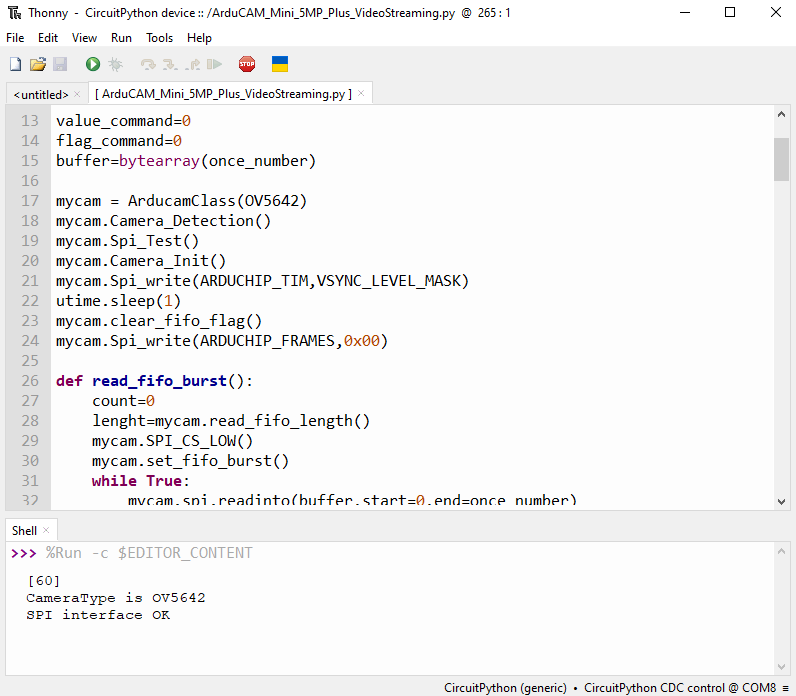
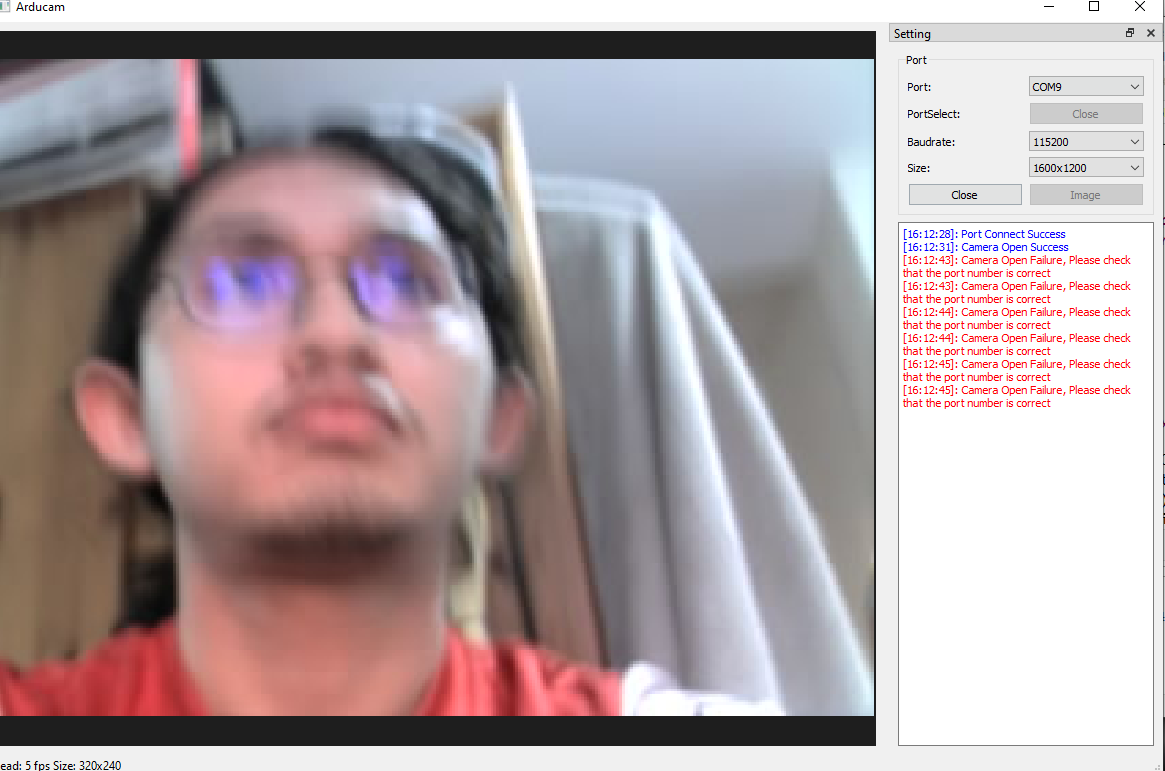
Hardware Logbook

# Testing components:

Set Thonny for circuitpython and raspberry pi pico (<https://learn.adafruit.com/circuitpython-libraries-on-micropython-using-the-raspberry-pi-pico/thonny-setup>)

## SPI Camera x2 (Work!)

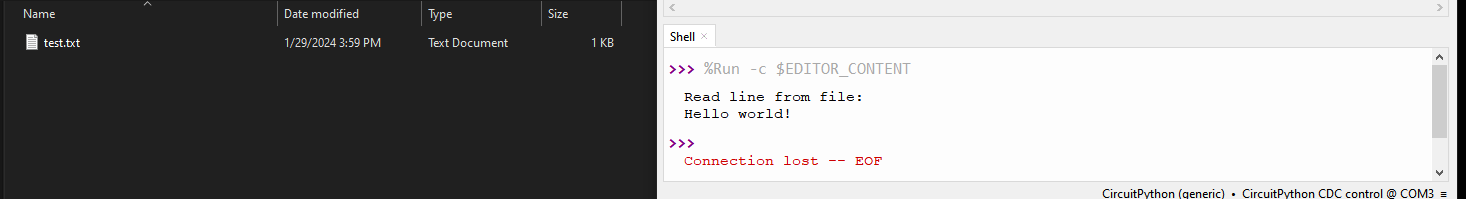
Using examples provided, [SPI Camera for Raspberry Pi Pico - Arducam Wiki](https://docs.arducam.com/Arduino-SPI-camera/Legacy-SPI-camera/Pico/Camera-Module/SPI-Camera/) and [ArduCAM/PICO\_SPI\_CAM (github.com)](https://github.com/ArduCAM/PICO_SPI_CAM), after weeks of struggling and even shorting one of my Pico (don’t use multimeter carelessly while Pico is powered!), finally got it working. The problem was the provided jumper wires are probably old, and rusted inside thus not having stable connection (or at all) getting SPI error when using the HOSTAPP.exe or even worse not even able to initialize the app properly when running the program through Thonny (when my SDA and SCK wire are not working, tested using multimeter connection test). I opened it and exposed some inside wire to it to work better but still not stable, so will just opt to soldering direct connection when assembling prototype, or try using university jumper wire to see if it’s better.

Very unstable connection, also to note is that the focal length (focus) is fixed depending on how tight I turn the lens in, and the default is the best (I think). Also, the demo app kinda shite, as I am unable to go higher res (even when the option is there) so I have no idea full capability

I had tested both camera module and can confirm they are working! Now time to test other components.

## SDIO breakout board and sd card

Following this tutorial: <https://learn.adafruit.com/adafruit-microsd-spi-sdio/circuitpython>.

It works!  Used pins from <https://www.instructables.com/Raspberry-Pi-Pico-Micro-SD-Card-Interface/>.

## Mic testing

This one seems pretty difficult to implement as my mic is Analog electret mic (thus needs ADC and some memory management stuff), not the usual and simpler I2S which is easier to program and have more resources especially in circuit python. Might revisit and implement later instead after stereo camera works tbh.

Put in last priority for now as deemed more complicated than worth it compared to other features (stereo and video), problems encountered is that only hearing static noises, maybe bad sample rate etc or when converting binary to waveform, my hypothesis its harder to sample properly because lack of clock, so might need external clock trigger etc idk.

## MONO camera (SD CARD + CAMERA + BUTTON)

Following [ferrapons](https://gist.github.com/ferranpons/faf789e7f69b37997567903037bbcad2) code, it works with new wires (better connections) for still image, but still taking too long for high res, so might need solder wire directly or prototype on perfboard to further test full capabilities of pico. NEED TO FIGURE OUT HOW TO TAKE AND SAVE VIDEO NEXT!

Wiring is as in code, and pull up resistor is used for button, refer image:

ALSO NEED TO CREATE CIRCUIT DIAGRAM (especially important for perfboard to reduce pin error etc)

February 6, 2024 – Finished soldering one side for mono testing

February 7, 2024 – Tested it, got ltos of photos, buffer size need to be bigger for optimization, python is too slow

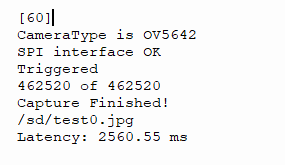
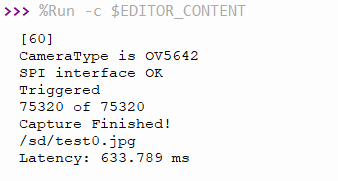
February 8, 2024 – Demonstrated to Tom (failed), after debugging in lab, it seems the culprit is due to incomplete initialization/library import?? Buffer not initialized??? Because after running the example app (the one with HostAPP.exe) on another PC then reflashing the main.py code, suddenly It works fine, now get some data regarding latency to take pictures and why change to C (python too slow, but need more concrete evidence)

* Todo from meeting:

1. Update Gantt chart (done)
2. Email second supervisor
3. Change to C library (and note the performance of Python library to justify)
4. Try implement stereo (optional)

Performance table of Python library with different config of resolution and buffer (using modified ferrapons code), ferrapons-with-latency.py:

Example, 1024x768 at 16384 buffer and 2592x1944 at 16384 buffer:



Orange are latency time in ms:

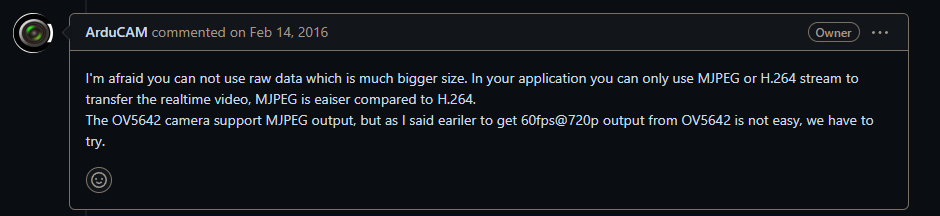
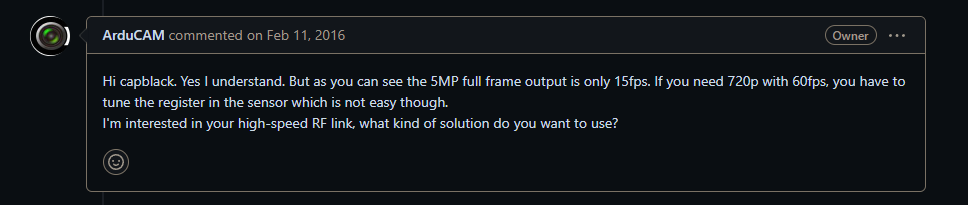
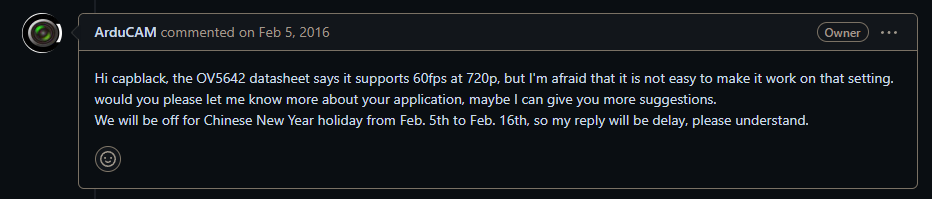
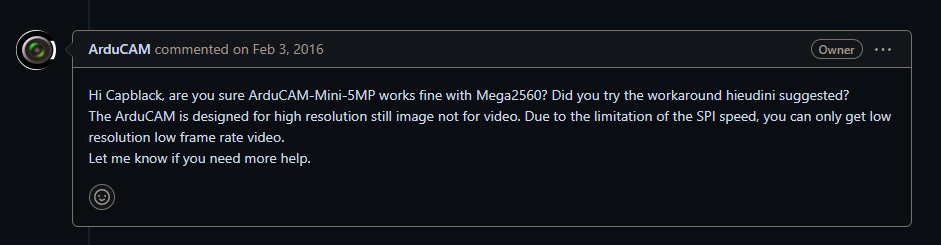
|  |  |  |  |
| --- | --- | --- | --- |
| Resolution/Buffer size | 1024 | 8192 | 16384 |
| 320x240 | 386.719 | 245.117 | 235.352 |
| 1280x960 | 1382.81 | 813.477 | 771.484 |
| 2592x1944 | 5664.06 | 2708.01 | 2560.55 |

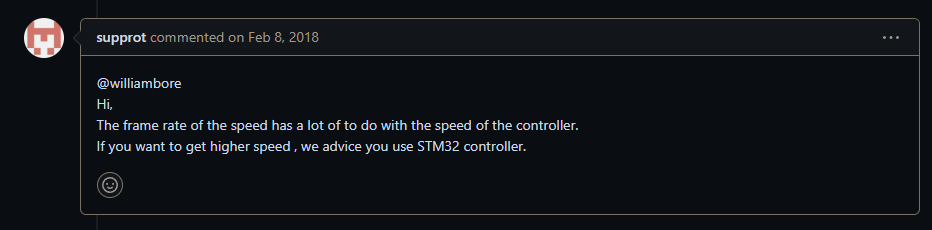
As seen above, all of the latency even on best case (high buffer, low res) have 235ms latency which is not suitable for smooth video (atleast 30fps), as the video is just bunch of images frames taken continuously thus would result in around 4fps video… The main culprit for slow performance seems to stem from python slow interpreter execution as it is running on the pico itself. Thus the need to change to C is needed to optimize performance. Another consideration is to use raspberry pi pico dual core to multithread the process to take picture and save to sd card in parallel.

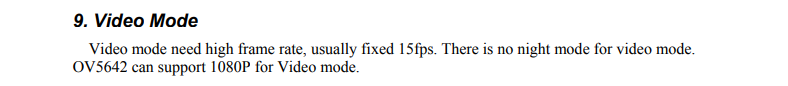
Tried flashing C compiled .uf2 files, no difference lol. Refer my new issue [Impossible to get video capture working at viable requirement, very low framerate (fps) and resolution · Issue #14 · ArduCAM/PICO\_SPI\_CAM (github.com)](https://github.com/ArduCAM/PICO_SPI_CAM/issues/14)

**VIDEO CAMERA NOT VIABLE WITH CURRENT HARDWARE WITHOUT ADVANCED TINKERING (STILL NOT GONNA REACH 60 FPS HD REGARDLESS) SO REDUCE SCOPE TO PICTURE ONLY UNFORTUNATELY**

**According to this github issue reply from official Arducam:** <https://github.com/ArduCAM/Arduino/issues/1#issuecomment-178943246>

****

**There was no further engagement regarding that topic unfortunately. And further research regarding fps and long capture time suggest microcontroller choice is most important:** [https://github.com/ArduCAM/Arduino/issues/298](https://github.com/ArduCAM/Arduino/issues/298)

[How to increase FPS on ESP32-Cam 🚀 : esp32 (reddit.com)](https://www.reddit.com/r/esp32/comments/vndbh7/how_to_increase_fps_on_esp32cam/)  
Even assuming it works, 15 fps is not enough for good video especially stereo video for VR media consumption, and lowering resolution also not a good idea, thus both drawback induces motion sickness: [OV5642\_camera\_module\_software\_application\_notes\_1.1.pdf (uctronics.com)](https://www.uctronics.com/download/OV5642_camera_module_software_application_notes_1.1.pdf)

15fps 1080p must be a lie if most users cant even get 15fps on lowest resolution lol…

**There a more examples of such issues which imo shows its too complicated for the scope and timeframe of my project. The alternative is to drop the video functionality. For future revisions, it seems it is better to use higher speed hardware that is built specifically for video with proven examples etc instead of trusting manufacturers advertised specifications… For example, using a raspberry pi 4 or higher with 2 usb webcam camera such as** [Arducam 1080P Low Light Wide Angle USB Camera Module with Microphone for Computer, 2MP 1/2.8" CMOS IMX291 100 Degree Mini UVC USB2.0 Webcam (uctronics.com)](https://www.uctronics.com/arducam-1080p-low-light-wide-angle-usb-camera-module-with-microphone-for-computer-2mp-1-2-8-cmos-imx291-100-degree-mini-uvc-usb2-0-webcam-board-with-3-3ft-1m-cable-for-windows-linux-and-mac-os.html).  
  
In hindsight, should’ve followed a proven tutorial or project such as these which I found during research phase for this project: [Low cost stereo camera | Hackaday.io](https://hackaday.io/project/19533-low-cost-stereo-camera/), [Making A Low-Cost Stereo Camera Using OpenCV | LearnOpenCV #](https://learnopencv.com/making-a-low-cost-stereo-camera-using-opencv/), or even kits such as [StereoPi | StereoPi - DIY stereoscopic camera based on Raspberry Pi](https://stereopi.com/) (although this is impossible as most are out of project budget). Atleast this is good learning experience and good to know and document so future peeps don’t make same mistakes and assumptions as I did.