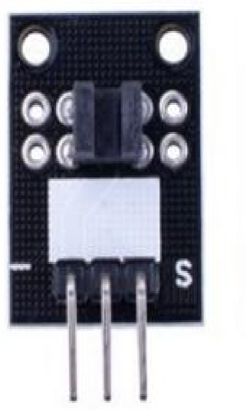
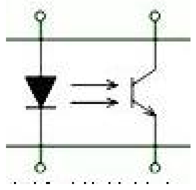
KUMAN
www.kumantech.com

Light-Blocking Sensor



Overview

A light-blocking sensor (also called a *photo interrupter*, and sometimes an *electric eye*) includes both a light emitter and light detector, separated by some channel of empty space. When the space is open (with nothing occupying it), the detector receives light from the emitter and electric current flows through the sensor's internal transistor. When the channel is blocked by some object in or passing through it, the detector stops receiving light, and the electric switch opens. By measuring the transistor's output, the Raspberry Pi can determine whether something is blocking the channel. The sensor can be represented as follows:



A light-emitting diode (left) emits light. When that light is received by the detector at the right, the switch is closed. If the light is blocked from the detector, the switch remains open.

The light-blocking sensor in this experiment has a small onboard channel that can be interrupted by a slip of paper or a playing card. Light-blocking sensors of all sizes have tremendous applications in manufacturing, security, robotics, logistics, speedometry, odometry, etc.

In this experiment, the Raspberry Pi will report whether the light-blocking sensor's channel is open or closed through an LED status indicator.

Experimental Materials

Raspberry Pi	x1
Breadboard	x1
Light-blocking sensor	x1
LED (3-pin)	x1
Resistor(330Ω)	x1
Dupont jumper wires	

Experimental Procedure

1. If you have not done so already, prepare your development system by installing the Python interpreter, RPi.GPIO library, and wiringPi library as described in READ_ME_FIRST.TXT.

2. Install the light-blocking sensor and LED on your breadboard, and use the resistor and Dupont jumper wires as illustrated in the Wiring Diagram below. Note you will connect only two of the three pins on the LED.

3. Execute the sample code stored in this experiment's subfolder.
   If using C, compile and execute the C code:

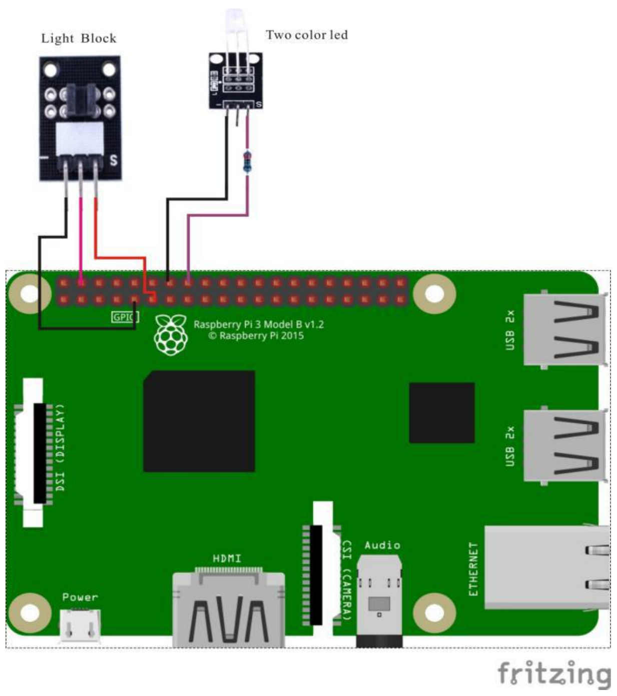
   ```
   cd Code/C
   gcc lightBlock.c -o lightBlock.out -lwiringPi
   ./lightBlock.out
   ```

   If using Python, launch the Python script:

   ```
   cd Code/Python
   python lightBlock.py
   ```

4. Make experimental observations while passing a piece of paper or light cardstock (a playing card or business card) between the two halves of the channel on the sensor. The code simply pushes the state of the sensor on to the state of the LED, so when you block the channel with a physical object, the LED illuminates.

Wiring Diagram



Light-blocking sensor pin position:

"S"	↔	Raspberry Pi pin 11
"+"	↔	Raspberry Pi +5V
"-"	↔	Raspberry Pi GND

LED pin position:

"S"	↔	Raspberry Pi pin 16 (through resistor)
"-"	↔	Raspberry Pi GND

Sample Code

Python Code

```
#!/usr/bin/env python
import RPi.GPIO as GPIO
import time

LightBreakPin = 11
LedPin = 16
```

```
def setup():
    GPIO.setmode(GPIO.BOARD)       # Numbers GPIOs by physical location
    GPIO.setup(LedPin, GPIO.OUT)   # Set LedPin's mode is output
    GPIO.setup(LightBreakPin, GPIO.IN, pull_up_down=GPIO.PUD_UP)
    GPIO.output(LedPin, GPIO.LOW) # Set LedPin low to off led


def loop():
    while True:
        if(GPIO.input(LightBreakPin) == 0):
            print 'Be covered....'
            GPIO.output(LedPin, GPIO.HIGH)
        else:
            GPIO.output(LedPin, GPIO.LOW)
        time.sleep(0.2)
def destroy():
    GPIO.output(LedPin, GPIO.LOW)     # led off
    GPIO.cleanup()                    # Release resource

if __name__ == '__main__':     # Program start from here
    setup()
    try:
        loop()
    except KeyboardInterrupt:  # When 'Ctrl+C' is pressed, the child
program destroy() will be  executed.
        destroy()
```

C Code

```
#include <wiringPi.h>
#include <stdio.h>

#define LightBreakPin     0
#define LedPin           4

int main(void)
{
    if(wiringPiSetup() == -1)
    {
        printf("setup wiringPi failed !");
        return -1;
    }
    pinMode(LightBreakPin, INPUT);
    pullUpDnControl(LightBreakPin, PUD_UP);
    pinMode(LedPin,  OUTPUT);

    while(1)
    {
        if(digitalRead(LightBreakPin) == LOW)
        {
            printf("Be covered....\n");
            digitalWrite(LedPin, HIGH);     //led on
        }
        else
        {
            digitalWrite(LedPin, LOW);      //led off
        }
    }

    return 0;
}
```