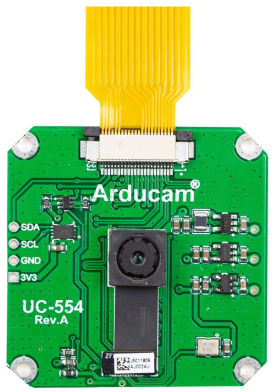
Arducam®

MIPI Camera Module for Raspberry Pi: 13MP Sony IMX135

SKU: B0163



INTRODUCTION

The official Raspberry Pi camera modules lack diverse camera options and camera driver support. Currently, only 5MP OV5647 and 8MP IMX219 are supported.

Arducam has released a series of MIPI camera modules for Raspberry PI ranging from global shutter cameras to high-definition cameras (up to 18MP) to fill this gap. Now you can connect a variety of MIPI camera modules directly to Raspberry Pi's native CSI camera port using our userland SDK. This guide will walk you through the steps of using Arducam MIPI camera modules on a Raspberry Pi.

If you need our help or want customize other models of MIPI cameras, feel free to contact us at support@arducam.com

Github: https://github.com/ArduCAM/MIPI_Camera

Website: https://www.arducam.com/docs/cameras-for-raspberry-pi/mipi-camera-modules/

PREREQUISITES

Enable i2c_vc

```
$ chmod +x ./enable_i2c_vc.sh
$ ./enable_i2c_vc.sh
```

Alter running the script, reboot will be required.

Install the SDK library

```
$ make install
```

Compile the Examples

```
$ make clean && make
```

Optional Settings

Edit /boot/config.txt file. Find gpu_mem=xxx line. Modify gpu_mem size with the proper size, recommend using gpu_mem=160 for 13MP camera board, gpu_mem=180 for 16MP or higher camera board.

RUNNING THE EXAMPLES

Preview Example

```
$ ./preview
```

In the preview.c example, it will demo how to do a preview in different resolution and camera control parameters.

Capture Example

```
$ ./capture
```

In the capture.c example, it will capture different resolution JPEG images.

```
$ ./capture_raw
```

In the capture_raw.c example, it will capture different resolution none interpolation raw format images, especially useful for monochrome sensors.

```
$ ./raw_callback
```

In the raw_callback.c example, it is callback version of capture_raw example.

Video Recording Example

```
$ ./video
```

In the video.c example, it will record the video in H246 format.

Camera Control Query Example

```
$ ./list_format
```

In the list_format.c example, it will list camera supported resolution and control functions.

Sensor Register Read/Write Example

```
$ ./read_write_sensor_reg
```

In the read_write_sensor_reg.c example, it illustrates how to directly read/write sensor registers. This example might need to be modified according to the correct sensor register address.

OpenCV Example

```
$ ./capture2opencv
```

In the capture2opencv.cpp example, it converts YUV data to OpenCV Mat format, and displays as is.

Gstreamer Example

In the video2stdout.c example, it outputs H.264 video stream to stdout, and uses gstreamer to push the stream to PC.

- Example 1:

Raspberry pi side command:

```
$ ./video2stdout | nc -l -p 5000
```

PC side command: (x.x.x.x is your Raspberry Pi IP address)

```
$   gst-launch-1.0   -v tcpclientsrc   host=x.x.x.x port=5000 ! decodebin ! autovideosink
```

- Example 2:

Raspberry pi side command: (x.x.x.x is your Raspberry Pi IP address)

```
$ ./video2stdout | gst-launch-1.0 -v fdsrc ! h264parse ! rtph264pay config-interval=1 pt=96 ! gdppay ! tcpserversink host=x.x.x.x port=5000
```

PC side command: (x.x.x.x is your Raspberry Pi IP address)

```
$   gst-launch-1.0 -v tcpclientsrc host=x.x.x.x port=5000 ! gdpdepay ! rtph264depay ! avdec_h264 ! autovideosink sync=false
```

QR Code Detection Example

```
$ ./qrcode_detection <exposure_value>
```

In the qrcode_detection.cpp example, it illustrates how to use global shutter camera like OV7251 or OV9281 to detect QR code using OpenCV. To run this demo you have to install the dependence

```
sudo apt-get update && sudo apt-get install libzbar-dev libopencv-dev
```

Dual Camera Demo

```
$ ./preview-camera0
```

or

```
$ ./capture-dualcam
```

In the preview-dualcam.c examle, it illustrates how to open the two camera ports on Raspberry pi compute module at the same time for preview. And the capture-dualcam.c examle, it illustrates how to do capture from each camera port on Raspberry pi compute module by switching between them.

A camera_interface struct should be constructed according to your hardware wiring.

For example camera port 0 is using sda_pin 28, scl_pin 29, led_pin 30, shutdown_pin 31, and camera port 1 is using sda_pin 0, scl_pin 1, led_pin 2, shutdown_pin 3.

More information about the compute module wiring please check:
https://www.raspberrypi.org/documentation/hardware/computemodule/cmio-camera.md

UTILITY

How to playback the H264 file

Compile hello_video.bin

```
$ cd /opt/vc/src/hello_pi && ./rebuild.sh
```

Play H264 file

```
$ /opt/vc/src/hello_pi/hello_video/hello_video.bin test.h264
```