IMX296 Sensor Module User Manual

Date	Revision	Change Details
2025/9/8	V2.0.0	First Released
2025/10/30	V2.0.1	Add Usage for external trigger and EEPROM
2026/02/04	V2.0.2	Add Description for Dual Camera Trigger, Strobe

Document Purpose

This Document is for innomaker imx296 mono version camera module and imx296 color version camera module.

Product Name	Sensor Name	Sensor Description
CAM-IMX296Mono-GS (Old Name: CAM-IMX296RAW)	**IMX296LLR**	Mono Sensor Sony Official Description
CAM-IMX296Color-GS	**IMX296LQR**	Color Sensor Sony Official Description

Normally We will update our development Mannual here
https://github.com/INNO-MAKER/cam-imx296raw-trigger
https://www.inno-maker.com/product/cam-mipi296raw-trigger/

1 Module Overview

InnoMaker IMX296 Sensor Module is a Global Shutter Camera with IMX296LLR-C CMOS Sensor Module for raspberry pi, compatible with raspberry pi build in driver. It supports up to 60fps at 1456×1088 Pixels operate with shorter exposure times down to 30μs, given enough light than a rolling shutter camera, which makes it useful for high-speed photography.

1.1 Module Features

- Support up to 60fps@1456×1088, totally Compatible with raspberry pi GS camera;
- Support M12 Len/ CS Len, comes with 1xM12 Wide angle lens/Cs Lens Seat.
- Support output format YUV with Resolution 1456*1088 up to 60fps.
- Support Hardware External Trigger.
- Support Strobe Function.

1.2 Sensor Overview:

This chip operates with analog 3.3V, digital 1.2V, and interface 1.8V triple power supply:

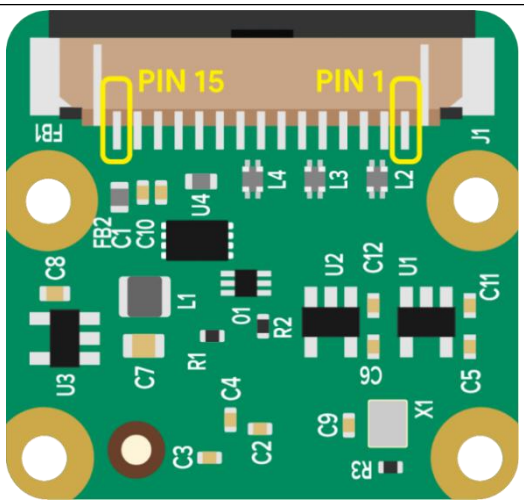
- **low power consumption.**

- **High sensitivity, low dark current and low PLS characteristics are achieved. (Applications: Sensing**

IMX296LLR / IMX296LQR-C	
FPS (Sensors):	60.3 fps
Pixel Size (Sensors):	3.4μm x 3.4μm
Resolution (Sensors):	1.58M
Scan/Series:	Pregius
Shutter (Sensors):	Global Shutter
Signal (Sensors):	**IMX296LLR** Mono **IMX296LQR** Color
Sensor Size	1/2.9 Diagonal 6.3 mm

2 Hardware Description

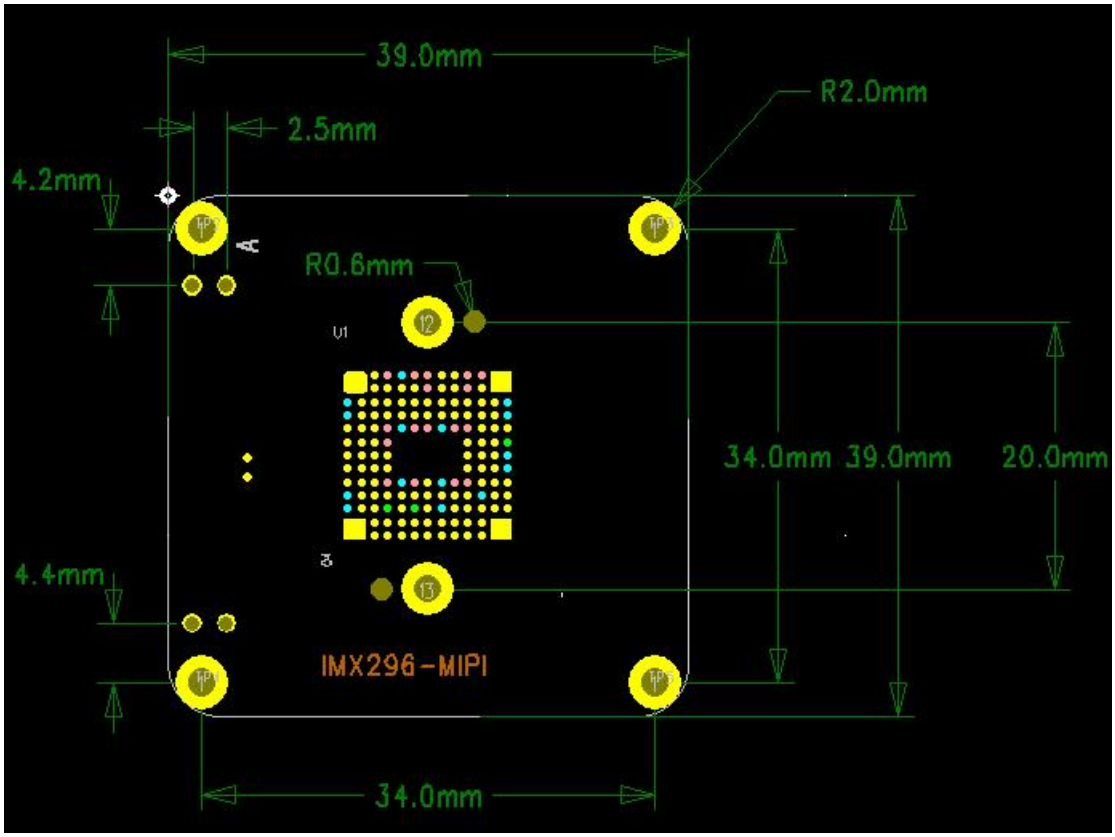
2.1 Pins Out Table(J1)



Pin #	Name	Description
1	GND	Ground

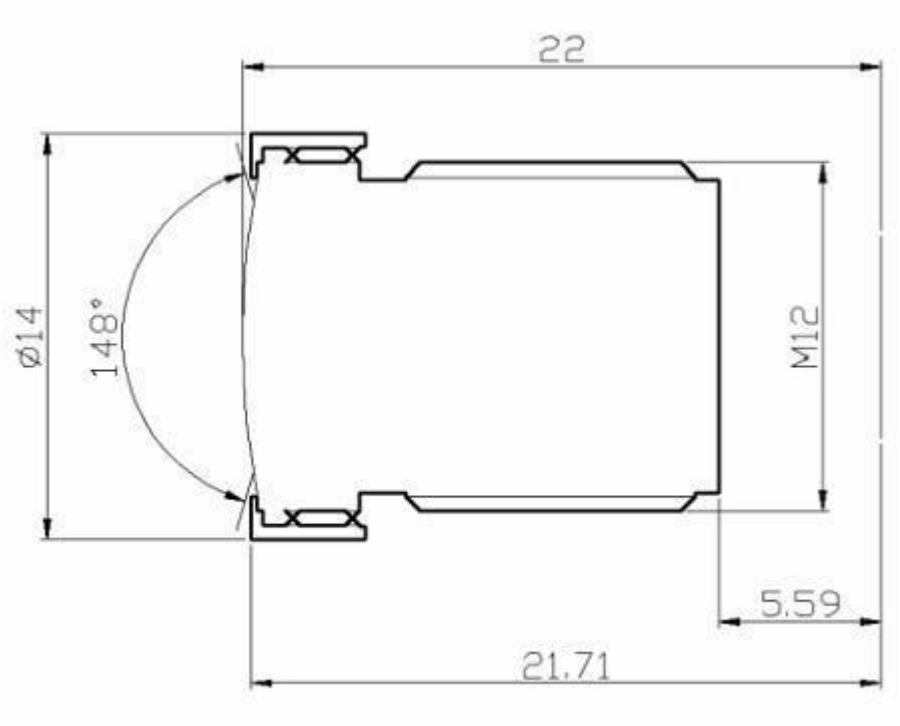
2	CAM_D0_N	MIPI Data Lane 0 Negative
3	CAM_D0_P	MIPI Data Lane 0 Positive
4	GND	Ground
5	CAM_D1_N	MIPI Data Lane 1 Negative
6	CAM_D1_P	MIPI Data Lane 1 Positive
7	GND	Ground
8	CAM_CK_N	MIPI Clock Lane Negative
9	CAM_CK_P	MIPI Clock Lane Positive
10	GND	Ground
11	CAM_IO0	Power Enable
12	CAM_IO1	LED Indicator
13	CAM_SCL	I2C SCL
14	CAM_SDA	I2C SDA
15	CAM_3V3	3.3V Power Input

2.2 Camera Size



2.3 Camera Lens

M12 Len

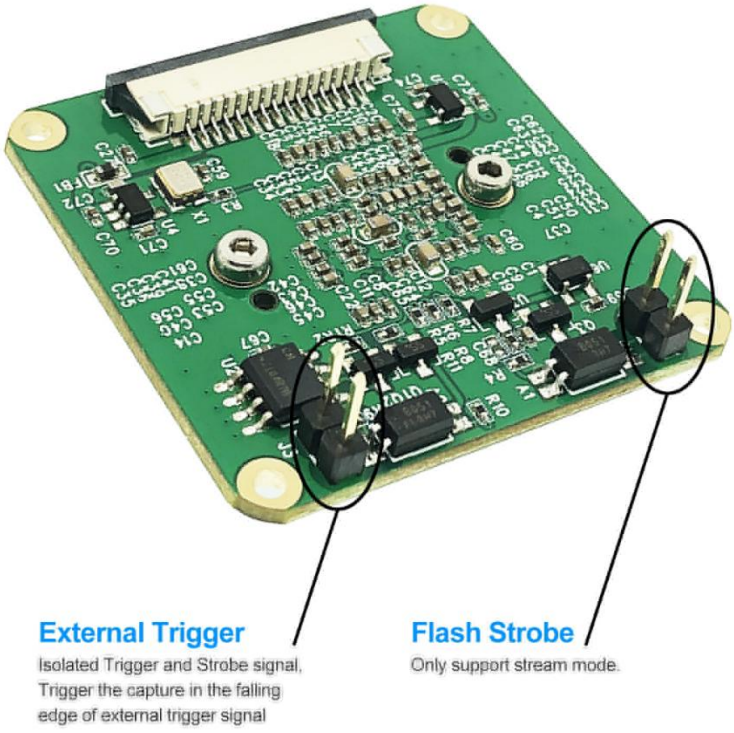


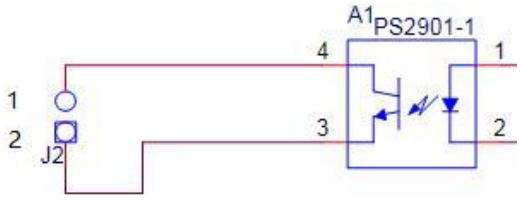
- Interface: M12
- Field of view Fov(D) = wide angle
- Focal Length 2.8 mm
- Focal Distance Adjustable
- TV DISTORTION <-17%
- F(N) /Aperture 2.2
- IR-Filter
  - Yes for Color Version
  - No for Mono Version

M12 LEN Seat

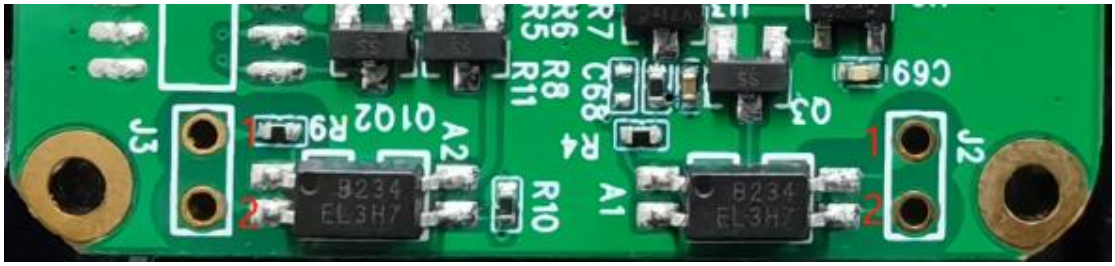
CS Len Seat

2.4 Flash Strobe Pins(J2)



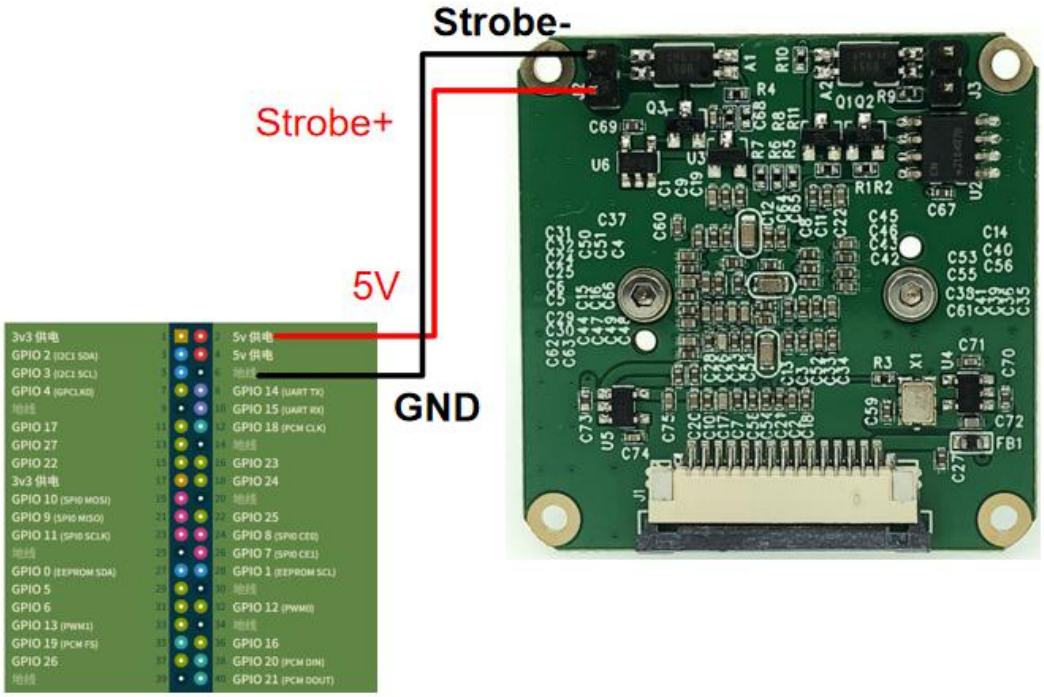


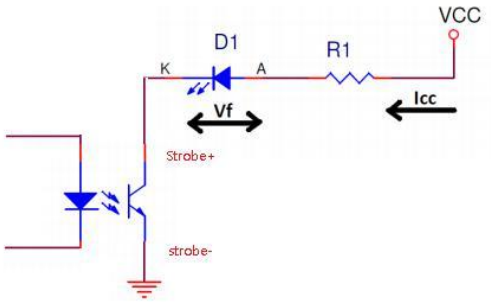
ISO FLASH



J2 PIN	Symbol
1	STROB+
2	STROB-

2.4.1 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

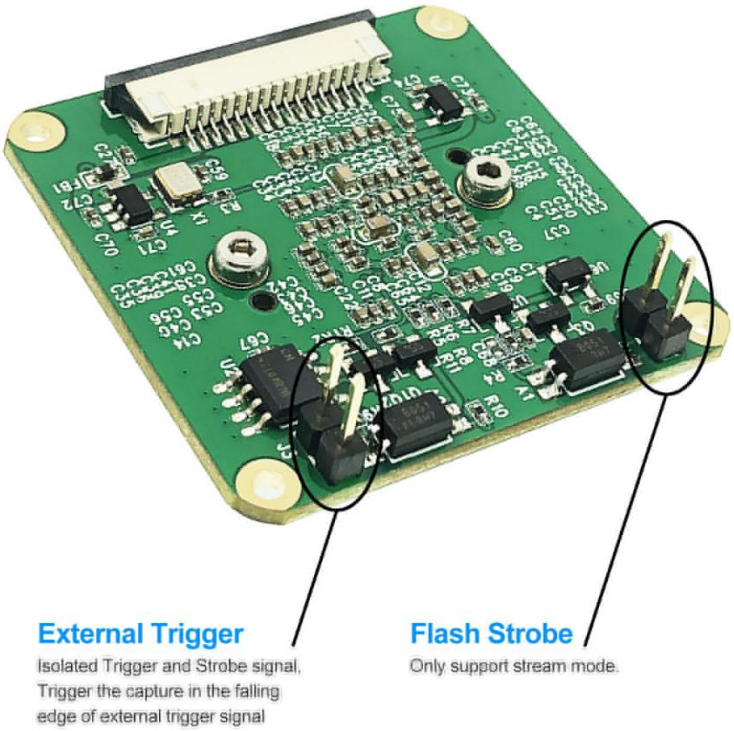
The value of series resistor: R1 = (VCC- Vf - VCE) / If

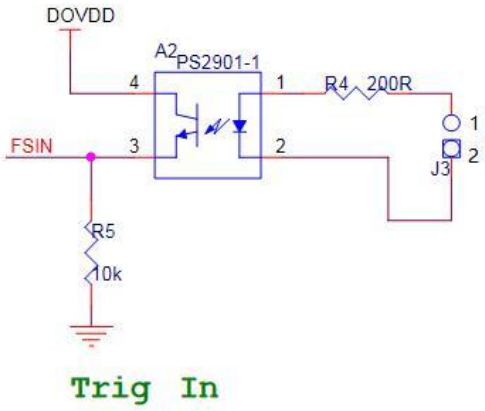
- VCC: system Voltage
- Vf: Forward voltage of Flash LED for current Icc
- VCE: Collection Emitter voltage, typical:0.1V

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.

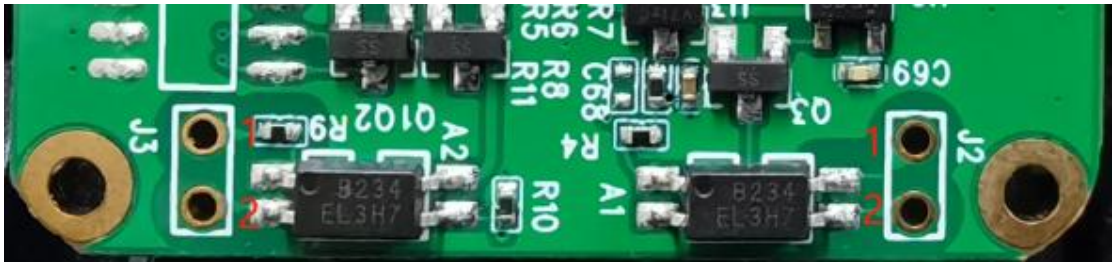
2.5 External TRIG Pins(J3)



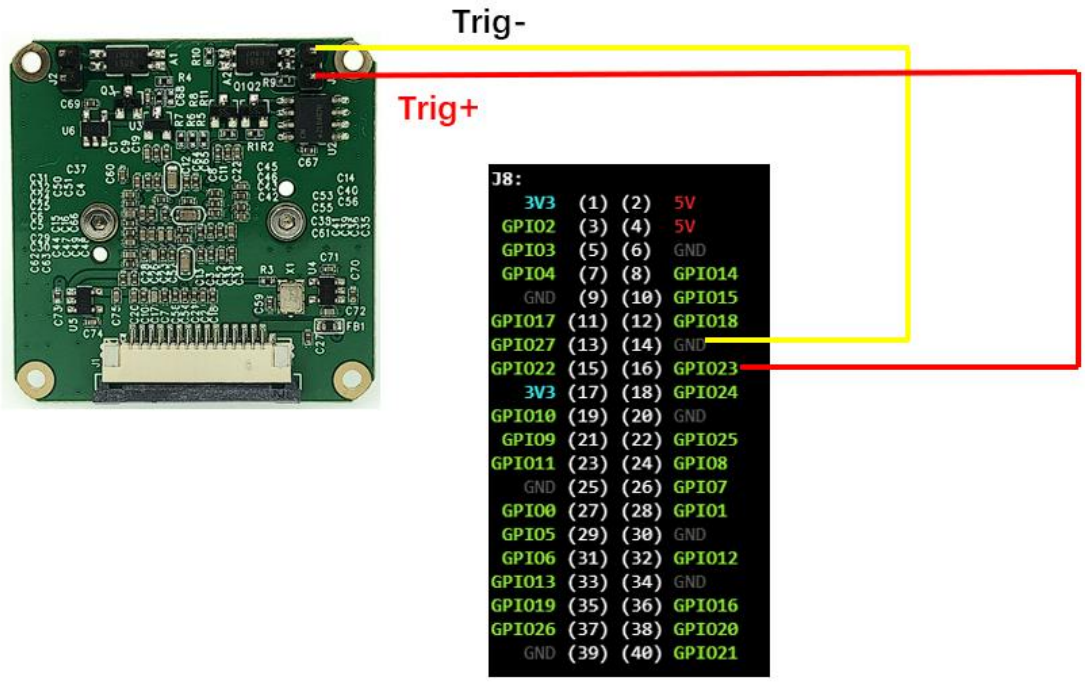


Trig In

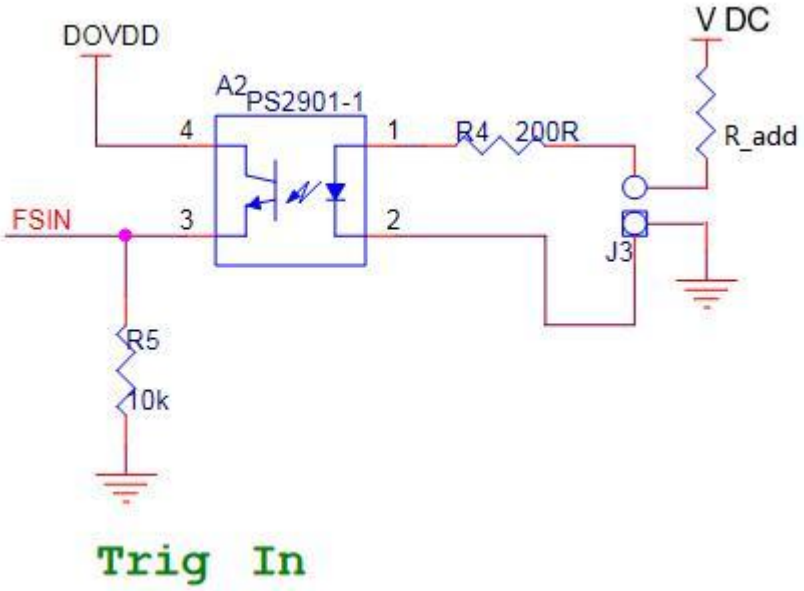
Support: support@inno-maker.com
Bulk Price: sales@inno-maker.com
Website: www.inno-maker.com
Github: https://github.com/INNO-MAKER



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



2.5.1 Reference Circuit



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

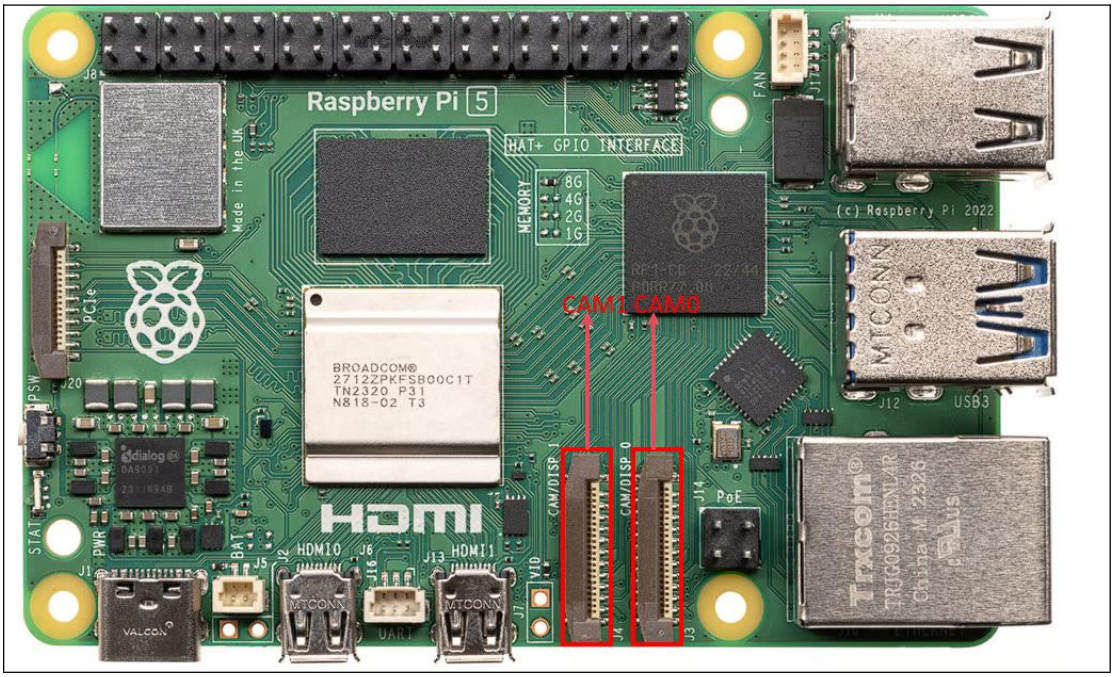
The calculations done here are based on 12VDC. Please do follow these calculations for other voltages like 24VDC.

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 = 200Ω), So the R_add = R1 - R4 = 537.5 - 200 = 337.5Ω

3 Quick Start Guide

Connection



3.1 Modify config.txt

```
sudo nano /boot/firmware/config.txt
```

older os it should be `sudo nano /boot/ config.txt`

3.2 Add dtoverlay

For CAM1 Interface

```
dtoverlay=imx296, cam1
```

For CAM0 Interface

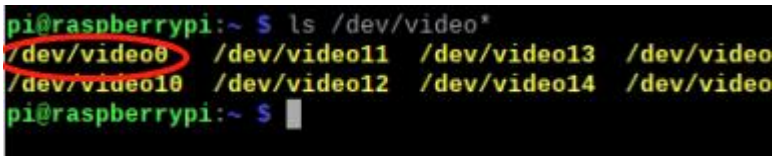
```
dtoverlay=imx296, cam0
```

Change camera_auto_detect=1 to camera_auto_detect=0
Save file and reboot.

3.2 Preview

Reboot and check camera Status

```
$ls /dev/video*
```




```
$rpicam-hello --width 1456 --height 1088 -t 0
```

3.3 libcamera-apps

More about libcamera and libcamera-apps Please Refer:

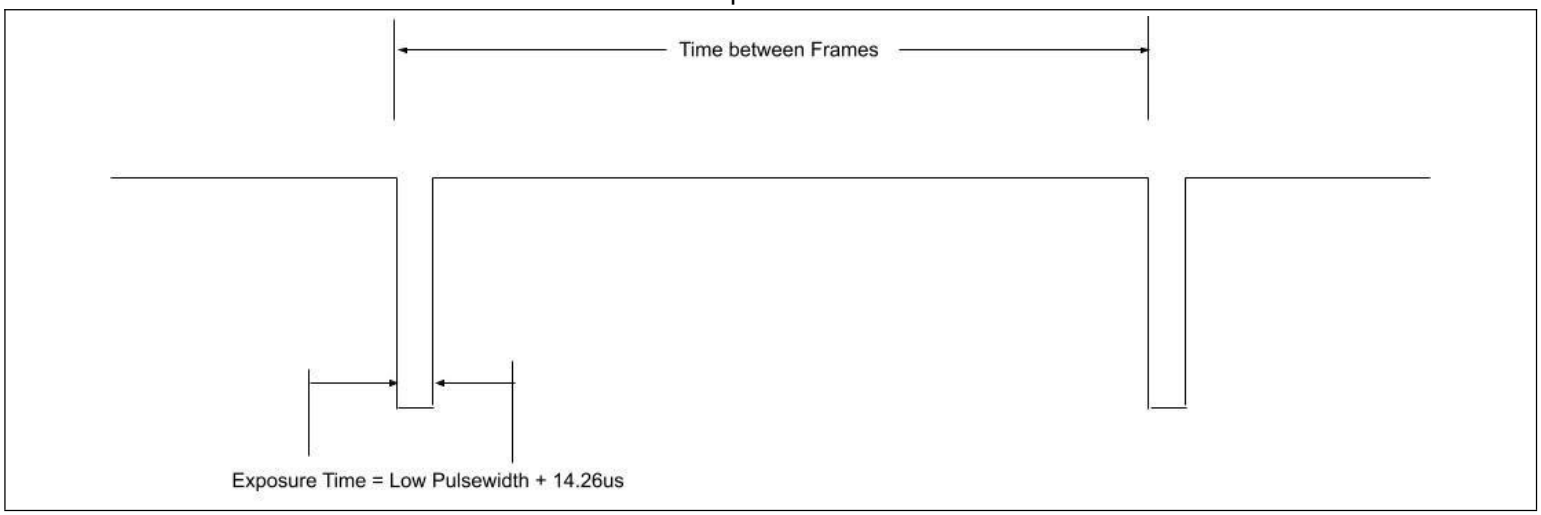
https://www.raspberrypi.com/documentation/computers/camera_software.html#libcamera-and-libcamera-apps

4 External Trigger

4.1 Description

- The Global Shutter (GS) camera can be triggered externally by pulsing the external trigger (denoted on the board as XTR（Trig+）,GND(Trig-)) connection on the board. Multiple cameras can be connected to the same pulse, allowing for an alternative way to synchronize two cameras.

- The exposure time is equal to the low pulse-width time plus an additional 14.26us. i.e. a low pulse of 10000us leads to an exposure time of 10014.26us. Framerate is directly controlled by how often you pulse the pin. A PWM frequency of 30Hz will lead to a framerate of 30 frames per second.

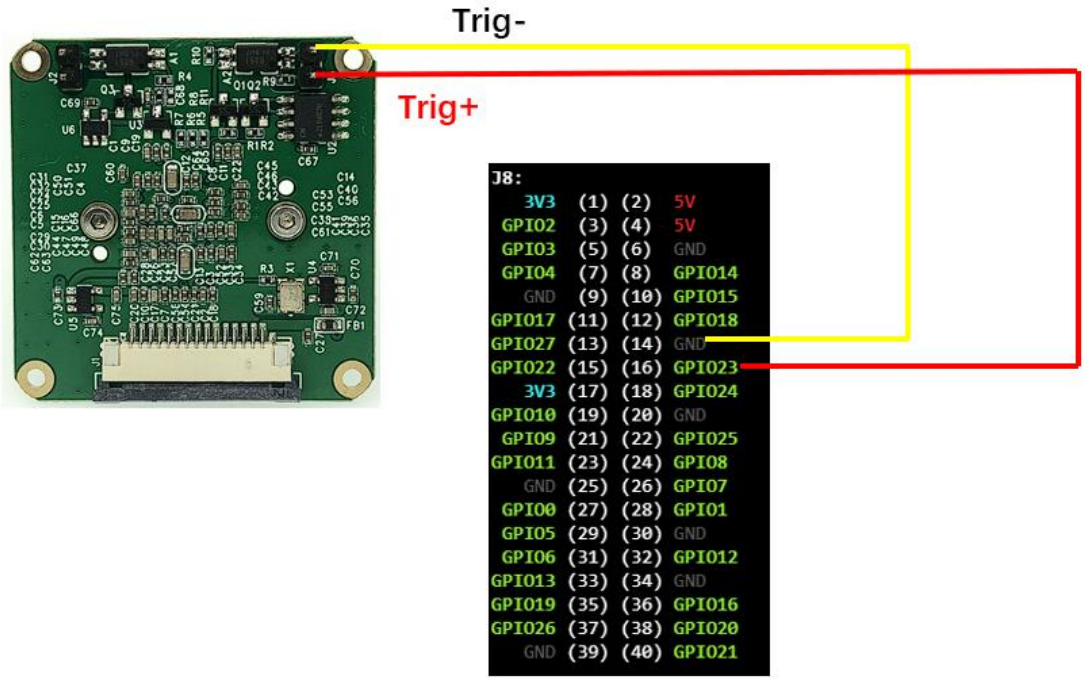


External Trigger Function

PULSE2_EN_NOR	79h (3079h)	[0]	0	Pulse2 enable in normal mode 0: disable 1: enable
PULSE2_EN_TRIG		[1]	0	Pulse2 enable in trigger mode 0: disable 1: enable
PULSE2_POL		[2]	0	Pulse2 polarity selection 0: High active 1: Low active
		[3]	0	Fixed to 1
PULSE2_UP [19:0]	7Ch (307Ch)	[7:0]	00000h	Pulse2 active period start timing setting Designated in line units from reference point
	7Dh (307Dh)	[7:0]		
	7Eh (307Eh)	[3:0]		
PULSE2_DN [19:0]	80h (3080h)	[7:0]	00000h	Pulse2 active period end timing setting Designated in line units from reference point
	81h (3081h)	[7:0]		
	82h (3082h)	[3:0]		

4.2 Trigger Pins Wire

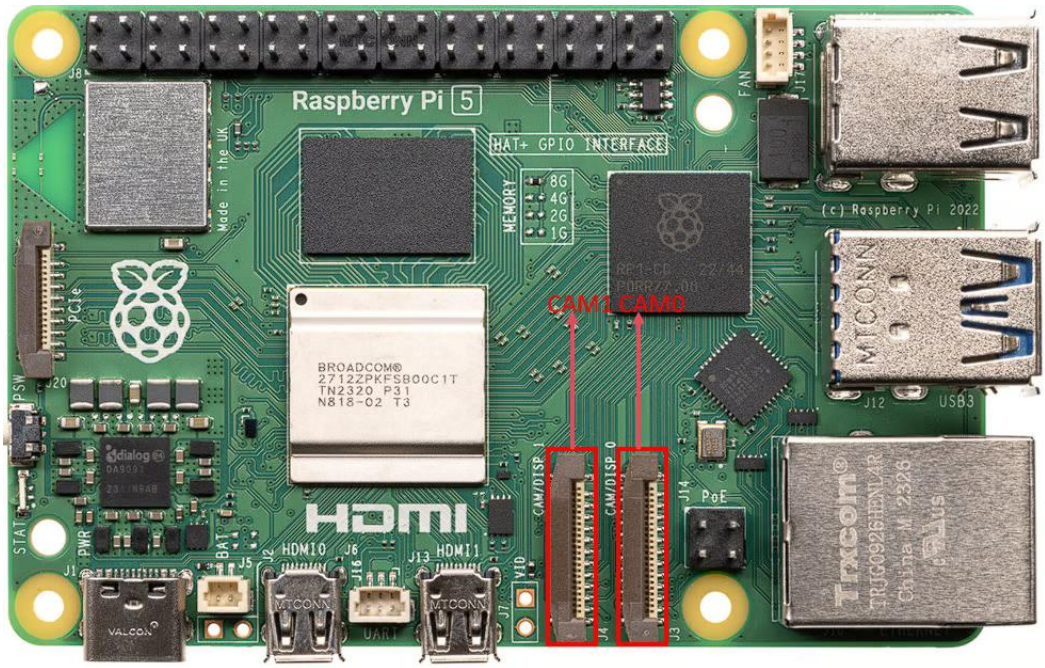
Follow Chapter 2.5



4.2 Download source from our GitHub

We use raspberry pi5 as example, showing dual Camera in free working:

```
sudo git clone https://github.com/INNO-MAKER/cam-imx296raw-trigger.git
cd cam-imx296raw-trigger
sudo chmod -R a+rwx *
```



4.3 Setting config.txt

```
sudo nano /boot/firmware/config.txt
```

Add below content to last line follow [all]

```
dtoverlay=imx296,always-on,cam0
dtoverlay=imx296,always-on,cam1
```

4.4 Timeout Setting

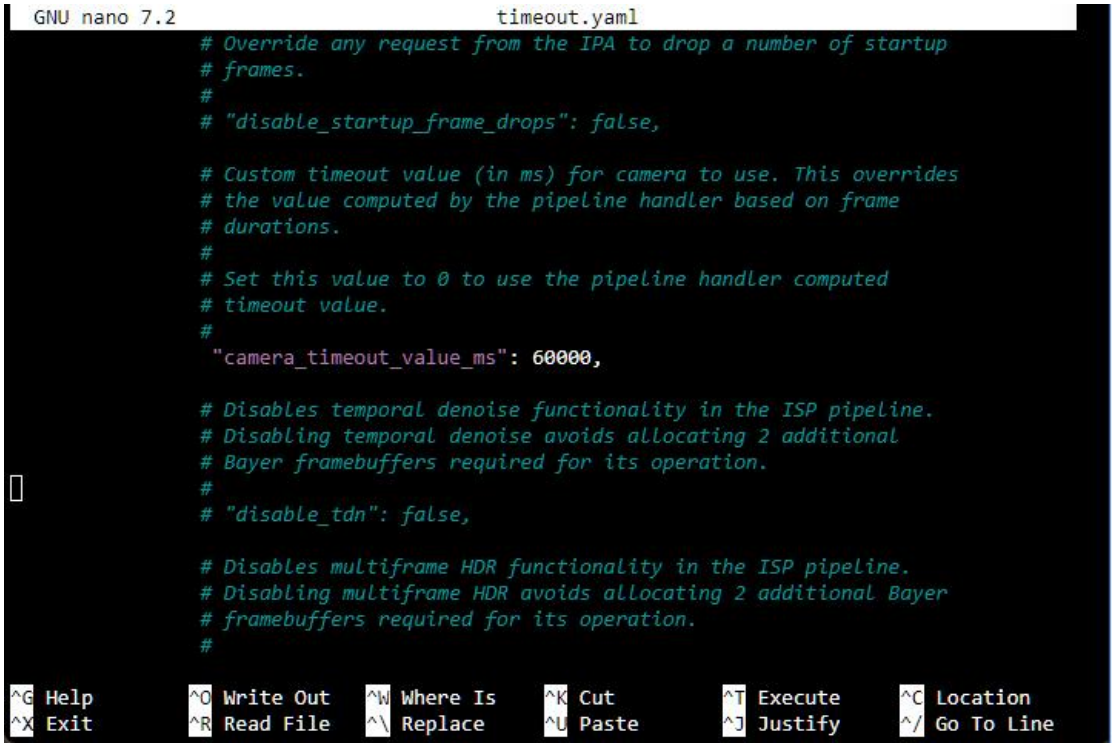
4.4.1 timeout.yaml

```
sudo cp /usr/share/libcamera/pipeline/rpi/pisp/example.yaml timeout.yaml
```

4.4.2 Edit timeout.yaml

```
sudo nano timeout.yaml
```

delete the # (comment) from the "camera_timeout_value_ms": line, and change the number from 0 to 60000,



4.4.3 Export timeout.yaml

```
export LIBCAMERA_RPI_CONFIG_FILE=timeout.yaml
```

4.5 Trigger Tool

InnoMaker provide Trigger tool which can enable dual imx296 sensor Module trigger mode individually.

4.5.1 Usage Method

```
./imx296_trigger    [i2c bus] [on/off]
```

4.5.2 PI5 Usage

Note: on pi5

```
Camera 1 i2c bus =4
camera 0 i2c bus =6
on=1
off=0
```

Example:

```
camera 1 trigger on:
./imx296_trigger 4    1

camera 1 trigger off:
./imx296_trigger 4    0

camera 0 trigger on:
./imx296_trigger 6    1

camera 0 trigger off:
./imx296_trigger 6    0
```

4.6 Dual Camera Trigger Mode Sample

Make two cameras work in free running mode/trigger Mode

4.6.1 Camera 0

Open terminal Window Enable timeout CAM0 Stream on (1)

```
cd cam-imx296raw-trigger    # timeout.yaml folder
export LIBCAMERA_RPI_CONFIG_FILE=timeout.yaml     #enable timeout setting (Must)
rpicam-hello -t 0 --camera 0      # free running
```

Open another terminal Window Set trigger Mode (2)

```
cd cam-imx296raw-trigger # trigger Tool folder
./imx296_trigger 6    1        # Set CAM0 trigger Mode
```

4.6.2 Camera 1

Open terminal Window Enable timeout CAM1 Stream on (3)

```
cd cam-imx296raw-trigger    # timeout.yaml folder
export LIBCAMERA_RPI_CONFIG_FILE=timeout.yaml     #enable timeout setting (Must)
rpicam-hello -t 0 --camera 0      # free running
```

Open another terminal Window Set trigger Mode (4)

```
cd cam-imx296raw-trigger # trigger Tool folder
./imx296_trigger 4 1          # Set CAM1 trigger Mode
```

4.6.3 Trigger signal

Open New terminal Window to give trigger signal (5)

For Trixie OS

```
sudo gpioset -t 30ms,2000ms -c 0 23=0      # 0.5fps
```

Or run scripts

For bookworm

We use Raspberry PI5 PIN23,GND GPIO as trigger signal and send signal by below script, Script code is as below, save it to .sh file.

```
while true;do
        gpioset gpiochip0 23=1
        sleep 1.9999
        gpioset gpiochip0 23=0
        sleep 0.0033
done
```

run the script

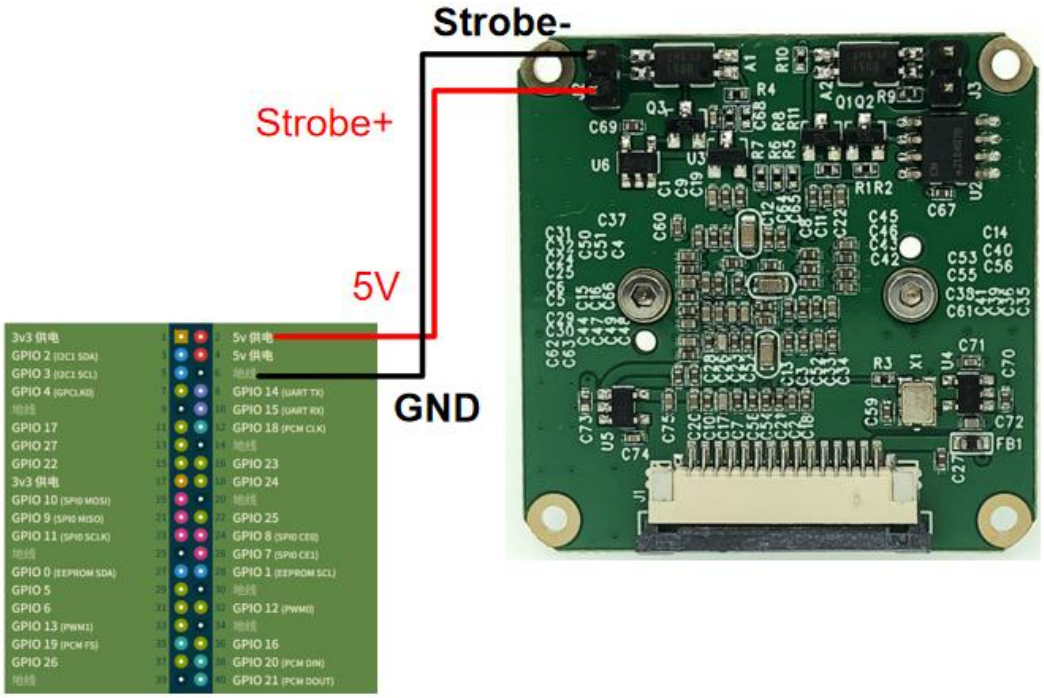
```
pi@raspberrypi:~/cam-imx296raw-trigger $ ./imx296.sh
```

5 Strobe Mode

- IMX296 official driver that provide by RPI default kernel not enable strobe by default.
- Imx296 can output strobe while work in normal or fast trigger mode, We can enable strobe by i2c tools.
- Strobe settings must be applied while the camera stream is off (no libcamera, rpicam, v4l2 streaming, or preview running).

5.1 Strobe Pins Wire

Follow chapter 2.4



5.2 I2c Tools download

```
sudo git clone https://github.com/INNO-MAKER/cam-imx296raw-trigger.git
cd cam-imx296raw-trigger
sudo chmod -R a+rwx *
```

Or Download from below link

i2c-tools-arch32.zip
i2c-tools-arch64.zip

5.3 I2C Tools Use Method

(confirm with ls /dev/i2c-* and i2cdetect -y 4 / i2cdetect -y 6 — you should see 0x1a).

Imx296 I2c address: 0x1a
Pi5 csi 1 i2c bus address : i2c-4
Pi5 csi 0 i2c bus address : i2c-6

5.3.1 Write common setup registers (always required):

./i2c_write 4 0x1a 0x3026 0x0F
./i2c_write 4 0x1a 0x3029 0x21

5.3.2 Choose one mode and write the enable registers:

A. Trigger mode + Strobe (recommended when using external trigger on J3):
./i2c_write 4 0x1a 0x306D 0x02 # trigger mode strobe enable
./i2c_write 4 0x1a 0x3079 0x0A # trigger mode strobe enable (second register)

B. Normal / Stream mode + Strobe (continuous streaming without external trigger):
./i2c_write 4 0x1a 0x306D 0x01 # normal mode strobe enable
./i2c_write 4 0x1a 0x3079 0x09 # normal mode strobe enable (second register)

Set Strobe timing (start and end points – default values from manual):
These define when the strobe pulse starts and ends relative to the frame (in line units).
First group (start point):
./i2c_write 4 0x1a 0x3070 0x00
./i2c_write 4 0x1a 0x3071 0x00
./i2c_write 4 0x1a 0x3072 0x00
First group (end point):
./i2c_write 4 0x1a 0x3074 0x2C
./i2c_write 4 0x1a 0x3075 0x01
./i2c_write 4 0x1a 0x3076 0x00

Second group (start point):
./i2c_write 4 0x1a 0x307C 0x00
./i2c_write 4 0x1a 0x307D 0x00
./i2c_write 4 0x1a 0x307E 0x00
Second group (end point):
./i2c_write 4 0x1a 0x3080 0x2C
./i2c_write 4 0x1a 0x3081 0x01
./i2c_write 4 0x1a 0x3082 0x00

5.3.3 Verify the writes

(optional but strongly recommended):

- **I2c tools write register:** ./i2c_write 4 0x1a <reg addr> <reg val>
- **I2c tools read register:** ./i2c_read 4 0x1a <reg addr> <num of regs regs to read>

./i2c_read 4 0x1a 0x306D 1 # should return 0x02 (trigger) or 0x01 (normal)
./i2c_read 4 0x1a 0x3079 1 # should return 0x0A or 0x09

5.3.4 Reference

Note: strobe setting must be done while camera stream is off.
Regs and setting values
0x3026: 0x0F
0X3029:0x21
0x306D: 0X02(trigger mode strobe enable) /0x01(normal mode strobe enable)
Strobe start point 3byte:
0x3070 :0x00
0x3071 :0x00
0x3072 :0x00
Strobe end point 3byte:
0x3074:0x2c

0x3075:0x01
0x3076:0x00

0x3079: 0X0A(trigger mode strobe enable) /0x09(normal mode strobe enable)
Strobe start point 3byte:
0x307c :0x00
0x307d :0x00
0x307e :0x00
Strobe end point 3byte:
0x3080 :0x2c
0x3081 :0x01
0x3082 :0x00

I2c tools write register:
./i2c_write 4 0x1a <reg addr> <reg val>

I2c tools read register:
./i2c_read 4 0x1a <reg addr> <num of regs regs to read>

6 On Board EEPROM

InnoMaker provide on board EEPROM, TYPE: FT24C08A(1KBYTE)

6.1 Operation env:

Hardware type: rpi5 ARCH64 OS , I2c bus on CSI : 4

6.2: Detect EEPROM on i2c bus:

```
i2cdetect -y 4
```

```
pi@raspberrypi:~ $ i2cdetect -y 4
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:                         -- -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- -- -- UU -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- 37 -- -- -- -- -- -- -- --
40: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
50: 50 51 52 53 -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- --
```

There should be 4 devices with address 0x50 0x51 0x52 0x53 detected.
Each device have 256 byte content.

6.3: Read EEPROM content

Take device address 0x51 sub address 0x00 as a example:

```
i2cget -y 4 0x51 0x00
```

```
pi@raspberrypi:~ $ i2cget -y 4 0x51 0x00
0xff
```

6.4: Write EEPROM and read the content

```
i2cset -y 4 0x51 0x00  0xaa
i2cget -y 4 0x51 0x00
```

```
pi@raspberrypi:~ $ i2cset -y 4 0x51 0x00  0xaa
pi@raspberrypi:~ $ i2cget -y 4 0x51 0x00
0xaa
```

7 Official Software Manual

- https://www.raspberrypi.com/documentation/computers/camera_software
- https://github.com/raspberrypi/documentation/tree/develop/documentation/asciidoc/accessories/camera

8 Preset System IMAGE

This is preset system IMG for raspberry pi 5.

https://www.jianguoyun.com/p/DY_2JXYQpdSrBxj-nf4FIAA

(Password : o1drfz)