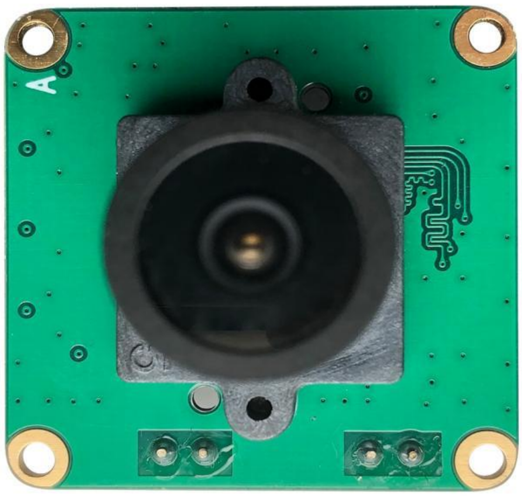
CAM-MIPIOV9281 V2 UserManual



目录

1. General

CAM-MIPIOV9281 module is a low-cost, monochrome(Black&White) global shutter camera module, designed for whole series Raspberry(P4/Pi3B+/PI3A+/PI3/CM4/CM3+). Plug into the CSI-2 Pi camera interface directly.

CAM-MIPIOV9281 module on board OmniVision's OV9281 is high-speed global shutter image sensors that bring 1-megapixel resolution to a wide range of consumer and industrial computer vision applications, including augmented reality (AR), virtual reality (VR), collision avoidance in drones, bar code scanning and factory automation. Built on OmniVision's OmniPixel®3-GS pixel technology, the OV9281 and OV9282 feature a high-speed global shutter pixel with best-in-class near-infrared (NIR) quantum efficiency (QE) to meet high-resolution and low-latency requirements.

Innomaker unique driver support 12 working mode, Suitable for various applications.

Mode	Resolution Ratio	Data Format	Frame Rate
Mode0	1280x800	Y10	120fps
Mode1	1280x800	Y8	144fps
Mode2	1280x800	Y10	EXT_TRIG
Mode3	1280x800	Y8	EXT_TRIG
Mode4	1280x720	Y10	120fps
Mode5	1280x720	Y8	144fps
Mode6	1280x720	Y10	EXT_TRIG
Mode7	1280x720	Y8	EXT_TRIG
Mode8	640x400	Y10	210fps
Mode9	640x400	Y8	253fps
Mode10	640x400	Y10	EXT_TRIG
Mode11	640x400	Y8	EXT_TRIG

2. Features

(1) CAM-MIPIOV9281 is an Industrial Camera Module for Raspberry Pi 4/3B+/3B/CM4/CM3+. Fully V4L2 (Video4Linux) compatible device. Support libcamera on Bullseye system.

(2) On-board OmniVision OV9281 Monochrome(Black&White) global shutter CMOS Sensor, 1M Pixel.Output RAW8/RAW10 choosable.Support from 640x400@253fps to 1280x800@144fps.

(3) Support for external trigger mode, LED and flash strobe mode and gain programmable. Uses the technique of optical coupling isolation TLP281.

(4) Match a wide angle fisheye Lens. Fov(D)=148 degrees, Fov(H)=118 degrees. Focal distance is adjustable.

(5) Comes with user manual, test demo and friendly technology support. We offer custom design service.

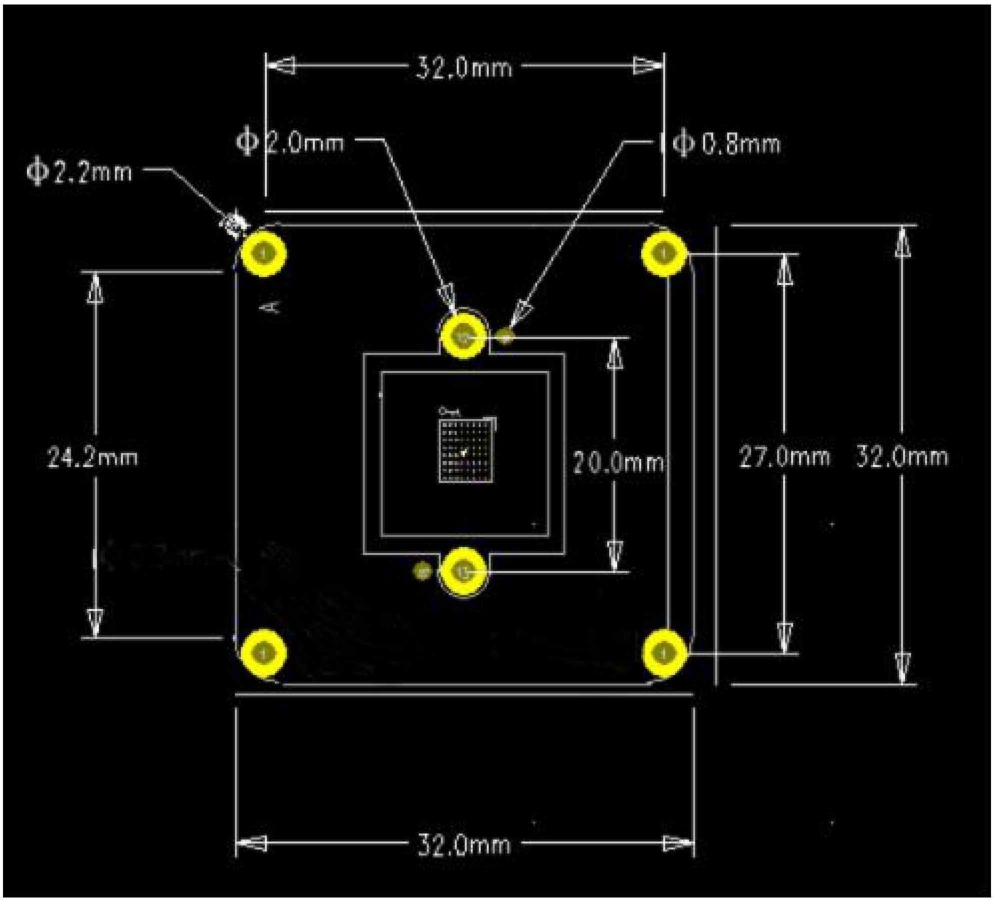
3. Hardware Description

3.1 Overview

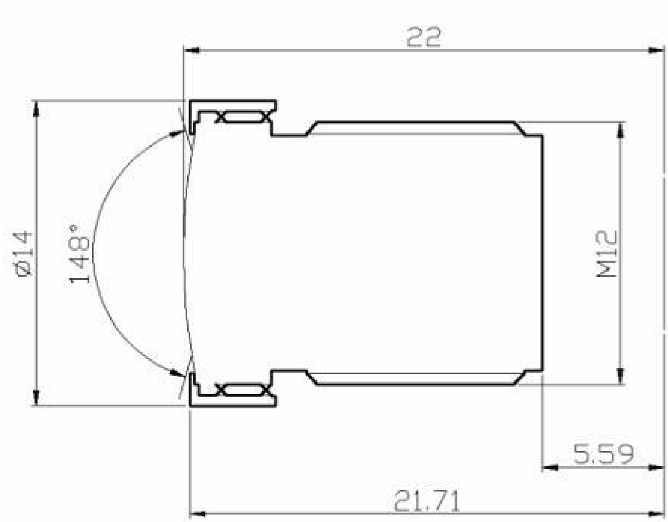
Sernor Board	
Size	32mm x 32mm
Weight	4g
Still Resolution	1 million pixels
Video Modes	Mode0: 1280x800, Y10, 120fps Mode1: 1280x800, Y8, 144fps Mode2: 1280x800, Y10, EXT_TRIG Mode3: 1280x800, Y8, EXT_TRIG Mode4: 1280x720, Y10, 120fps Mode5: 1280x720, Y8, 144fps Mode6: 1280x720, Y10, EXT_TRIG Mode7: 1280x720, Y8, EXT_TRIG Mode8: 640x400, Y10, 210fps Mode9: 640x400, Y8, 253fps Mode10: 640x400, Y10, EXT_TRIG Mode11: 640x400, Y8, EXT_TRIG Mode12: 320x200 Y8 453fps
Linux integration	V4L2 driver available
Sensor	Monochrome global shutter OV9281
Sensor Resolution	1280*800 pixels
Sensor image area	3896µm x 2453µm
Pixel size	3 µm x 3 µm
Optical size	1/4"
S/N ratio	38 dB
Dynamic range	68 dB
Output interface	2-lane MIPI Interface
Output formats	8/10-bit B&W RAW
Field of view	Fov(D) = 148 degrees , Fov(H) = 118 degrees
Focal Length	2.8 mm
Focal Distance	Adjustable
TV DISTORTION	<-17%
F(N) /Aperture	2.2

3.2 Size

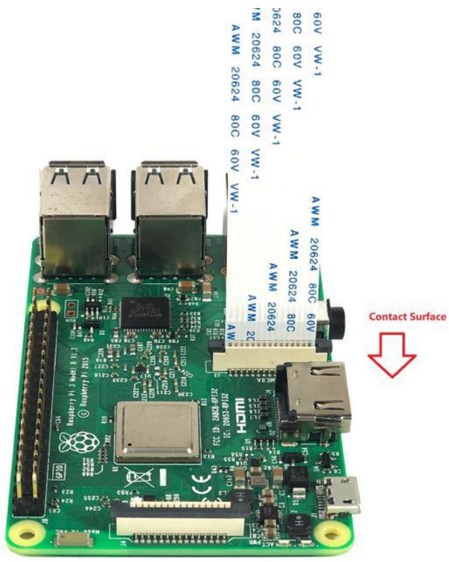
3.2.1 PCB Size

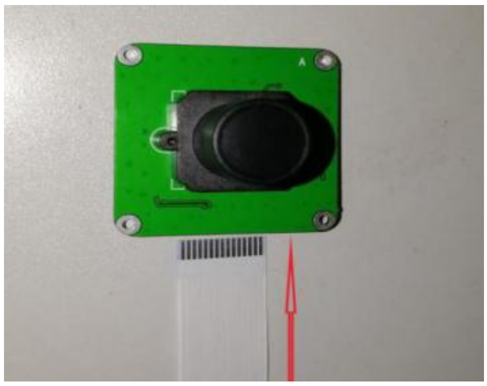


3.2.2 Len Size



3.3 Connection Of The Hardware

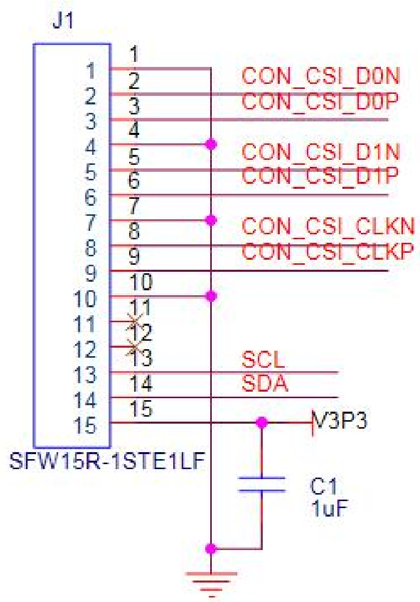




3.4 Pin-Out

3.4.1 Signal/Power Connector J1

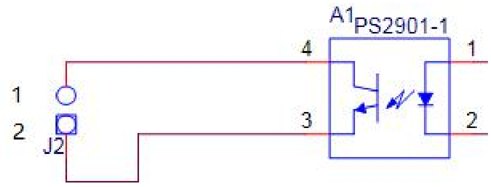
The J1 pin map is same Raspberry Pi camera.



PIN	Symbol	Description
1	GND	Ground Pin
2	CON_CSI_D0N	Pixel Data Lane0 Negative
3	CON_CSI_D0P	Pixel Data Lane0 Positive
4	GND	Ground Pin
5	CON_CSI_D1N	Pixel Data Lane1 Negative
6	CON_CSI_D1P	Pixel Data Lane1Positive
7	GND	Ground Pin
8	CON_CSI_CLKN	Pixel Clock Output Form Sensor Negative
9	CON_CSI_CLKP	Pixel Clock Output Form Sensor Positive
10	GND	Ground Pin
11	None	None
12	None	None
13	SCL	CLK input, SIO_C of SCCB
14	SDA	DATA input,SIO_D of SCCB
15	3.3V Power	Power Supply

3.4.2 STROB Connector J2

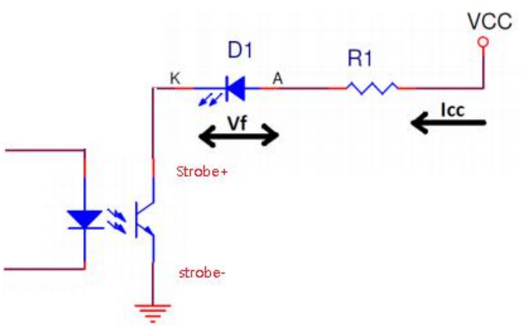
(1)Pin Description



ISO FLASH

J2 PIN	Symbol
1	STROB+
2	STROB-

(2)Reference Circuit



On-board TLP281 optocoupler isolation, Notice the max collector current is 50mA.

Output Specifications

S. No	Parameter	Test Condition	Value			Unit
			Min	Typ	Max	
1	Driver Voltage (VCC)			12	24	V
2	Drive current (Icc)			10	50	mA
3	Collector Emitter Breakdown Voltage				80	V
4	Collector Emitter Saturation Voltage	Icc = 1 mA		0.1	0.2	V
5	Power Dissipation				150	mW

Collector-Emitter Saturation Voltage	$V_{CE(sat)}$	$I_F = 10mA, I_C = 1mA$		0.1	0.2	V

So If the current required to drive the Flash LED is no more than 50mA

The value of series resistor: R1 = (VCC- Vf - VCE) / If

VCC: system Voltage

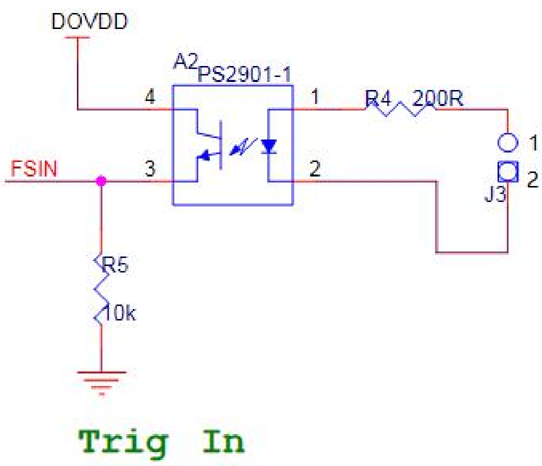
Vf: Forward voltage of Flash LED for current Icc

VCE: Collection Emitter voltage, typical:0.1V

If the current required to drive the flash exceeds 50mA, then it is required to drive it with the help of LED driver circuit, and LED driver circuit can be controlled by using the strobe output pin.

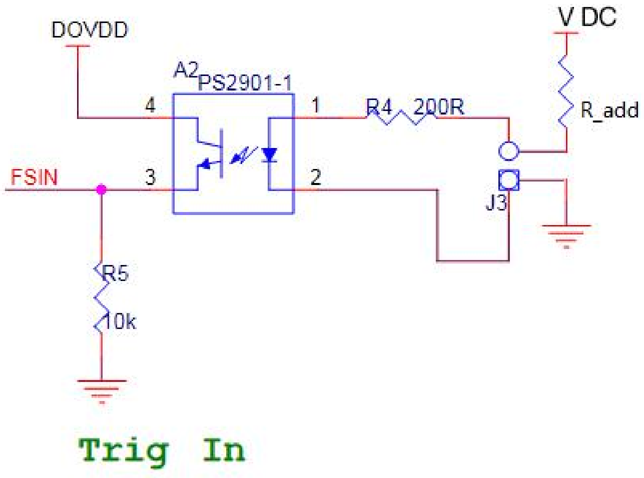
3.4.3 EXT TRIG Connector J3

(1)Pin Description



J3 PIN	Symbol	Description
1	TRIG+	3.3V-5.0V External Trigger Input
2	TRIG-	External GND

(2) Reference Circuit



For example, VCC = 12V, Vf = 1.25V

The calculations done here are based on 12VDC. Please do follow these calculations for other voltages like 24VDC.

Let's take the current through IR LED I_f = 20mA.
Voltage drop across the IR LED = 1.25V
The value of Resistor $R_1 = (V_{cc}-V_f)/I_f = (12 – 1.25)/0.02 = 537.5\ \Omega$
Wattage of resistor $R_1 > I_f^2 * R_1 = 0.02^2*537.5 = 0.215W$
Wattage of the resistor R_1 selected should be greater than 0.215W.

And there is a resistor on board(R4 = 200 Ω), So the R_add = R1 - R4 = 537.5 - 200 = 337.5 Ω

4. Using Innomaker Unique Driver

4.1 Load Raspberry Pi image

Prepare a capacity of more than 8GB TF card(16Gb Class10 is better) and a card reader. Load the image file on to the SD card, using the instructions provided on the Raspberry Pi website for Linux, Mac or PC:
https://www.raspberrypi.org/documentation/installation/installing-images/README.md

Raspbian Image download:
https://www.raspberrypi.org/downloads/

4.2 Tools/Driver Download

There are two ways to get the tools and drivers into Raspberry Pi.

Step 1: Use Raspberry Pi terminal get from github directly. And check whether is download successful. Make sure your Raspberry Pi is connect to network.

sudo git clone https://github.com/INNO-MAKER/CAM-OV9281RAW-V2.git

```
pi@raspberrypi:~ $ sudo git clone https://github.com/INNO-MAKER/CAM-OV9281RAW-V2.git
Cloning into 'CAM-OV9281RAW-V2'...
remote: Enumerating objects: 40, done.
remote: Counting objects: 100% (40/40), done.
remote: Compressing objects: 100% (19/19), done.
remote: Total 40 (delta 17), reused 40 (delta 17), pack-reused 0
Receiving objects: 100% (40/40), 229.05 KiB | 1.47 MiB/s, done.
Resolving deltas: 100% (17/17), done.
pi@raspberrypi:~ $
```

Step 2: Download it into your computer

Download from below link, and copy them to your Raspberry Pi by U disk or telnet.
https://github.com/INNO-MAKER/CAM-OV9281RAW-V2.git

Step 3: Packet Instructions
The are contain three parts in the link:

```
pi@raspberrypi:~/CAM-OV9281RAW-V2 $ ls
autoinstall_driver.sh  Linux_5.10.92  tools
```

Linux_5.xx.xx	**Linux core version of Driver**
tools	**All test demo and tools**
autoinstall driver.sh	**auto installation script for driver**

4.3 Tools/Driver Automatic Install

We provide a automatic install script for user install the OV9281 driver convenient and fast. It's only for Pi4 and Pi3 now.

Step 1: chmod

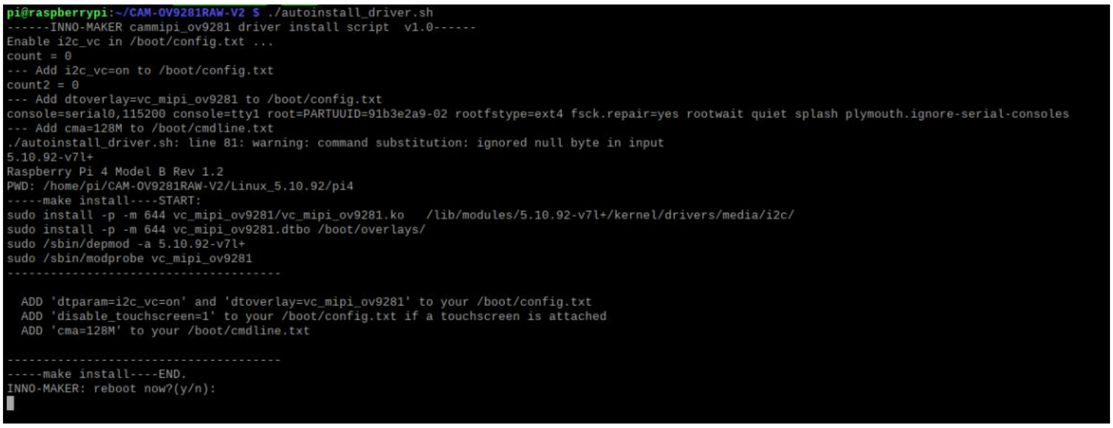
Using chmod command set all the read, write, and execute permissions for these file.

sudo chmod -R a+rwx *

Step 2: Execute

Execute this script in terminal and input 'y' to reboot.

./autoinstall_driver.sh



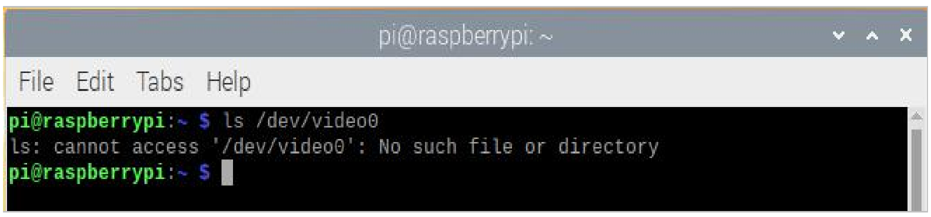
Step 3: Checkout Device

after reboot, use below command to check the camera is ready.

ls /dev/video0

Successful:

Unsuccessful:



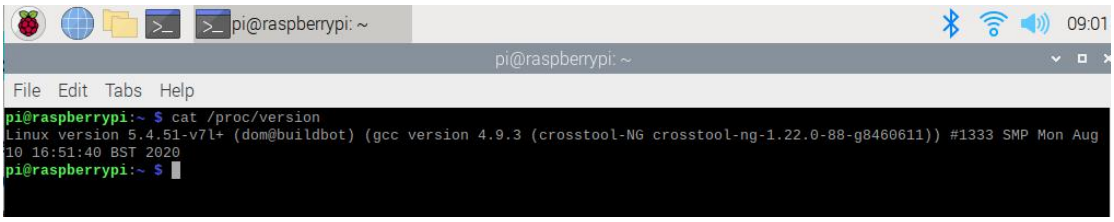
4.4 Tools/Driver Manual Install

Step 1: Check Basic Information

Check the basic information of your Raspberry Pi to choose the right driver. We take Raspberry Pi 4 + 5.10.92-v7l+(Release data 2022-01-28) as an example in this document.

Check the kernel version of your Raspbian.

cat /proc/version



Check the hardware version of your Raspberry Pi

cat /proc/device-tree/model

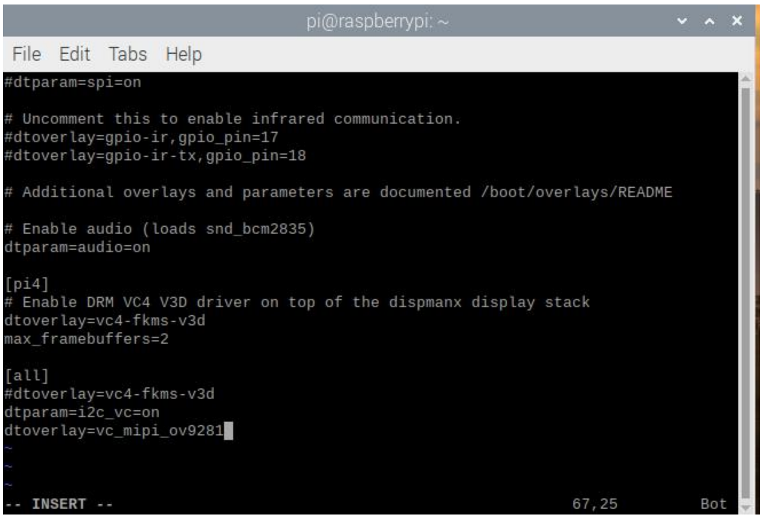
Step 2: Modify 'config.txt'

sudo nano /boot/config.txt

Open 'config.txt', and then add two line in the bottom,finally save and exit.

dtparam=i2c_vc=on

dtoverlay=vc_mipi_ov9281



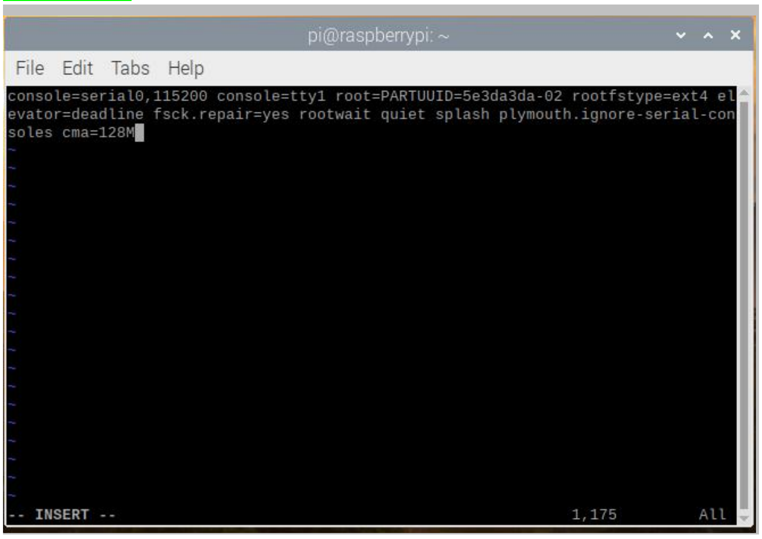
```
pi@raspberrypi: ~
File  Edit  Tabs  Help
#dtparam=spi=on

# Uncomment this to enable infrared communication.
#dtoverlay=gpio-ir,gpio_pin=17
#dtoverlay=gpio-ir-tx,gpio_pin=18

# Additional overlays and parameters are documented /boot/overlays/README

# Enable audio (loads snd_bcm2835)
dtparam=audio=on

[pi4]
# Enable DRM VC4 V3D driver on top of the dispmanx display stack
dtoverlay=vc4-fkms-v3d
max_framebuffers=2

[all]
#dtoverlay=vc4-fkms-v3d
dtparam=i2c_vc=on
dtoverlay=vc_mipi_ov9281
~
~
~
-- INSERT --                                          67,25         Bot
```

Step 3: Modify 'cmdline.txt'

sudo vim /boot/cmdline.txt

Allocate memory to GPU, add below line in the end of file, finally save and exit.

cma=128M



```
pi@raspberrypi: ~
File  Edit  Tabs  Help
console=serial0,115200 console=tty1 root=PARTUUID=5e3da3da-02 rootfstype=ext4 el
evator=deadline fsck.repair=yes rootwait quiet splash plymouth.ignore-serial-con
soles cma=128M
~
-- INSERT --                                          1,175         All
```

Step 5:Restart to enable the configuration

sudo reboot

Step 6: Chmod

Using chmod command set all the read, write, and execute permissions for these file.

sudo chmod -R a+rwx *

```
pi@raspberrypi: ~/cam-mipiov9281
File  Edit  Tabs  Help
pi@raspberrypi:~ $ cd cam-mipiov9281/
pi@raspberrypi:~/cam-mipiov9281 $ sudo chmod -R a+rwx *
pi@raspberrypi:~/cam-mipiov9281 $
```

Step 7:Choose the right driver to install

Refer to chapter 4.3, we must choose the right driver path to match your Raspberry Pi hardware and system version. In this example, we choose ‘Linux version 5.10.92’ and ‘pi4’

Please double check the hardware version(Pi3 or Pi4) and Linux-Kernel are 100% match the driver version. Otherwise the camera won’t work.

```
pi@raspberrypi:~ $ ls
Bookshelf          Desktop    Downloads  Pictures  Templates
CAM-OV9281RAW-V2   Documents  Music      Public    Videos
pi@raspberrypi:~ $ cd CAM-OV9281RAW-V2/
pi@raspberrypi:~/CAM-OV9281RAW-V2 $ ls
autoinstall_driver.sh  Linux_5.10.92  tools
pi@raspberrypi:~/CAM-OV9281RAW-V2 $ cd Linux_5.10.92/
pi@raspberrypi:~/CAM-OV9281RAW-V2/Linux_5.10.92 $ ls
arm64  cm4-dual  pi3  pi4
pi@raspberrypi:~/CAM-OV9281RAW-V2/Linux_5.10.92 $ cd pi4/
pi@raspberrypi:~/CAM-OV9281RAW-V2/Linux_5.10.92/pi4 $ ls
Makefile  vc_mipi_ov9281  vc_mipi_ov9281.dtbo
pi@raspberrypi:~/CAM-OV9281RAW-V2/Linux_5.10.92/pi4 $
```

Step 8:Install and reboot

sudo make install

```
pi@raspberrypi:~/CAM-OV9281RAW-V2/Linux_5.10.92/pi4 $ sudo make install
sudo install -p -m 644 vc_mipi_ov9281/vc_mipi_ov9281.ko  /lib/modules/5.10.92-v7l+/kernel/drivers/media/i2c/
sudo install -p -m 644 vc_mipi_ov9281.dtbo /boot/overlays/
sudo /sbin/depmod -a 5.10.92-v7l+
sudo /sbin/modprobe vc_mipi_ov9281
--------------------------------------

  ADD 'dtparam=i2c_vc=on' and 'dtoverlay=vc_mipi_ov9281' to your /boot/config.txt
  ADD 'disable_touchscreen=1' to your /boot/config.txt if a touchscreen is attached
  ADD 'cma=128M' to your /boot/cmdline.txt

--------------------------------------
pi@raspberrypi:~/CAM-OV9281RAW-V2/Linux_5.10.92/pi4 $
```

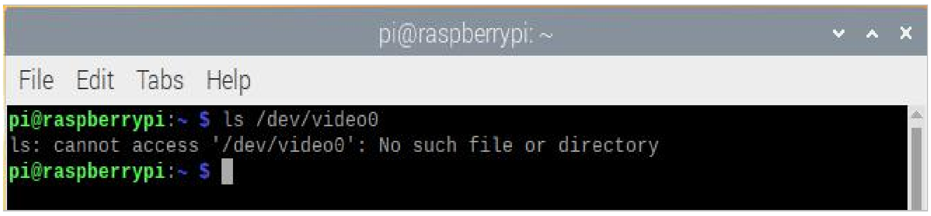
Step 9: Check the device:

Use below command to check the camera is ready,after reboot.

ls /dev/video0

Successful:

Unsuccessful:



4.5 Setting Mode

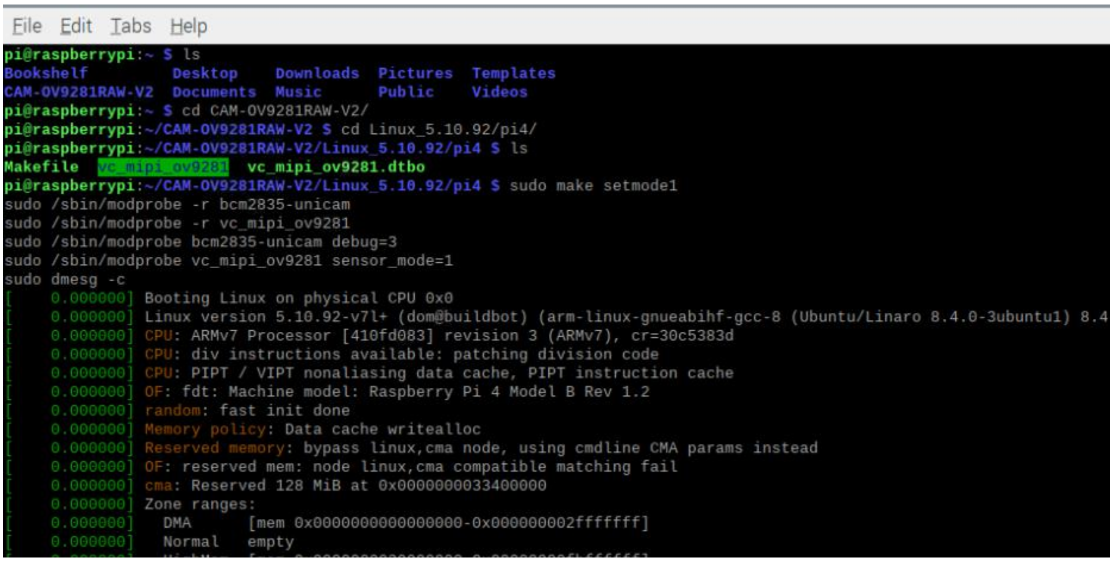
4.5.1 Set Mode

CAM-MIPIOV9281 unique driver can support below working modes now. If you need other resolution/frame rate mode. Please contract us by e-mail(support@inno-maker.com).

Mode	Resolution Ratio	Data Format	Frame Rate
Mode0	1280x800	Y10	120fps
Mode1	1280x800	Y8	144fps
Mode2	1280x800	Y10	EXT_TRIG
Mode3	1280x800	Y8	EXT_TRIG
Mode4	1280x720	Y10	120fps
Mode5	1280x720	Y8	144fps
Mode6	1280x720	Y10	EXT_TRIG
Mode7	1280x720	Y8	EXT_TRIG
Mode8	640x400	Y10	210fps
Mode9	640x400	Y8	253fps
Mode10	640x400	Y10	EXT_TRIG
Mode11	640x400	Y8	EXT_TRIG

In the driver folder, use below command to set the working mode, I suggest you keep one terminal separately for easy to change the mode.

sudo make setmode1



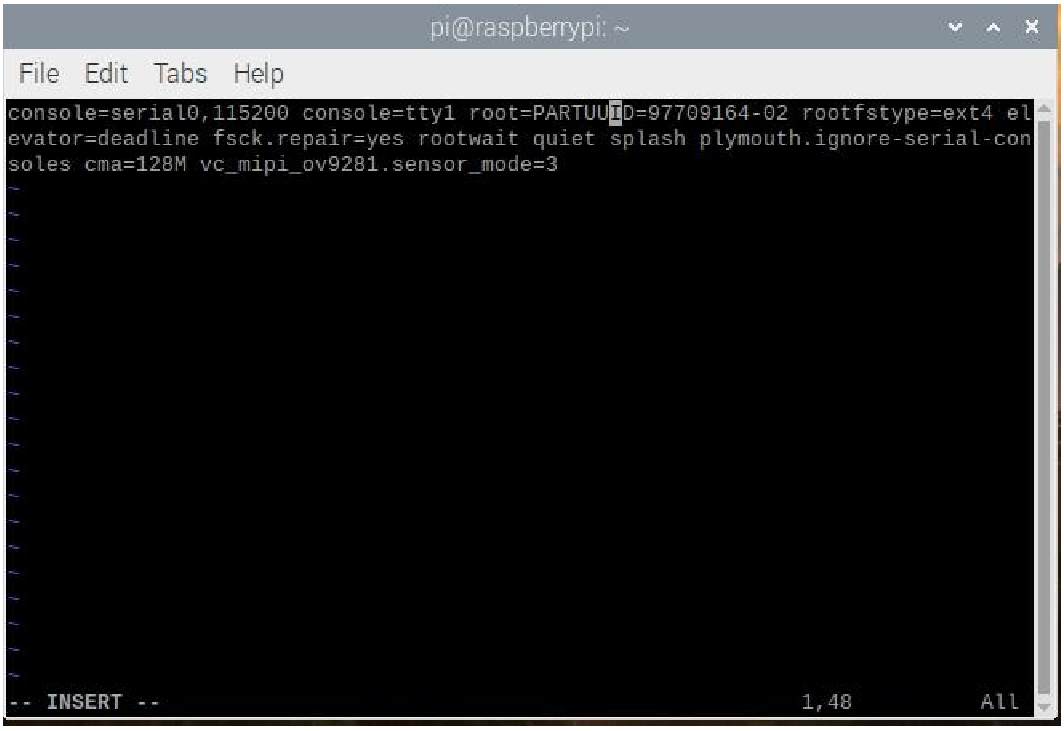
4.5.2 Change Default Mode

Step 1: Open cmdline.txt

sudo vim /boot/cmdline.txt

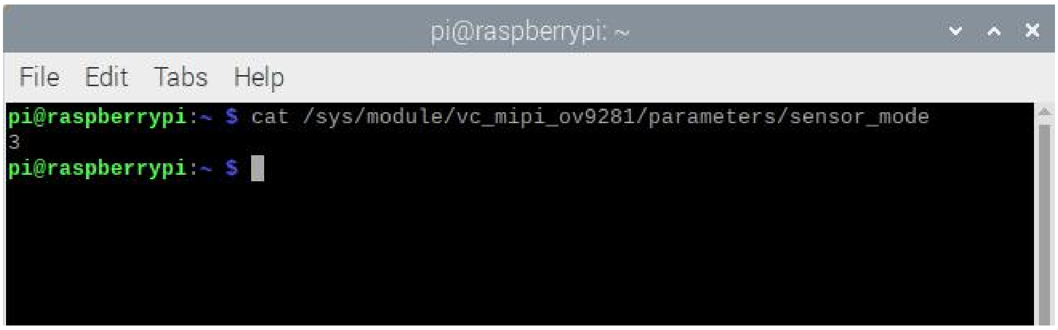
Step 2: Add default mode

vc_mipi_ov9281.sensor_mode=3



Step 3: Save and reboot, You will check the default value.

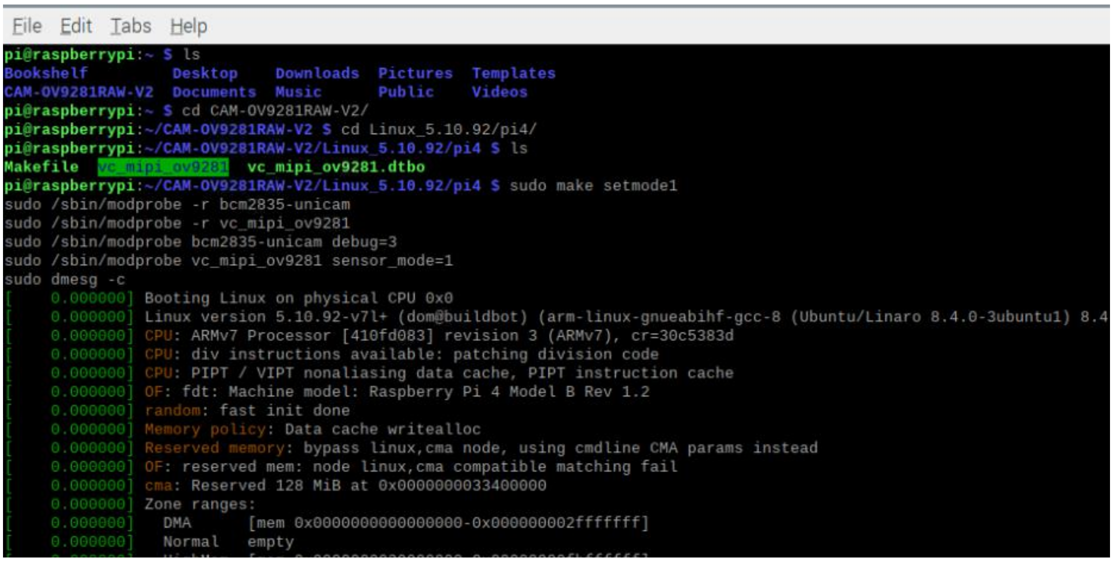
cat /sys/module/vc_mipi_ov9281/parameters/sensor_mode



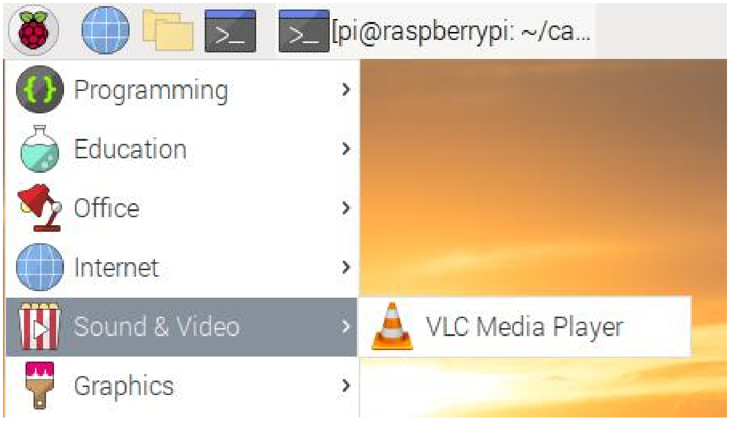
4.6 Qucik Test By VLC Tool

Step 1: Set mode 1

VLC only support the Y8 data format.

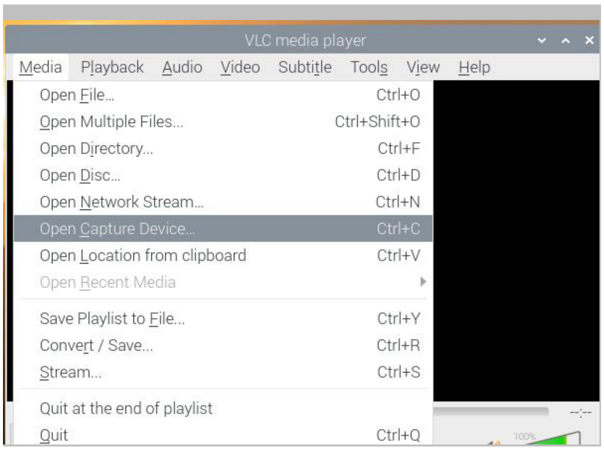


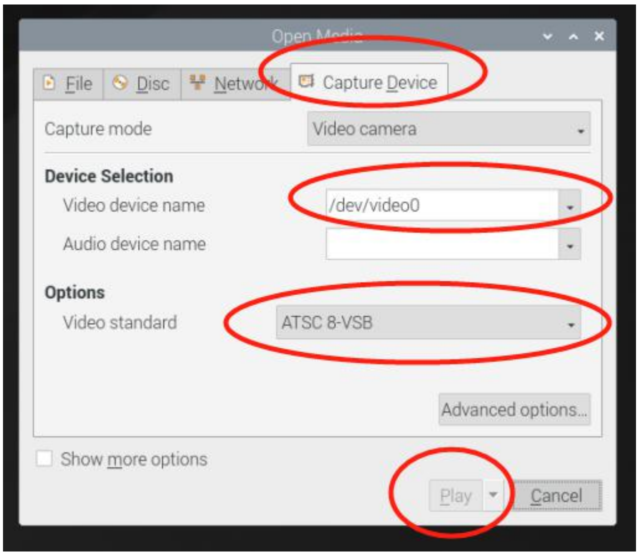
Step 2: Open VLC



Step 3: Setting VLC

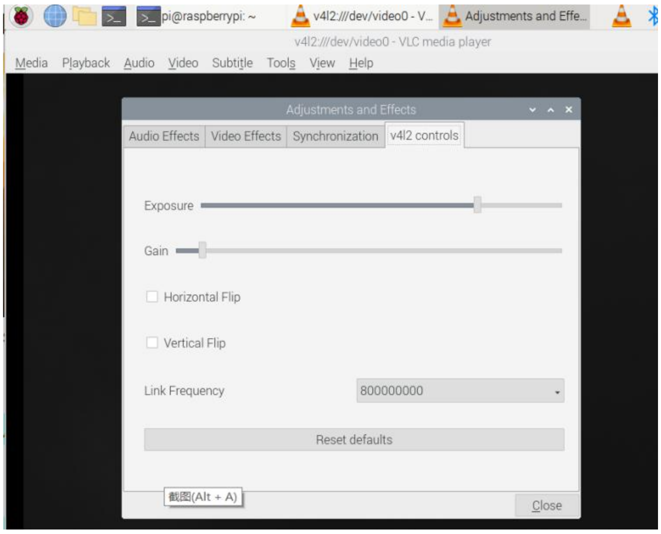
Click 'Media' → 'OpenCapture Device'→'Capture_Device', choose 'video0'. And click 'Play' you will see the image that collected by camera.





Step 4: Exposure/Gain

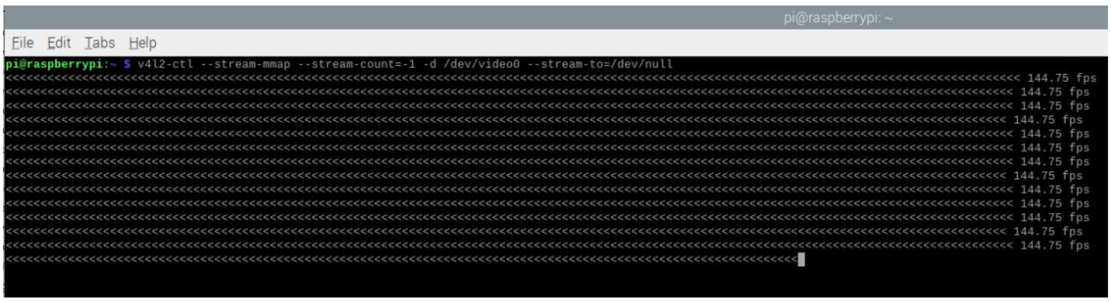
If you want to set the exposure and gain, click 'Tools' → 'effects and Filters' → 'V4l2 controls'



4.7 Frame Rate(fps) Test

Use below command, you can see frames-per-second information of your camera. You can set different modes to get different frame rate.

```
v4l2-ctl --stream-mmap --stream-count=-1 -d /dev/video0 --stream-to=/dev/null
```



4.8 Preview Function

Step 1: Set Mode0 or Mode1 (Mode4,Mode5,Mode8,Mode9)

Refer to the Chapter 4.5

Step 2: Read vcmipidemo tool help

Enter tools folder, .you could see 4 vcmipidemo files.

vcmipidemo-pi4 is for Pi4 + 32-bit system

vcmipidemo-pi4-arch64 is for Pi4 + 64-bit system

vcmipidemo-pi3 is for Pi3 + 32-bit system

vcmipidemo-pi3-arch64 is for Pi3 + 64-bit system

Use below command to read the vcmipidemo tool help

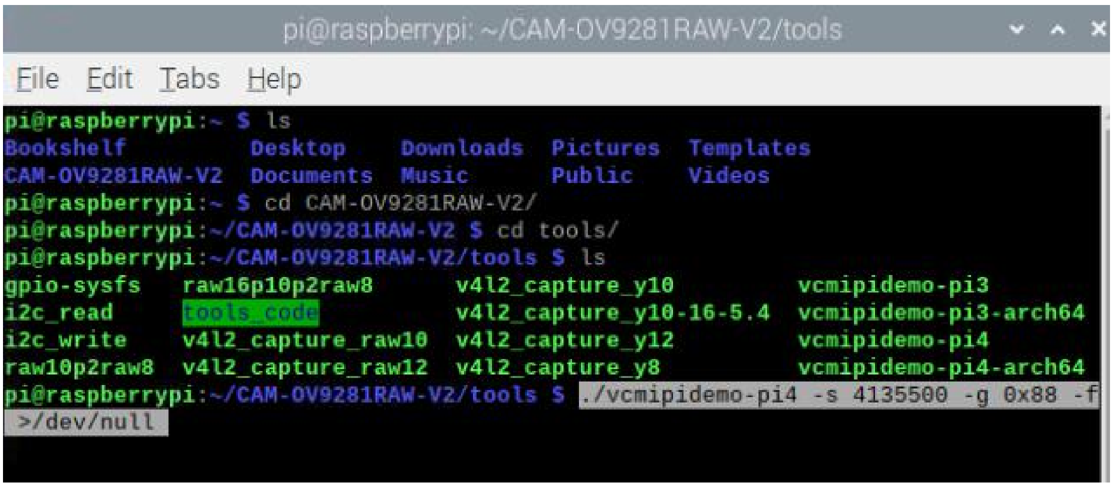
./vcmipidemo-pi4 --help

Usage: ./vcmipidemo [-s sh] [-g gain] [-f] [-a]

-s	Shutter Time. Value is from 8721ns to 8721*885ns, must be integral multiple of 8721ns . 8721xN(N =1,2,3,4,5.....855)
-g	Gain Value (0-254d)
-b	Buffer Count to use
-f	Output Capture to framebuffer /dev/fb0
-o	Output Captures to file in PGM or PPM format (openable by e.g. GIMP)
-a	Suppress ASCII capture at stdout.

Step 3: Use vcmipidemo tool.

./vcmipidemo-pi4 -s 4135500 -g 0x88 -f >/dev/null



Set shutter time = 4135500ns = 8721ns * 500cnt

Set gain = 0x88 db= 136 db

Note: If you are using Raspberry Pi 4**,** you need to Press 'Ctrl+Alt+F1' to start the preview windows (come back is Ctrl+Alt+F7) after you do follow step3 . Otherwise you can't see the preview windows. If you are using Raspberry Pi 3, no need to do that.

It case by the frame buffer difference between Pi3 and Pi4. Refer to the below link:
https://www.raspberrypi.org/forums/viewtopic.php?f=29&t=250564&hilit=Framebuffer+difference+in+RPi+4

4.11 Strobe Setting And Register Read/Write

Strobe function is also turn on by default in all modes.Strobe generates a pulse with a reference starting point at the time when the pixel array starts integration. Following a delay after the reference starting point, which is controlled by :

0x3921	PWM_CTRL_21	0x00	RW	Bit[7]: Shift direction Bit[6:0]: strobe_frame_shift[30:24]

address	register name	default value	R/W	description
0x3922	PWM_CTRL_22	0x00	RW	Bit[7:0]: strobe_frame_shift[23:16]
0x3923	PWM_CTRL_23	0x00	RW	Bit[7:0]: strobe_frame_shift[15:8]
0x3924	PWM_CTRL_24	0x05	RW	Bit[7:0]: strobe_frame_shift[7:0]

a width of strobe_frame_span[31:0] is generated.

0x3925	PWM_CTRL_25	0x00	RW	Bit[7:0]: strobe_frame_span[31:24]
0x3926	PWM_CTRL_26	0x00	RW	Bit[7:0]: strobe_frame_span[23:16]
0x3927	PWM_CTRL_27	0x00	RW	Bit[7:0]: strobe_frame_span[15:8]
0x3928	PWM_CTRL_28	0x1A	RW	Bit[7:0]: strobe_frame_span[7:0]

Step 1: Go Into The Tools Folders

```
pi@raspberrypi:~/cam-mipiov9281 $ cd tools
pi@raspberrypi:~/cam-mipiov9281/tools $ ls
 1.png          Capture2.png  'Tools code'          v4l2_capture_y8
 2.png          gpio-sysfs     v4l2_capture_raw10   vcmipidemo
 3.png          i2c_read       v4l2_capture_raw12
 Capture0.png   i2c_write      v4l2_capture_y10
 Capture1.png   raw10p2raw8    v4l2_capture_y12
pi@raspberrypi:~/cam-mipiov9281/tools $
```

Step 2: I2c tool read Register

$./i2c_read 0 0x60 [start addr of reg] [num of regs]

For example, Read value of register 0x3928

./i2c_read 10 0x60 0x3928 1

```
pi@raspberrypi:~/cam-mipiov9281/tools $ ./i2c_read 0 0x60 0x3928 1
Using i2C device /dev/i2c-0
====I2C read:<0x60> <0x3928> <0x1>====
Read i2c addr 60
addr 3928 : value 1a
pi@raspberrypi:~/cam-mipiov9281/tools $
```

Step 3: I2c tool write Register

$./i2c_write 10 0x60 [reg addr] [reg value]

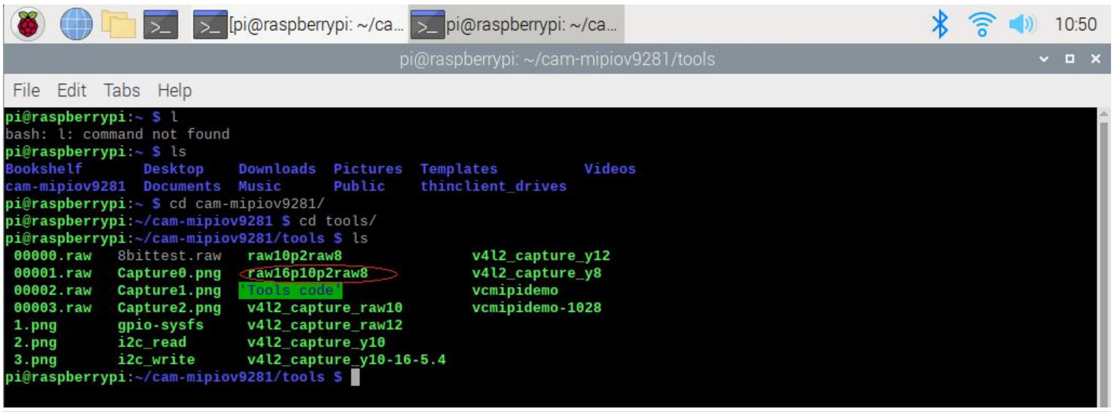
For example, Write 0x32 to register 0x3928.

./i2c_write 10 0x60 0x3928 0x32

```
pi@raspberrypi:~/cam-mipiov9281/tools $ ./i2c_write 0 0x60 0x3928 0x32
====I2C write:<0x60> <0x3928> <0x32>====
pi@raspberrypi:~/cam-mipiov9281/tools $
```

4.12 Raw 10 Change into Raw 8 And Display the image in Windows System

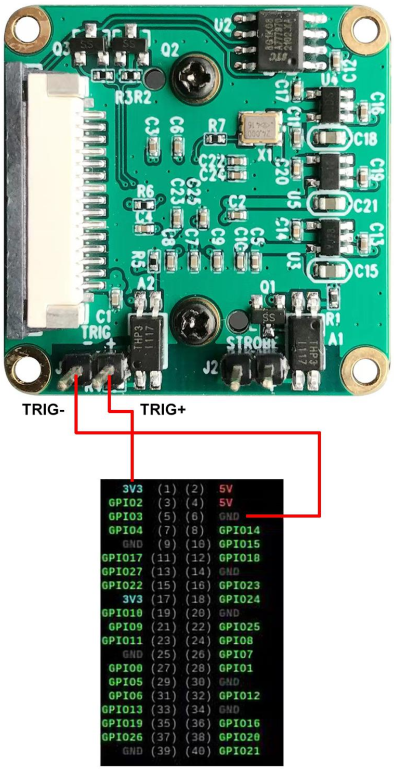
We provide a tools to help you change RAW10 to RAW8 in many applications.

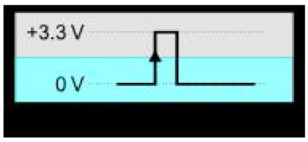


4.9 Capture Function

Step 1: Simulate the Trigger Signal

You can connect the TRIG- to the GND Pin and connect the TRIG+ to 3.3V Pin of Raspberry Pi to simulate a trigger signal. This test function will comes with repeated trigger signal sometime.





Step 2: Y8 Date Format Capture Example

Usage: ./v4l2_capture_y8 [-s sh] [-g gain] [-h f] [-v f] [-c cnt]

-s	Shutter Time. Value is from 8721ns to 8721*885ns, must be integral multiple of 8721ns . 8721xN(N =1,2,3,4,5.....855)
-g	Gain Value (0-254d)
-h	horizen flip 1: Enable 0:Disable
-v	vertical flip 1: Enable 0:Disable
-c	capture count

Step 1: Set Mode3(Y8, EXT_TRIG)

make setmode3

Refer to the chapter 4.5

Step 2: Enter capture setting:

./v4l2_capture_y8 -s 4135500 -g 0x88 -h 1 -v 1 -c 5

Set shutter time = 4135500ns = 8721ns * 500cnt

Set gain = 0x88 db= 136 db

For more detail please use below command

./v4l2_capture_y8 --help

```
pi@raspberrypi:~/cam-mipiov9281/tools $ ./v4l2_capture_y8 -s 4135500 -g 0x88 -h
1 -v 1 -c 5
sensor_set_parameters():  Old Gain Value: 16.
sensor_set_parameters():  Requested New Gain Value: 136.
sensor_set_parameters():  New Gain Value: 136.
sensor_set_parameters():  Old Exposure Value: 5939001.
sensor_set_parameters():  Requested New Exposure Value: 5000.
sensor_set_parameters():  New Exposure Value: 8721.
sensor_set_parameters():  Old Hflip Value: 0.
sensor_set_parameters():  Requested New Hflip Value: 0.
sensor_set_parameters():  New Hflip Value: 0.
sensor_set_parameters():  Old Vflip Value: 0.
sensor_set_parameters():  Requested New Vflip Value: 0.
sensor_set_parameters():  New Vflip Value: 0.
cam_init:113, req.count: 3
cam_init:133, buffer.length: 1024000
cam_init:134, buffer.m.offset: 0
cam_init:133, buffer.length: 1024000
cam_init:134, buffer.m.offset: 1024000
cam_init:133, buffer.length: 1024000
cam_init:134, buffer.m.offset: 2048000
cam_init:161, cam init done.
cam_get_image:188, dequeue done, index: 0
cam_get_image:190, copy done.
cam_get_image:198, enqueue done.
```

Step 3: Give a trigger signal voltage to J3 connector.

```
---
14 15 15 15 15 14 13 15 15 15 14 15 14 15 15 16
cam_get_image:188, dequeue done, index: 1
cam_get_image:190, copy done.
cam_get_image:198, enqueue done.
---
18 15 15 15 15 15 15 14 15 17 14 15 14 16 14 14
cam_get_image:188, dequeue done, index: 2
cam_get_image:190, copy done.
cam_get_image:198, enqueue done.
```

Step 4: Check the formed files.

You will see two files named '00000.raw' and '00001.raw'.

```
pi@raspberrypi:~/cam-mipiov9281/tools $ ls
00000.raw  1.png  3.png         Capture1.png  gpio-sysfs   v4l2_capture_raw10  v4l2_capture_y10  v4l2_capture_y8
00001.raw  2.png  Capture0.png  Capture2.png  raw10p2raw8  v4l2_capture_raw12  v4l2_capture_y12  vcmipidemo
pi@raspberrypi:~/cam-mipiov9281/tools $
```

Step 3: Y10 Date Format Capture Example

Usage: ./v4l2_capture_y10-16-5.4 [-s sh] [-g gain] [-h f] [-v f] [-c cnt]

-s	Shutter Time. Value is from 8721ns to 8721*885ns, must be integral multiple of 8721ns . 8721xN(N =1,2,3,4,5.....855)
-g	Gain Value (0-254d)
-h	horizen flip 1: Enable 0:Disable
-v	vertical flip 1: Enable 0:Disable
-c	capture count

Step 1: Set Mode2(Y10, EXT_TRIG)

make setmode2

Refer to the chapter 4.5

Step 2: Enter capture setting:

./v4l2_capture_y10-16-5.4 -s 4135500 -g 0x88 -h 1 -v 1 -c 5

Set shutter time = 4135500ns = 8721ns * 500cnt

Set gain = 0x88 db= 136 db

For more detail please use below command:

./v4l2_capture_y10-16-5.4 --help

```
pi@raspberrypi:~/Desktop/tools $ ./v4l2_capture_y10 -s 4135500 -g 0x88 -h 1 -v 1 -c 5
Setting Shutter Value to 4135500.
Setting Gain Value to 136.000000.
Horizen flip the captured image.
Vertical flip the captured image.
Capture 5 frame.
sensor_set_parameters():  Old Gain Value: 137.
sensor_set_parameters():  Requested New Gain Value: 136.
sensor_set_parameters():  New Gain Value: 136.
sensor_set_parameters():  Old Exposure Value: 24813.
sensor_set_parameters():  Requested New Exposure Value: 4135500.
sensor_set_parameters():  New Exposure Value: 4135500.
sensor_set_parameters():  Old Hflip Value: 1.
sensor_set_parameters():  Requested New Hflip Value: 1.
sensor_set_parameters():  New Hflip Value: 1.
sensor_set_parameters():  Old Vflip Value: 1.
sensor_set_parameters():  Requested New Vflip Value: 1.
sensor_set_parameters():  New Vflip Value: 1.
cam_init:111, req.count: 3
cam_init:131, buffer.length: 1280000
cam_init:132, buffer.m.offset: 0
cam_init:131, buffer.length: 1280000
cam_init:132, buffer.m.offset: 1282048
cam_init:131, buffer.length: 1280000
cam_init:132, buffer.m.offset: 2564096
cam_init:159, cam init done.
```

Step 3: Give a trigger signal voltage to J3 connector

```
cam_get_image:186, dequeue done, index: 0
cam_get_image:188, copy done.
cam_get_image:196, enqueue done.
---
1f 1f 1e 1c a4 1b 17 1e 1d fb 21 1d 1e 20 49 1e
cam_get_image:186, dequeue done, index: 1
cam_get_image:188, copy done.
cam_get_image:196, enqueue done.
---
19 1e 1f 1d 83 1e 17 1c 1d a9 1f 1e 1f 1a ce 1a
cam_get_image:186, dequeue done, index: 2
cam_get_image:188, copy done.
cam_get_image:196, enqueue done.
---
1d 1b 1a 1a b9 1c 15 1c 1c 6b 1e 1b 1c 1c 29 1a
```

Step 4: Check the formed files.

You will see two files named '00000.raw' and '00001.raw'.

```
pi@raspberrypi:~/Desktop/tools $ ls
00000.raw  00001.raw  00002.raw  v4l2_capture_raw10  v4l2_capture_raw12  v4l2_capture_y10  v4l2_capture_y12  v4l2_capture_y8
```

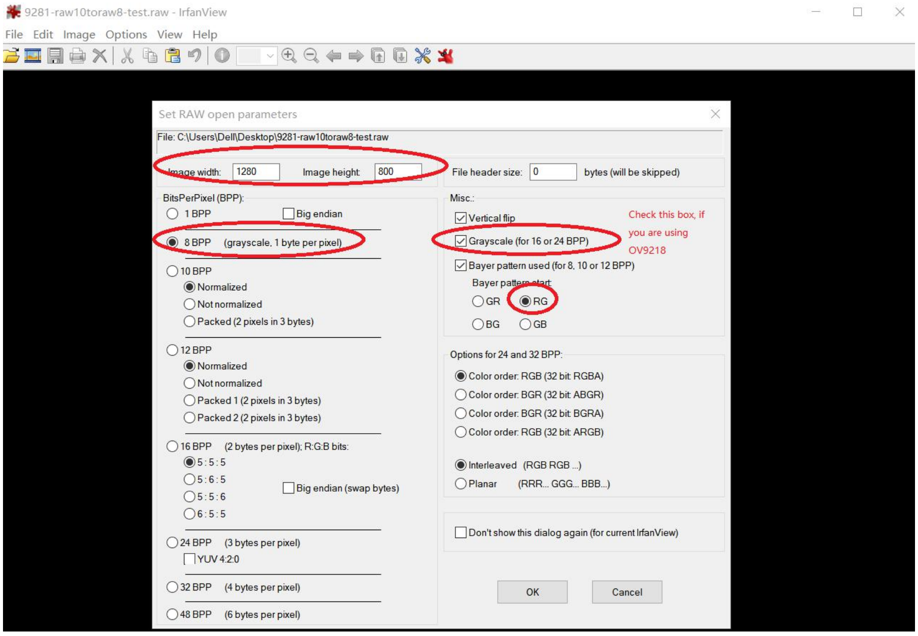
Step 1:Using the conversion tools

./raw16p10p2raw8 [raw 10 name] [raw8 name]

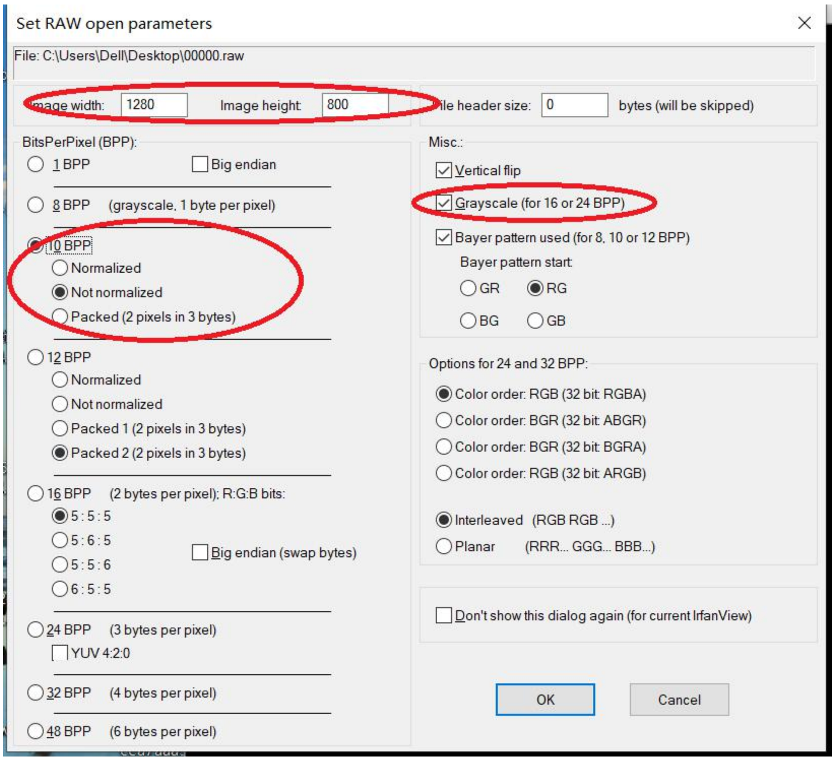
```
pi@raspberrypi: ~/ca...   pi@raspberrypi: ~/ca...
pi@raspberrypi: ~/cam-mipiov9281/tools
File  Edit  Tabs  Help
pi@raspberrypi:~ $ cd cam-mipiov9281/tools/
pi@raspberrypi:~/cam-mipiov9281/tools $ ls
 00000.raw   1.png         Capture1.png   i2c_write        v4l2_capture_raw10        v4l2_capture_y12
 00001.raw   2.png         Capture2.png   raw10p2raw8      v4l2_capture_raw12        v4l2_capture_y8
 00002.raw   3.png         gpio-sysfs     raw16p10p2raw8   v4l2_capture_y10          vcmipidemo
 00003.raw   Capture0.png  i2c_read       'Tools code'     v4l2_capture_y10-16-5.4   vcmipidemo-1028
pi@raspberrypi:~/cam-mipiov9281/tools $ ./raw16p10p2raw8 00000.raw 9281-raw10toraw8-test.raw
pi@raspberrypi:~/cam-mipiov9281/tools $ ls
 00000.raw   2.png                          Capture2.png   raw16p10p2raw8       v4l2_capture_y10-16-5.4
 00001.raw   3.png                          gpio-sysfs     'Tools code'         v4l2_capture_y12
 00002.raw   9281-raw10toraw8-test.raw      i2c_read       v4l2_capture_raw10   v4l2_capture_y8
 00003.raw   Capture0.png                   i2c_write      v4l2_capture_raw12   vcmipidemo
 1.png       Capture1.png                   raw10p2raw8    v4l2_capture_y10     vcmipidemo-1028
pi@raspberrypi:~/cam-mipiov9281/tools $
```

Step 2:Using the IrfanView 64 tools

Copy the '9281-raw10toraw8-test.raw' to Windows, use the IrfanView 64 set as below to get the image.

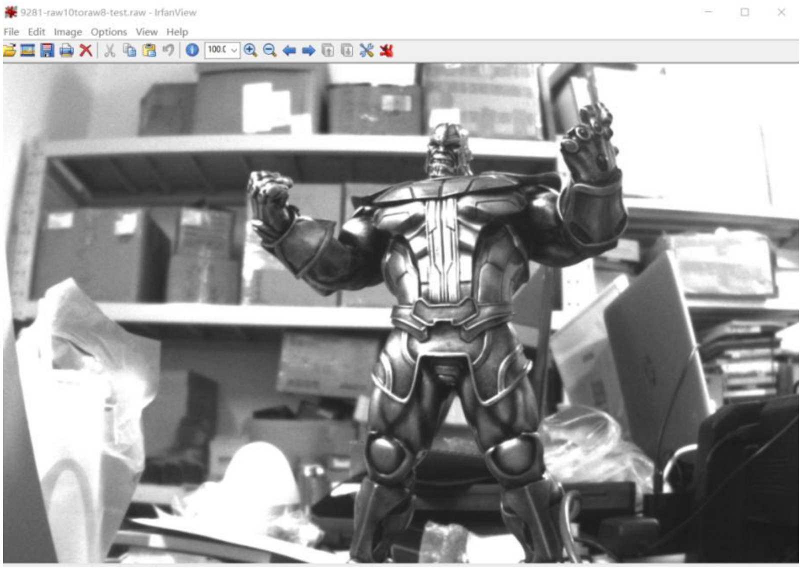


If you want to display the RAW10 image directly. Please set as below picture.



Step 3: View the results

We provide the raw10 format named '9281-raw10.raw' and raw8 format named '9281-raw10toraw8-test.raw' (After the conversion) on our wiki for you to test. below is the correct result.



5. Using Build-in Driver On Bullseye

5.1 Load Raspberry Pi image

Prepare a capacity of more than 8GB TF card(16Gb Class10 is better) and a card reader. Load the image file on to the SD card, using the instructions provided on the Raspberry Pi website for Linux, Mac or PC:
https://www.raspberrypi.org/documentation/installation/installing-images/README.md

Raspbian Image download:
https://www.raspberrypi.org/downloads/

5.2 Driver Sources Codes

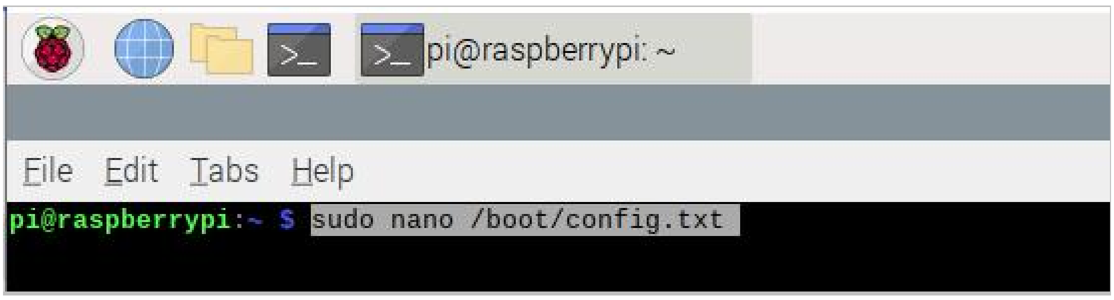
The open source driver on Raspbian:
https://github.com/raspberrypi/linux/blob/rpi-5.10.y/drivers/media/i2c/ov9281.c

This driver support 1280x800 ,1280x720,640x400 three resolution now and do not support trigger mode.

5.3 Dtoverlay

(1) Open the config.txt on Raspbian:

sudo nano /boot/config.txt



(2) Add the dtoverlay into the config.txt file,

dtoverlay=ov9281

```
[cm4]
# Enable host mode on the 2711 built-in XHCI USB controller.
# This line should be removed if the legacy DWC2 controller is required
# (e.g. for USB device mode) or if USB support is not required.
otg_mode=1

[all]

[pi4]
# Run as fast as firmware / board allows
arm_boost=1

[all]
dtoverlay=ov9281
```

(3) And then press ctrl+ x to exit nad press 'y' to save.

```
Save modified buffer?
 Y Yes
 N No           ^C Cancel
```

(4) Rebooted your Pi

sudo reboot

(5) Use below command to check the camera is ready.

ls /dev/video0

Successful:

```
pi@raspberrypi: ~
File  Edit  Tabs  Help
pi@raspberrypi:~ $ ls /dev/video0
/dev/video0
pi@raspberrypi:~ $
```

Unsuccessful:

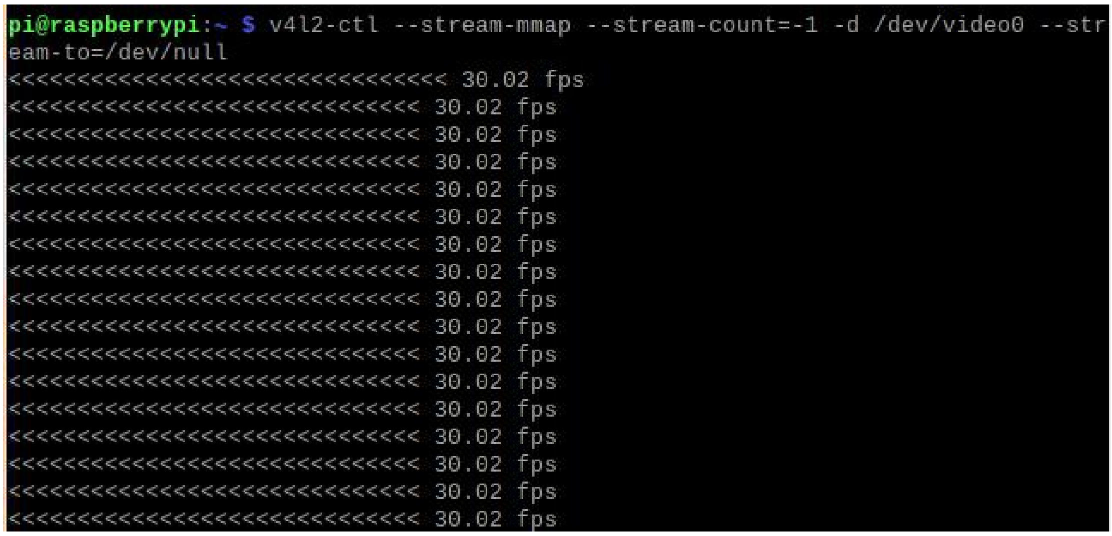
```
pi@raspberrypi: ~
File  Edit  Tabs  Help
pi@raspberrypi:~ $ ls /dev/video0
ls: cannot access '/dev/video0': No such file or directory
pi@raspberrypi:~ $
```

5.4 Frame Rate(fps) Test

Use below command, you can see frames-per-second information of your camera.

`v4l2-ctl --stream-mmap --stream-count=-1 -d /dev/video0 --stream-to=/dev/null`

30 fps:



```
pi@raspberrypi:~ $ v4l2-ctl --stream-mmap --stream-count=-1 -d /dev/video0 --str
eam-to=/dev/null
<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<< 30.02 fps
<<<<<<<<<<<<<<<<<<<<<<<<<<<<<< 30.02 fps
<<<<<<<<<<<<<<<<<<<<<<<<<<<<<< 30.02 fps
<<<<<<<<<<<<<<<<<<<<<<<<<<<<<< 30.02 fps
<<<<<<<<<<<<<<<<<<<<<<<<<<<<<< 30.02 fps
<<<<<<<<<<<<<<<<<<<<<<<<<<<<<< 30.02 fps
<<<<<<<<<<<<<<<<<<<<<<<<<<<<<< 30.02 fps
<<<<<<<<<<<<<<<<<<<<<<<<<<<<<< 30.02 fps
<<<<<<<<<<<<<<<<<<<<<<<<<<<<<< 30.02 fps
<<<<<<<<<<<<<<<<<<<<<<<<<<<<<< 30.02 fps
<<<<<<<<<<<<<<<<<<<<<<<<<<<<<< 30.02 fps
<<<<<<<<<<<<<<<<<<<<<<<<<<<<<< 30.02 fps
<<<<<<<<<<<<<<<<<<<<<<<<<<<<<< 30.02 fps
<<<<<<<<<<<<<<<<<<<<<<<<<<<<<< 30.02 fps
<<<<<<<<<<<<<<<<<<<<<<<<<<<<<< 30.02 fps
<<<<<<<<<<<<<<<<<<<<<<<<<<<<<< 30.02 fps
<<<<<<<<<<<<<<<<<<<<<<<<<<<<<< 30.02 fps
```

5.5 Libcamera

libcamera is an open source Linux community project. More information is available at the libcamera website:
https://libcamera.org/

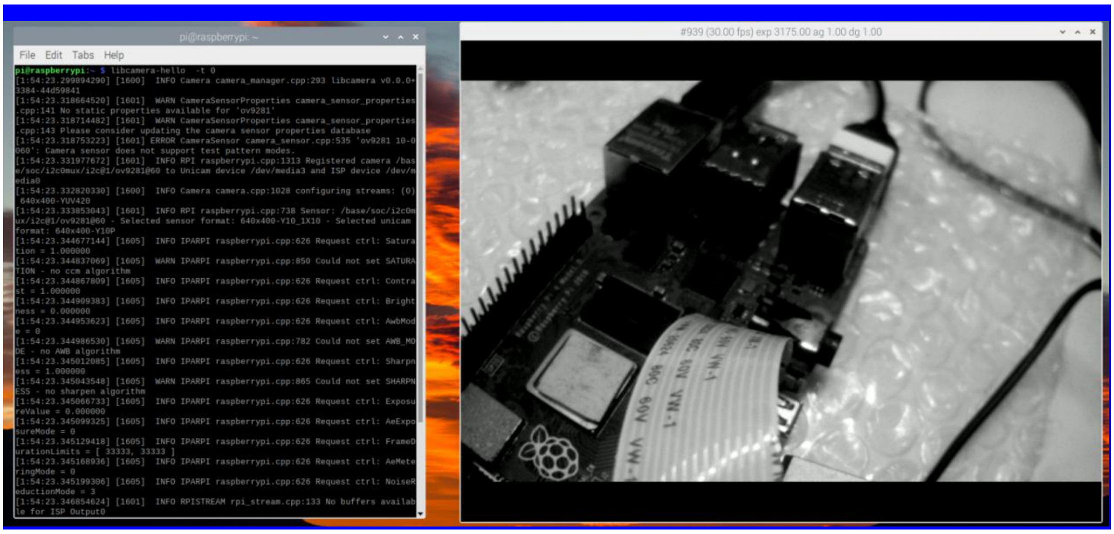
The libcamera source code can be found and checked out from the official libcamera repository.
https://git.linuxtv.org/libcamera.git/

When running a Raspberry Pi OS based on Bullseye, the 5 basic libcamera-apps are already installed. In this case, official Raspberry Pi cameras will also be detected and enabled automatically. Below we only take 'libcamera-hello' for example. For more information, please refer to below link：

https://www.raspberrypi.com/documentation/accessories/camera.html#binary-packages

```
libcamera-hello
```

```
libcamera-hello -t 0
```



6. Versions Description

Version	Description	Date	E-mail
V1.0		2022.01.02	support@inno-maker.com calvin@inno-maker.com

If you have any suggestions, ideas, codes and tools please feel free to email to me. I will update the user manual and record your name and E-mail in list. Look forward to your letter and kindly share.