**0, Basis**

coordinate frame

**A picture containing text, electronics

Description automatically generated**

**1,Set Up Camera on Windows**

for Windows machine you need to install the camera driver before normal operation

**(1) clone Arducam github**

use your terminal (such as the one in Vscode)

git clone <https://github.com/ArduCAM/ArduCAM_USB_Camera_Shield.git>

the driver is located at Drivers/Win10/x64

**(2) install driver**

plug in your camera

go to device manager

find this unknown device under "Other devices"

click "properties"

click"Driver-Update Driver-Browse my computer for driver software"

and navigate Navigate to the driver path "x64" where you clone it

**(3) Use USB camera demo**

the demo is inside the git repository

ArduCAM\_USB\_Camera\_Shield\Windows\GUI\USBTest\Release\USBTest.exe

guide:  <https://www.arducam.com/downloads/shields/USB_Shield/USB2/ArduCAM_USB2_Camera_User_Guide.pdf>in 'Devic

in "Device" press "scan"

at "Camera Set" load "MT9V022" for 'CfgFile"

if there is noting in CfgFile list, go to "File" - "Open" and load the configuration manually

press "open"

click "Play" to play real time video

click 'Shot" to save BMP images to "Shot" folder

**2, Set Up Camera on Raspberry Pi**

**(1) install libusb**

<https://www.raspberrypi.org/forums/viewtopic.php?t=34363#p669717>

apt-get install libusb-dev

**(2) Install opencv**

sudo apt-get install python-opencv : cv2 for python2

python3 -m pip install opencv-python : install opencv for python 3

this might install the package at the wrong path so that your codes cannot find cv2, do the following instead

python3 -m pip install --target=/usr/lib/python3/dist-packages opencv-python

there maybe some some dependencies problem, do the following

pip3 install opencv-python

sudo apt-get install libcblas-dev

sudo apt-get install libhdf5-dev

sudo apt-get install libhdf5-serial-dev

sudo apt-get install libatlas-base-dev

sudo apt-get install libjasper-dev

sudo apt-get install libqtgui4

sudo apt-get install libqt4-test

**(3) clone Arducam github**

git clone <https://github.com/ArduCAM/ArduCAM_USB_Camera_Shield.git>

go to "RaspberryPi - Python"

**(3) Demos**

go to RaspberryPi/Python in the Arducam repository

we look into Streaming demo

go to PIXEL (otherwise cannot display), open LXTerminal

run (../ means one directory above)

sudo python3 ArduCam\_Py\_Demo.py ../../../python\_config/MT9V022\_VGA.json

press "s"/ "c"/ "q" + enter to save images/ stop recording/ quit

Display is not working, but we can still save images, which is fine to me.

image is a 3d array (480, 640, 1)

numpy array of image, (0,0) is always the top left corner

sample image:

<https://www.nasa.gov/feature/jpl/astrophysics-cubesat-demonstrates-big-potential-in-a-small-package/>

**3, Write your own codes for 2 cameras**

**Python sdk guide:**

<https://www.arducam.com/downloads/shields/USB_Shield/ArduCAM_USB_Camera_Python_SDK_Guide_V1.3.pdf>

**Notices:**

**use sudo python3 to run codes**

need to sleep(2) between scan and open

**Software trigger**:

We based on external trigger demo, but use Py\_AruCam\_softTrigger instead, which trigger the camera to take image by software rather than from external trigger input

**About external trigger:**

<https://www.arducam.com/downloads/modules/industrial/ArduCAM_USB_Camera_MT9V022AN.pdf>

<https://www.arducam.com/docs/camera-breakout-board/0-36mp-mt9v034-mt9v022-global-shutter/hardware-guide/>

you set a pin on camera board to high voltage and it will trigger the camera to capture an image

**Board configuration:**

configBoard(handle,config["board\_parameter\_dev2"])

And then use Py\_ArduCam\_setBoardConfig()

It sets correct register value for different working modes (here, it’s USB 2 or 3).

It uses vendor command codes (see python SDK)

For mt9v022, board parameters are:

Diagram

Description automatically generated with medium confidenceDiagram

Description automatically generated with medium confidence

which reset the camera and do something else

**Write register:**

writeSensorRegs(handle,config["register\_parameter"])

And then Py\_ArduCam\_writeSensorReg(handle, reg, Val)

It changes values in the register to do certain functions

In MT9V022, it’s setting height (3) and width(4)

A picture containing text

Description automatically generatedA picture containing text

Description automatically generated

**Set integration time and Tune gain:**

Get detailed reg information here:

<https://www.onsemi.com/pub/Collateral/MT9V022-D.PDF>

This is mt9v032, probably the same (different image size), but even with default register value:

<http://edge.rit.edu/edge/P10662/public/old/Specs/MT9V032_data_sheet.pdf>

**Reg values are reset to default after power on-off**

Register measured in bits, directly interacts with CPU:

<https://techdifferences.com/difference-between-register-and-memory.html>

Bit N in register:

What does it mean by Set 0xAF[1] to 1?

<http://rfmw.em.keysight.com/wireless/helpfiles/n519xa-vector/Content/Programming%20Guide/Status%20Register%20Bit%20Values.htm>

You can use **Py\_ArduCam\_readSensorReg( handle, reg, val)** to figure out current register setting

reg and val are all int

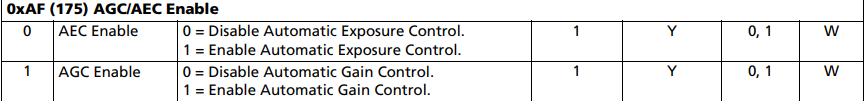
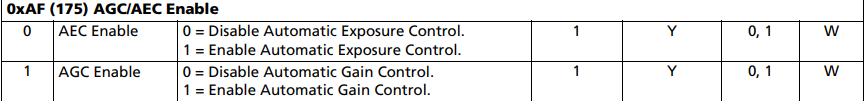
**-turn off AEC and AGC**

**turn 0x00AF = 0**

bit 0 =0, bit 1 =0  -> 00 -> 0x0000

default: binary 11 -> 3 in decimal





turn off  AEC and AGC before tune integration time and gain

**-Integration time( check the link for why)**

from MTV032, but V022 datasheet is the same

Text

Description automatically generatedText

Description automatically generated



0x0004 is Window Width Contex

0x0005 is horizontal blanking

**change 0x0B  to change time (0x0B default val is 0x1E0, 480 in decimal, 64FPS)**

**10ms : val\_0x0B = 362= 0x16A (64 FPS)**

**30 ms val\_0x0B = 1097 = 0x449 (30FPS)**

**1s : va\_0x0B = 36239= 0x8D8F (1 FPS)**



Text

Description automatically generatedText

Description automatically generated

for example:

int\_time = 1 *# ms*

master\_clock\_period = (1 / (26.6E6))\*1E3 *# ms*

R0x04 =  640 *# default decimal*

R0x05 = 94 *# default decimal*

overhead = (R0x04 + R0x05 - 255) \* master\_clock\_period

row\_time = (R0x04 + R0x05) \* master\_clock\_period

val\_R0x0B = (int\_time - overhead) / row\_time

print(val\_R0x0B) *# decimal*

result is 36 (0x24)

default 0x0B (480) result in 13ms integration time

maximum exposure time is one Frame time. Basically it equals to total vertical resolution times 1 Row Time.

default fps is ~60, 17ms per frame

fps will drop based on your int time ( for streaming mode)

**-tune gain:**

**Diagram

Description automatically generated**Diagram

Description automatically generated

****

Graphical user interface, text

Description automatically generatedGraphical user interface, text

Description automatically generated

Reg: 0X0035

Val:

Try

0x0010

0x0025

0x0040 (64)

see

<https://www.arducam.com/docs/camera-breakout-board/0-36mp-mt9v034-mt9v022-global-shutter/software-guide/>

And

<https://www.arducam.com/downloads/modules/industrial/ArduCAM_USB_Camera_MT9V022AN.pdf>