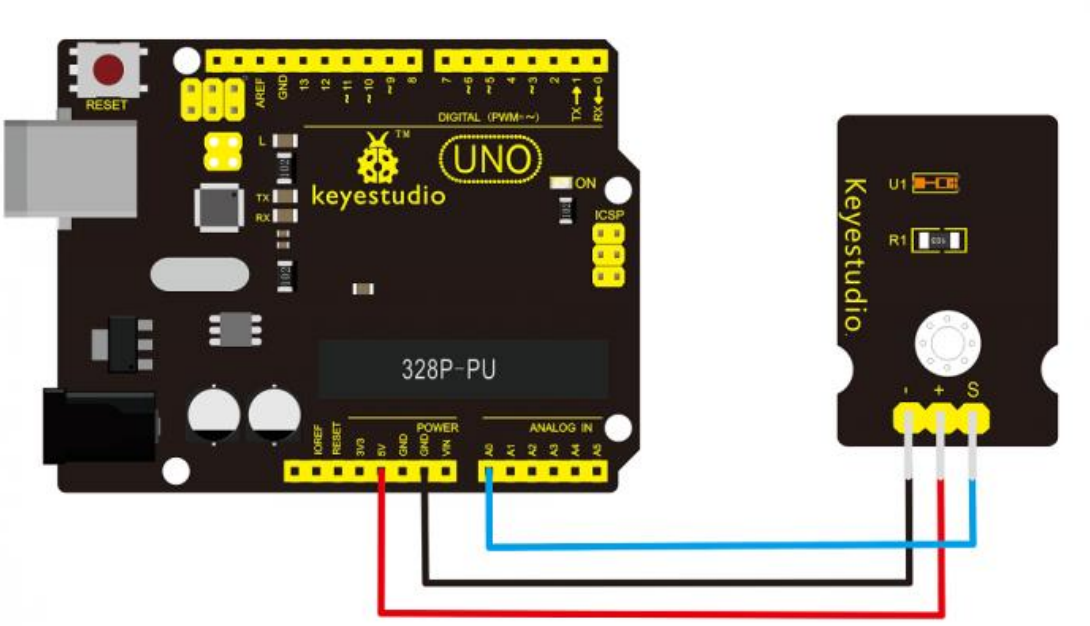
Size: 36.5*16mm

Weight: 4g

32.3 Connection diagram

This one is quite simple and the user simply connects the power, ground, and signal pin to the favorite analog input and when ready, the sensor will output analog voltage, which increases when it gets brighter.

It can power it from 3.3v as the user wishes and then, the output price will simply be lower.



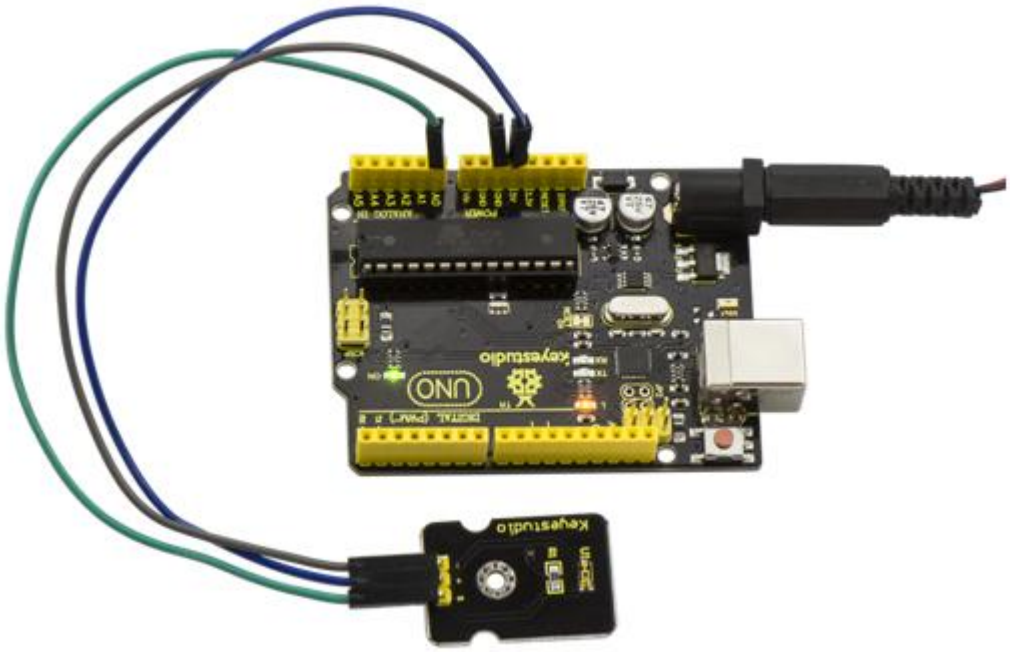
32.4Sample Code

It can't get simpler than him. This simply reports the reading from the sensor on the serial terminal: 1023 is too bright and 0 is too dark.

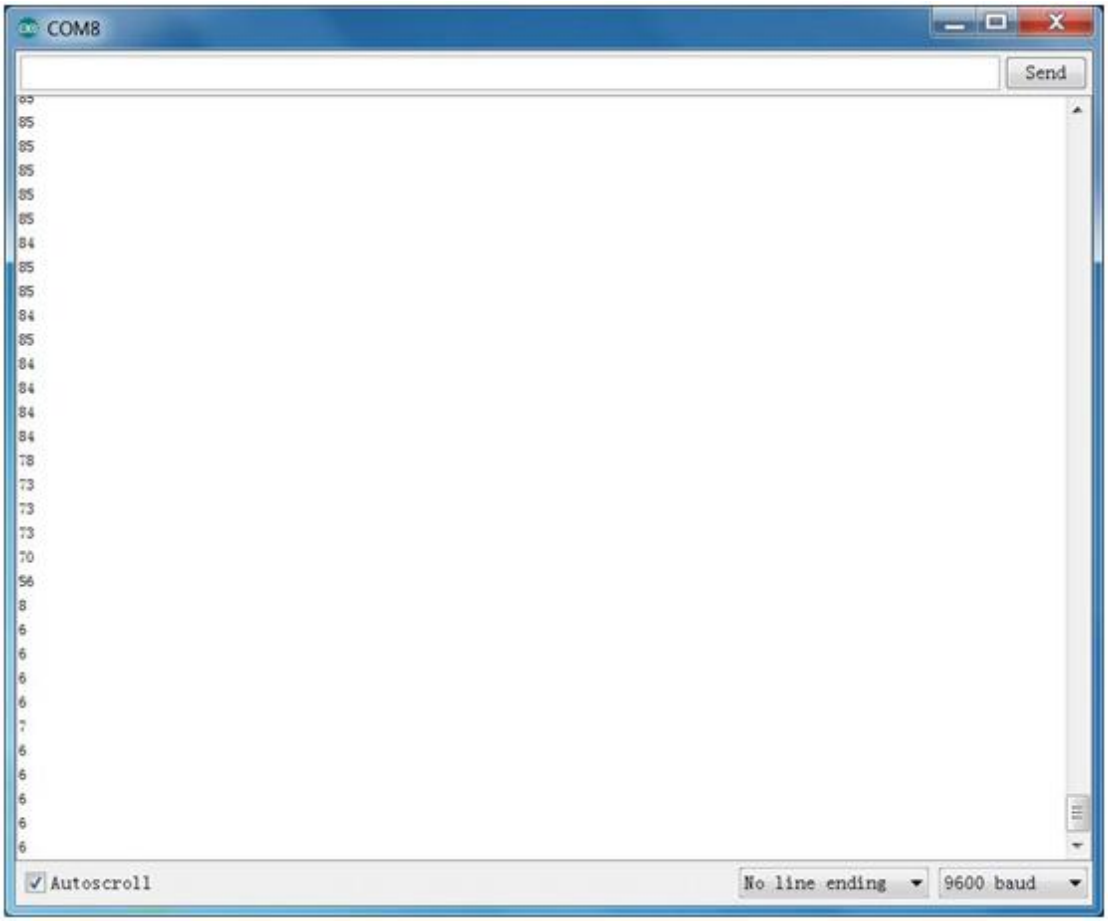
```
int temt6000Pin = 0;
void setup() {
 Serial.begin(9600);
}
void loop() {
 int value = analogRead(temt6000Pin);
 Serial.println(value);
 delay(100); //only here to slow down the output so
it is easier to read
}
```

32.5 Result

By wiring and uploading the above code, it opens the serial screen of the Arduino software.



Then, he covers the sensor by hand or with a piece of paper, the light fades and eventually the user will "read" the value displayed on the screen to decrease.


COM8
Send
85
85
85
85
85
85
84
85
85
84
85
84
84
84
84
78
73
73
73
70
56
8
6
6
6
6
7
6
6
6
6
6
Autoscroll
No line ending
9600 baud

CHAPTER THIRTY-THREE

PROJECT 33: SR01 ULTRASONIC SENSOR

33.1 Introduction

Keyestudio's SR01 ultrasonic sensor is a very affordable proximity/distance sensor that has been mainly used to avoid objects in various robotics projects.

It essentially gives the Arduino eyes/attention and can prevent some robot from crashing or falling off a table. It has also been used in turret, water level sensing, and even parking sensor applications.

This simple project will use the Keyestudio SR01 Ultrasonic Sensor with an Arduino and a Sketch Processing to provide a neat little interactive display on your computer screen.

33.2 Specifications

Operating Voltage: Operating Voltage: Operating Voltage: DC 5V

Working Current: 15mA

Working Frequency: 40Hz

Maximum Range: 5m

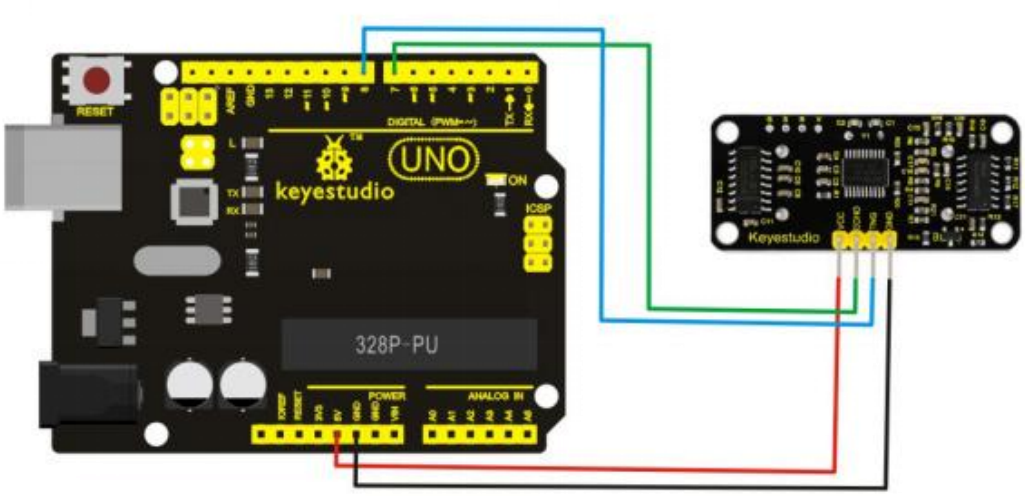
Minimum Range: 2cm

Measuring Angle: 15 degrees

Trigger input signal: 10μS TTL pulse

Echo Output Signal TTL Lever Input Signal and Range in Ratio

33.3Connection diagram



SR01 Ultrasonic UNO Sensor

VCC 5V

GND GND

Digital echo pin 7

Digital Pin 8

33.4 Sample Code

VCC to Arduino 5V

GND to Arduino GND

Echo on Arduino Pin 7

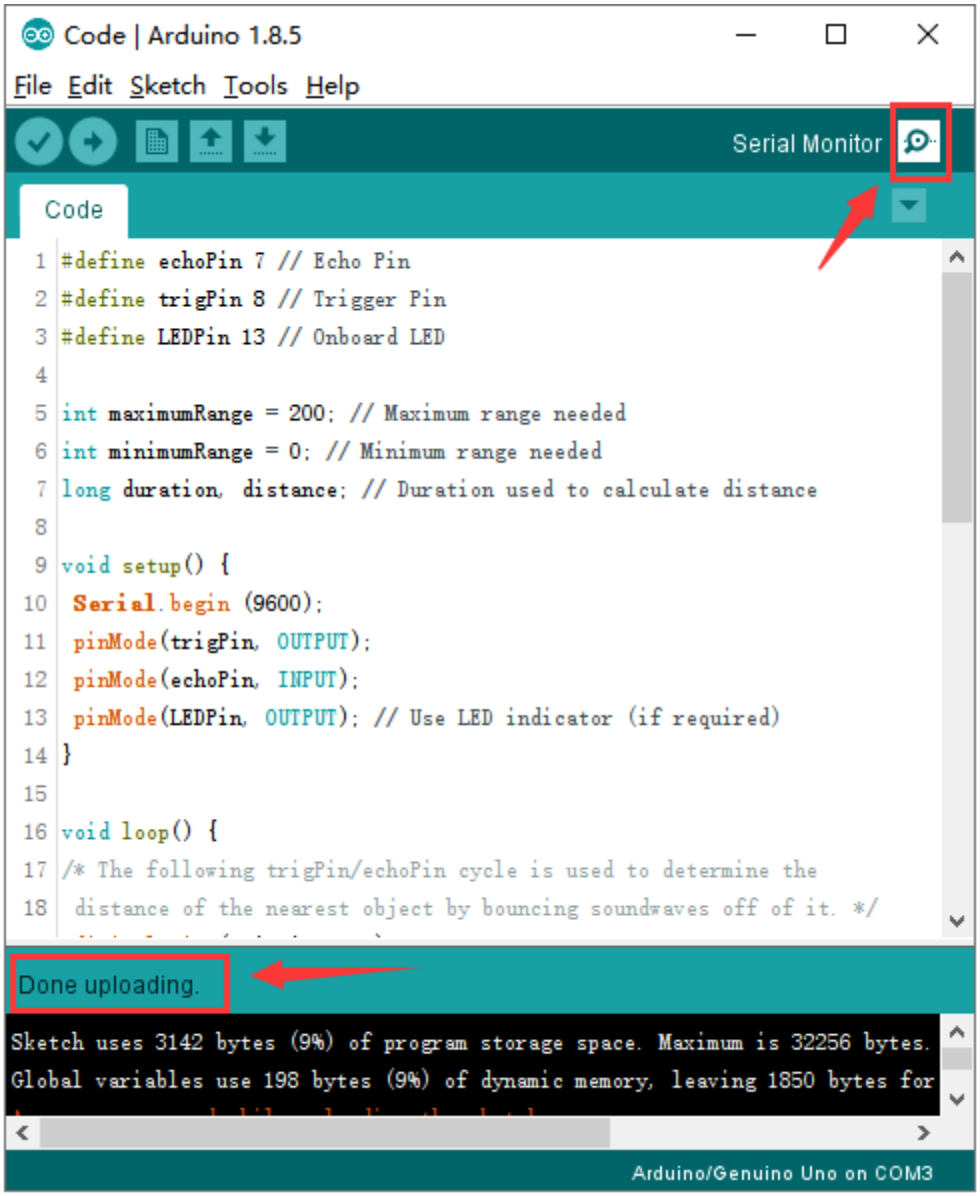
Trig on Arduino Pin 8

```
#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);
```

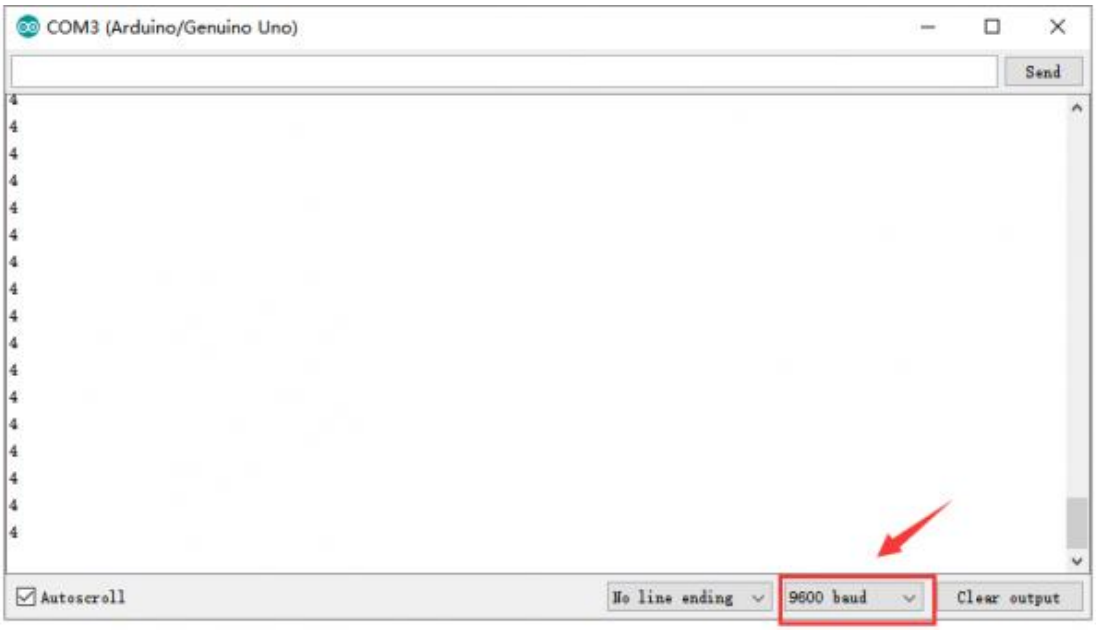
```
 digitalWrite(trigPin, HIGH);
 delayMicroseconds(10);
 digitalWrite(trigPin, LOW);

 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);
}
```

33.5 Result



After loading the code onto the board, it opens the serial display of the Arduino IDE, it can "read" the value of the distance measured by the ultrasonic sensor.


COM3 (Arduino/Genuino Uno)
Send
4
Autoscroll
No line ending
9600 baud
Clear output

CHAPTER THIRTY-FOUR

PROJECT 34: JOYSTICK MODULE

34.1 Introduction

Many robot projects need joysticks. This section provides an affordable solution. By simply connecting to two analog inputs, the robot is in the commands with X, Y control.

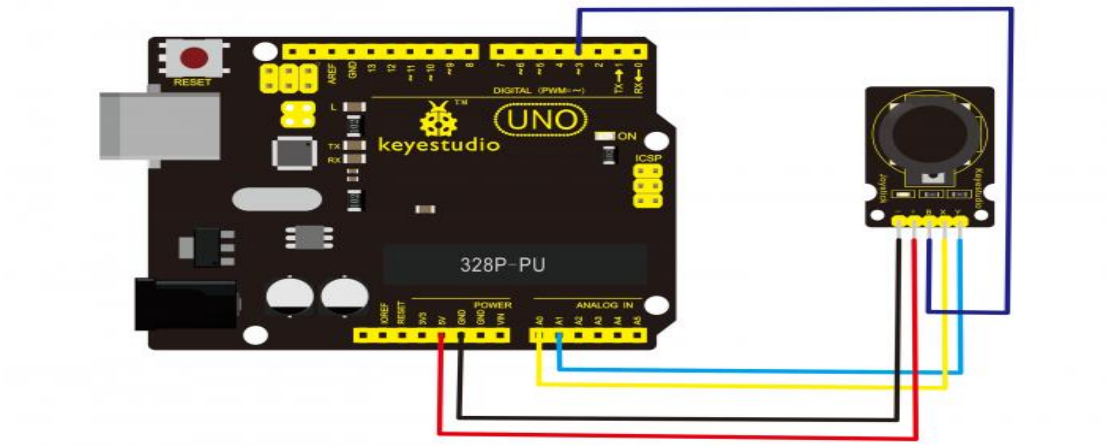
It also has a switch that plugs into a digital terminal. This joystick module can be easily connected to the Arduino via the IO shield.

34.2 Specifications

Supply voltage: 3.3V to 5V

Interface: x2, digital x1

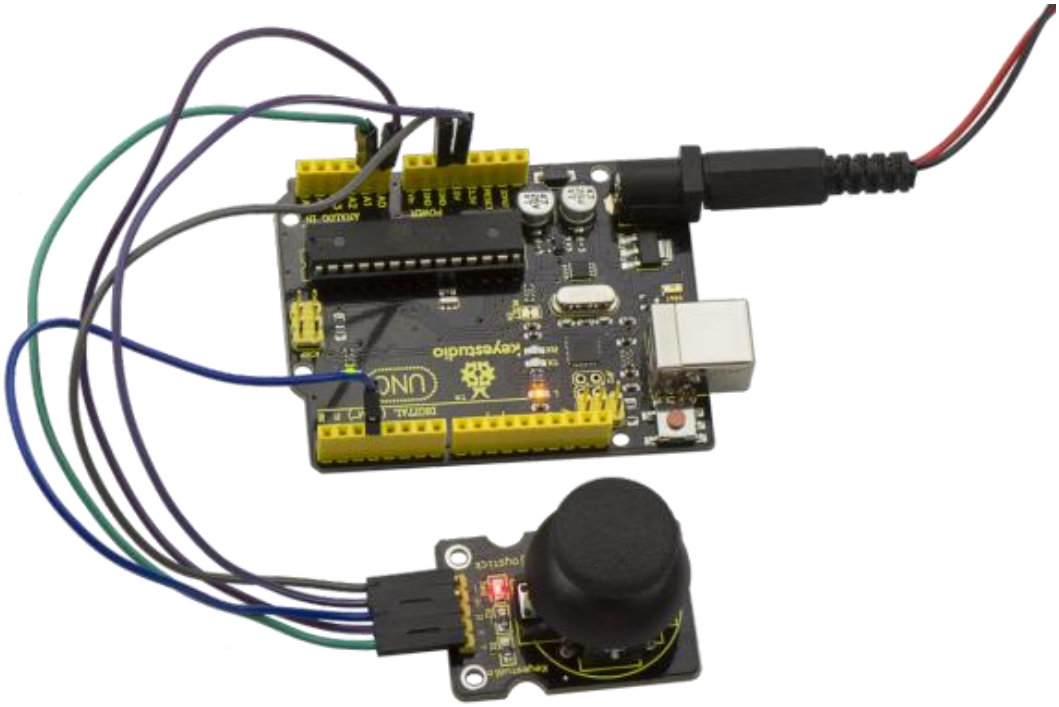
34.3 Connection diagram



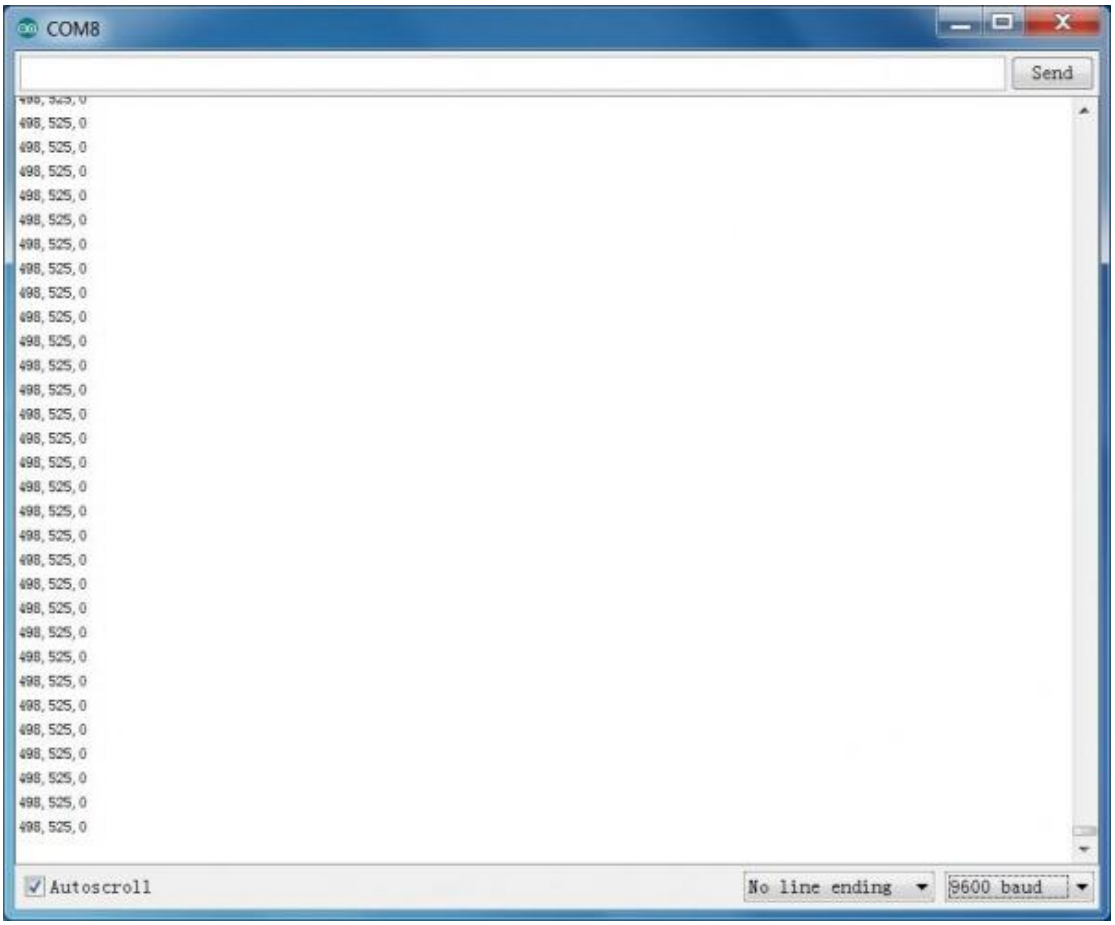
34.4 Sample Code

```
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);
}
```

34.5 Result



After wiring and uploading the code, the user opens the serial display and sets the baud rate to 9600, presses the joystick, and "reads" the value shown below.


COM8
Send
498, 525, 0
Autoscroll
No line ending
9600 baud

CHAPTER THIRTY-FIVE

PROJECT 35: DS3231 WATCH UNIT

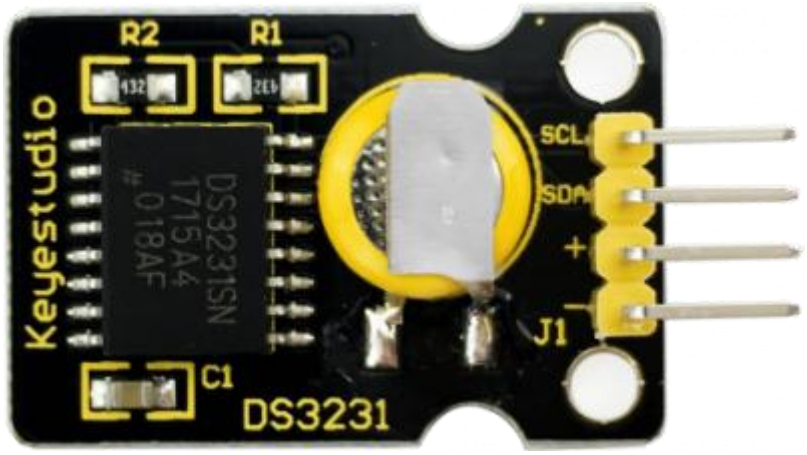
35.1 Introduction

The DS3231 is equipped with built-in TCXO and crystal, which make it an economical I2C real-time watch with high precision. The device carries a battery input, so if it disconnects the main power supply, it can maintain accurate timing.

The built-in oscillator ensures the long-term accuracy of the device and reduces the number of parts.

The DS3231 provides both commercial and industrial temperature ranges and supports 16-pin (300mil) small contour packaging.

The unit itself can be adapted to the 3.3V and 5V system without a level switch, which is quite convenient!



35.2 Specifications

Timing accuracy: ± 5ppm (±0.432 seconds/day)

Providing a backup battery for continuous timekeeping

Low power consumption

Device packaging and DS3231 compatible mode

Full clock calendar function contains seconds and minutes, hour, week, date, month, and year of timekeeping, and provides leap year compensation up to 2100.

Two calendar clocks

Output: 768kHz: 1Hz and 32,768kHz

Reset output and button input bounce

High-speed (400kHz), I2C serial bus

Supply voltage: +3.3V to +5.5V

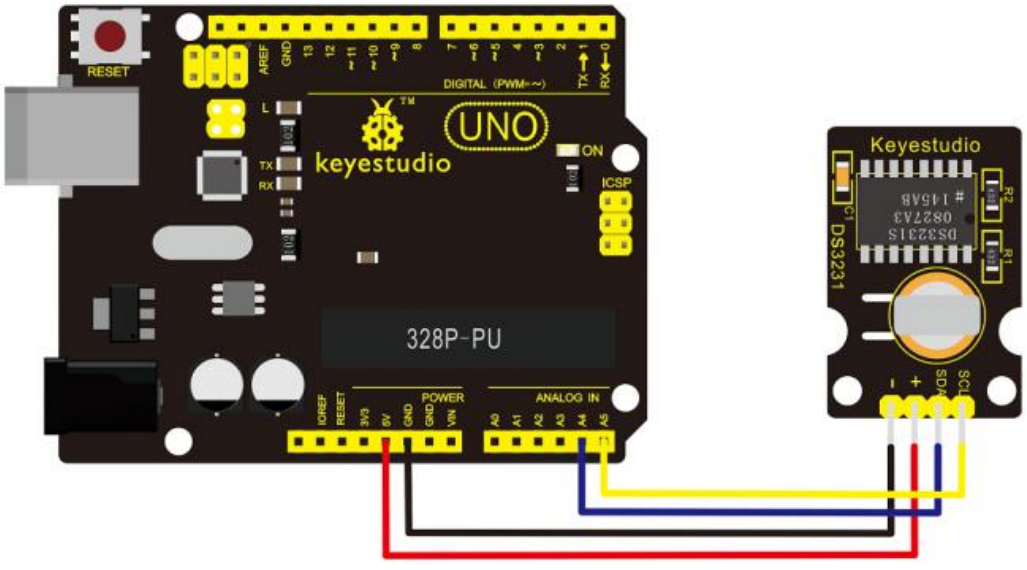
Digital temperature sensor with an accuracy of ±3°C

Operating Temperature: -40~C to +85~C

16 pin (300mil) small contour package

35.3Connection diagram

This module adopts the IIC test method, so we only need to connect the SDA to the Arduino A4- the SCL to the A5- the positive pin to the VCC- the negative pin to the GND.



35.4 Sample Code

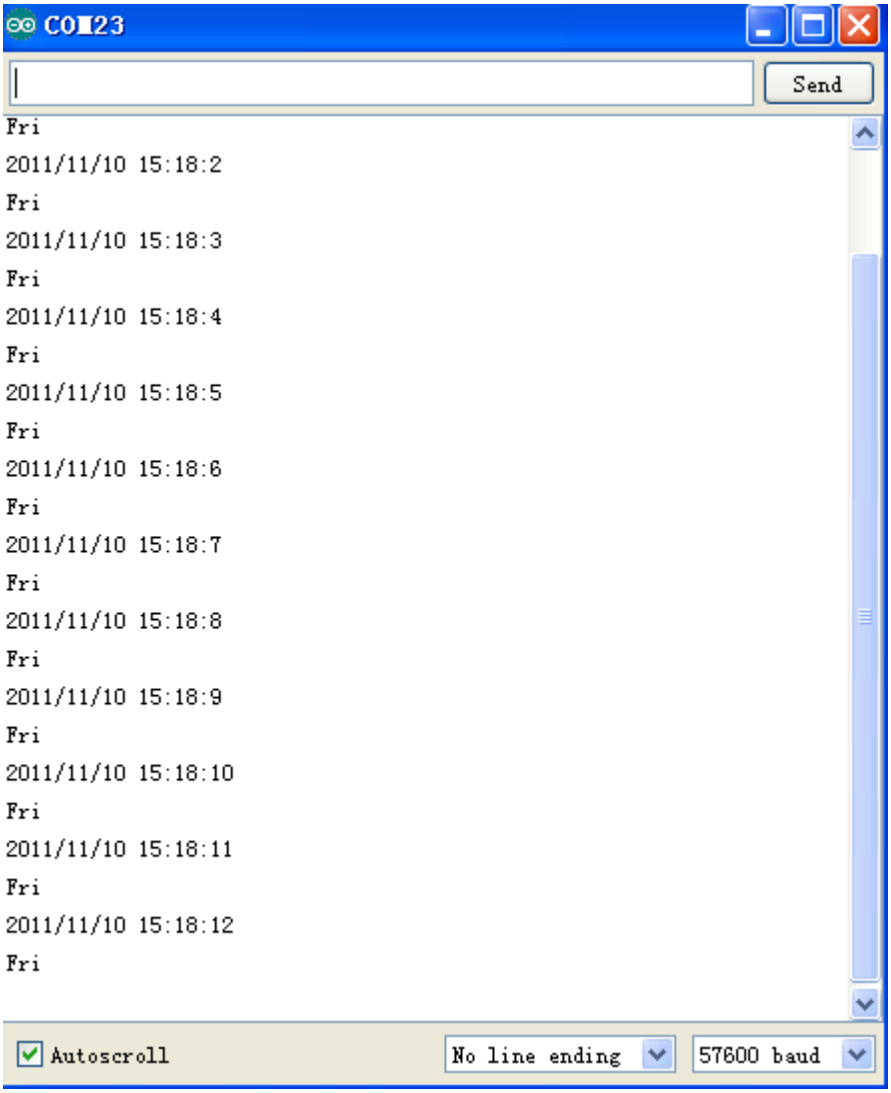
Before compiling the code, it would be better for the user to put the DS3231 library under the file in the Arduino directory.

```
#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.
//Take care of week-day also.
DateTime dt(2011, 11, 10, 15, 18, 0, 5);//open the
serial 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, it is better for the user to put the DS3231 library under the file in the Arduino directory.

When the above steps are completed, the user can load the code into the Arduino and open the serial display and get the following results:


COM23
Send
Fri
2011/11/10 15:18:2
Fri
2011/11/10 15:18:3
Fri
2011/11/10 15:18:4
Fri
2011/11/10 15:18:5
Fri
2011/11/10 15:18:6
Fri
2011/11/10 15:18:7
Fri
2011/11/10 15:18:8
Fri
2011/11/10 15:18:9
Fri
2011/11/10 15:18:10
Fri
2011/11/10 15:18:11
Fri
2011/11/10 15:18:12
Fri
Autoscroll
No line ending
57600 baud

CHAPTER THIRTY-SIX

PROJECT 36: 5V RELAY MODULE

36.1 Introduction

This single relay module can be used in interactive projects making up an active HIGH level. The unit in question uses high-quality SONGLE 5v relays.

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 output of the relay is through a light-emitting diode. It can be controlled via digital IO port, such as solenoid valves, bulbs, motors, and other high-current or high-voltage devices.

36.2 Specifications

Type: Digital

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

Maximum Switching Voltage: 150VAC 24VDC

Digital Interface

Control Signal: TTL

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

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

Contact Action Time: 10ms

36.3 Connection diagram

First he needs to prepare the following parts before testing.

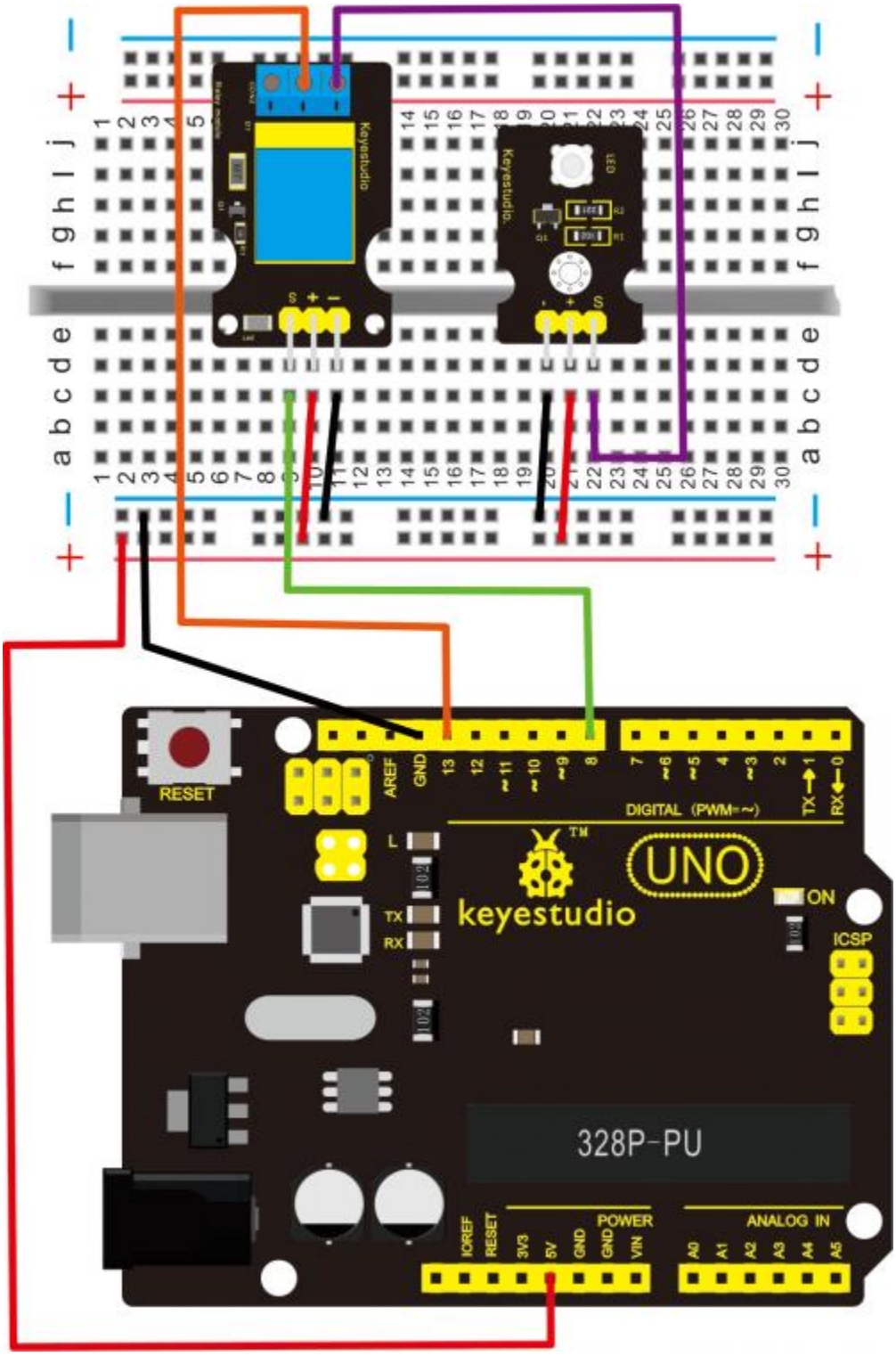
Arduino Board*1

Single Relay Unit*1

LED Module *1

USB Cable*1

Jumper Cable*8


RESET
AREF
GND
DIGITAL (PWM=~)
TX
RX
keyestudio
UNO
ON
ICSP
328P-PU
POWER
ANALOG IN
IOREF
RESET
3V3
5V
GND
GND
VIN
A0
A1
A2
A3
A4
A5

36.4 Sample Code

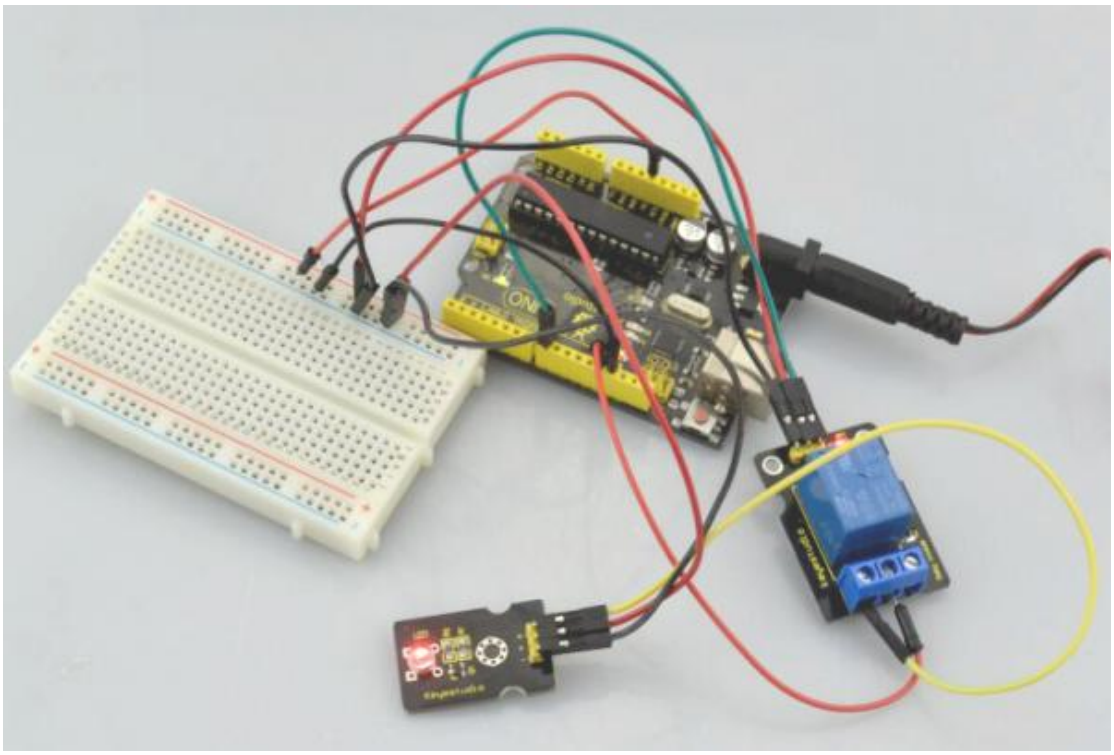
It needs to copy and paste the following code into the Arduino software.

```
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);
}
```

36.5 Test Result

This relay module is active at a high level. It wires the cable, supplies it with power, and then loads the above code onto the board. It will "see" that the relay is turned on (ON connected, NC disconnected)for two seconds, then turned off for two seconds (NC closed, ON disconnected), repeatedly and cyclically.

When the relay is turned on, the external LED lights up. If the relay is turned off, the external LED is turned off.

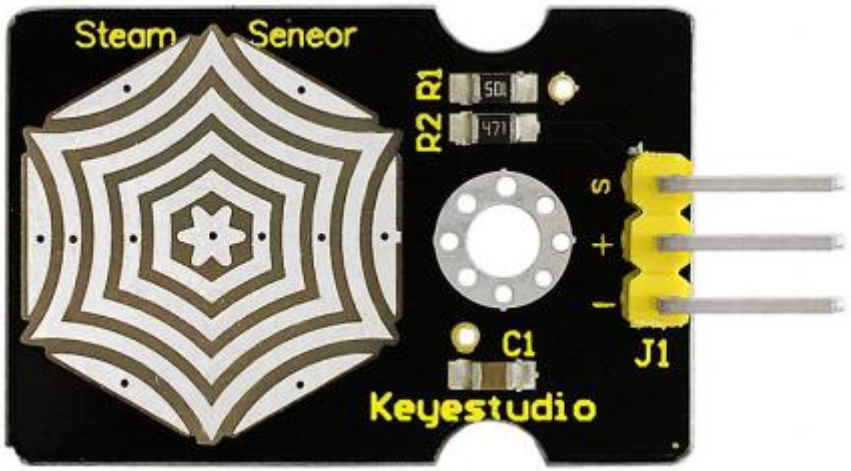


CHAPTER THIRTY-SEVEN

PROJECT 37: VAPOR SENSOR

37.1 Introduction

The vapor sensor is an analog sensor and can be made as a simple rainwater detector and liquid level switch. When the humidity on the surface of this sensor increases, the output voltage will increase. Attention: The connecting parts are not waterproof, so it is better not to place them in water.



37.2 Performance Parameters

1. Working voltage: 3.3V or 5V

2: <20 mA

3.Working temperature range: -10°C~+70°C

4: Mode Type: Analog Signal Output

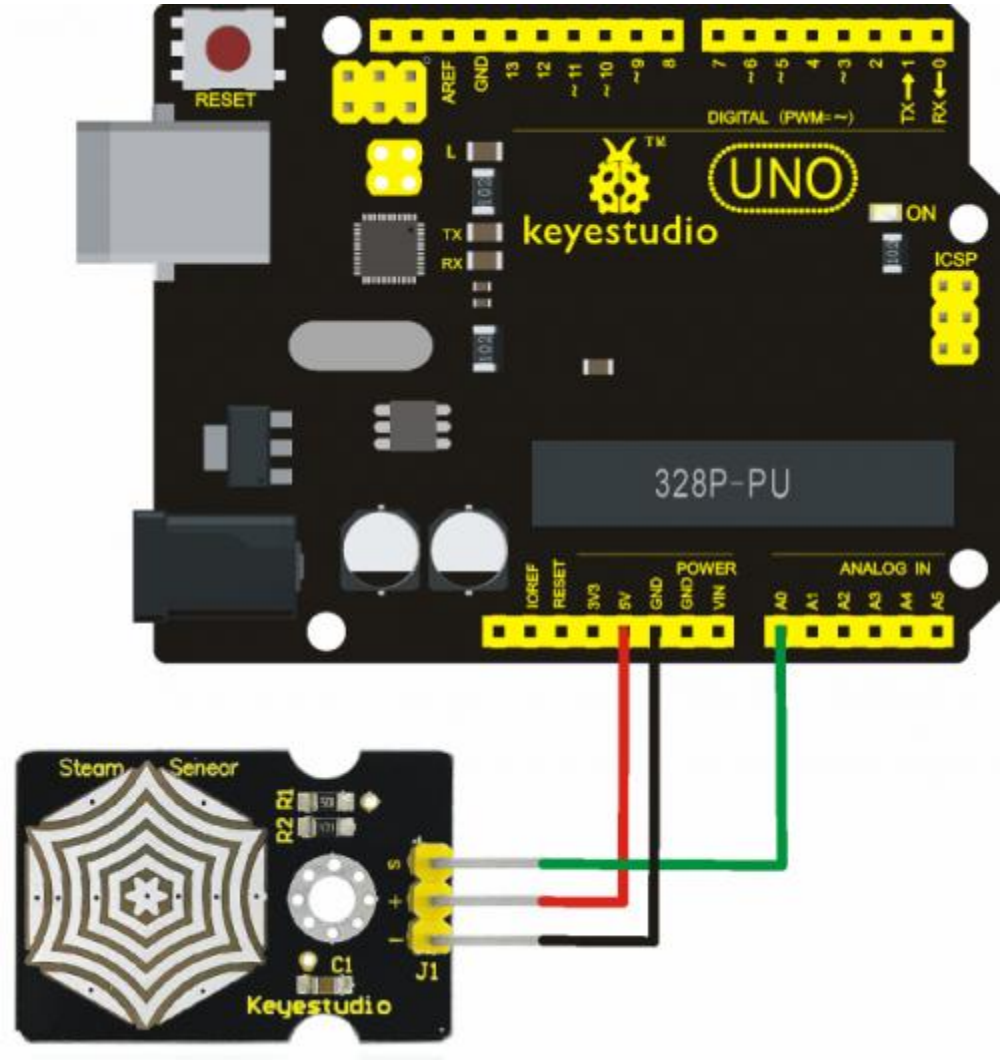
37.3 Defining Terminals

S:Signal Output

+:Power Supply (VCC)

-:Ground (GND)

37.4 Connection diagram



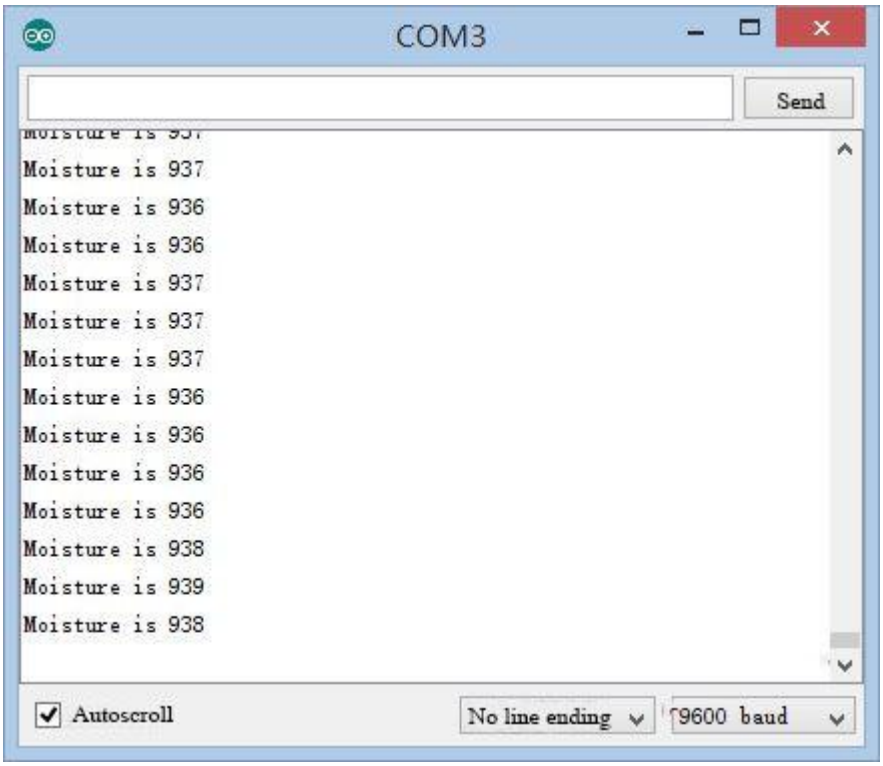
37.5 Test Code

```
void setup()
{
Serial.begin(9600); //open serial port, and set
baud rate at 9600bps
}
void loop()
{
int val;
val=analogRead(0); //plug vapor sensor into analog
port 0
Serial.print("Moisture is ");
Serial.println(val,DEC); //read analog value
through serial port printed
delay(100);
}
```

37.6 Result

When detecting different degrees of humidity, the sensor will get the feedback of different current value. It appears as the following image.

When the sensor detects the steam of boiled water, the humidity value is displayed on the serial display of the Arduino software.



ELECTRONIC SOURCES

Resources

Video:

http://video.keyestudio.com/KS0068/

Libraries:

https://fs.keyestudio.com/KS0068