Two-axis game rocker module

Overview

We will uses the Raspberry Pi to catch the joystick's signal and determines the behavior of the joystick based on the signal caught.

Material Needed

RaspberryPi \*1

Breadboard \*1

Joystick \*1

ADC0832 \*1

Dupont Line

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 wiringPi library in your Raspberry Pi system

See the attached “Installing a Python Interpreter and Corresponding Libraries in a Raspberry Pi System”for details.

Product Description

Brief Introduction

The cross rocker is a bidirectional 10K resistor. The resistance of the tap changes with the direction of the rocker changes. This module uses 5V power supply. In the original state, the readout voltage of X and Y is about 2.5V. When the joystick is shaken in different directions, the readout voltage value changes along with it, maximum to 5V, and the minimum is 0V. When the raspberry pi detects this level change, it can determine how the rocker has acted.



Characteristic Parameters

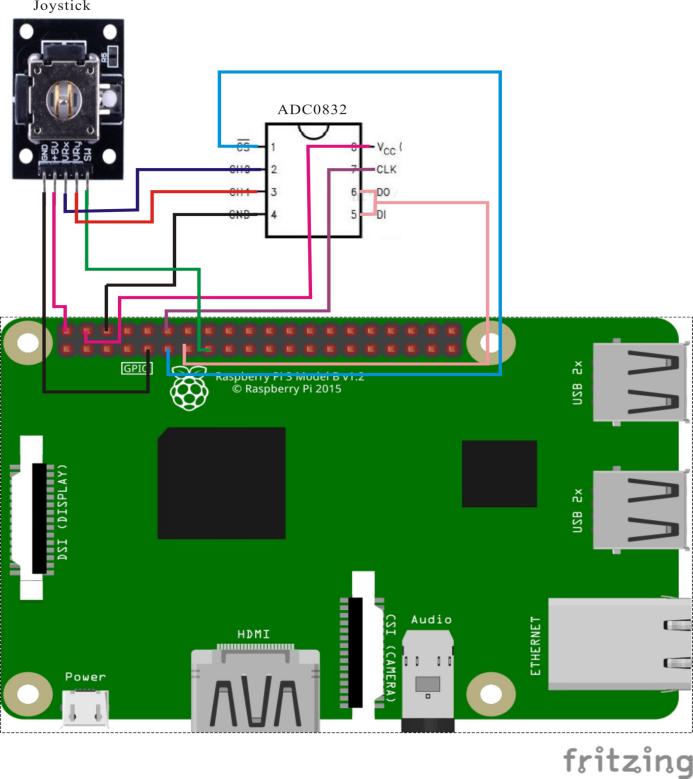
◆Operating voltage: 5V

◆Operating current: less than 20mA

◆Sensor Type: Analog

◆Production process: FR4 double-sided spray tin

Wiring diagram



Sample Code

1. Python Code

#!/usr/bin/env python

#

# This is a program for JoystickPS2 Module.

# This program depend on ADC0832 ADC chip.

#

import ADC0832

import RPi.GPIO as GPIO

import time

btn = 15

xFlag = 0

yFlag = 0

def setup():

ADC0832.setup() # Setup ADC0832

GPIO.setmode(GPIO.BOARD) # Numbers GPIOs by physical location

GPIO.setup(btn, GPIO.IN, pull\_up\_down=GPIO.PUD\_UP) # Setup button pin as input an pull it up

def getResult():

#get joystick result

global xFlag, yFlag

if ADC0832.getResult(0) == 0:

xFlag = 1 #down

if ADC0832.getResult(0) == 255:

xFlag = 2 #up

if ADC0832.getResult(1) == 0:

yFlag = 1 #left

if ADC0832.getResult(1) == 255:

yFlag = 2 #right

if GPIO.input(btn) == 0:

print 'Button is pressed!' # Button pressed

def loop():

while True:

getResult()

if xFlag == 1:

print 'down'

elif xFlag == 2:

print 'up'

if yFlag == 1:

print 'left'

elif yFlag == 2:

print 'right'

def destory():

GPIO.cleanup() # Release resource

if \_\_name\_\_ == '\_\_main\_\_': # Program start from here

setup()

try:

loop()

except KeyboardInterrupt:

destory()

2. 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 JoyStick\_Z 3

#define UP 1

#define DOWN 2

#define LEFT 1

#define RIGHT 2

uchar get\_ADC\_Result(uchar xyVal)

{

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);

if(xyVal == 'x'){

digitalWrite(ADC\_DIO,0); delayMicroseconds(2);

}

if(xyVal == 'y'){

digitalWrite(ADC\_DIO,1); 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 xFlag, yFlag;

uchar xVal = 0, yVal = 0, zVal = 0;

if(wiringPiSetup() == -1)

{

printf("setup wiringPi failed !");

return -1;

}

pinMode(ADC\_CS, OUTPUT);

pinMode(ADC\_CLK, OUTPUT);

pinMode(JoyStick\_Z, INPUT);

pullUpDnControl(JoyStick\_Z, PUD\_UP);

while(1)

{

xFlag = 0;

yFlag = 0;

xVal = get\_ADC\_Result('x');

if(xVal == 0)

{

xFlag = DOWN; //down

}

if(xVal == 255)

{

xFlag = UP; //up

}

yVal = get\_ADC\_Result('y');

if(yVal == 0)

{

yFlag = LEFT; //left

}

if(yVal == 255)

{

yFlag = RIGHT; //right

}

zVal = digitalRead(JoyStick\_Z);

if(zVal == 0)

{

printf("Button is pressed !\n");

}

switch(xFlag)

{

case UP:

printf("up\n");

break;

case DOWN:

printf("down\n");

break;

default:

break;

}

switch(yFlag)

{

case LEFT:

printf("left\n");

break;

case RIGHT:

printf("right\n");

break;

default:

break;

}

delay(200);

}

return 0;

}

Experimental phenomena

When the joystick is rotated, the command line interface of the Raspberry Pi system will print out the current rotation direction of the joystick; when the joystick is pressed, the command line interface of the Raspberry Pi system will print out the notice of “button pressed”.