**Microphone sensor**

Overview

This course will use the Raspberry Pi to control the microphone sensor module to capture the microphone sensor signal and print it to the command line of Raspiberry Pi.

Experimental Materials

Raspberry Pi \*1

Breadboard \*1

Microphone sensor \*1

ADC0832 \*1

Dupont Line

The Preparatory Work

1. Install python interpreter in your Raspberry Pi system

2. Install the RPi.GPIO library in your Raspberry Pi system

3. Install the wiring Pi library in your Raspberry Pi system

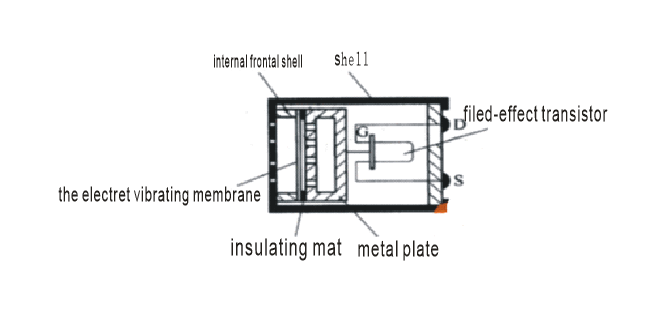
For more details, please refer to the attachment for installing the python interpreter and corresponding libraries in the Raspberry Pi system.

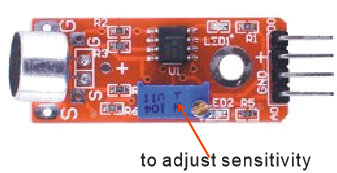
Product Description

Brief Introduction

Microphone sensor is a kind of sensor that can detect sound. It is widely used in daily life, military, medical, industrial, territorial waters, aerospace, etc. And it has become an indispensable part of the development of modern society. The microphone sensor incorporates a sound-sensitive capacitive electret microphone. Acoustic waves cause the electret thin film vibration in the microphone to cause a change in the capacitance and generate a tiny voltage corresponding it.

This voltage is then converted to 0-5v, which is accepted by the data collector through A/D conversion and transmitted to the controller. In this course, we use Raspberry Pi to detect the signal of sound sensor module and make corresponding action according to this signal.





**Characteristic Parameters**

◆3.6mm Mounting Screw Holes

◆ Use 5v DC power supply

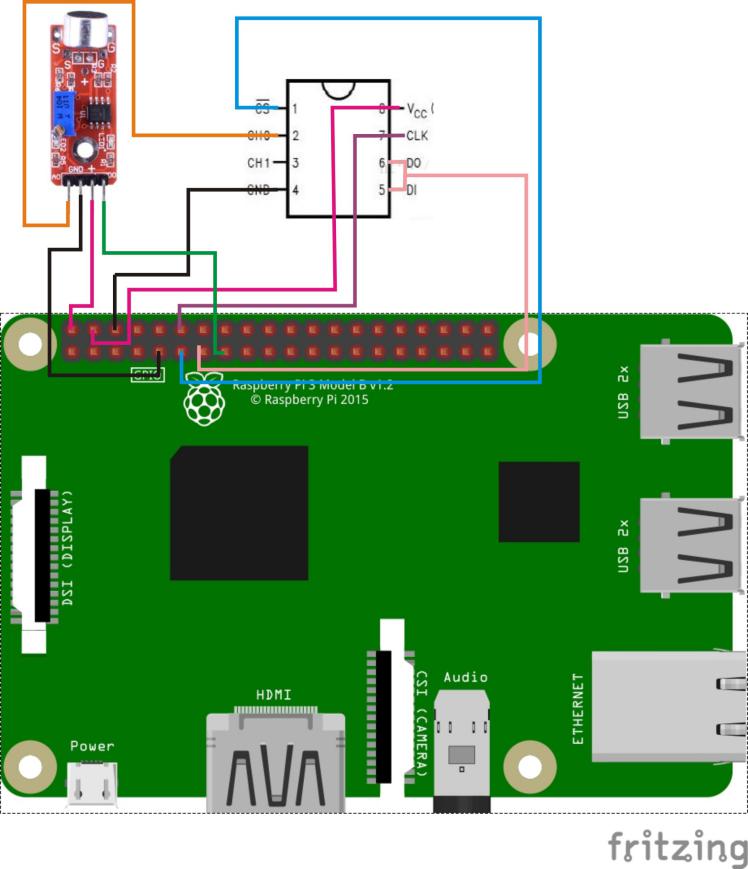
◆ AO analog output, real-time output microphone voltage signal

◆ There is a D0 threshold level output, when the sound intensity reaches a certain threshold, It will output high and low signal, the threshold - sensitivity can be adjusted by the potentiometer.

◆ Power Indicator

◆ There is a led to indicate Comparator output

Wiring diagram



**The Sample Code**

1. **Python Code**

#!/usr/bin/env python

import RPi.GPIO as GPIO

import ADC0832

import time

MIC\_DO\_PIN = 15

def init():

GPIO.setmode(GPIO.BOARD)

GPIO.setup(MIC\_DO\_PIN, GPIO.IN, pull\_up\_down=GPIO.PUD\_UP)

ADC0832.setup()

def loop():

while True:

global digitalVal

digitalVal = GPIO.input(MIC\_DO\_PIN)

if(digitalVal == 0):

print 'DO is %d' % digitalVal

print "voice in..."

print 'Current analog value is %d'% ADC0832.getResult(0)

else:

pass

if \_\_name\_\_ == '\_\_main\_\_':

init()

try:

loop()

except KeyboardInterrupt:

ADC0832.destroy()

print 'The end !'

2**、C Code**

#include <wiringPi.h>

#include <stdio.h>

#include <string.h>

#include <errno.h>

#include <stdlib.h>

#define ADC\_CS 0

#define ADC\_CLK 1

#define ADC\_DIO 2

#define MIC\_DO\_Pin 3

typedef unsigned char uchar;

typedef unsigned int uint;

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 digitalVal = 1;

uchar analogVal = 0;

if(wiringPiSetup() == -1){ //when initialize wiring failed,print messageto screen

printf("setup wiringPi failed !");

return 1;

}

pinMode(ADC\_CS, OUTPUT);

pinMode(ADC\_CLK, OUTPUT);

pinMode(MIC\_DO\_Pin, INPUT);

pullUpDnControl(MIC\_DO\_Pin, PUD\_UP);

printf("Please speaking...\n");

while(1){

if(!(digitalVal = digitalRead(MIC\_DO\_Pin)))

{

printf("Do is %d.\n", digitalVal);

printf("Voice in...");

printf("Current analog value is %d.\n", get\_ADC\_Result());

}

else

{

;

}

}

return 0;

}

**Experimental Phenomena**

When the microphone sensor detects the sound and the sound intensity reaches the set threshold, the value of the current microphone sensor after ADC conversion is printed in the command line interface of the Raspberry Pi system.