Adjust the brightness of LED light_Jetson

1.Experiment purpose

Use the pin output pwm of jetson NANO to control the brightness of LED light, and also achieve the effect of breathing light.

2.Experiment preparation

Jetson NANO board *1

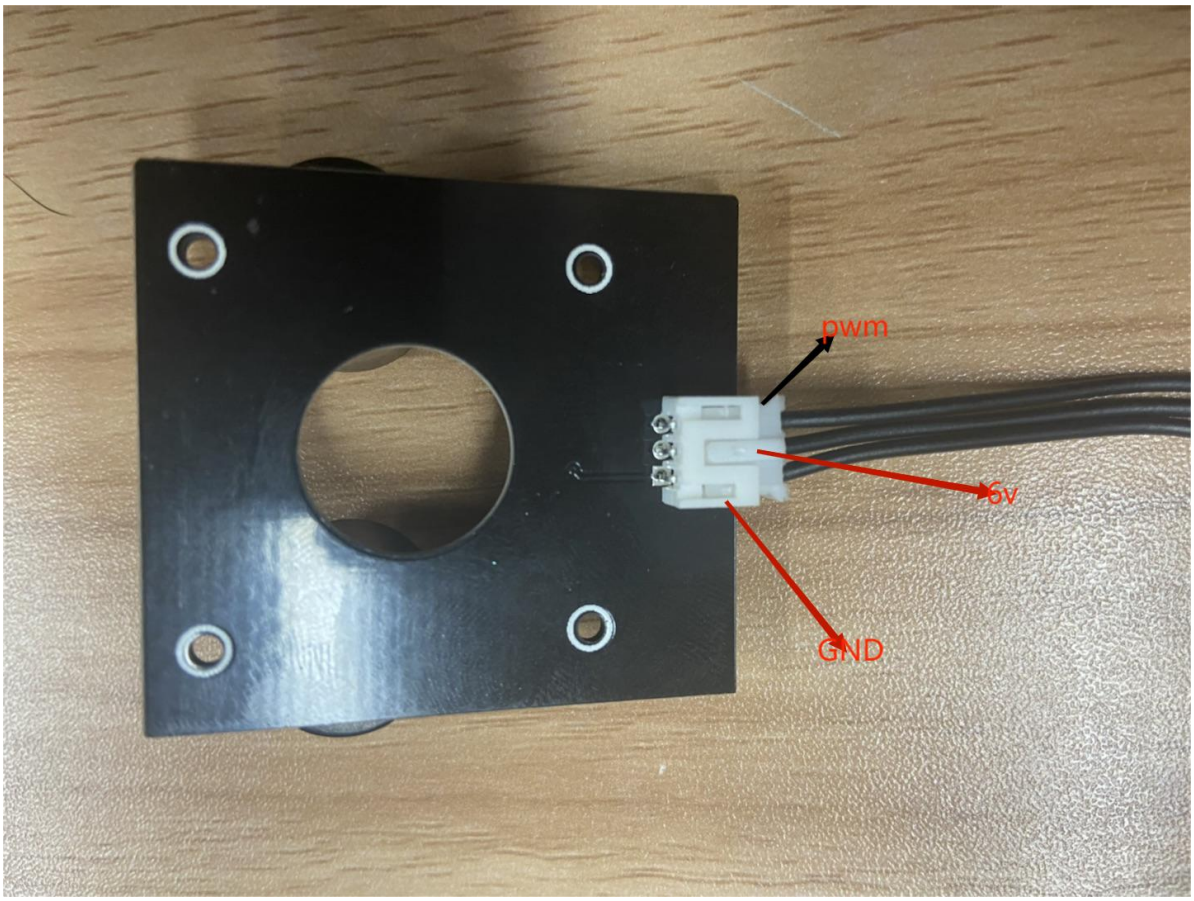
Searchlight module(working voltage:6V~30V) *1

External power supply with 6V output voltage *1

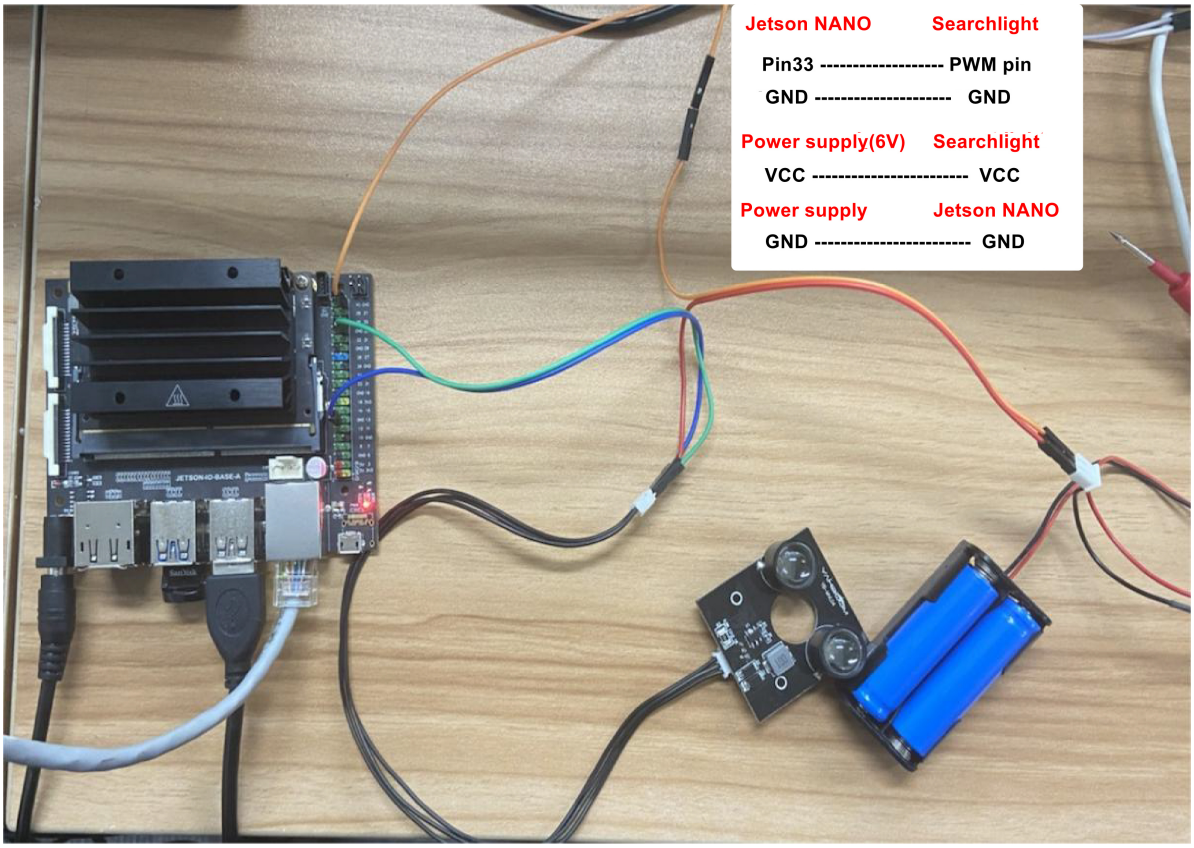
DuPont lines *N

The following is the connection diagram:

Pin of searchlight



The wiring is shown in the figure below:



3. About code

Because the 40pin Jetson.GOIO library of the Jetson nano board does not support direct output pwm. We need to use ROS to simulate pin output pwm, so we need to install ROS before running.

You can find a tutorial to install ROS system.

Step1: Create a pwm. py file

```
sudo touch pwm.py
```

Step 2: Write code

```
import Jetson.GPIO as GPIO
import time
import rospy
from multiprocessing import Process, Queue
from geometry_msgs.msg import Point,Twist

class PWM():
    def __init__(self, channel, frequency, level=1):
#Channel and frequency
        self.channel = channel
        self.frequency = frequency
        if level == 1:
            self.HIGH = GPIO.HIGH
            self.LOW = GPIO.LOW
        elif level == 0:
            self.HIGH = GPIO.LOW
            self.LOW = GPIO.HIGH
```

```
        else:
            raise ValueError("ERROR: level's err")
#Set pin mode
        GPIO.setmode(GPIO.BOARD)
        GPIO.setup(self.channel, GPIO.OUT, initial=self.LOW)

        self.if_pwm = True

    def ros_pwm(self):
        rospy.init_node("pwm")
        a = rospy.Subscriber("/pwm",Point,self.Change)   #subscribe

        f = 50.0
        c = 0.5         #Analog high and low level time
        t = 1/f
        t_h = t*c
        t_l = t*(1-c)
        print(t, t_h, t_l)
        while True:
            if self.if_pwm:           #Analog pwm
                GPIO.output(self.channel, self.HIGH)
                time.sleep(t_h)
                GPIO.output(self.channel, self.LOW)
                time.sleep(t_l)
                print('f and c: ', f, '  ',c)
            if rospy.is_shutdown():
                exit()
        rospy.spin()

if __name__=="__main__":
    a = PWM(33,50)
    a.ros_pwm()
```

The 33 and 50 in the initialization represent the pin and frequency, which here represents the pin 33 and 50 hz.

You can set the parameter c (0-1) in the code to represent the duty cycle of analog pwm, that is, the high and low level time. 0 is off, and 1 is the brightest

```
  c = 0.5
```

Step 3: Save and run code

```
python3 pwm.py
```

4.Experimental effect

After running the program, you can see that the searchlight is on for a long time.