

ECE 387 – Project Report

Spring 2021  Section C

Instructed by: Dr. Peter Jamieson

By: Zhihao Lin (’22)

May 10th, 2021

**Introduction:**

In this project, the originals I mainly used are Arduino UNO and camera OV7670. I hope to be able to use Arduino to drive the camera to achieve the effect of taking pictures. After consulting the information, I learned that the ov7670 is a light-sensitive array produced by Omni Vision, with 30W pixels (Like the following Figure 1-1). What we are using this time is the OV7670 module with FIFO. (FIFO chip -> back view, the largest 28-pin chip)



Figure 1-1

**Mathematical Model**

According to the diagram in Figure2-1, we can understand the principle of OV7670. And we can follow this idea to connect the devices.

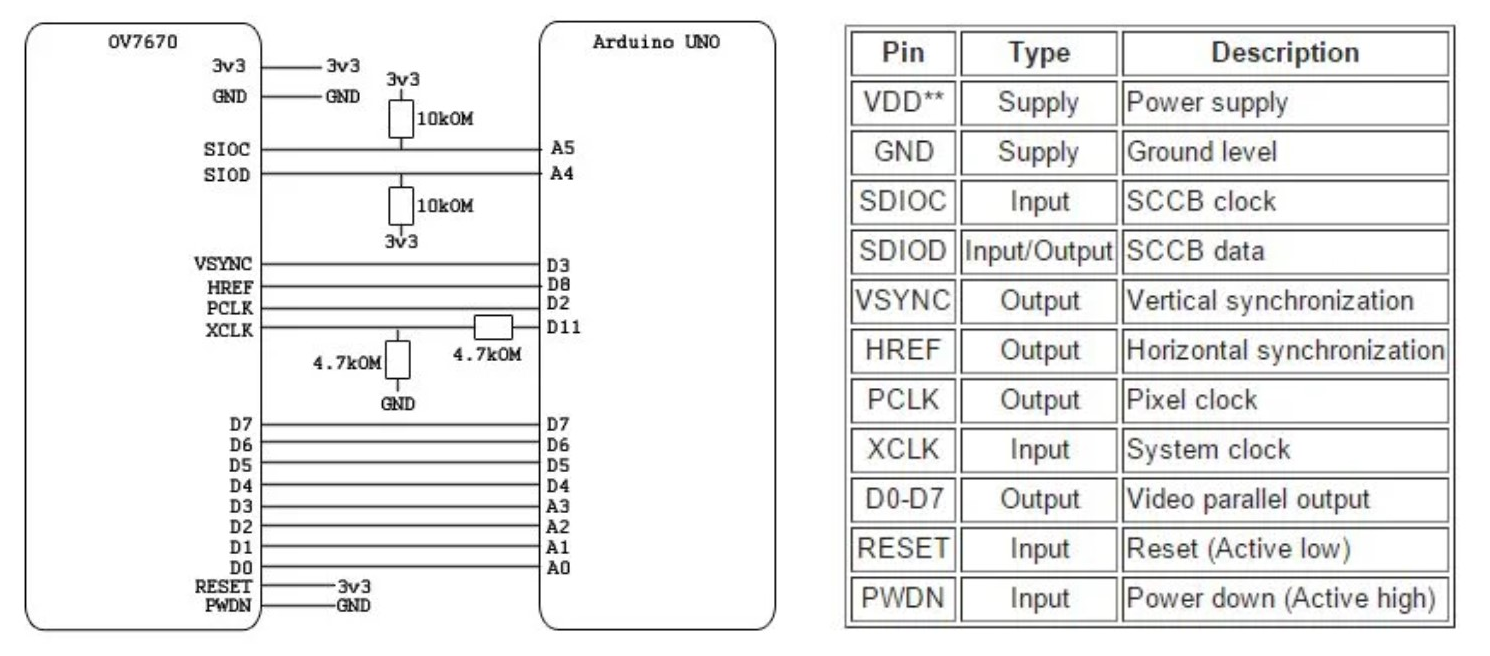


Figure 2-1 Pin Connection

We also can see the model in Figure 3-1, The camera module is powered by a + 3.3V single power supply.

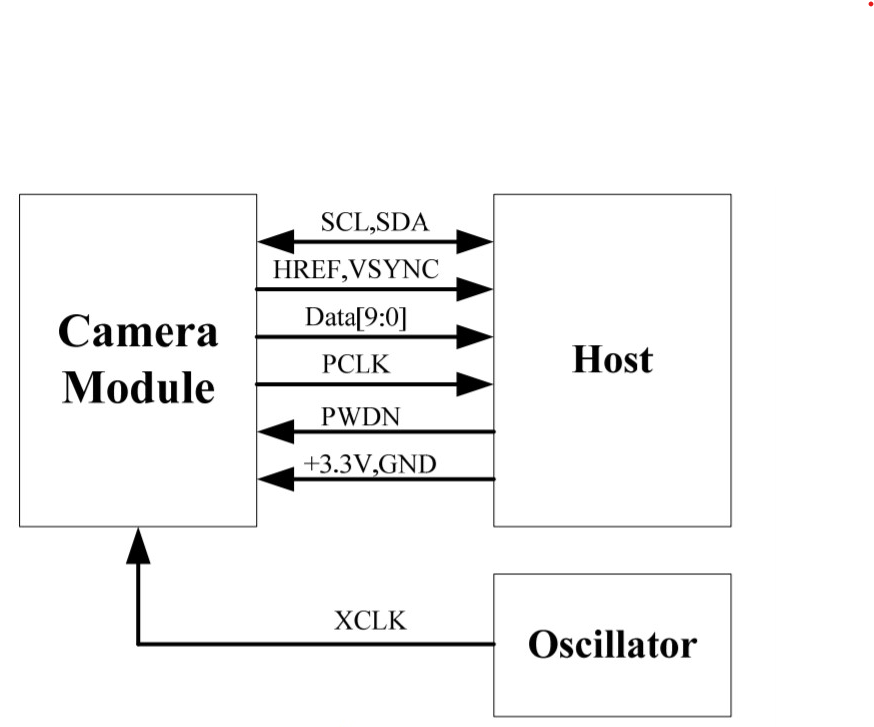


Figure 3-1

We are going to use the module with FIFO here, because the speed of Arduino UNO is relatively slow. With FIFO, the data taken by the camera can be temporarily stored in the FIFO, and then our Arduino UNO will slowly read out the taken data and send it through the serial port. Display on the host computer of the serial port.

According to the Figure 4-1, The image data is output when the HREF is high. When the HREF is high, one byte of data is output for each PCLK clock. For example, we use VGA timing, RGB565 format output, every two bytes constitute a pixel color (high byte first, low byte second), so each line output has a total of 640\*2 PCLK cycles, output 640\*2 bytes.

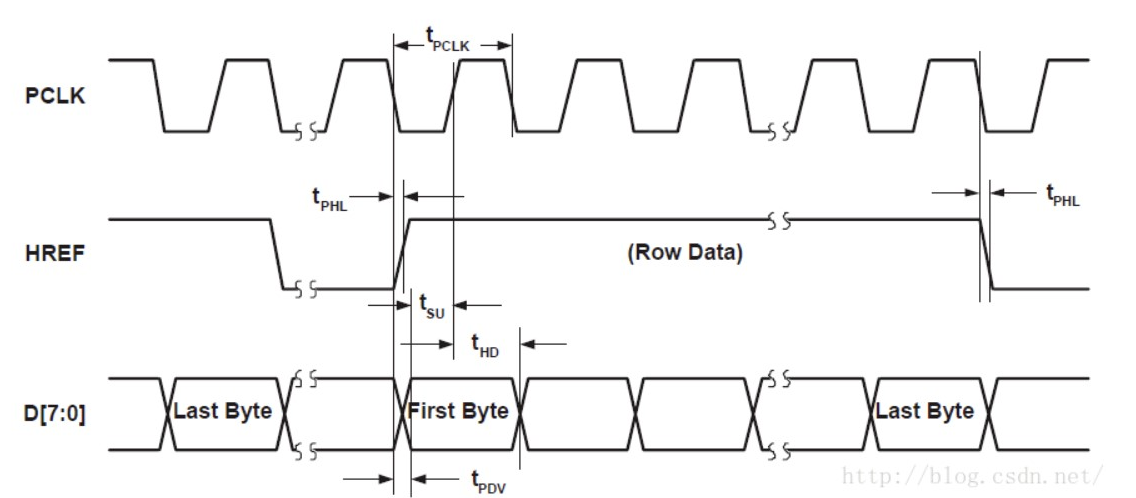


Figure 4-1 OV7670 line output timing

FIFO, the abbreviation of first in first out. Here, the speed of the FIFO is very fast, and the data of the camera can be temporarily stored. So that our slow CPU will slowly take out and process the acquired data to achieve a reservoir-like effect.

We can see that the resources of Arduino UNO are limited. Except for the serial port (pin 0, 1) and pin 13 of the camera, the other pins are used.

According to the Figure 5-1, VSYNC generates one frame synchronization signal when the bit is high, so when two frame synchronization signals are generated, the data output of one frame is completed. Note: The HSYNC and HREF in the figure are signals generated by a single pin, but under different circumstances, they use different signals.

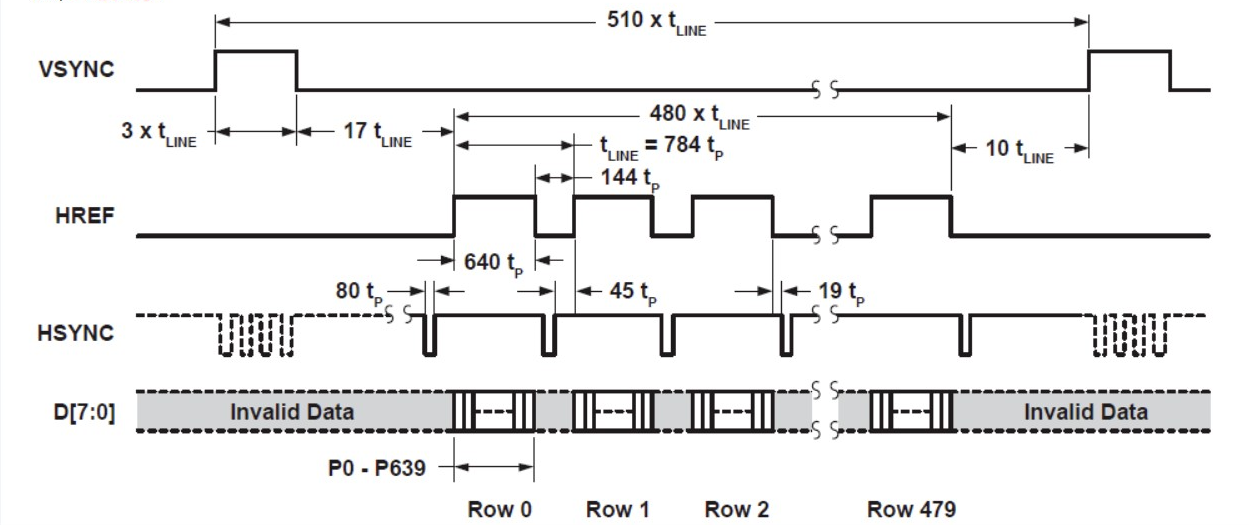


Figure 5 -1 OV7670 frame sequence

**Controller Design**

So， the data can only be uploaded to the computer through the serial port for display.

OV7670 D0 ->Arduino 3

OV7670 D1 ->Arduino 4

OV7670 D2 ->Arduino 5

OV7670 D3 ->Arduino 6

OV7670 D4 ->Arduino 7

OV7670 D5 ->Arduino 8

OV7670 D6 ->Arduino 9

OV7670 D7 ->Arduino 10

OV7670 FIFIO\_RCK -> Arduino A0

OV7670 FIFIO\_OE -> Arduino A1

OV7670 FIFIO\_WR -> Arduino A2

OV7670 FIFIO\_RRST -> Arduino A3

OV7670 SIO\_C -> Arduino A5 (pins cannot be adjusted to other pins)

OV7670 SIO\_D -> Arduino A4 (pins cannot be adjusted to other pins)

OV7670 FIFIO\_WRST -> Arduino 11

OV7670 VSYNC -> Arduino 2 (The pin cannot be adjusted to other pins)

OV7670 RESET -> Arduino 12

OV7670 3.3V -> Arduino 3.3V

OV7670 GND -> Arduino GND

The other camera pins can be left unconnected (floating).

The approximate process of writing a driver is:

1) Initialize all IO ports of Arduino.

2) Hardware reset OV7670 module

3) Detect the I2C communication with the module, and configure the module through the I2C interface. (That is to initialize the module, for example, configure the module to be color data instead of black and white, and the resolution is 320\*240, etc.).

4) Turn on the interrupt and wait for the start of 1 frame of camera data. The first frame of data is discarded and is only used to align the FIFO address. At the beginning of the second frame of data, open the FIFO, take out the data captured by the camera and upload it to the computer through the serial port.

5) End of program

**Operating procedures**

The final purpose of the Arduino program is to transmit the image data to the computer through the serial port for display. This requires corresponding software on our computer.

I searched for a long time and found a software called "Multifunctional Debugging Assistant". First, we need to plug the Arduino UNO board to our computer through the USB cable. We need to adjust the baud rate to 256000, the stop bit to 1, and the data bit to 8. We need to set the width and height to 320 and 240 respectively. RGB565 little endian display. Finally click the start button, the Arduino board will start to upload data automatically, wait about 10 seconds, the picture will be displayed on the right side of the software. If you want to refresh the picture again, click the close serial port button of the software. Then click to open the serial port. Wait about 10 seconds, the new picture will overwrite the old picture.

**Conclusion**

I also encountered many difficulties during the experiment. For example, in the earliest attempts, the photos taken by the camera several times were blurred. So， in the end I used a welding process to ensure the quality of the photos. In my opinion, one of the major reason is that there are too many cables in this project, which is caused by interference.

Reference

1. <https://blog.csdn.net/HouQi02/article/details/51712994>
2. <https://zhuanlan.zhihu.com/p/65250414>
3. <https://www.arducam.com/wp-content/uploads/2019/11/Arducam_ov7670_cmos_camera_module_revc_ds.pdf>
4. <https://www.instructables.com/OV7670-Arduino-Camera-Sensor-Module-Framecapture-T/>