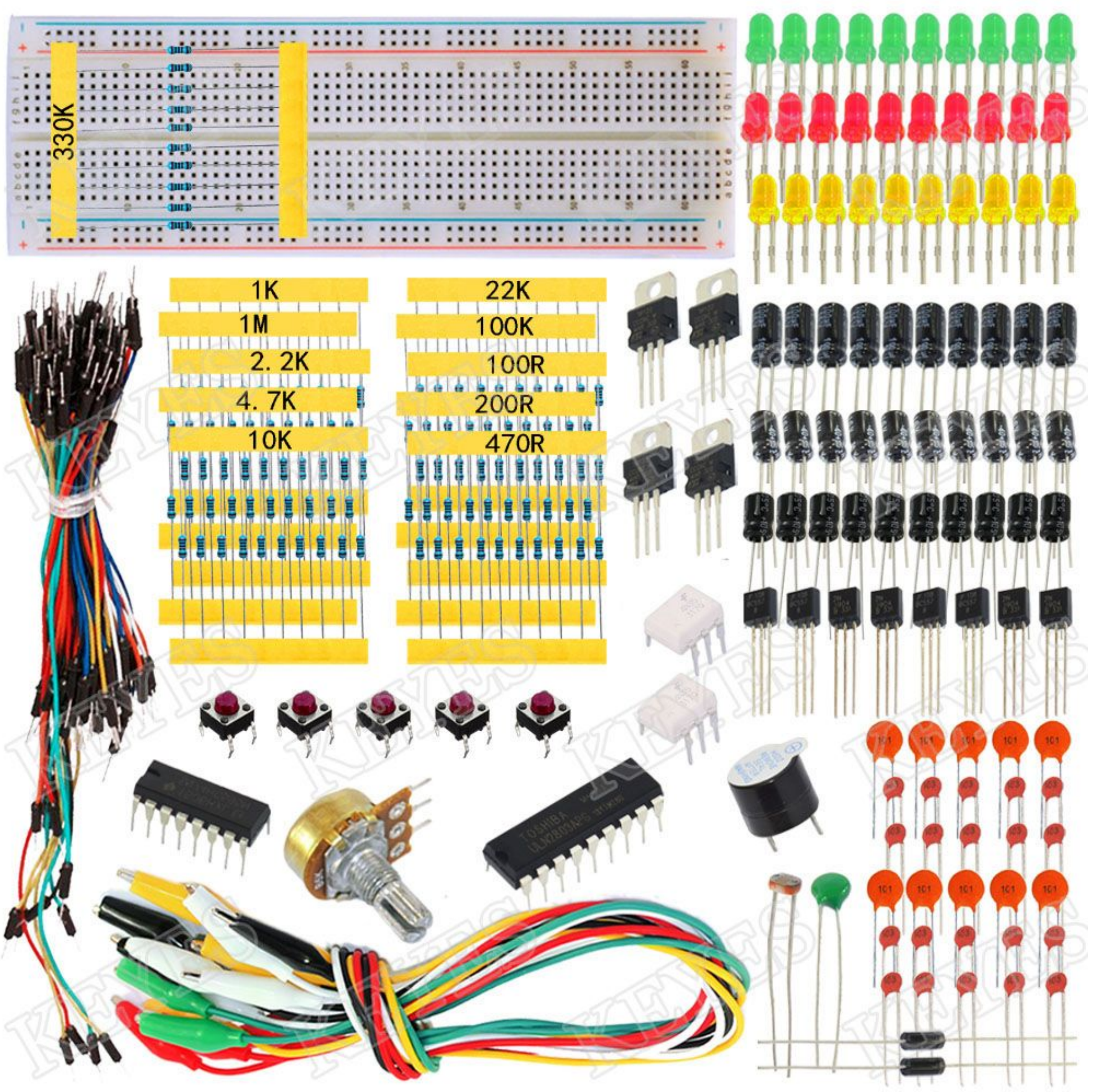
keyes

Arduino 电子元件包套件 含电阻卡



Project 1: Hello World

Introduction

As for starters, we will begin with something simple. In this project, you only need an Arduino and a USB cable to start the "Hello World!" experiment. This is a communication test of your Arduino and PC, also a primer project for you to have your first try of the Arduino world!

Hardware required

Arduino board *1
USB cable *1

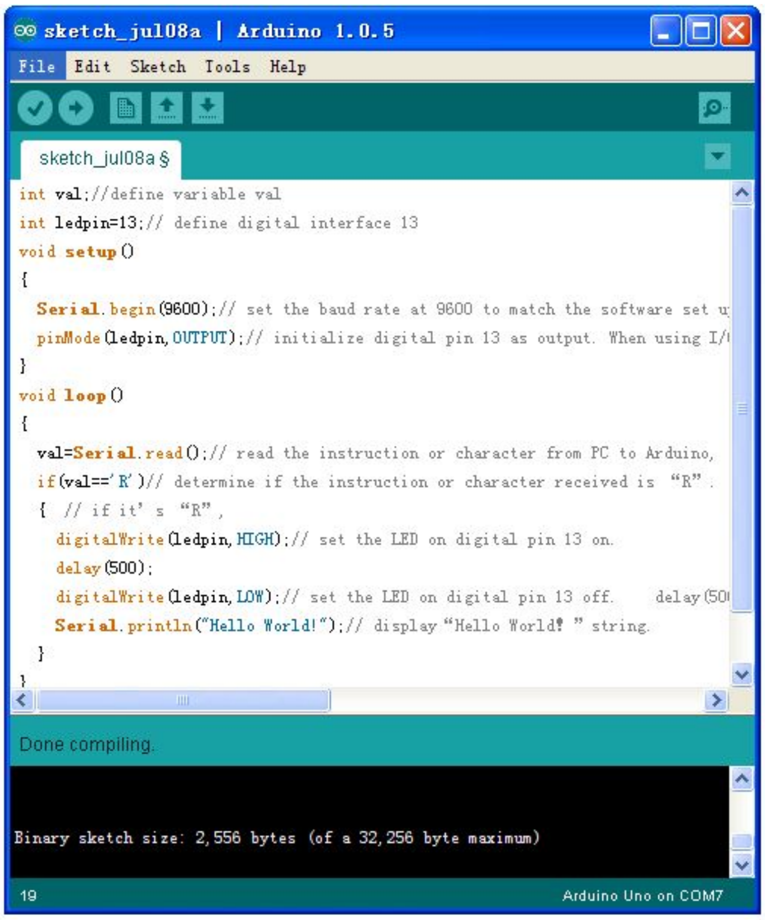
Sample program

After installing driver for Arduino, let's open Arduino software and compile code that enables Arduino to print "Hello World!" under your instruction. Of course, you can compile code for Arduino to continuously echo "Hello World!" without instruction. A simple If () statement will do the instruction trick. With the onboard LED connected to pin 13, we can instruct the LED to blink first when Arduino gets an instruction and then print "Hello World!".

```
//////////////////////////////////////////////////////////
int val;//define variable val
int ledpin=13;// define digital interface 13
void setup()
{
   Serial.begin(9600);// set the baud rate at 9600 to match the software set up. When connected to
a specific device, (e.g. bluetooth), the baud rate needs to be the same with it.
   pinMode(ledpin,OUTPUT);// initialize digital pin 13 as output. When using I/O ports on an
Arduino, this kind of set up is always needed.
}
void loop()
{
   val=Serial.read();// read the instruction or character from PC to Arduino, and assign them to
Val.
   if(val=='R')// determine if the instruction or character received is "R".
   {   // if it's "R",
      digitalWrite(ledpin,HIGH);// set the LED on digital pin 13 on.
      delay(500);
      digitalWrite(ledpin,LOW);// set the LED on digital pin 13 off.       delay(500);
      Serial.println("Hello World!");// display"Hello World！ "string.
   }
}
```

//

Result



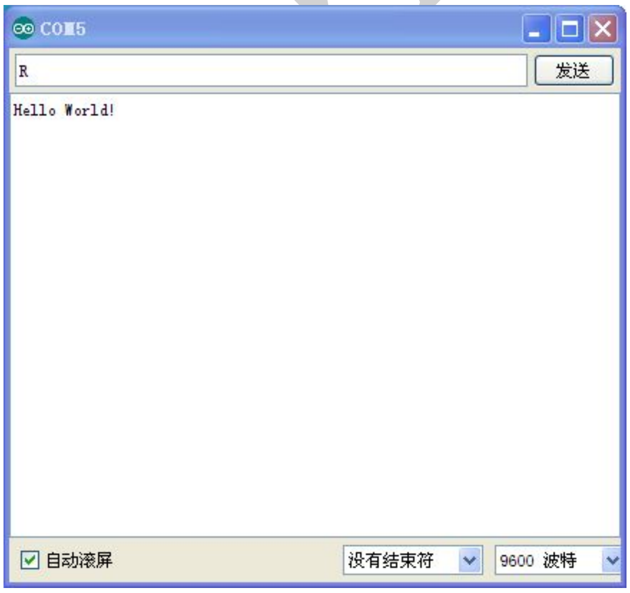
```
int val;//define variable val
int ledpin=13;// define digital interface 13
void setup()
{
  Serial.begin(9600);// set the baud rate at 9600 to match the software set u
  pinMode(ledpin,OUTPUT);// initialize digital pin 13 as output. When using I/
}
void loop()
{
  val=Serial.read();// read the instruction or character from PC to Arduino,
  if(val=='R')// determine if the instruction or character received is “R”.
  {  // if it’s “R”,
    digitalWrite(ledpin,HIGH);// set the LED on digital pin 13 on.
    delay(500);
    digitalWrite(ledpin,LOW);// set the LED on digital pin 13 off.    delay(50
    Serial.println("Hello World!");// display“Hello World! ” string.
  }
}
```

```
Done compiling.

Binary sketch size: 2,556 bytes (of a 32,256 byte maximum)

19                                              Arduino Uno on COM7
```

Click serial port monitor,Input R,LED 13 will blink once,PC will receive information from Arduino: Hello World

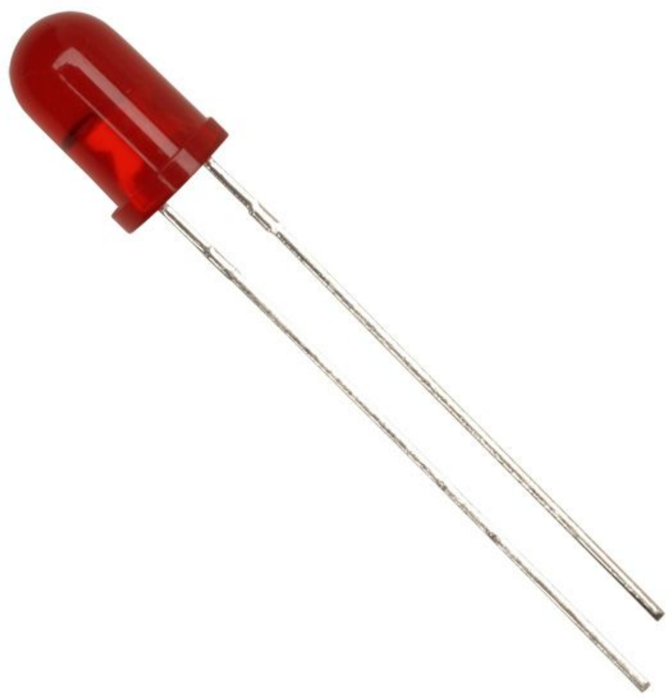


```
R

Hello World!
```

After you choosing the right port， the experiment should be easy for you!

Project 2: LED blinking



Introduction

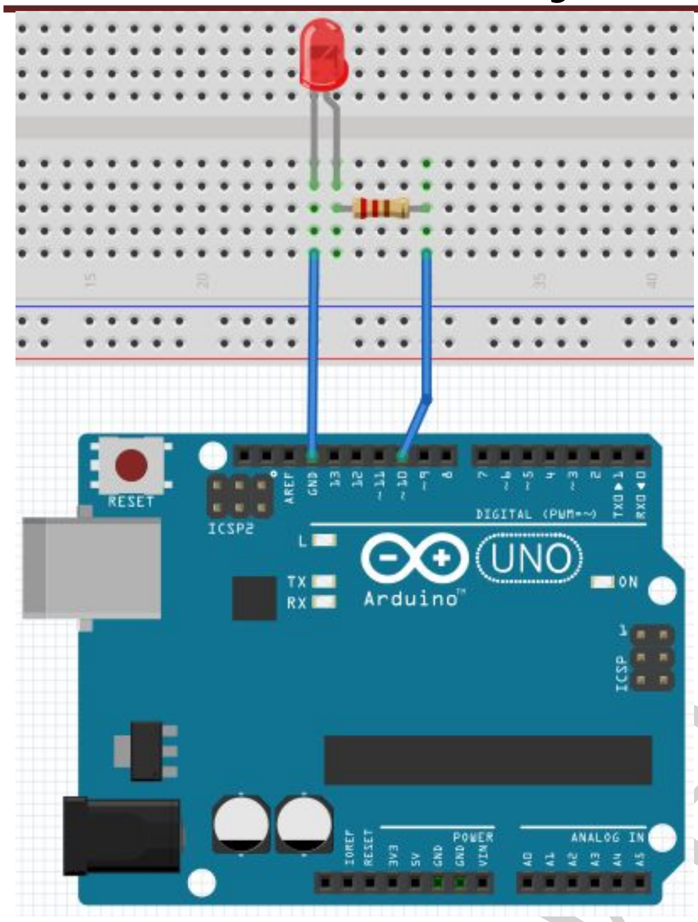
Blinking LED experiment is quite simple. In the "Hello World!" program, we have come across LED. This time, we are going to connect an LED to one of the digital pins rather than using LED13, which is soldered to the board. Except an Arduino and an USB cable, we will need extra parts as below:

Hardware required

Red M5 LED*1
220Ω resistor*1
Breadboard*1
Breadboard jumper wires

Circuit connection

We follow below diagram from the experimental schematic link. Here we use digital pin 10. We connect LED to a 220 ohm resistor to avoid high current damaging the LED.



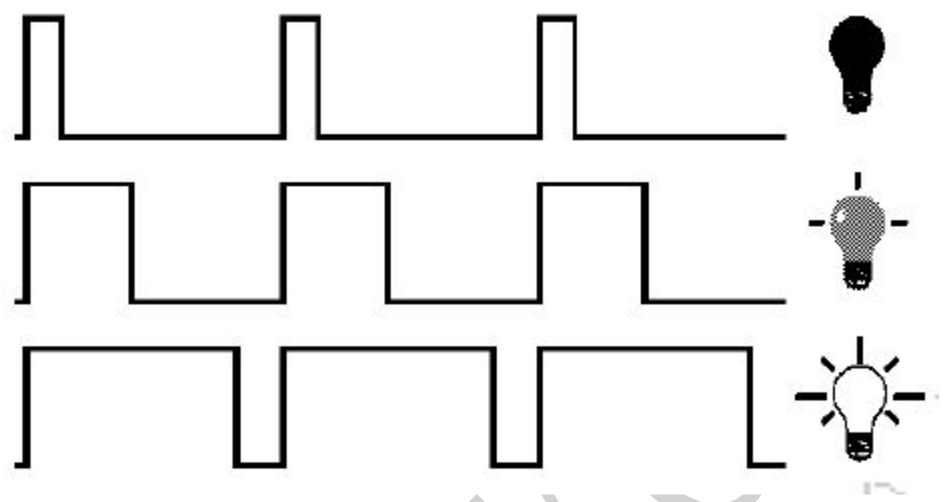
Sample program

```
//////////////////////////////////////////////////////////
int ledPin = 10; // define digital pin 10.
void setup()
{
pinMode(ledPin, OUTPUT);// define pin with LED connected as output.
}
void loop()
{
digitalWrite(ledPin, HIGH); // set the LED on.
delay(1000); // wait for a second.
digitalWrite(ledPin, LOW); // set the LED off.
delay(1000); // wait for a second
}
//////////////////////////////////////////////////////////
```

Result

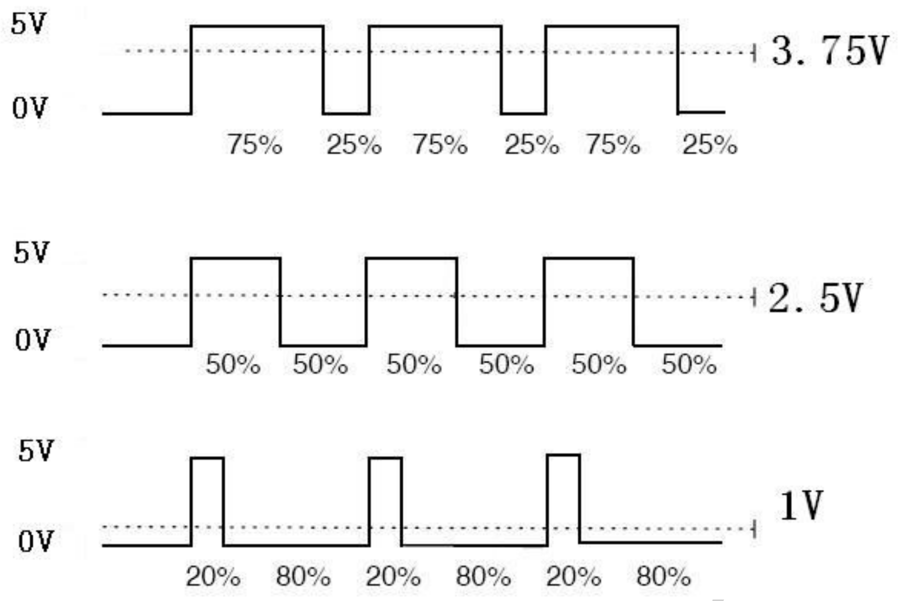
After downloading this program, in the experiment, you will see the LED connected to pin 10 turning on and off, with an interval approximately one second.
The blinking LED experiment is now completed. Thank you!
****

Project 3: PWM light control

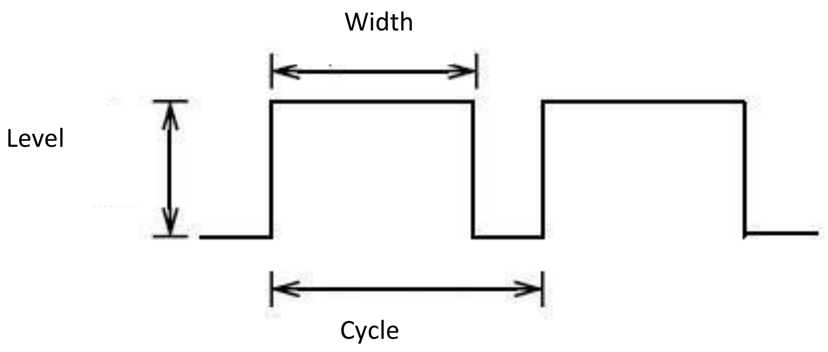


Introduction

PWM, short for Pulse Width Modulation, is a technique used to encode analog signal level into digital ones. A computer cannot output analog voltage but only digital voltage values such as 0V or 5V. So we use a high resolution counter to encode a specific analog signal level by modulating the duty cycle of PMW. The PWM signal is also digitalized because in any given moment, fully on DC power supply is either 5V (ON), or 0V (OFF). The voltage or current is fed to the analog load (the device that uses the power) by repeated pulse sequence being ON or OFF. Being on, the current is fed to the load; being off, it's not. With adequate bandwidth, any analog value can be encoded using PWM. The output voltage value is calculated via the on and off time. Output voltage = (turn on time/pulse time) * maximum voltage value



PWM has many applications: lamp brightness regulating, motor speed regulating, sound making, etc.
The following are the three basic parameters of PMW:



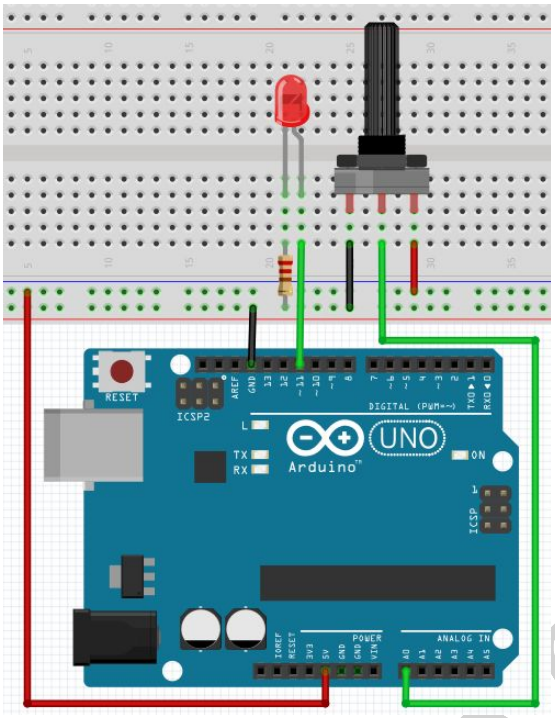
1. The amplitude of pulse width (minimum / maximum)
2. The pulse period (The reciprocal of pulse frequency in 1 second)
3. The voltage level（such as：0V-5V）
There are 6 PMW interfaces on Arduino, namely digital pin 3, 5, 6, 9, 10, and 11. In previous experiments, we have done "button-controlled LED", using digital signal to control digital pin, also one about potentiometer. This time, we will use a potentiometer to control the brightness of the LED.

Hardware required

Potentiometer*1
Red M5 LED*1
220Ω resistor*1
Breadboard*1
Breadboard jumper wires

Circuit connection

The input of potentiometer is analog, so we connect it to analog port, and LED to PWM port. Different PWM signal can regulate the brightness of the LED.

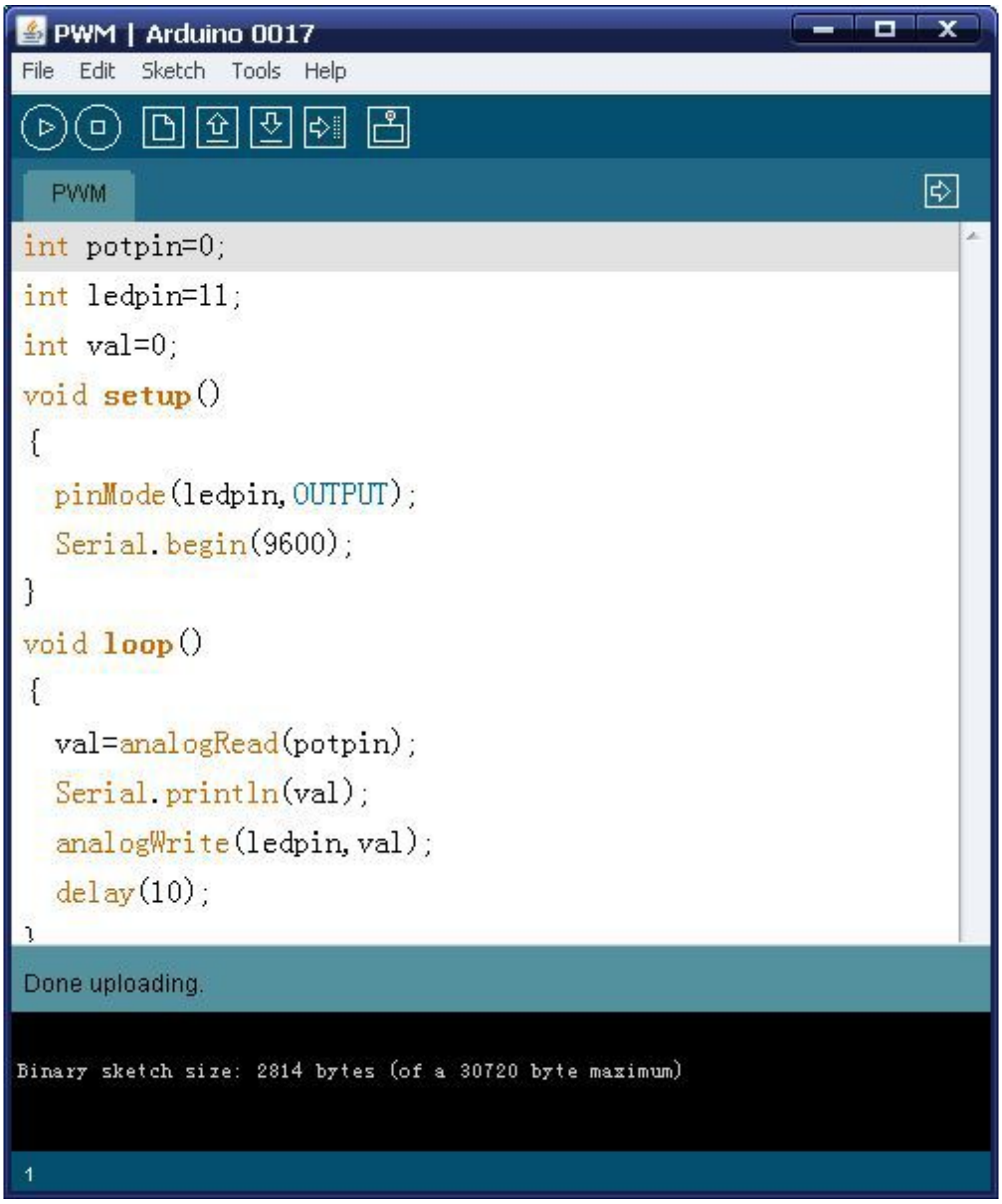


Sample program

In the program compiling process, we will use the analogWrite (PWM interface, analog value) function. In this experiment, we will read the analog value of the potentiometer and assign the value to PWM port, so there will be corresponding change to the brightness of the LED. One final part will be displaying the analog value on the screen. You can consider this as the "analog value reading" project adding the PWM analog value assigning part. Below is a sample program for your reference.

```
////////////////////////////////////////////////////////
int potpin=0;// initialize analog pin 0
int ledpin=11;//initialize digital pin 11（PWM output）
int val=0;// Temporarily store variables' value from the sensor
void setup()
{
pinMode(ledpin,OUTPUT);// define digital pin 11 as “output”
Serial.begin(9600);// set baud rate at 9600
// attention: for analog ports, they are automatically set up as “input”
}
void loop()
{
val=analogRead(potpin);// read the analog value from the sensor and assign it to val
Serial.println(val);// display value of val
```

analogWrite(ledpin,val/4);// turn on LED and set up brightness（maximum output of PWM is 255）
delay(10);// wait for 0.01 second
}
//



```
int potpin=0;
int ledpin=11;
int val=0;
void setup()
{
  pinMode(ledpin,OUTPUT);
  Serial.begin(9600);
}
void loop()
{
  val=analogRead(potpin);
  Serial.println(val);
  analogWrite(ledpin,val);
  delay(10);
}
```

Done uploading.

```
Binary sketch size: 2814 bytes (of a 30720 byte maximum)
```

1

Result

After downloading the program, when we rotate the potentiometer knob, we can see changes of the displaying value, also obvious change of the LED brightness on the breadboard.
****

Project 4: Traffic light

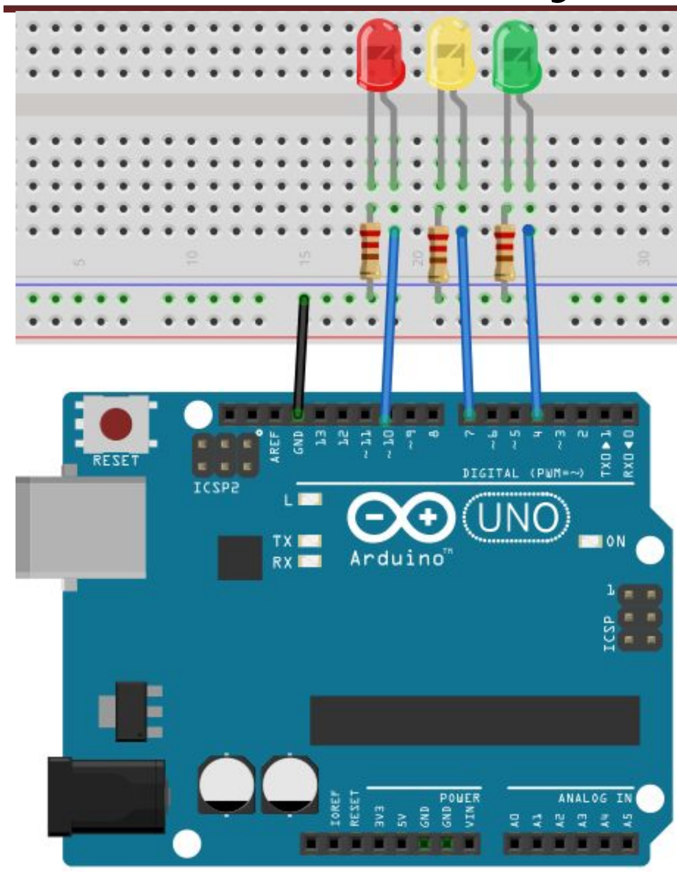
Introduction

In the previous program, we have done the LED blinking experiment with one LED. Now, it's time to up the stakes and do a bit more complicated experiment-traffic lights. Actually, these two experiments are similar. While in this traffic lights experiment, we use 3 LEDs with different color other than 1 LED.

Hardware required

Arduino board *1
USB cable *1
Red M5 LED*1
Yellow M5 LED*1
Green M5 LED*1
220Ω resistor *3
Breadboard*1
Breadboard jumper wires

Circuit connection



Sample program

Since it is a simulation of traffic lights, the blinking time of each LED should be the same with those in traffic lights system. In this program, we use Arduino delay () function to control delay time, which is much simpler than C language.

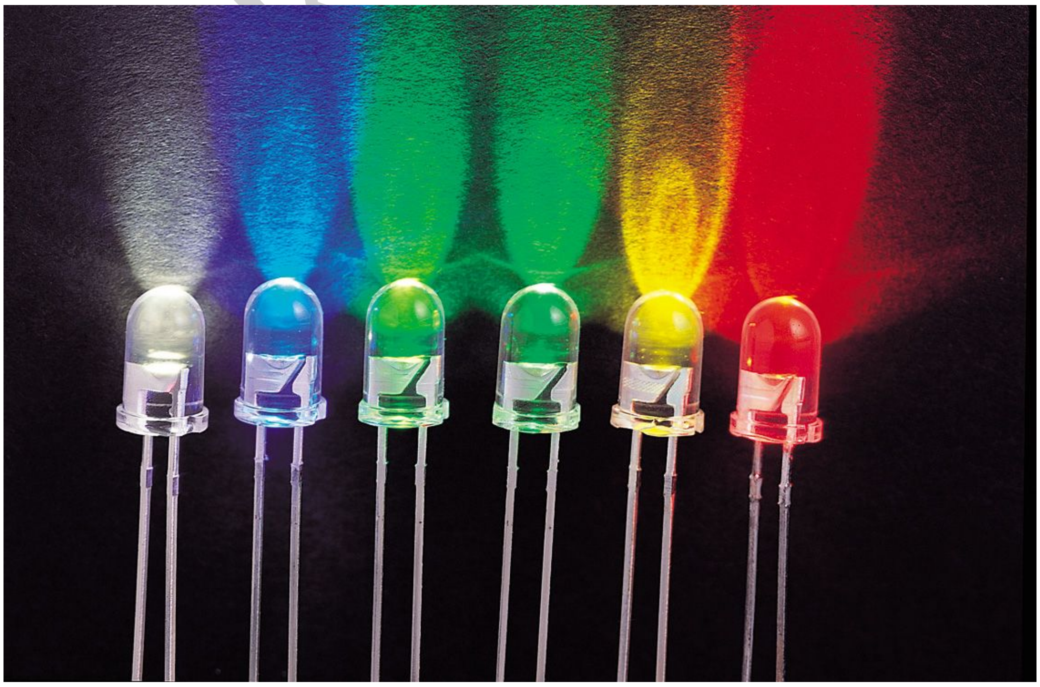
```
////////////////////////////////////////////////////////
int redled =10; // initialize digital pin 8.
int yellowled =7; // initialize digital pin 7.
int greenled =4; // initialize digital pin 4.
void setup()
{
pinMode(redled, OUTPUT);// set the pin with red LED as "output"
pinMode(yellowled, OUTPUT); // set the pin with yellow LED as "output"
pinMode(greenled, OUTPUT); // set the pin with green LED as "output"
}
void loop()
{
digitalWrite(greenled, HIGH);//// turn on green LED
delay(5000);// wait 5 seconds
digitalWrite(greenled, LOW); // turn off green LED
for(int i=0;i<3;i++)// blinks for 3 times
{
```

```
delay(500);// wait 0.5 second
digitalWrite(yellowled, HIGH);// turn on yellow LED
delay(500);// wait 0.5 second
digitalWrite(yellowled, LOW);// turn off yellow LED
}
delay(500);// wait 0.5 second
digitalWrite(redled, HIGH);// turn on red LED
delay(5000);// wait 5 second
digitalWrite(redled, LOW);// turn off red LED
}
////////////////////////////////////////////////////////////
```

Result

When the uploading process is completed, we can see traffic lights of our own design.
Note: this circuit design is very similar with the one in LED chase effect.
The green light will be on for 5 seconds, and then off., followed by the yellow light blinking for 3 times, and then the red light on for 5 seconds, forming a cycle. Cycle then repeats.
Experiment is now completed, thank you.
****

Project 5: LED chasing effect



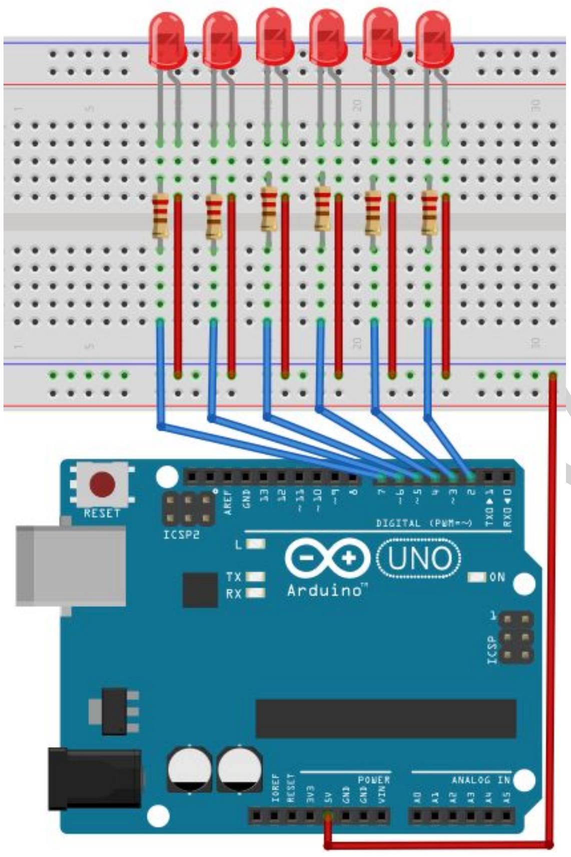
Introduction

We often see billboards composed of colorful LEDs. They are constantly changing to form various effects. In this experiment, we compile a program to simulate chase effect.

Hardware required

Led *6
220Ω resistor *6
Breadboard jumper wires

Circuit connection



Sample program

```
////////////////////////////////////////////////////////
int BASE = 2 ;    // the I/O pin for the first LED
int NUM = 6;      // number of LEDs

void setup()
{
    for (int i = BASE; i < BASE + NUM; i ++)
    {
      pinMode(i, OUTPUT);      // set I/O pins as output
    }
}

void loop()
{
    for (int i = BASE; i < BASE + NUM; i ++)
    {
        digitalWrite(i, LOW);        // set I/O pins as "low", turn off LEDs one by one.
        delay(200);                 // delay
    }
    for (int i = BASE; i < BASE + NUM; i ++)
    {
        digitalWrite(i, HIGH);        // set I/O pins as "high", turn on LEDs one by one
        delay(200);                 // delay
    }
}
////////////////////////////////////////////////////////////
```

Result

You can see the LEDs blink by sequence.
****

Project 6: Button-controlled LED

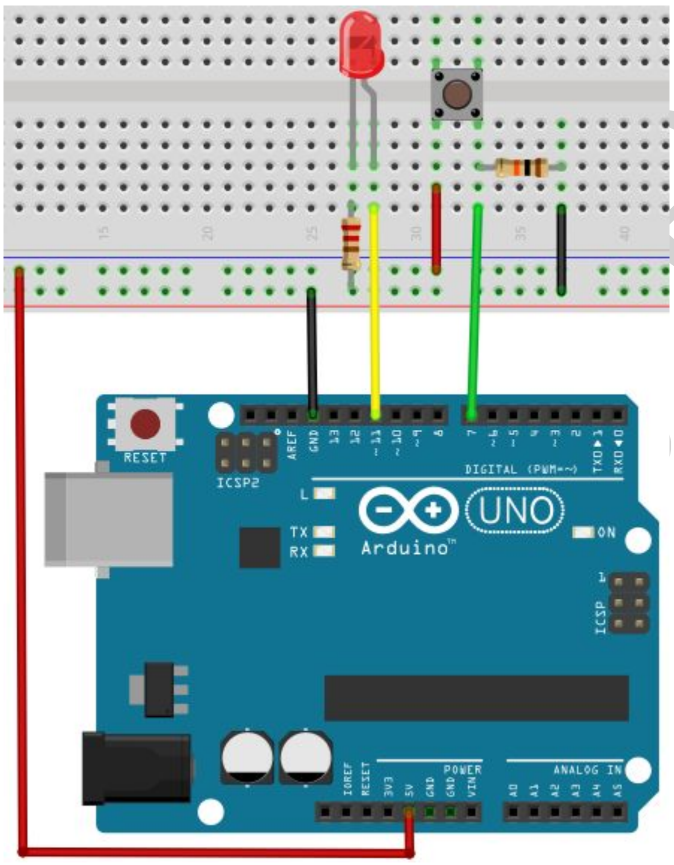
Introduction

I/O port means interface for INPUT and OUTPUT. Up until now, we have only used its OUTPUT function. In this experiment, we will try to use the input function, which is to read the output value of device connecting to it. We use 1 button and 1 LED using both input and output to give you a better understanding of the I/O function. Button switches, familiar to most of us, are a switch value (digital value) component. When it's pressed, the circuit is in closed (conducting) state.

Hardware required

Button switch*1
Red M5 LED*1
220Ω resistor*1
10KΩ resistor*1
Breadboard*1
Breadboard jumper wires

Circuit connection



Sample program

Now, let's begin the compiling. When the button is pressed, the LED will be on. After the previous study, the coding should be easy for you. In this program, we add a statement of judgment. Here, we use an if () statement.
Arduino IDE is based on C language, so statements of C language such as while, switch etc. can certainly be used for Arduino program.

When we press the button, pin 7 will output high level. We can program pin 11 to output high level and turn on the LED. When pin 7 outputs low level, pin 11 also outputs low level and the LED remains off.

```
////////////////////////////////////////////////////////
int ledpin=11;// initialize pin 11
int inpin=7;// initialize pin 7
int val;// define val
void setup()
{
pinMode(ledpin,OUTPUT);// set LED pin as “output”
pinMode(inpin,INPUT);// set button pin as “input”
}
void loop()
{
val=digitalRead(inpin);// read the level value of pin 7 and assign if to val
if(val==LOW)// check if the button is pressed, if yes, turn on the LED
{ digitalWrite(ledpin,LOW);}
else
{ digitalWrite(ledpin,HIGH);}
}
////////////////////////////////////////////////////////
```

Result

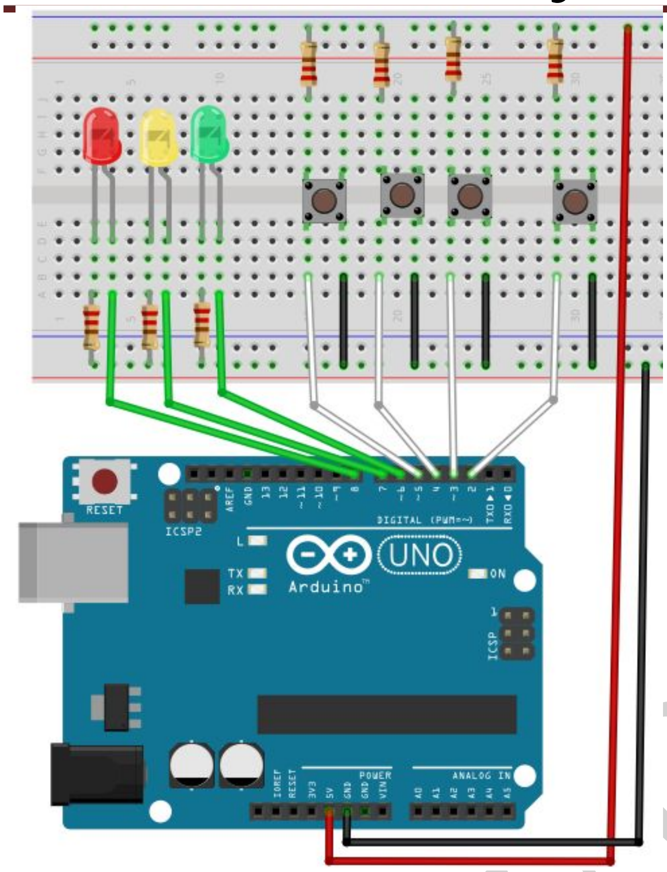
When the button is pressed, LED is on, otherwise, LED remains off. After the above process, the button controlled LED experiment is completed. The simple principle of this experiment is widely used in a variety of circuit and electric appliances. You can easily come across it in your every day life. One typical example is when you press a certain key of your phone, the backlight will be on.
****

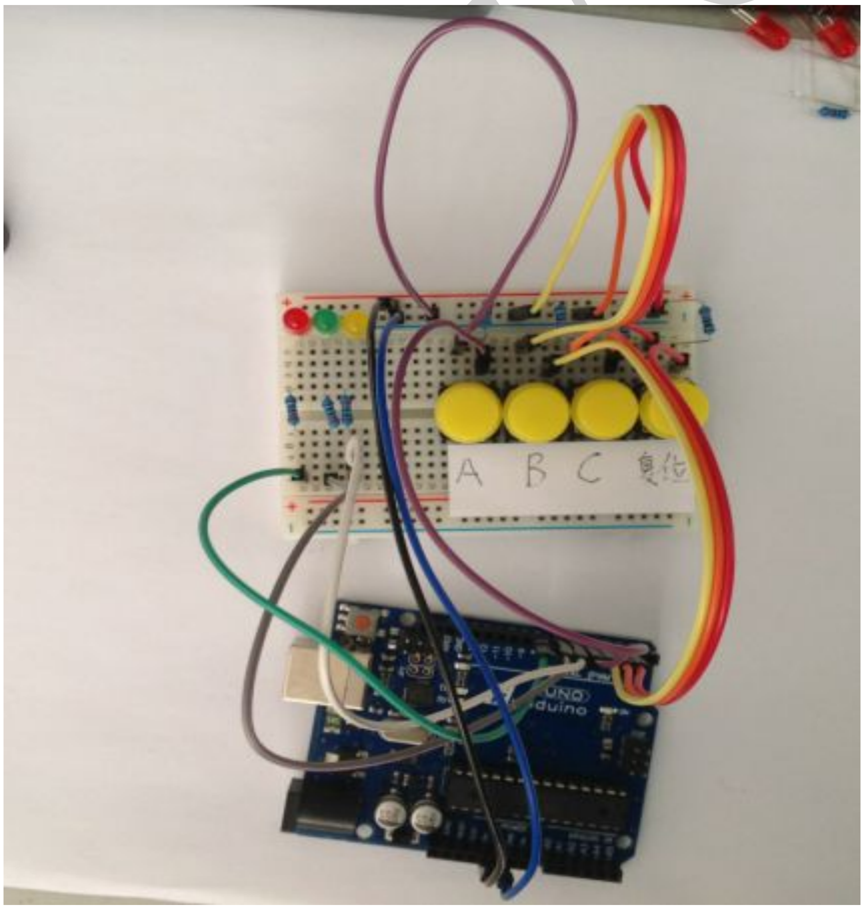
Project 7: Responder experiment

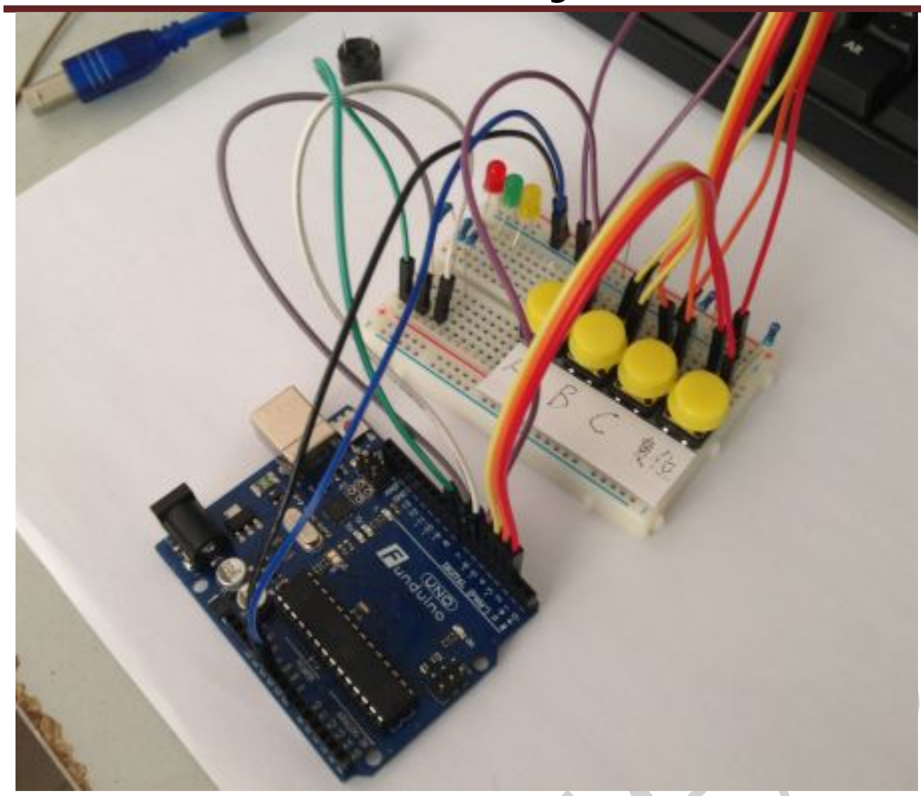
Introduction

After completing all the previous experiments, we believe you will find this one easy. In this program, we have 3 buttons and a reset button controlling the corresponding 3 LEDs, using 7 digital I/O pins.

Circuit connection


RESET
ICSP2
AREF
GND
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
TX▶1
RX◀0
DIGITAL (PWM=~)
L
TX
RX
UNO
Arduino™
ON
ICSP
1
IOREF
RESET
3V3
5V
GND
GND
VIN
POWER
ANALOG IN
A0
A1
A2
A3
A4
A5


A
B
C
复位



Sample program

```
/////////////////////////////////////////////////////////////
int redled=8;          // set red LED as "output"
int yellowled=7;    // set yellow LED as "output"
int greenled=6;      // set green LED as "output"
int redpin=5;          // initialize pin for red button
int yellowpin=4;    // initialize pin for yellow button
int greenpin=3;      // initialize pin for green button
int restpin=2;      // initialize pin for reset button
int red;
int yellow;
int green;
void setup()
{
pinMode(redled,OUTPUT);
pinMode(yellowled,OUTPUT);
pinMode(greenled,OUTPUT);
pinMode(redpin,INPUT);
pinMode(yellowpin,INPUT);
pinMode(greenpin,INPUT);
}
```

```
void loop()    // repeatedly read pins for buttons
{
red=digitalRead(redpin);
yellow=digitalRead(yellowpin);
green=digitalRead(greenpin);
if(red==LOW)RED_YES();
if(yellow==LOW)YELLOW_YES();
if(green==LOW)GREEN_YES();
}

void RED_YES()// execute the code until red light is on; end cycle when reset button is pressed
{
   while(digitalRead(restpin)==1)
   {
     digitalWrite(redled,HIGH);
     digitalWrite(greenled,LOW);
     digitalWrite(yellowled,LOW);
   }
   clear_led();
}
void YELLOW_YES()// execute the code until yellow light is on; end cycle when reset button is
pressed
{
   while(digitalRead(restpin)==1)
   {
   digitalWrite(redled,LOW);
   digitalWrite(greenled,LOW);
   digitalWrite(yellowled,HIGH);
   }
   clear_led();
}
void GREEN_YES()// execute the code until green light is on; end cycle when reset button is
pressed
{
   while(digitalRead(restpin)==1)
   {
   digitalWrite(redled,LOW);
   digitalWrite(greenled,HIGH);
   digitalWrite(yellowled,LOW);
   }
   clear_led();
}
void clear_led()// all LED off
```

```
{
   digitalWrite(redled,LOW);
   digitalWrite(greenled,LOW);
   digitalWrite(yellowled,LOW);
}
////////////////////////////////////////////////////////////
```

Result

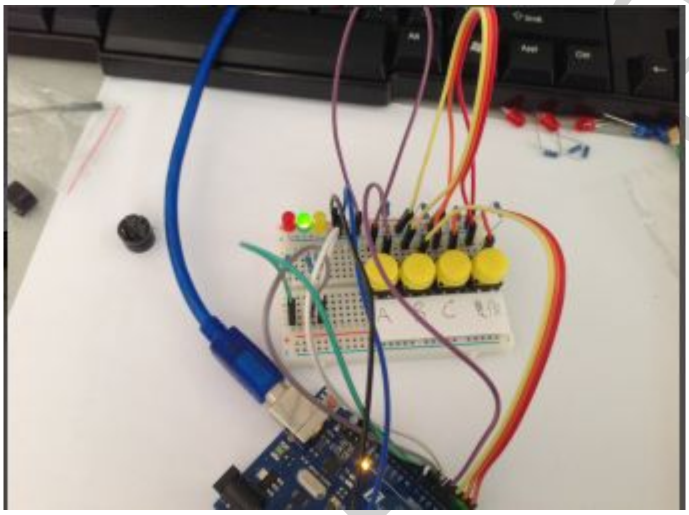
Whichever button is pressed first, the corresponding LED will be on!

Then press the REST button to reset.

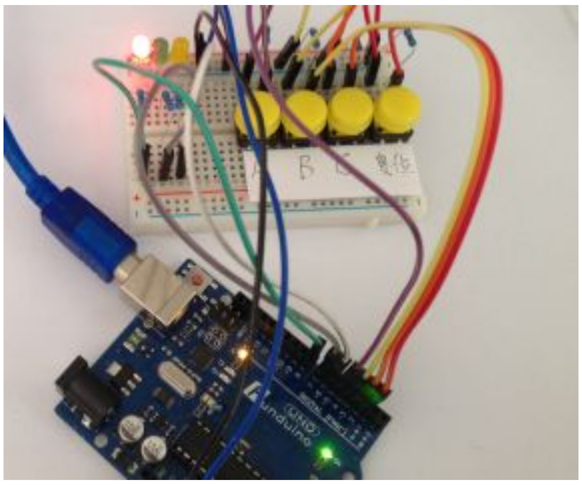
After the above process, we have built our own simple responder.

Screen shot:

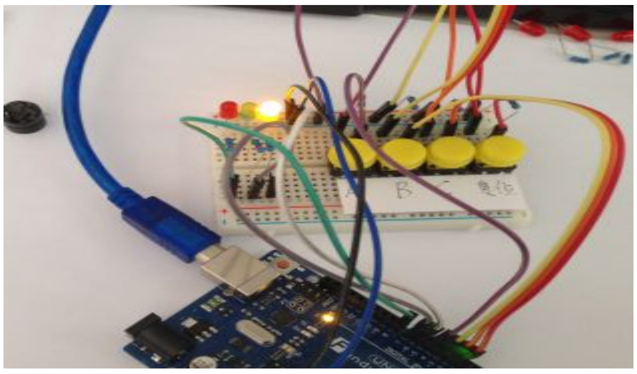
A Response successfully：Green light on



B Response successfully：Red light on



C Response successfully：Yellow light on



Project 8: Active buzzer

Introduction

Active buzzer is widely used on computer, printer, alarm, electronic toy, telephone, timer etc as a sound making element. It has an inner vibration source. Simply connect it with 5V power supply, it can buzz continuously.

Hardware required

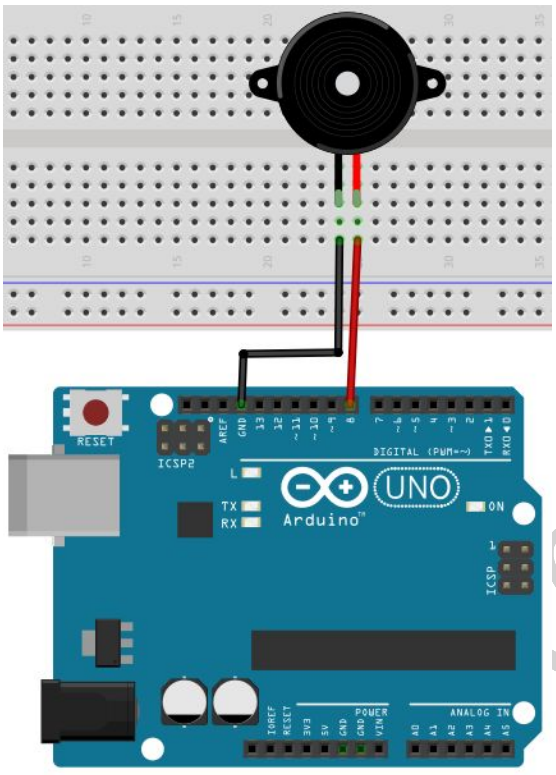
Buzzer*1

Key *1

Breadboard*1

Breadboard jumper wires

Circuit connection



When connecting the circuit, pay attention to the positive & the negative poles of the buzzer. In the photo, you can see there are red and black lines. When the circuit is finished, you can begin programming.

Sample program

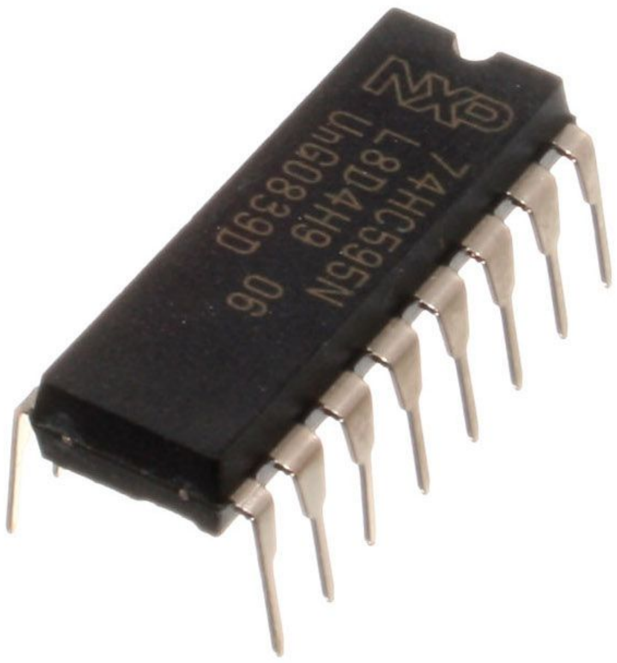
Program is simple. You control the buzzer by outputting high/low level.

```
////////////////////////////////////////////////////////
int buzzer=8;// initialize digital IO pin that controls the buzzer
void setup()
{
   pinMode(buzzer,OUTPUT);// set pin mode as "output"
}
void loop()
{
digitalWrite(buzzer, HIGH); // produce sound
}
////////////////////////////////////////////////////////
```

Result

After downloading the program, the buzzer experiment is completed. You can see the buzzer is ringing.
****

Project 9: 74HC595



Introduction

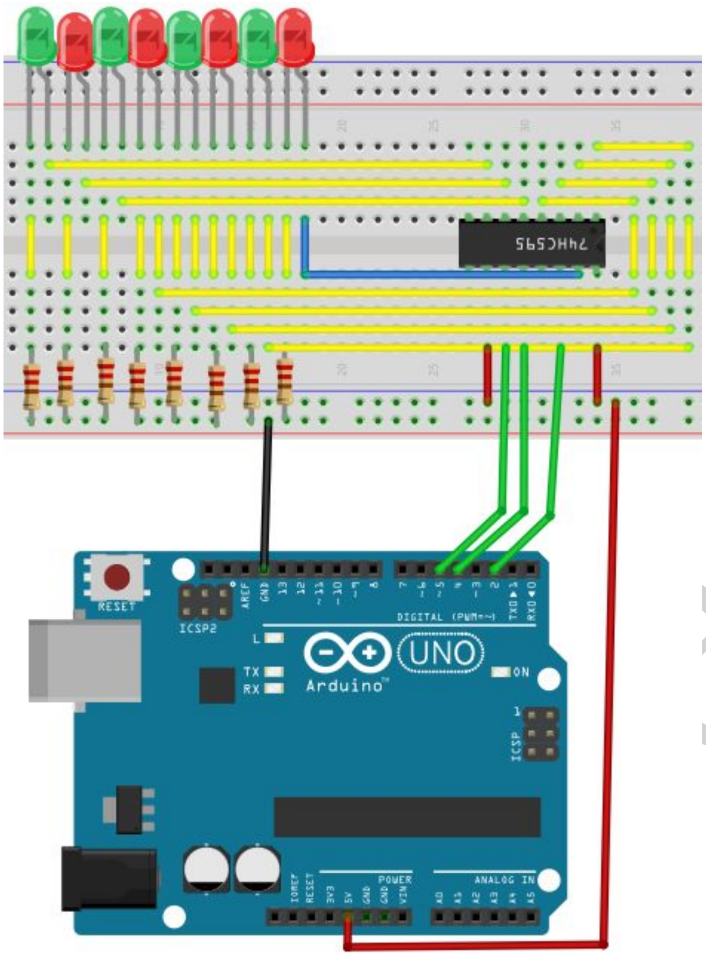
To put it simply, 74HC595 is a combination of 8-digit shifting register, memorizer and equipped with tri-state output. Here, we use it to control 8 LEDs. You may wonder why use a 74HC595 to control LED? Well, think about how many I/O it takes for an Arduino to control 8 LEDs? Yes, 8. For an Arduino 168, it has only 20 I/O including analog ports. So, to save port resources, we use 74HC595 to reduce the number of ports it needs. Using 74HC595 enables us to use 3 digital I/O port to control 8 LEDs!

Hardware required

74HC595 chip*1
Red M5 LED*4
Green M5 LED*4
220Ω resistor*8
Breadboard*1
Breadboard jumper wires

Note: for pin 13 OE port of 74HC595, it needs to be connected to GND

Circuit connection



The circuit may seem complicated, but once you give it a good look, you will find it easy!

Sample program

```
////////////////////////////////////////////////////////
int data = 2;// set pin 14 of 74HC595as data input pin SI
int clock = 5;// set pin 11 of 74hc595 as clock pin SCK
int latch = 4;// set pin 12 of 74hc595 as output latch RCK
int ledState = 0;
const int ON = HIGH;
const int OFF = LOW;
void setup()
{
pinMode(data, OUTPUT);
pinMode(clock, OUTPUT);
pinMode(latch, OUTPUT);
}
```

```
void loop()
{
for(int i = 0; i < 256; i++)
{
updateLEDs(i);
delay(500);
}
}
void updateLEDs(int value)
{
digitalWrite(latch, LOW);//
shiftOut(data, clock, MSBFIRST, ~value);// serial data “output”, high level first
digitalWrite(latch, HIGH);// latch
}
//////////////////////////////////////////////////////////////
```

Result

After downloading the program, you can see 8 LEDs displaying 8-bit binary number.
****

Project 10: Analog value reading

Introduction

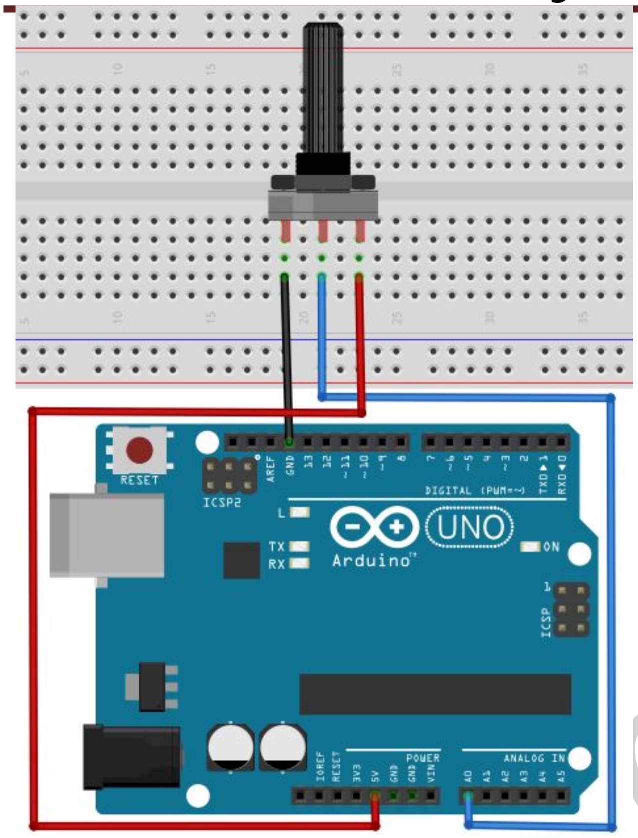
In this experiment, we will begin the learning of analog I/O interfaces. On an Arduino, there are 6 analog interfaces numbered from 0 to 5. These 6 interfaces can also be used as digital ones numbered as 14-19. After a brief introduction, let's begin our project. Potentiometer used here is a typical output component of analog value that is familiar to us.

Hardware required

Potentiometer *1
Breadboard*1
Breadboard jumper wires * several

Circuit connection

In this experiment, we will convert the resistance value of the potentiometer to analog ones and display it on the screen. This is an application we need to master well for our future experiments. Connection circuit as below:



We use the analog interface 0.
The analog interface we use here is interface 0.

Sample program

The program compiling is simple. An analogRead () Statement can read the value of the interface. The A/D acquisition of Arduino 328 is in 10 bits, so the value it reads is among 0 to 1023. One difficulty in this project is to display the value on the screen, which is actually easy to learn. First, we need to set the baud rate in voidsetup (). Displaying the value is a communication between Arduino and PC, so the baud rate of the Arduino should match the the one in the PC's software set up. Otherwise, the display will be messy codes or no display at all. In the lower right corner of the Arduino software monitor window, there is a button for baud rate set up. The set up here needs to match the one in the program. The statement in the program is Serial.begin(); enclosed is the baud rate value, followed by statement for displaying. You can either use Serial.print() or Serial.println() statement.

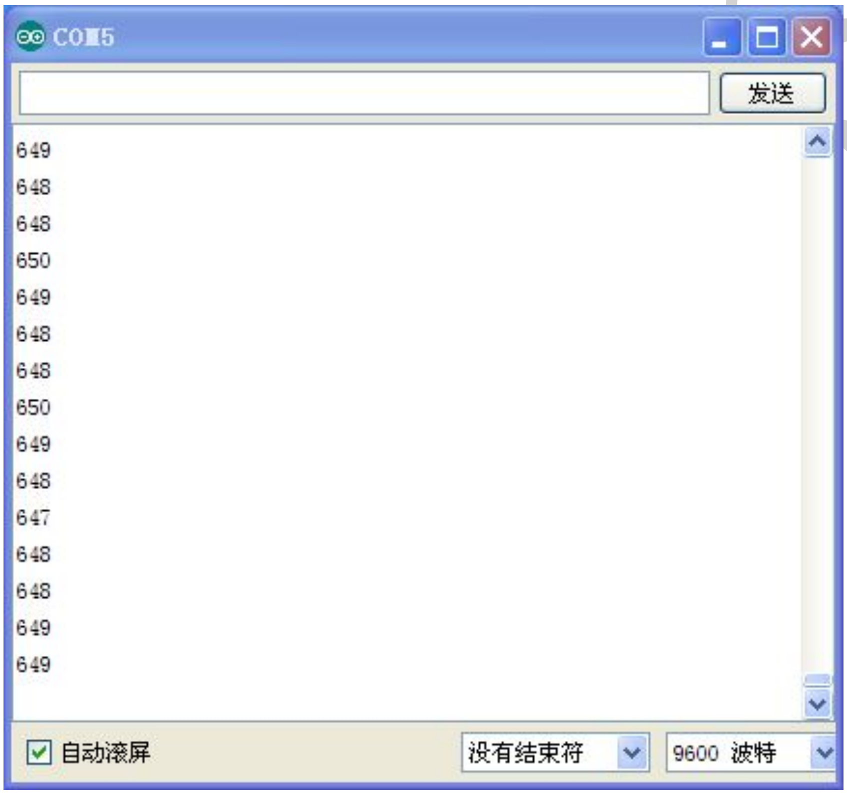
```
//////////////////////////////////////////////////////////
int potpin=0;// initialize analog pin 0
int ledpin=13;// initialize digital pin 13
int val=0;// define val, assign initial value 0
void setup()
{
pinMode(ledpin,OUTPUT);// set digital pin as “output”
Serial.begin(9600);// set baud rate at 9600
}
```

```
void loop()
{
digitalWrite(ledpin,HIGH);// turn on the LED on pin 13
delay(50);// wait for 0.05 second
digitalWrite(ledpin,LOW);// turn off the LED on pin 13
delay(50);// wait for 0.05 second
val=analogRead(potpin);// read the analog value of analog pin 0, and assign it to val
Serial.println(val);// display val's value
}
////////////////////////////////////////////////////////
```

Result

The sample program uses the built-in LED connected to pin 13. Each time the device reads a value, the LED blinks.

Below is the analog value it reads.



```
649
648
648
650
649
648
648
650
649
648
647
648
648
649
649
```

When you rotate the potentiometer knob, you can see the displayed value changes. The reading of analog value is a very common function since most sensors output analog value. After calculation, we can have the corresponding value we need.

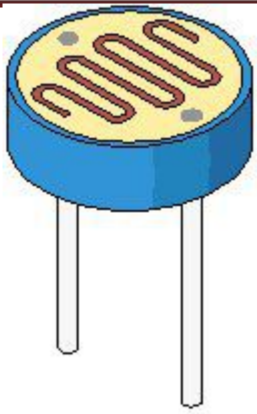
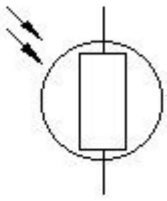
The experiment is now completed, thank you.

****

Project 11: Photo resistor

Introduction

After completing all the previous experiments, we acquired some basic understanding and knowledge about Arduino application. We have learned digital input and output, analog input and PWM. Now, we can begin the learning of sensors applications.
Photo resistor (Photovaristor) is a resistor whose resistance varies according to different incident light strength. It's made based on the photoelectric effect of semiconductor. If the incident light is intense, its resistance reduces; if the incident light is weak, the resistance increases. Photovaristor is commonly applied in the measurement of light, light control and photovoltaic conversion (convert the change of light into the change of electricity).
Photo resistor is also being widely applied to various light control circuit, such as light control and adjustment, optical switches etc.

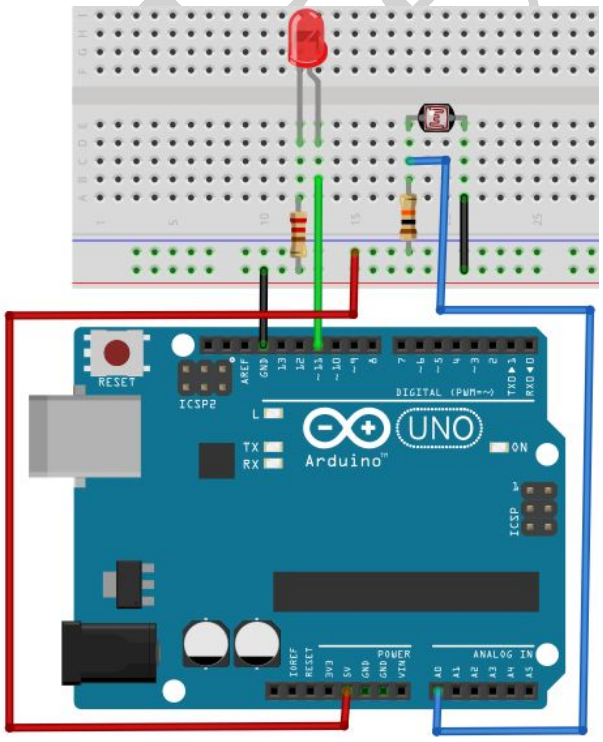



We will start with a relatively simple experiment regarding photovaristor application. Photovaristor is an element that changes its resistance as light strenth changes. So we will need to read the analog values. We can refer to the PWM experiment, replacing the potentiometer with photovaristor. When there is change in light strength, there will be corresponding change on the LED.

Hardware required

Photo resistor*1
Red M5 LED*1
10KΩresistor*1
220Ωresistor*1
Bread board*1
Bread board jumper wires

Circuit connection



Sample program

After the connection, let's begin the program compiling. The program is similar to the one of PWM. For change detail, please refer to the sample program below.

```
//////////////////////////////////////////////////////////
int potpin=0;// initialize analog pin 0, connected with photovaristor
int ledpin=11;// initialize digital pin 11, output regulating the brightness of LED
int val=0;// initialize variable va
void setup()
{
pinMode(ledpin,OUTPUT);// set digital pin 11 as “output”
Serial.begin(9600);// set baud rate at “9600”
}
void loop()
{
val=analogRead(potpin);// read the analog value of the sensor and assign it to val
Serial.println(val);// display the value of val
analogWrite(ledpin,val);// turn on the LED and set up brightness（maximum output value 255）
delay(10);// wait for 0.01
}
//////////////////////////////////////////////////////////
```

Result

After downloading the program, you can change the light strength around the photovaristor and see corresponding brightness change of the LED. Photovaristors has various applications in our everyday life. You can make other interesting interactive projects base on this one.
****

Project 12: Analog temperature (thermistor)

Introduction

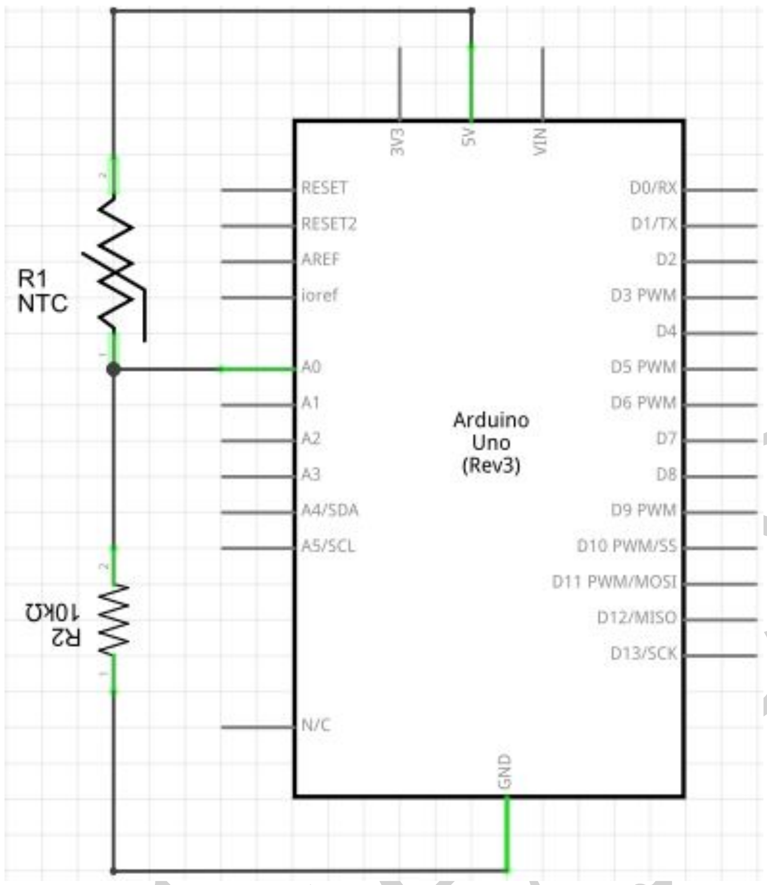
Thermistor is a temperature measuring component base on the principle that a conductor changes in resistance with a change in its body temperature. As a result, it requires the temperature coefficient and the resistivity of the conductor to be as large and stable as possible. It is best that the resistance is in linear relationship with temperature. And it should also have stable physical and chemical properties in a wide range. Currently, the most used thermal resistance materials are platinum, nickel and copper.

Hardware required

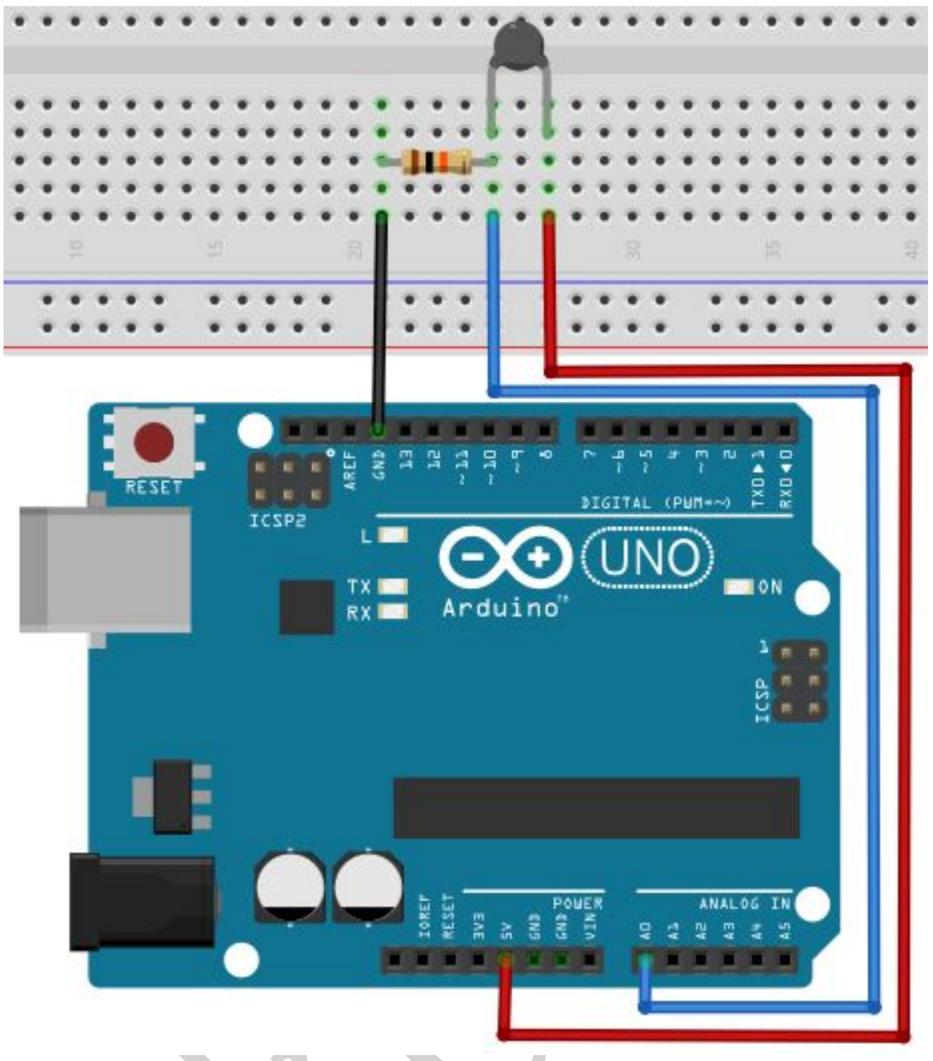
Thermistor *1
10KΩ resistor *1
Breadboard *1

Breadboard jumper wires * several

Schematic diagram



Circuit connection



Sample program

```
/////////////////////////////////////////////////////////
int pin = 7;    //attach to the third pin of NE555
unsigned long duration;    //the variable to store the length of the pulse

void setup()

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

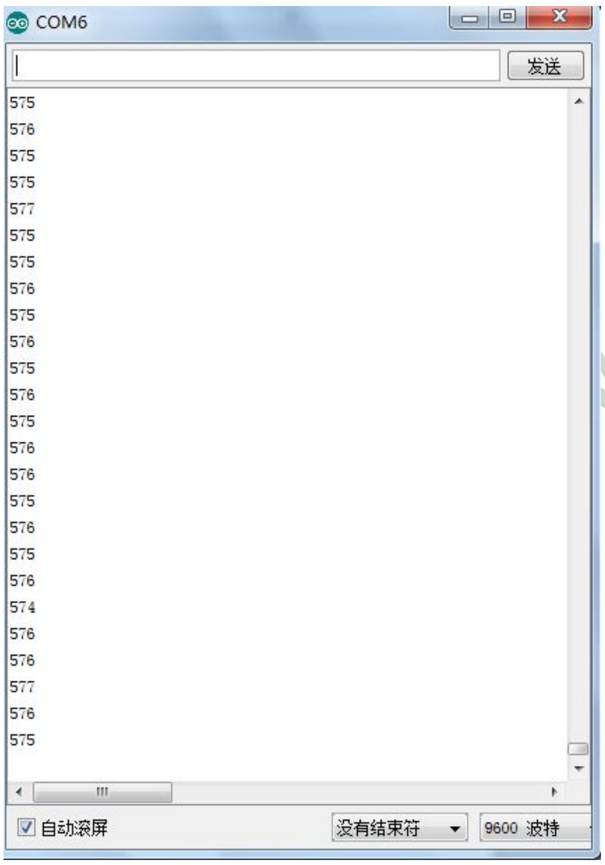
delay(100);

}

//

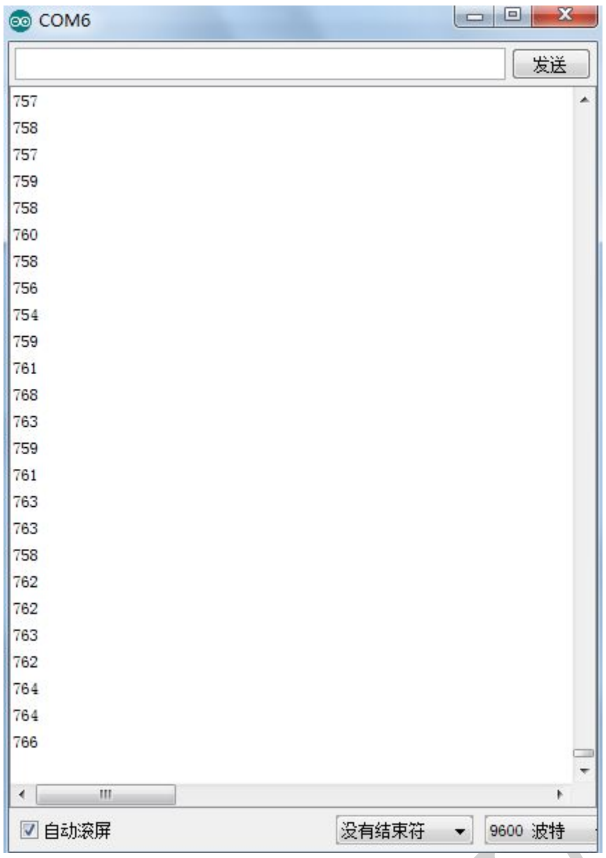
Result

Shown in pic 1 is data displayed by serial port monitor in room temperature. After the temperature is changed (thermistor wrapped and put into hot water), the data changes as shown in pic 2.



```
575
576
575
575
577
575
575
576
575
576
575
576
575
576
576
575
576
575
576
574
576
576
577
576
575
```

Pic 1



```
757
758
757
759
758
760
758
756
754
759
761
768
763
759
761
763
763
758
762
762
763
762
764
764
766
```

Pic 2
****

Project 13: 4N35

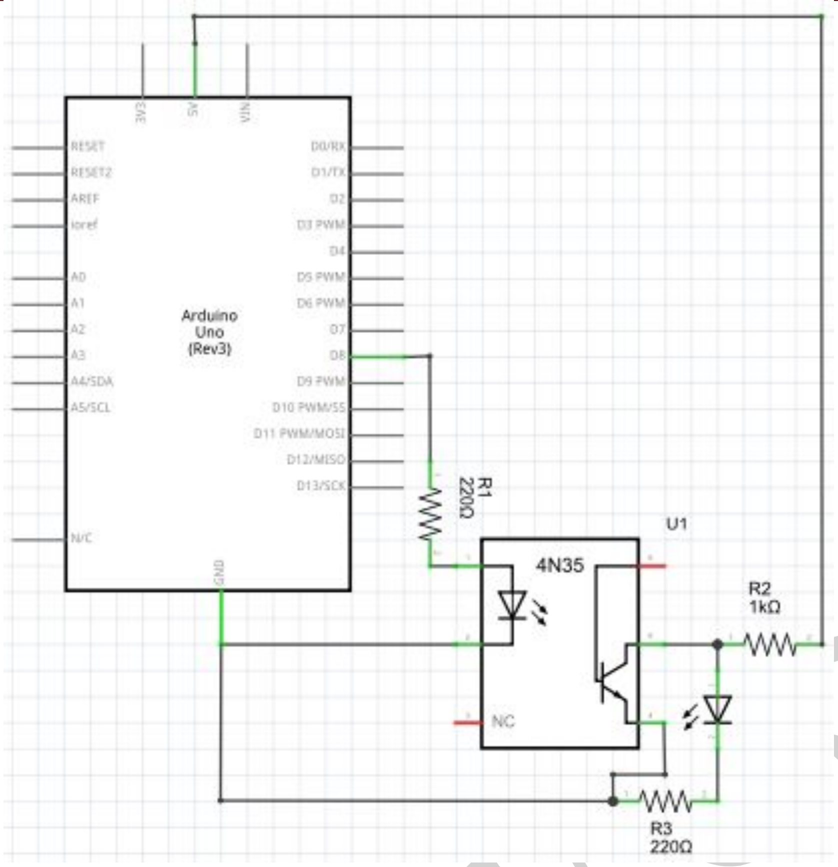
Introduction

4N35 is a general photoelectric coupler, containing a gallium arsenide infrared LED, and use the LED to drive silicon photoelectric transistor diode. The package of 4N35 is 6-pin dual-in-line Package.

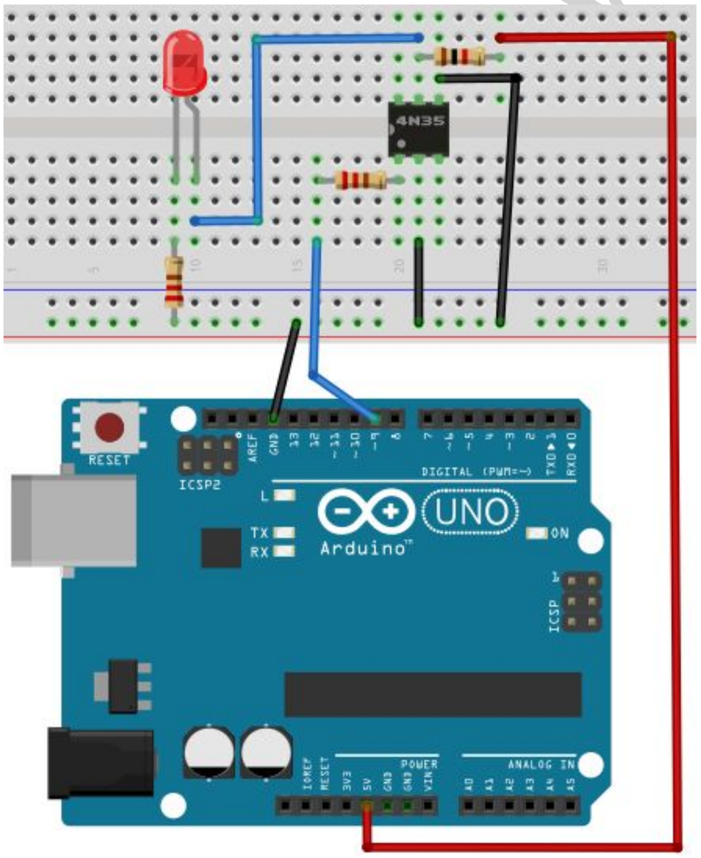
Hardware required

4n35 *1
1KΩ resistor *1
220Ω resistor *2
Red M5 LED *1
Breadboard *1
Breadboard jumper wires *several

Schematic diagram



Circuit connection



Sample program

```
////////////////////////////////////////////////////////////

int Optocoupler=8;

void setup()

{

  pinMode(Optocoupler,OUTPUT);

}


void loop()

{

    digitalWrite(Optocoupler,LOW);

    delay(1000);

    digitalWrite(Optocoupler,HIGH);

    delay(1000);

}

////////////////////////////////////////////////////////////
```

Result

Red light blinks, on for 1 second and off for 1 second.
****