Gesture recognition

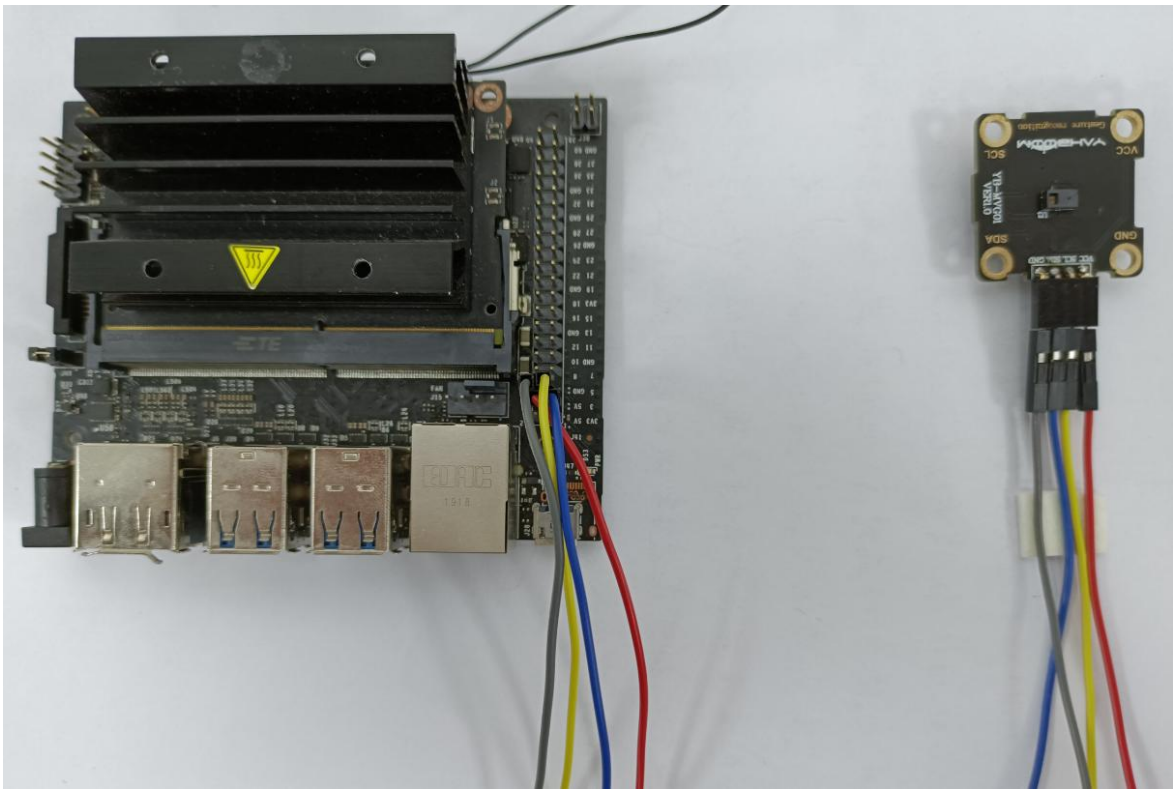
1. Purpose

In this course, we mainly learn to use Jetson Nano and gesture recognition module.

2. Preparation

Wiring diagram as shown below.

Gesture recognition module	Jetson NANO
SCL	SCL
SDA	SDA
VCC	5V
GND	GND



Before use this module, we need open I2C service of Jetson NANO.

Jetson Nano J41 Header					
Sysfs GPIO	Name	Pin	Pin	Name	Sysfs GPIO
	3.3 VDC *Power*	1	2	**5.0 VDC** *Power*	
	I2C_2_SDA *I2C Bus 1*	3	4	**5.0 VDC** *Power*	
	I2C_2_SCL *I2C Bus 1*	5	6	**GND**	
gpio216	**AUDIO_MCLK**	7	8	**UART_2_TX** */dev/ttyTHS1*	
	GND	9	10	**UART_2_RX** */dev/ttyTHS1*	
gpio50	**UART_2_RTS**	11	12	**I2S_4_SCLK**	gpio79
gpio14	**SPI_2_SCK**	13	14	**GND**	
gpio194	**LCD_TE**	15	16	**SPI_2_CS1**	gpio232
	3.3 VDC *Power*	17	18	**SPI_2_CS0**	gpio15
gpio16	**SPI_1_MOSI**	19	20	**GND**	

gpio17	SPI_1_MISO	21	22	SPI_2_MISO	gpio13
gpio18	SPI_1_SCK	23	24	SPI_1_CS0	gpio19
	GND	25	26	SPI_1_CS1	gpio20
	I2C_1_SDA *I2C Bus 0*	27	28	I2C_1_SCL *I2C Bus 0*	
gpio149	CAM_AF_EN	29	30	GND	
gpio200	GPIO_PZ0	31	32	LCD_BL_PWM	gpio168
gpio38	GPIO_PE6	33	34	GND	
gpio76	I2S_4_LRCK	35	36	UART_2_CTS	gpio51
gpio12	SPI_2_MOSI	37	38	I2S_4_SDIN	gpio77
	GND	39	40	I2S_4_SDOUT	gpio78

2.2 You need to open the IIC service of Jetson NANO board.

2.3 Install I2Ctool
Input following command in command terminal,
sudo apt-get update
sudo apt-get install -y i2c-tools
Wait patiently for the successful installation to complete.

2.4 Check whether the installation is successfully
Input following command in command terminal,
apt-cache policy i2c-tools

If system output is as follows, the installation is successful.
i2c-tools:
Installed: 4.0-2
Candidate: 4.0-2
Version list:
*** 4.0-2 500

500 http://ports.ubuntu.com/ubuntu-ports bionic/universe arm64 Packages
100 /var/lib/dpkg/status

2.5 Scan all i2c devices on a certain bus, and print out the device i2c bus address.

sudo i2cdetect -y -r -a 1

```
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
10:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
20:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
30:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
40:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
50:  50  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
60:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
70:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
```

2.6 Install smbus

Input following command in command terminal,

sudo apt-get update

sudo apt-get install -y python3-smbus

3. About code

Please check PAJ7620U2.py file.

3.1 Define the device address and register address of the module.

```
#i2c address
PAJ7620U2_I2C_ADDRESS   = 0x73
#Register Bank Selection
PAJ_BANK_SELECT         = 0xEF          #Bank0== 0x00,Bank1== 0x01
#Register Bank 0
PAJ_SUSPEND             = 0x03      #I2C suspend command (write= 0x01Enter the suspended state)
PAJ_INT_FLAG1_MASK      = 0x41      #Gesture detection interrupt flag mask
PAJ_INT_FLAG2_MASK      = 0x42      #Gesture /PS detects interrupt flag mask
PAJ_INT_FLAG1           = 0x43      #Gesture detects interrupt flags
PAJ_INT_FLAG2           = 0x44      #Gesture /PS detects interrupt flags
PAJ_STATE               = 0x45      #Gesture detection status indicator (only in gesture detection mode)
PAJ_PS_HIGH_THRESHOLD   = 0x69      #PS hysteresis high threshold (only in proximity detection mode)
PAJ_PS_LOW_THRESHOLD    = 0x6A      #PS hysteretic low threshold (only effective in proximity detection mode)
PAJ_PS_APPROACH_STATE   = 0x6B      #PS approaching state, approaching = 1
PAJ_PS_DATA             = 0x6C      #PS 8-bit data (valid only in gesture detection mode)
PAJ_OBJ_BRIGHTNESS      = 0xB0      #Object brightness (maximum 255)
PAJ_OBJ_SIZE_L          = 0xB1      #Object size (low 8 bits)
PAJ_OBJ_SIZE_H          = 0xB2      #Object size (high 8 bits)
#Register Bank 1
PAJ_PS_GAIN             = 0x44      #PS Gain setting (only available in proximity detection mode)
PAJ_IDLE_S1_STEP_L      = 0x67      #Idle S1 step size, used to set S1, response coefficient (low 8 bits)
PAJ_IDLE_S1_STEP_H      = 0x68      #Idle S1 step size, used to set S1, response coefficient (high 8 bits)
PAJ_IDLE_S2_STEP_L      = 0x69      #Free S2 step size for setting S2, response factor (low 8 bits)
PAJ_IDLE_S2_STEP_H      = 0x6A      #Free S2 step size, used to set S2, response factor (high 8 bits)
PAJ_OPTOS1_TIME_L       = 0x6B      #OPtoS1 Step, The OPtoS1 time used to set the operation state to standby 1 (low 8 bits)
PAJ_OPTOS2_TIME_H       = 0x6C      #OPtoS1 Step, Use to set OPtoS1 runtime to standby 1 stateHigh 8 bits)
PAJ_S1TOS2_TIME_L       = 0x6D      #S1toS2 step, S1toS2 time used to set standby state 1to standby state 2 (low 8 bits)
PAJ_S1TOS2_TIME_H       = 0x6E      #S1toS2 step, Set the S1toS2 time in standby 1to 8 bits higher in standby 2)
PAJ_EN                  = 0x72      #Enable/Disable PAJ7620U2
#Gesture detection interrupt flag mask
PAJ_RIGHT               = 0x01
PAJ_LEFT                = 0x02
PAJ_UP                  = 0x04
PAJ_DOWN                = 0x08
PAJ_FORWARD             = 0x10
PAJ_BACKWARD            = 0x20
PAJ_CLOCKWISE           = 0x40
PAJ_COUNT_CLOCKWISE     = 0x80
PAJ_WAVE                = 0x100
```

3.2 Define initialization array, register array, gesture register address.

```
Init_Register_Array = (
    (0xEF,0x00),
    (0x37,0x07),
    (0x38,0x17),
    (0x39,0x06),
    (0x41,0x00),
    (0x42,0x00),
    (0x46,0x2D),
    (0x47,0x0F),
    (0x48,0x3C),
    (0x49,0x00),
    (0x4A,0x1E),
    (0x4C,0x20),
```

```
#Register init array
Init_PS_Array = (
    (0xEF,0x00),
    (0x41,0x00),
    (0x42,0x00),
    (0x48,0x3C),
    (0x49,0x00),
    (0x51,0x13),
    (0x83,0x20),
    (0x84,0x20),
    (0x85,0x00),
    (0x86,0x10),
    (0x87,0x00),
    (0x88,0x05),
    (0x89,0x18),
```

```
#Gesture register init array
Init_Gesture_Array = (
    (0xEF,0x00),
    (0x41,0x00),
    (0x42,0x00),
    (0xEF,0x00),
    (0x48,0x3C),
    (0x49,0x00),
    (0x51,0x10),
    (0x83,0x20),
    (0x9F,0xF9),
    (0xEF,0x01),
    (0x01,0x1E),
    (0x02,0x0F),
    (0x03,0x10),
    (0x04,0x02),
```

....

3.3 Through I2C, the value of the initializing array and the initializing array of gesture register are written into the corresponding registers to start and initialize the gesture recognition module.

```
def __init__(self,address=PAJ7620U2_I2C_ADDRESS):
    self._address = address
    self._bus = smbus.SMBus(1)
    time.sleep(0.5)
    if self._read_byte(0x00) == 0x20:
        print("\nGesture Sensor OK\n")
        for num in range(len(Init_Register_Array)):
            self._write_byte(Init_Register_Array[num][0],Init_Register_Array[num][1])
    else:
        print("\nGesture Sensor Error\n")
    self._write_byte(PAJ_BANK_SELECT, 0)
    for num in range(len(Init_Gesture_Array)):
            self._write_byte(Init_Gesture_Array[num][0],Init_Gesture_Array[num][1])
```

3.4 Gesture recognition function: judge the currently recognized gesture by reading the value of the gesture recognition storage register and print out the corresponding gesture name.

```
def check_gesture(self):
    Gesture_Data=self._read_u16(PAJ_INT_FLAG1)
    if Gesture_Data == PAJ_UP:
        print("Up\r\n")
    elif Gesture_Data == PAJ_DOWN:
        print("Down\r\n")
    elif Gesture_Data == PAJ_LEFT:
        print("Left\r\n")
    elif Gesture_Data == PAJ_RIGHT:
        print("Right\r\n")
    elif Gesture_Data == PAJ_FORWARD:
        print("Forward\r\n")
    elif Gesture_Data == PAJ_BACKWARD:
        print("Backward\r\n")
    elif Gesture_Data == PAJ_CLOCKWISE:
        print("Clockwise\r\n")
    elif Gesture_Data == PAJ_COUNT_CLOCKWISE:
        print("AntiClockwise\r\n")
    elif Gesture_Data == PAJ_WAVE:
        print("Wave\r\n")
    return Gesture_Data
```

3.5 After successful initialization, the gesture recognition function is cycled to judge the current gesture.

```python
if __name__ == '__main__':

    import time

    print("\nGesture Sensor Test Program ...\n")

    paj7620u2=PAJ7620U2()

    while True:
        time.sleep(0.05)
        paj7620u2.check_gesture()
```

4. Running code

Input following command in command terminal of jetson nano.

python3 PAJ7620U2.py

5. Experimental phenomena

After the program running, if the module is initialized successfully, Jetson NANO system will print "Gesture Sensor OK", otherwise it will print "Gesture Sensor Error". If the initialization fails, we need to run code again.

After the initialization is successful, the module will start judge the value of gesture recognition, and different gestures will print out different action names through the serial port.

Put the gesture recognition module in the vertical direction, open your palm to face the module,
Swing over your palm from left to right in front of the module, Jetson NANO system will print "Left".
Swing over your palm from left to right in front of the module, Jetson NANO system will print "Right".
Swing over your palm from bottom to top in front of the module, Jetson NANO system will print "Up".
Swing over your palm from top to buttom in front of the module, Jetson NANO system will print "Down".
Approach from back to front directly in front of the module, Jetson NANO system will print "Forward".
Approach from front to back directly in front of the module, Jetson NANO system will print "Backward".
Make a fist and stretch out two or three fingers to point to the front of the module, then circle it clockwise for a while, Jetson NANO system will print "Clockwise".
Make a fist and stretch out two or three fingers to point to the front of the module, then circle it counterclockwise for a while, Jetson NANO system will print "AntiClockwise".
Wave your hand in front of the module for a while, Jetson NANO system will print "Wave".