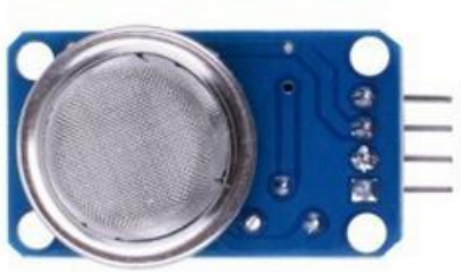
Smoke Sensor



Overview

Smoke sensors are vital technologies for fire prevention safety mechanisms as well as a variety of ventilation controllers. The material inside this MQ-2 gas sensor is tin dioxide (SnO2), which has low conductivity in clean air. When a combustible gas—such as liquefied gas, propane, butane, methane, alcohol or hydrogen—is present in the environment, the conductivity of the sensor increases as the concentration of combustible gas in the air increases. The sensor contains a simple circuit that converts that change in conductivity to an output signal corresponding to the gas concentration, and packages it in a convenient hybrid circuit design that reports both the intensity of the signal (as an analog output), and whether that signal has exceeded some user-adjustable threshold (as a digital output). In this experiment, you'll use the Raspberry Pi to capture the smoke sensor outputs to control an LED light indicating the presence of smoke or combustible vapors.

Experimental Materials

Raspberry Pi	x1
Breadboard	x1
Smoke sensor	x1
ADC0832	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 ADC0832 analog/digital converter IC, smoke sensor, three-pin LED and resistor on your breadboard, and use Dupont jumper wires to connect them to each other and your Raspberry Pi as illustrated in the Wiring Diagram below. Note you will connect only two of the three pins on the LED.

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

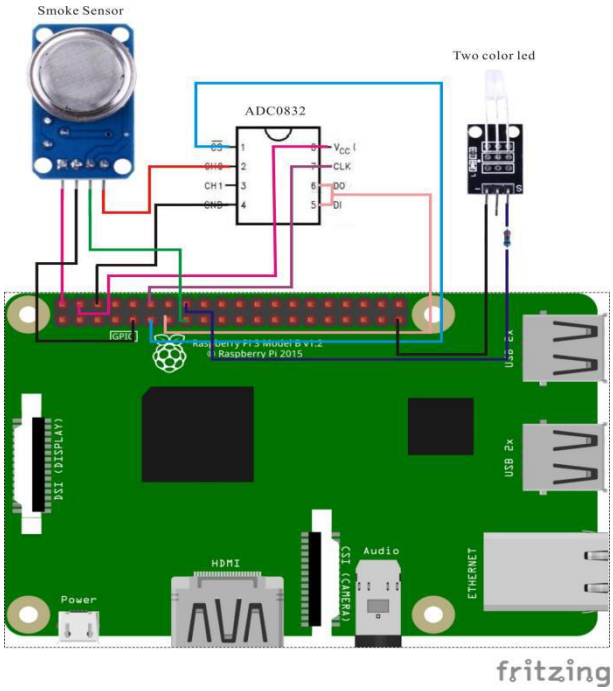
```
cd Code/C
gcc smokeSensor.c -o smokeSensor.out -lwiringPi
./smokeSensor.out
```

   If using Python, launch the Python script:

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

4. Make experimental observations. The sample code periodically reports the analog value of the current "smoke intensity" to the Raspberry Pi's command line output. (It may require ~20 seconds for the unit to reach stable measurements after powering up, and the sensor may become warm (not hot!) to the touch in normal operation.) If that measured value of combustible gas concentration exceeds a set threshold value, the LED lamp will light up, and when the gas concentration falls below the threshold value, the LED lamp will turn off.

Wiring Diagram



ADC0382 pin position:

CS	↔	Raspberry Pi Pin 11
CLK	↔	Raspberry Pi Pin 12
DI	↔	Raspberry Pi Pin 13
D0	↔	Raspberry Pi Pin 13
CH0	↔	Flame Sensor Pin A0
VCC	↔	Raspberry Pi +5V
GND	↔	Raspberry Pi GND

Flame Sensor pin position:

A0	↔	ADC0382 Pin CH0
D0	↔	Raspberry Pi Pin 15
GND	↔	Raspberry Pi GND
"+"	↔	Raspberry Pi +5V

LED pin position:

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

Sample Code

Python Code

```
#!/usr/bin/env python

#
# This is a program for MQ-2 Gas Sensor Module.
# It could detect danger gas and smokes.
```

```
# This program depends on ADC0832 ADC chip. Follow
# the instruction book to connect the module and
# ADC0832 to your Raspberry Pi.
#

import RPi.GPIO as GPIO
import ADC0832
import time

SensorDoPin = 15
LedPin = 16

def setup():
   GPIO.setmode(GPIO.BOARD)      # Numbers GPIOs by physical
location
   GPIO.setup(LedPin, GPIO.OUT)
   GPIO.setup(SensorDoPin, GPIO.IN)
   ADC0832.setup()

def loop():
    while True:
        # Get analog value from ADC0832
       analogVal = ADC0832.getResult(0)
       print analogVal     # Print analog value

       if not GPIO.input(SensorDoPin):
              print '    ****************'
              print '    * ! DANGER ! *'
              print '    ****************'
              print ''

              GPIO.output(LedPin, GPIO.LOW)
              time.sleep(0.1)
              GPIO.output(LedPin, GPIO.HIGH)

       else:
             GPIO.output(LedPin, GPIO.LOW)
       time.sleep(1)
```

```
def destroy():
    GPIO.cleanup()          # Release resource

if __name__ == '__main__':    # Program start from here
    setup()
    try:
        loop()
    except KeyboardInterrupt:
        destroy()
```

C Code

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

typedef unsigned char uchar;
typedef unsigned int  uint;

#define     ADC_CS    0
#define     ADC_CLK   1
#define     ADC_DIO   2
#define      SensorDoPin 3
#define     LedPin      4


uchar get_ADC_Result(void)
{

    uchar i;
    uchar dat1=0, dat2=0;

    digitalWrite(ADC_CS, 0);

    digitalWrite(ADC_CLK,0);
    digitalWrite(ADC_DIO,1);   delayMicroseconds(2);
    digitalWrite(ADC_CLK,1);   delayMicroseconds(2);
    digitalWrite(ADC_CLK,0);

    digitalWrite(ADC_DIO,1);    delayMicroseconds(2);
    digitalWrite(ADC_CLK,1);   delayMicroseconds(2);
    digitalWrite(ADC_CLK,0);

    digitalWrite(ADC_DIO,0);   delayMicroseconds(2);
```

```
    digitalWrite(ADC_CLK,1);
    digitalWrite(ADC_DIO,1);    delayMicroseconds(2);
    digitalWrite(ADC_CLK,0);
    digitalWrite(ADC_DIO,1);    delayMicroseconds(2);

    for(i=0;i<8;i++)
    {
        digitalWrite(ADC_CLK,1);   delayMicroseconds(2);
        digitalWrite(ADC_CLK,0);    delayMicroseconds(2);

        pinMode(ADC_DIO, INPUT);
        dat1=dat1<<1 | digitalRead(ADC_DIO);
    }

    for(i=0;i<8;i++)
    {
        dat2 = dat2 | ((uchar)(digitalRead(ADC_DIO))<<i);
        digitalWrite(ADC_CLK,1);   delayMicroseconds(2);
        digitalWrite(ADC_CLK,0);    delayMicroseconds(2);
    }

    digitalWrite(ADC_CS,1);

    pinMode(ADC_DIO, OUTPUT);

    return(dat1==dat2) ? dat1 : 0;
}

int main(void)
{
    uchar analogVal;

    if(wiringPiSetup() == -1)
    {
        printf("setup wiringPi failed !\n");
        return -1;
    }

    pinMode(ADC_CS,  OUTPUT);
    pinMode(ADC_CLK, OUTPUT);
    pinMode(LedPin,  OUTPUT);
    pinMode(SensorDoPin, INPUT);
```

```
    while(1)
    {
        analogVal = get_ADC_Result();
        printf("%d\n", analogVal);

        if(!digitalRead(SensorDoPin))
        {
            digitalWrite(LedPin, LOW);
            delay(100);
            digitalWrite(LedPin, HIGH);
            printf("\n**********   \
                \n  Danger!    \
                \n**********  \
                \n");
            delay(1000);
        }
        else
        {
            digitalWrite(LedPin, LOW);
        }
        delay(100);
    }

    return 0;
}
```

Characteristic Parameters

- Product Model: MQ-2
- Input Voltage: DC5V
- Product Type: Semiconductor Sensors
- Detection gas concentration: 300ppm ~ 10000ppm
- Analog output voltage increases with higher gas concentration.
- Good sensitivity to liquefied gas, natural gas, city gas, smoke.
- Product Dimensions: 32(L)*20(W)*22(H), four screw holes for easy positioning
- Long service life and reliable stability