

SKU:FIT0731 (<https://github.com/jacksonliam/mjpg-streamer>)



](Product Link)

Introduction

This is a 2-DOF Pan-Tilt HAT based on Raspberry Pi 40PIN GPIO. It can directly be used once plugged into Pi board's GPIO. Meanwhile, the board makes the Raspberry Pi GPIO pins led out for your convenient use. With I2C interface adopted, the module allows your Pi to control the camera movement and light intensity detection only using 2 signal lines. The on-board PCA9685 chip can output 12-bit resolution PWM to control the rotation of the pan-tilt. In addition, it comes with onboard TSL25911FN and built-in ADC that could make responses to light similar to human eyes, which is helpful for enhancing module performance; The on-board voltage level translators make it compatible with 3.3V/5V operating level. There is a specific acrylic board for fixing the Raspberry Pi camera on this product.

Through actual tests, it is proven that this module is compatible with all series types of Raspberry Pi and works well with Jetson Nano board. It can do 2-DOF motion in the horizontal and vertical directions, which is suitable for image monitoring/recognition and tracking when installed a camera. Besides, when combining an IR sensor or Ultrasonic distance sensor, it can be turned into a detection device to make a robot sense and avoid the surrounding obstacles. Of course, you can use it with other various sensors to complete your interactive works by Raspberry Pi or

Jetson Nano.

Note:

1. The Pan-Tilt HAT is unassembled, you need to get it done by yourself.
2. Raspberry Pi board and Camera are not included.

Specification

- Operating Voltage: 3.3V/5V
- Control Chip: PCA9685
- PWM Resolution: 12-bit
- Ambient Light Sensor: TSL25911FN
- Ambient Light Resolution: 12-bit
- Logic Voltage: 3.3V
- Communication: I2C
- Dimension: 56.665mm/2.232.56"

Controller

The controller of the product is PCA9685, an I2C-bus controlled 12-bit 16 channel PWM output chip. The Pan-Tilt HAT integrates an onboard TSL2581 light sensor that can be used to detect light and work with camera, and it uses I2C interface as well, saving pin resources.

Communication Protocol

This product employs I2C communication. I2C-bus has one data line(SDA) and one clock line(SCL). There are three kinds of signals in the process of communicating: Start signal, Stop signal and Answer signal.



- Start signal: SCL is High, SDA changes from High to Low, start to transmit data
- Stop signal: SCL is High, SDA changes from Low to High, stop transmitting
- Answer signal: The receiver will answer a specific Low level pulse to sender after receiving 8-Bit data as ACK.
 - I2C Write Data

When working, Raspberry Pi (hereafter named as Master) will first send a Start signal, then send a byte to TSL2581 sensor(hereafter named as Slaver), whose first 7bits are I2C address of Master and 1 bit write bit. Slave response with Answer signal every time it receives any data. Master send command register address to Slaver, then data of command register. Stop signals is sent to slave to stop communicating.

- I2C Read Data

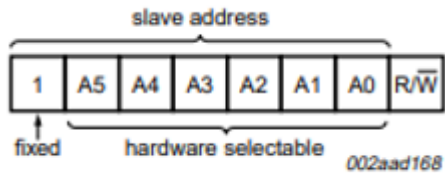
When working, Master will first send a Start signal, then send a byte to Slaver, whose first 7bits are I2C address of Master and 1 bit write bit. Slave response with Answer signal every time it receives any data. Master send command register address to Slave. After that. Master will send a

slave response with master signal every time it receives any data master send command register address to slave master send master then send a Start signal again, and then send a byte (7bits address and 1bit read bit) to Slaver. Slaver response and send data of the register to Master, master answer as well. Communication stops when Master sends Stop signal.

I2C Address

- PCA9685

PCA9685 and TSL2581 both use I2C communication. So we need to set different I2C addresses for them to ensure proper communication.



- TSL2581

Note: The default I2C address pins of PCA9685 are set as A5=A4=A3=A2=A1=0, address is 0x40. I2C address pins of TSL2581 are set as Float and its I2C address is 0x39 by default. If you use the module with other development boards, for example, STM32, please add R/W bit to Low bit.

Tutorial Based on Raspberry Pi

Note: Before you assemble servos to Pan-Tilt HAT, please test the servo independently with test codes, otherwise, the servo may get stuck or even damaged since it is not at its original position. Refer to the instructions below:

HardWare Configuration

Enable I2C Interface:

Operate at the end:

1. `sudo raspi-config`
2. Select Interfacing Options -> I2C ->yes Enable i2C Kernel Module

Raspberry Pi Software Configuration Tool (raspi-config)

- 1 Change User Password Change password for the current user
- 2 Network Options Configure network settings
- 3 Boot Options Configure options for start-up
- 4 Localisation Options Set up language and regional settings to match your location

Restart Raspberry Pi:

```
1. sudo reboot
```

Install Library

- Install BCM2835, open Raspberry Pi end and run the following commands:

```
1. wget http://www.airspayce.com/mikem/bcm2835/bcm2835-1.68.tar.gz
2. tar zxvf bcm2835-1.68.tar.gz
3. cd bcm2835-1.68/
4. sudo ./configure && sudo make && sudo make check && sudo make install
5. # For more information, please refer to: http://www.airspayce.com/mikem/bcm2835/
```

- Install wiringPi

```
1. sudo apt-get install wiringpi
2. # For Raspberry Pi system after may 2019, an upgrade is required.
3. wget https://project-downloads.drogon.net/wiringpi-latest.deb
4. sudo dpkg -i wiringpi-latest.deb
5. gpio -v
6. # run gpio -v version 2.52 will appear, if not, installation error happens
```

Download Program

```
1. sudo apt-get install p7zip-full
2. wget http://www.waveshare.net/w/upload/9/96/Pan-Tilt_HAT_code.7z
```

```
3. 7z x Pan-Tilt_HAT_code.7z -r -o./Pan-Tilt_HAT_code
4. sudo chmod 777 -R Pan-Tilt_HAT_code
5. cd Pan-Tilt_HAT_code/RaspberryPi/
```

Test

Connect:

Please Connect them correctly.

Brown	GND
Red	5V
Orange	S1/S0

Pay attention to the mount position of the tilt servo. Wrong place may cause servo damage. Do not assemble servos when testing. Make sure the servo can rotate 360 degrees freely and then connect hardware: pan servo to S1, tilt servo to S0. Open the test file in program folder.

```
1. cd test
2. make clean
3. make
4. sudo ./main
```

Assembly

Assembled:



A: pan servo

B: tilt servo

- How to assemble Camera?

The module comes with a M2 screw package independently. It has 5 screws and 10 nuts. Screw the camera on with M2 screws, and then attach the Acrylic plate onto it, and fasten them together with nuts.

Demo

- Basic Demo

```
1. #Execute the following actions under the directory Pan-Tilt-HAT/RaspberryPi/:  
2. #Execute this if using ambient light  
3. cd Light_Sensor/  
4. #Execute this if using pan servo  
5. cd Servo_Driver
```

- BCM2835 Example

```
1. cd bcm2835  
2. make clean  
3. make  
4. sudo ./main
```

- wiringpi Example

```
1. cd wiringpi
2. make clean
3. make
4. sudo ./main
```

- python Example

```
1. cd python
2. sudo python main.py
```

MJPEG-STREAMER Software Monitoring in Real-time

MJPEG-streamer can be used to collect image from a camera and transfer image to a browser by IP-based network in the form of stream.

- Run raspi-config command to start the camera before using

```
1. sudo raspi-config
2. Select Enable Camera, select YES
```

For CSI interface camera, the system cannot find the device node of /dev/video0. It is required to add bcm2835-v4l2 in /etc/modules file.

```
1. sudo nano /etc/modules
```

Add:

```
1. bcm2835-v4l2
```

This module will be loaded when the system starts. Restart the system:

```
1. sudo reboot
2. ls /dev/video*
```

The device node of video0 can be found now:

Note: after the above steps finished, check whether the operation and command are correct.

- Run the command below:

```
1. sudo apt-get update
2. sudo apt-get install libjpeg8-dev cmake
3. cd Pan-Tilt-HAT/RaspberryPi/web_Python
4. git clone https://github.com/jacksonliam/mjpg-streamer
5. cd mjpg-streamer/mjpg-streamer-experimental/
6. sudo make clean all
7. sudo ./start.sh
```

Input the Raspberry Pi address into Google address bar (other browsers may not support this). Port COM 8080. Click stream, the content captured by the camera will be displayed in real-time. For more information, please refer to: <https://github.com/jacksonliam/mjpg-streamer> (<https://github.com/jacksonliam/mjpg-streamer>).

Remote Control via BOTTLE

Bottle is a simple and efficient micro Python web framework that follows WSGI, through which web control can be quickly realized. It only supports Python2.

Install Library:

```
1. sudo apt-get install python-bottle
2. cd Pan-Tilt-HAT/RaspberryPi/web_Python/
3. sudo python main.py
```

Input the Raspberry Pi address into Google address bar(other browsers may not support this). Port COM 8001. The following interface will appear, click the button to control remotely.

Support controlling by mobile phone and PC without using APP.

Controlled by mobile phone:

Controlled by PC:

Tutorial Based on Jetson Nano

Install Library

Install the function library

- Open the end interface, and input the following commands to install the library.

```
1. sudo apt-get update
2. sudo apt-get install python3-pip
3. sudo pip3 install Jetson.GPIO
4. sudo groupadd -f -r gpio
5. sudo usermod -a -G gpio your_user_name
6. sudo cp /opt/nvidia/jetson-gpio/etc/99-gpio.rules /etc/udev/rules.d/
7. sudo udevadm control --reload-rules && sudo udevadm trigger
```

Note: your_user_name is your user name, like Tom.

- Install I2C

```
1. sudo apt-get install python-smbus
```

- Install Image Processing Library

```
1. sudo apt-get install python3-pil
2. sudo apt-get install python3-numpy
```

Download Program

```
1. sudo apt-get install p7zip
2. wget http://www.waveshare.net/w/upload/9/96/Pan-Tilt_HAT_code.7z
3. 7zr x Pan-Tilt_HAT_code.7z -r -o./Pan-Tilt_HAT_code
4. sudo chmod 777 -R Pan-Tilt_HAT_code
5. cd Pan-Tilt\ HAT Code/Pan-Tilt\ HAT/
```

```
2. cd Pan-Tilt-HAT/pan_servo/pan_tilt_hat/
```

Also you can download from the Github:

1. `sudo git clone https://github.com/waveshare/Pan-Tilt-HAT`
2. `cd Pan-Tilt-HAT/JetsonNano/`

Basic Demo

1. `# Execute this if using ambient light`
2. `cd 1_Light_Sensor/`
3. `#Execute this if using pan servo`
4. `cd 2_Servo_Driver/`

- Python Program

1. `#python2`
2. `cd python2`
3. `sudo python main.py`
4. `#python3`
5. `cd python3`
6. `sudo python3 main.py`

Test the Camera

1. `DISPLAY=:0.0 gst-launch-1.0 nvarguscamerasrc ! \ 'video/x-raw(memory:NVMM), width=1920, height=1080, format=(string)NV12, framerate=`

If everything works fine, the data captured by the camera will be displayed on the screen. Press Ctrl+C to stop.

Control the pan module.

Run at the end:

1. `cd 3_Pan-Tilt+CAM`
2. `sudo python main.py`


```
ezio@ezio-desktop:~/code/Pan-Tilt_HAT/App-Controls$ sudo python main.py
```

```
[sudo] password for ezio:
```

```
Sorry, try again.
```

```
[sudo] password for ezio:
```

```
192.168.0.141
```

```
server is running....
```

```
No protocol specified
```

```
uhubf vti0a: Could not get X11 display connection
```

```
Setting pipeline to FRAMED ...
```

```
Pipeline is live and does not need PRERUN ...
```

```
Setting pipeline to PLAYING ...
```

```
New clock: GstSystemClock
```

```
OST_AUDIO: Creating output stream
```

```
No protocol specified
```

```
CONSUMER: waiting until producer is connected...
```

```
OST_AUDIO: Available Sensor Index :
```

```
OST_AUDIO: 1280 x 1024 FR = 21.000000 fps Duration = 47618048 ; Analog Gain range min 1.000000, max 18.625000; Exposure Range min 13000, max 98370000;
```

```
OST_AUDIO: 1280 x 1024 FR = 28.000000 fps Duration = 35714288 ; Analog Gain range min 1.000000, max 18.625000; Exposure Range min 13000, max 98370000;
```

```
OST_AUDIO: 1824 x 1024 FR = 29.000000 fps Duration = 33133334 ; Analog Gain range min 1.000000, max 18.625000; Exposure Range min 13000, max 98370000;
```

```
OST_AUDIO: 1280 x 720 FR = 50.000000 fps Duration = 18888887 ; Analog Gain range min 1.000000, max 18.625000; Exposure Range min 13000, max 98370000;
```

```
OST_AUDIO: 1280 x 720 FR = 120.000000 fps Duration = 8333333 ; Analog Gain range min 1.000000, max 18.625000; Exposure Range min 13000, max 98370000;
```

```
OST_AUDIO: Running with following settings:
```

```
Camera Index = 0
```

```
Camera mode = Z
```

```
Output Stream W = 1024 H = 1024
```

```
seconds to Run = 8
```


```
Frame Rate = 29.000000
```

```
OST_AUDIO: PowerService: requested clock 40-2000000
```

```
OST_AUDIO: Setup Complete. Starting capture for 8 seconds
```

```
OST_AUDIO: Starting repeat capture requests.
```

```
CONSUMER: Producer has connected; continuing.
```




Data captured by the camera is displayed on the screen in real-time.

FAQ

For any questions, advice or cool ideas to share, please visit the **DFRobot Forum** (<https://www.dfrobot.com/forum/>).

More Documents

 Get **Pan-Tilt HAT For Raspberry Pi** (<https://www.dfrobot.com/product-203.html>) from DFRobot Store or **DFRobot Distributor**.
(<https://www.dfrobot.com/index.php?route=information/distributorslogo>)

Turn to the Top