VIET NAM NATIONAL UNIVERSITY - HO CHI MINH CITY
HO CHI MINH CITY UNIVERSITY OF TECHNOLOGY
DEPARTMENT OF ELECTRONICS

INTERNSHIP REPORT

HARDWARE DESIGN FOR STREAMING VIDEO

Supervisor **ME. HIEU NGUYEN TRUNG**

Students

Sang Truong Tan	1813818
Quoc Nguyen Anh	1813733

Ho Chi Minh city, Septemper 6th, 2021

Introduction

First of all, to complete this topic well, We get a lot of help from the teachers in the department. Especially, We would like to thank teacher **Hieu Nguyen Trung**. He helped us during the time of making the report. We sincerely thank the teaches.
About this topic, the field of Drone in particular and embedded field in general, Camera streaming is important to supervisors and processing data. IP camera is too big and consumes more power. Therefore, the embedded camera is more flexible and doesn't need a high-performance microprocessor. At this topic, We are using the OV7670 camera module and STM32F466RE to processing raw data. Raw data after compress to jpeg format which will transfer to PCs through Uart and display on the screen.

Contents

1 System specifications **1**
- 1.1 Product requirements . . . 1
- 1.2 Design specification . . . 3

2 System Design **4**
- 2.1 System behavior . . . 4
- 2.2 Hardware description . . . 6
  - 2.2.1 Board Nucleo STM32 . . . 6
  - 2.2.2 Camera OV7670 . . . 7
  - 2.2.3 Uart to MicroUSB CP2102 . . . 8
- 2.3 Design Hardware . . . 9
  - 2.3.1 OV7670 block and connection . . . 9
  - 2.3.2 Hardware connection . . . 10
  - 2.3.3 MCU pinout . . . 10
  - 2.3.4 System Architecture . . . 11
  - 2.3.5 Schematic . . . 11
- 2.4 The clock configuration . . . 12
  - 2.4.1 Main clock tree . . . 12
  - 2.4.2 The clock source for the camera module . . . 12
- 2.5 Software . . . 13
  - 2.5.1 Interface . . . 13
  - 2.5.2 Memory management . . . 14
  - 2.5.3 Software achitecture . . . 16
  - 2.5.4 Software implementation . . . 17

3 Design Issues **18**
- 3.1 Constraint issues . . . 18
- 3.2 Function issues . . . 18
- 3.3 Realtime issues . . . 18
- 3.4 Concurrent issues . . . 18
- 3.5 Reactive issues . . . 18

4 Conclusion and Plan **19**
- 4.1 Conclusion . . . 19
- 4.2 Plan . . . 19

5 References **20**

List of Figures

1.1 Use case diagram 2
1.2 *System Architecture* 3

2.1 *System block diagram* 4
2.2 *Dataflow Structure* 5
2.3 *System state diagrams* 6
2.4 *Board NUCLEO STM32F4 F466RE* 6
2.5 *Camera OV7670 without FIFO* 7
2.6 *UART to MicroUSB CP2102* 8
2.7 *Functional block diagram* 9
2.8 *Hardware connection* 10
2.9 *MCU pinout* 10
2.10 *System Architecture* 11
2.11 *Schematic* 11
2.12 *Main clock tree of the CPU* 12
2.13 *Clock source for camera module using MCO* 12
2.14 *Serial camera control bus connection* 13
2.15 *Serial camera control bus waveform* 13
2.16 *Display data bus waveform* 14
2.17 *One bytes data transfer* 14
2.18 *Memory management* 15
2.19 *RGB565 output waveform* 15
2.20 *YCbCr colour format arrange in the memory cell* 16
2.21 *Software architecture* 16
2.22 *Capture and transmission timing* 17

List of Tables

1.1 *List of device and costs* 1

Chapter 1

System specifications

1.1 Product requirements

- Name: Streaming video with STM32 MCU.
- Purpose: Streaming video or other system using the embedded system.
- Inputs and outputs:
  - Inputs:
    - OV7670 camera module.
  - Outputs:
    - Display the video on the Pc's screen.
- Use case:
  - Streaming video with 160x120 resolutions(at RGB565 format).
  - Streaming video with 160x120 higher qualification(at YCbCr format).
- Use case diagram
- Functions: Streaming video jpg format can decrease bandwidth of COM port or TCP connection.
- Performance: about 5fps[1].
- Munufacturing costs:

Table 1.1: *List of device and costs*

Devices	Quantity	Costs
STM32 Nucleo Board	1	503.000đ
OV7670 Camera (Non Buffer)	1	62.000đ
Uart to USB CP2102	1	39.000đ
Total		604.000đ

[1]frame per second

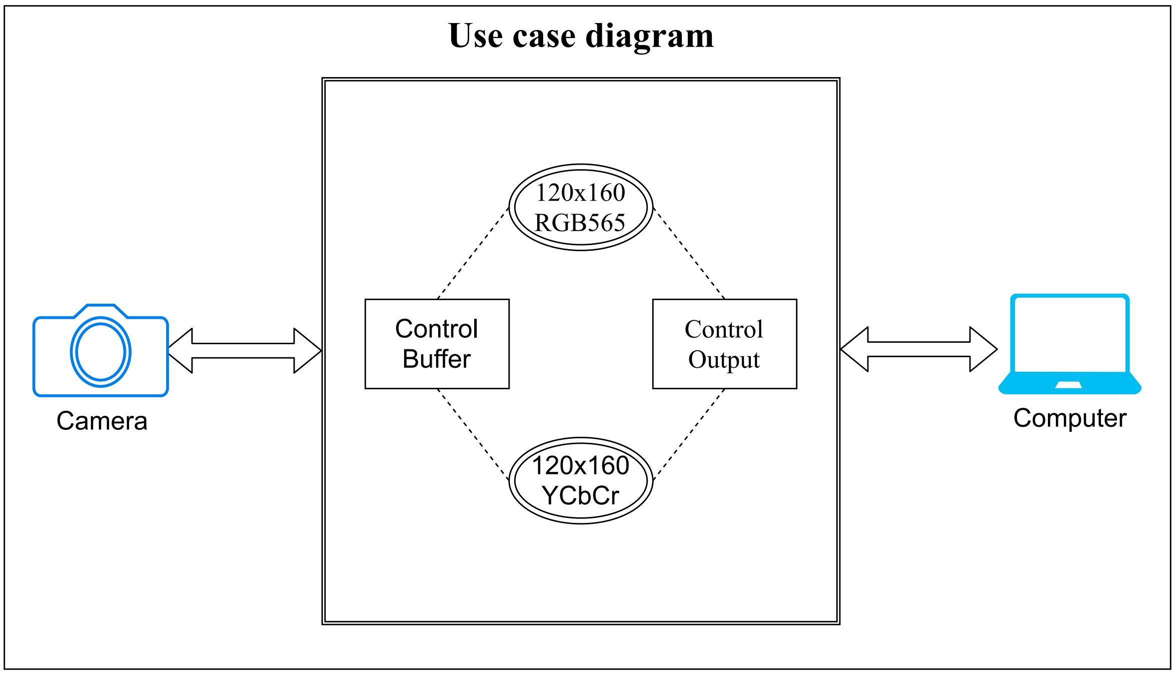


Figure 1.1: Use case diagram

- Power supply: USB Cable 5V
- Physical size and weight:
- Installation and working enviroment:
  -
  -

1.2 Design specification

In several points of view, the system contains three main blocks: Camera block, Stm32 board, and screen display (Pc [2]). The embedded camera, OV7670 has 640x480 resolutions. However, the main core of stm32f446re (Cortex M4) has a maximum 180Mhz clock, and We don't use SRAM. Therefore, the image resolution output is 160x120 with RGB565 formatted color or YCbCr formatted. The image rate is about 5fps. The resolution and Color format can control by Pc GUI[3]. The system is powered by a USB 5v connecter.

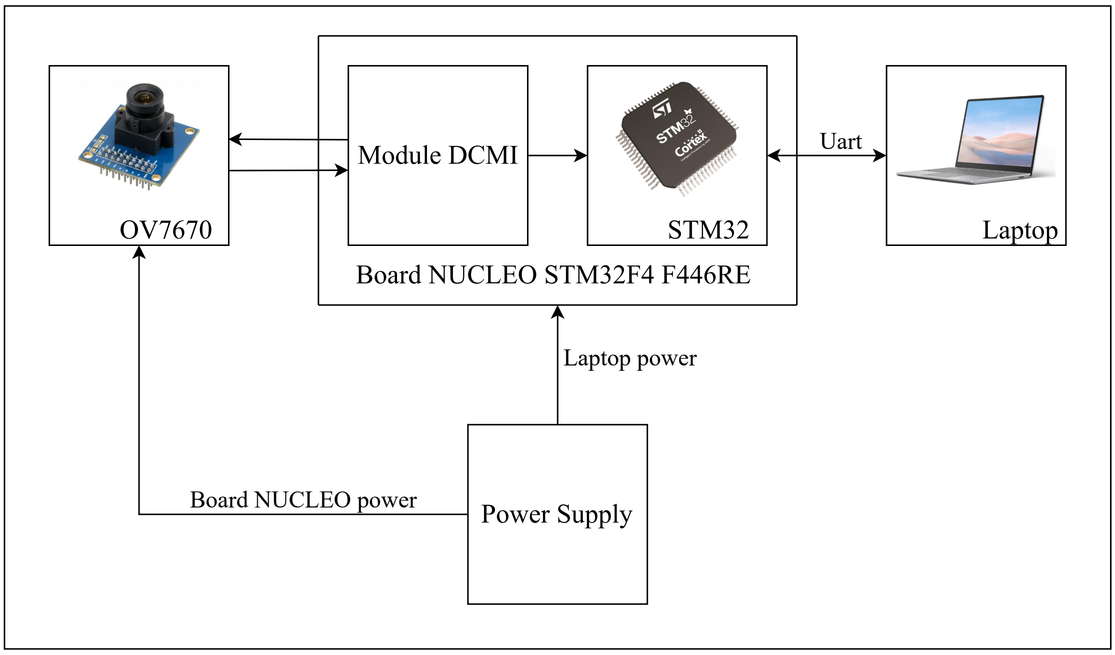


Figure 1.2: *System Architecture*

[2]Personal Computer
[3]Graphical User Interface

Chapter 2

System Design

2.1 System behavior

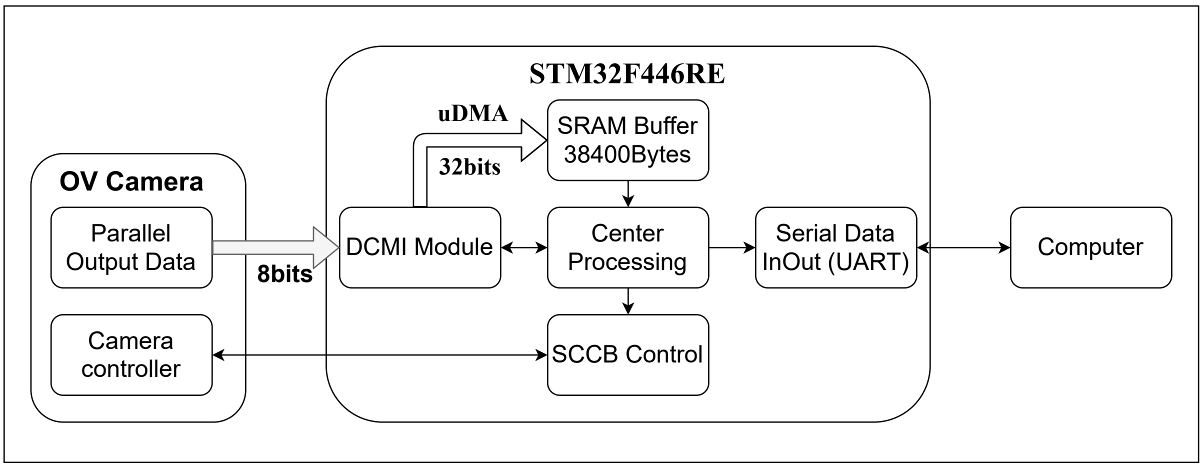


Figure 2.1: *System block diagram*

- Control flow: The OV7670 Camera module uses an SCCB interface to Control and uses an 8-bits parallel bus to transfer image data. The SCCB interface is similar I2C[1] interface, so we can use the I2C module which has built-in MCU[2] to control the camera. To choose the color format (RGB565 or YCbCr) using Computer through a UART connection.
- Dataflow: The Data from the Camera module transfer to DCMI [3] module through 8-bits parallel bus. The FIFO of DCMI can store 4 bytes of data, then the data move to SRAM buffer using μDMA[4]. The data in the buffer will be compressed to jpeg format and send to the Computer using a UART connection.

[1]Inter-Intergrated Circuit
[2]Micro-Controller Unit
[3]Digital Camera Interface
[4]Direct Memory Access

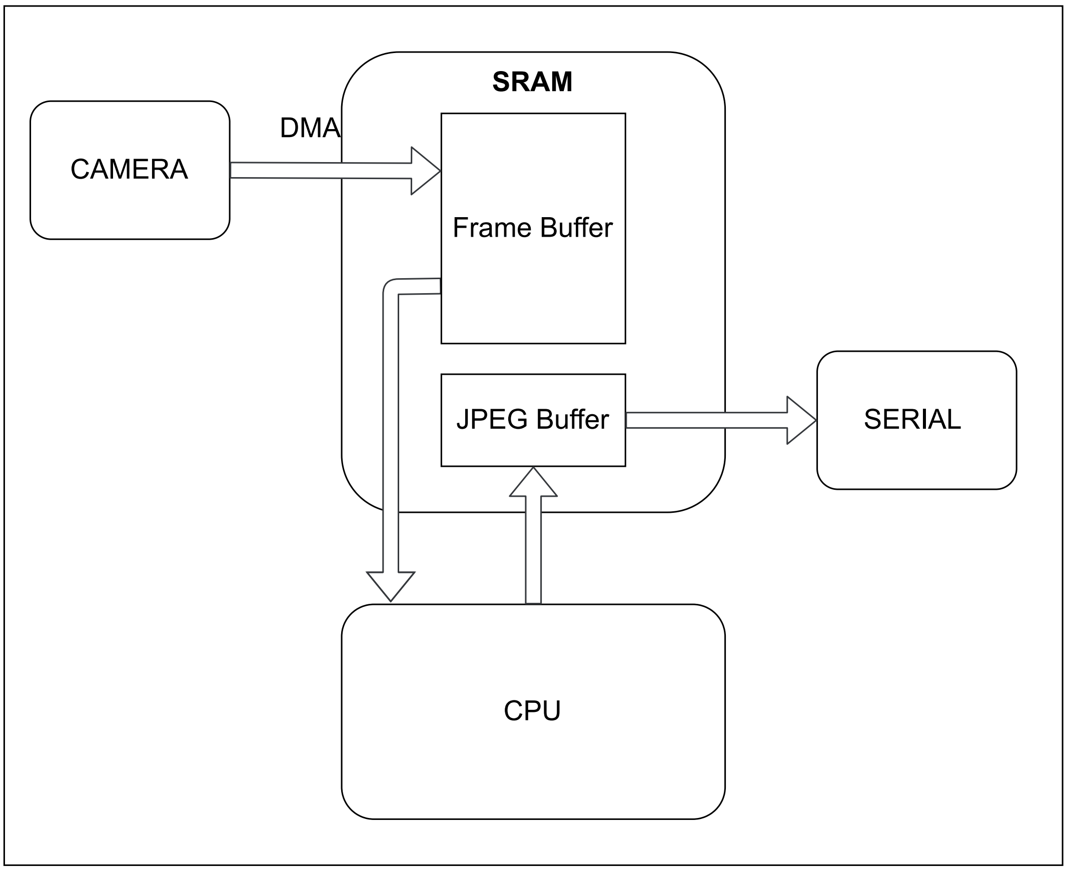


Figure 2.2: *Dataflow Structure*

- Timing requirements: System timing is constrained by the frame rate. The OV7670 Camera support 30fps, but Cortex M4 is too slow to process this big data flow. Because the time to receive the data from the camera and the time to transmit UART is too fast compared to the time compress jpeg. Therefore, The time to process jpeg is the time per frame.
  - Time per frame $= \frac{1}{30}s$
  - Jpeg process time per frame:
    - 160x120: $t \simeq 160ms \Rightarrow fps \simeq 6fps$
- State diagrams:
  - Idle state: This state will start when reset system or after set DMA for the output buffer.
  - Start capture state: When the VSYNC[5] signal is detected, The CPU [6] will allow the DCMI module to capture the frame.
  - Process Line by Line state: To reduce the time process, The image must be process line by line. The time for capture one is smaller than the time for process one line. Therefore, we just using the simple synchronous signal.
  - Transmit data state: Because the jpeg compression has a different size, depending on the complexity of the photo. The jpeg data must be sent when the compression is done.

[5]Vertical Synchronous
[6]Central Processing Unit

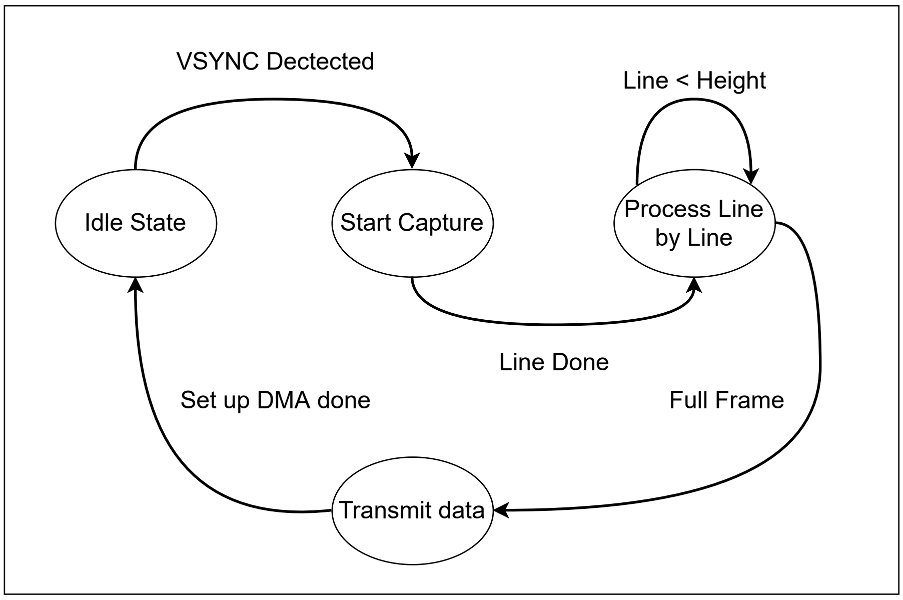


Figure 2.3: *System state diagrams*

2.2 Hardware description

2.2.1 Board Nucleo STM32

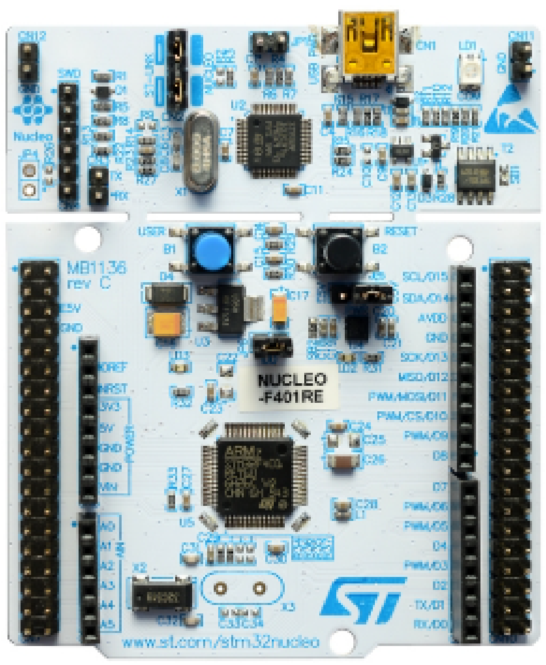


Figure 2.4: *Board NUCLEO STM32F4 F466RE*

- Specifications:
  - STM32 microcontroller with LQFP64 package.
  - Two types of extension resources

    - Arduino Uno Revision 3 connectivity
    - STMicroelectronics Morpho extension pin headers for full access to all STM32 I/Os
  - On-board ST-LINK/V2-1 debugger/programmer with SWD connector
    - selection-mode switch to use the kit as a standalone ST-LINK/V2-1
  - Flexible board power supply
    - USB VBUS or external source (3.3V, 5V, 7-12V)
    - Power management access point
  - Three LEDs
    - USB communication (LD1), user LED (LD2), power LED (LD3)
  - Two push buttons: USER and RESET
  - USB re-enumeration capability: three different interfaces supported on USB
    - Virtual Com port
    - Mass storage
    - Debug port
  - Supported by wide choice of Integrated Development Environments (IDEs) including *IAR*^{TM}, *Keil*®, GCC-based IDEs

2.2.2 Camera OV7670

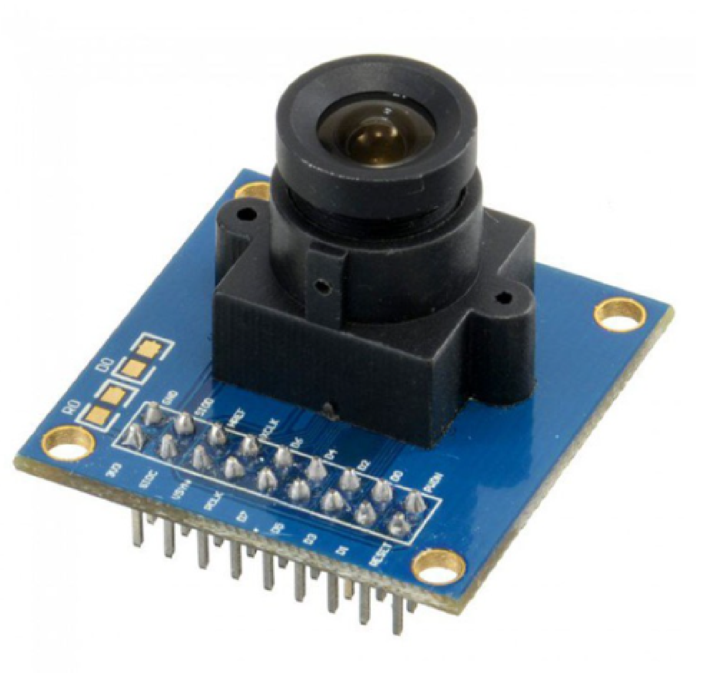


Figure 2.5: *Camera OV7670 without FIFO*

- Specifications:
  - Photosensitive Array: 640x480
  - IO Voltage: 2.5V to 3.0V

- Operating Power: 60mW/15fps
- Sleeping Mode: <20μA
- Operating Temperature: -30 to 70 deg C
- Output Format: YUV/YCbCr4:2:2 RGB565/555/444 GRB4:2:2 Raw RGB Data (8 bits)
- Lens Size: 1/6"
- Vision Angle: 25 degree
- Max Frame Rate: 30fps VGA
- Sensitivity: 1.3V / (lux-sec)
- Signal to Noise Ratio: 46dB
- DynamicRange: 52dB
- Browse Mode: By row
- Electronic Exposure: 1 to 510 row
- Pixel Coverage: 3.6μm x 3.6μm
- Dark Current: 12mV/s at 60 deg C
- PCB Size (L x W): Approx. 1.4x1.4inch / 3.5x3.5cm

2.2.3 Uart to MicroUSB CP2102

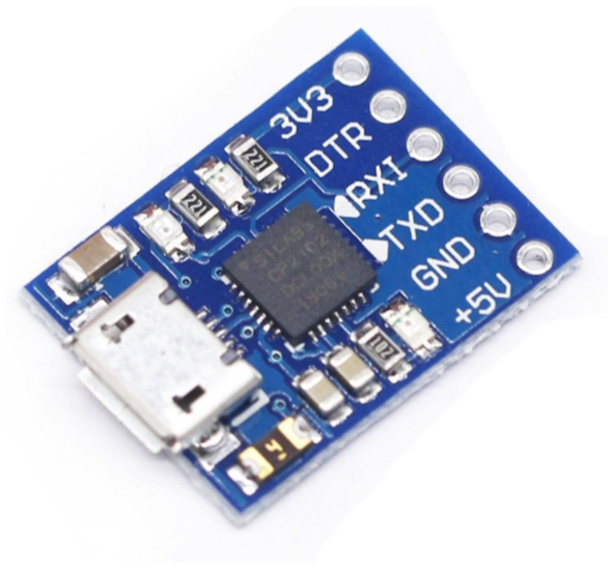


Figure 2.6: *UART to MicroUSB CP2102*

- Features:
  - Embedded USB transceiver, no external circuit device
  - Containing clock circuit, no external circuit device
  - Contains power-on reset circuit
  - The on-chip voltage regulator within the 3.3V output
  - Meet the USB2.0 specification requirements

- SUSPEND pins support USB suspend state
- Asynchronous serial data bus compatible with all handshakes and modulation controller interface signals
- Support data format is 8 data bits, 1 stop bit and the parity bit
- Connotation 512 byte receive buffer and 512 byte transmit buffer
- Supports hardware or X-ON / X-OFF Handshake
- Size: 21x16mm

2.3 Design Hardware

2.3.1 OV7670 block and connection

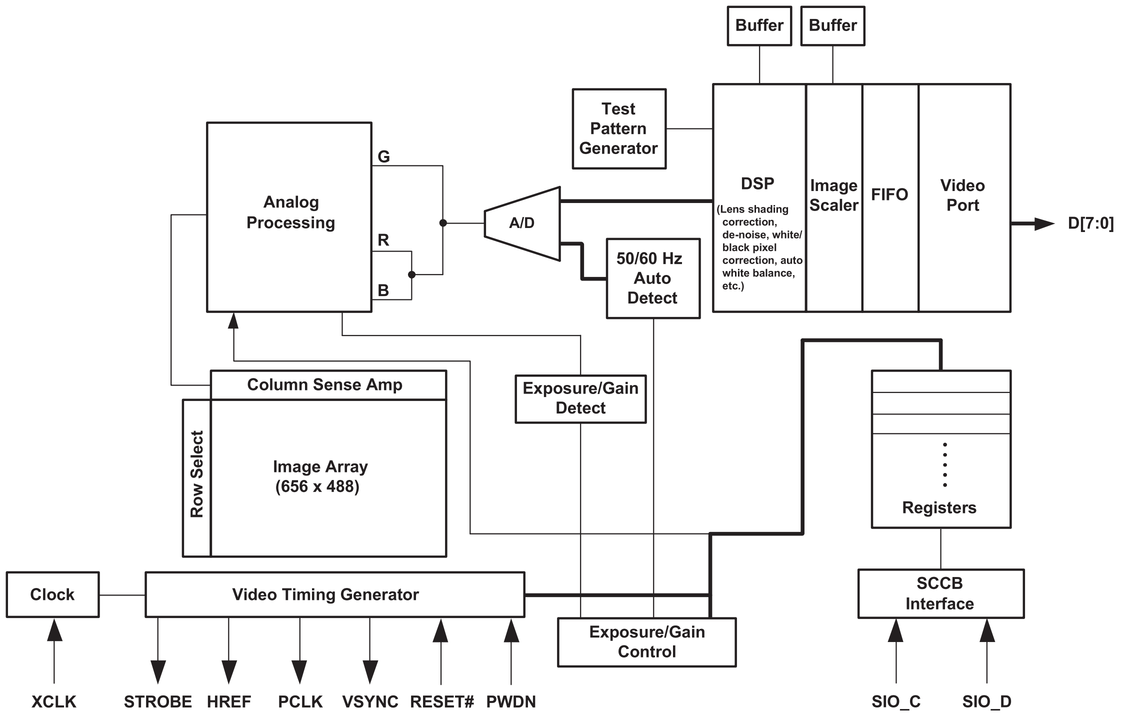


Figure 2.7: *Functional block diagram*

- Control I/F
  - SCCB (SIO_C, SIO_D)
- Clock supply
  - Supply around 10 to 48MHz
- Synchronization
  - OV7670 outputs PCLK, HREF and VSYNC
- Output data length
  - OV7670 outputs 8bit data D[7:0]

2.3.2 Hardware connection

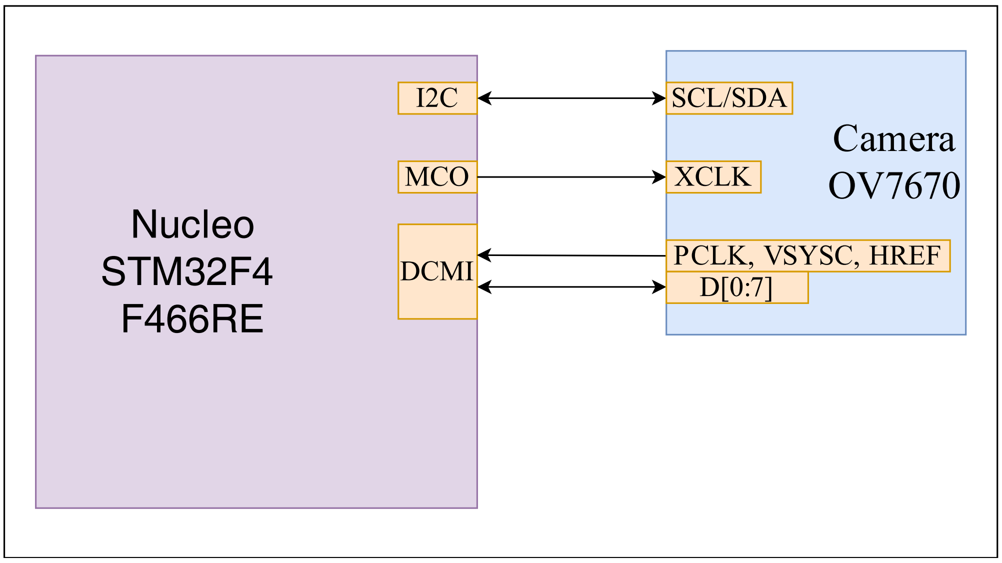


Figure 2.8: *Hardware connection*

2.3.3 MCU pinout

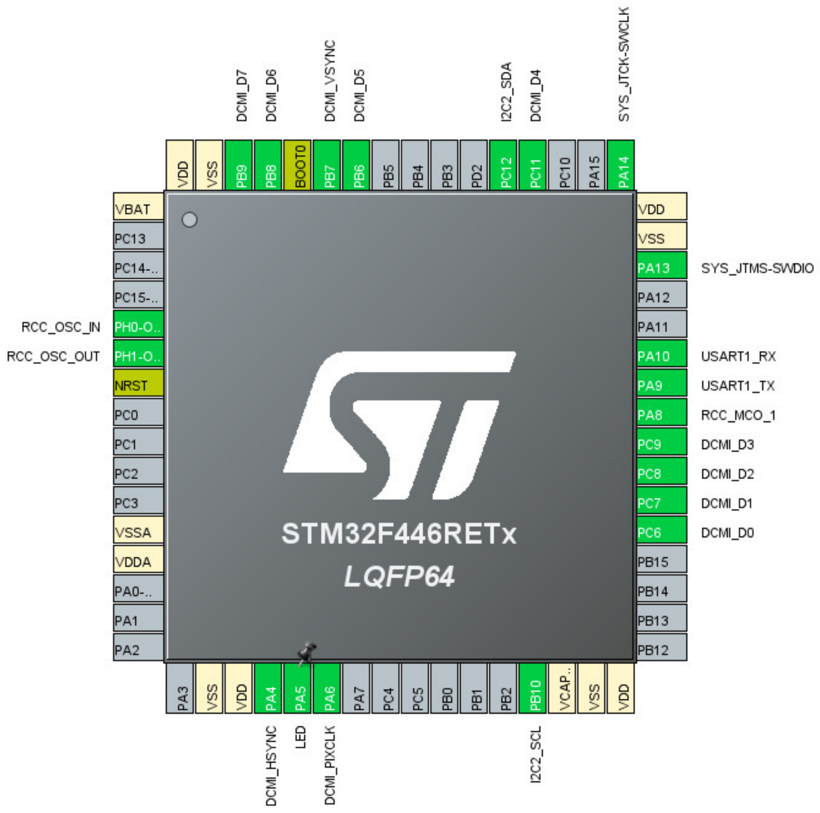


Figure 2.9: *MCU pinout*

2.3.4 System Architecture

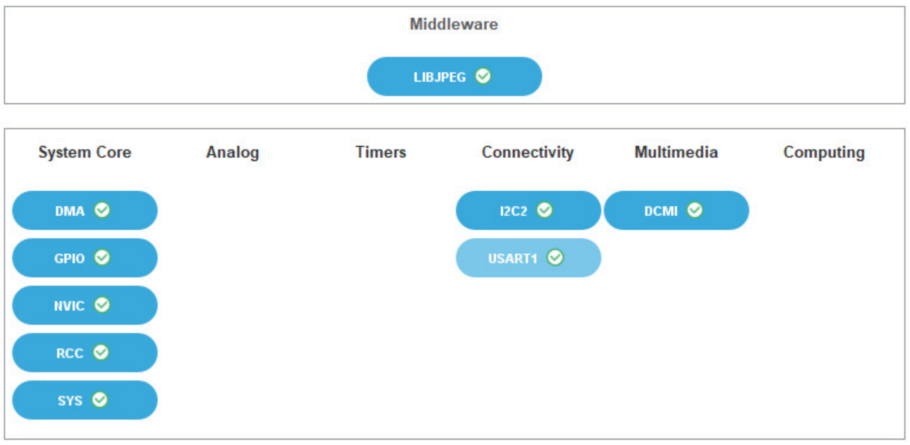


Figure 2.10: *System Architecture*

2.3.5 Schematic

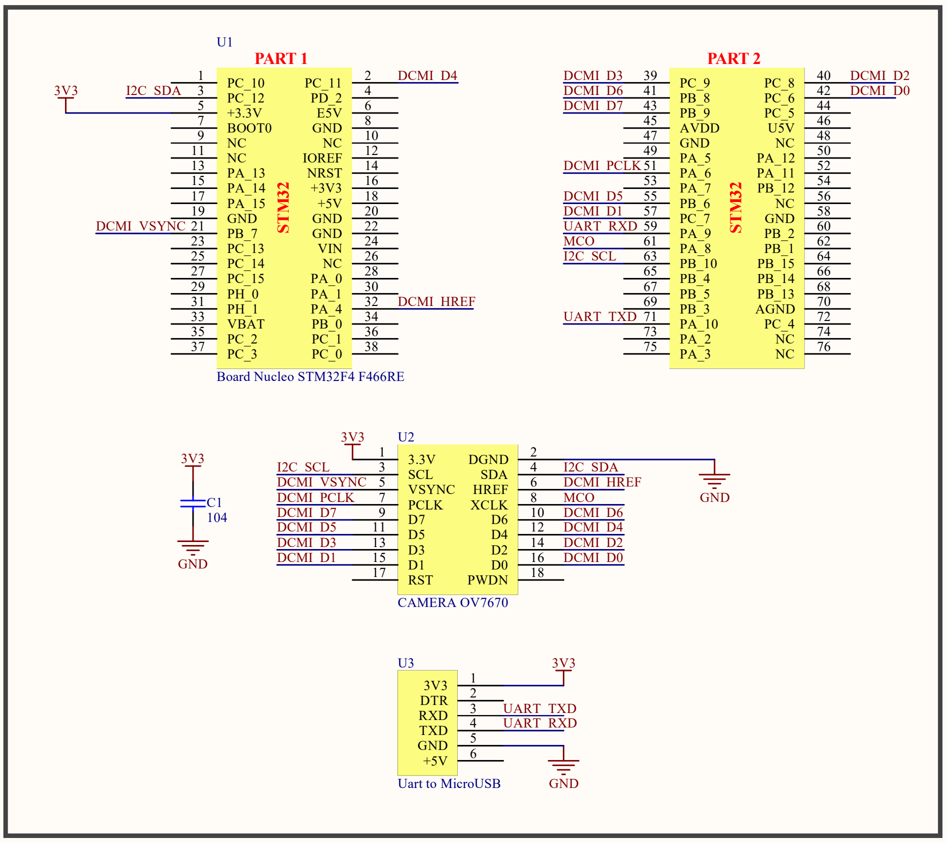


Figure 2.11: *Schematic*

2.4 The clock configuration

2.4.1 Main clock tree

Many systems need high performance, the clock configuration is important. To reduce the time's processing is smallest, the core clock of stm32f446re is setup to maximum (180Mhz). The CubeMX tool makes clock setup easier. Figure 2.12 below shows the clock tree of this project.

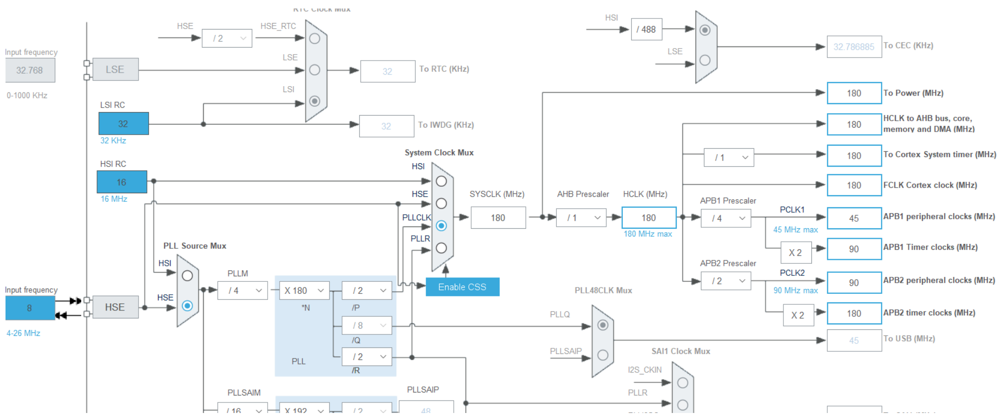


Figure 2.12: *Main clock tree of the CPU*

2.4.2 The clock source for the camera module

The OV7670 camera does not contain the crystal clock. If using an external crystal is complex, so using a built-in MCO is a good choice. The camera's clock source has ranged from 10 to 48 Mhz. The MCO of Cotex M4 support from 1 - 5 divider. Therefore, the PLL clock (180Mhz) is divided to 5 and getting 36Mhz (in of range). Clock selection is shown in figure 2.13 below.

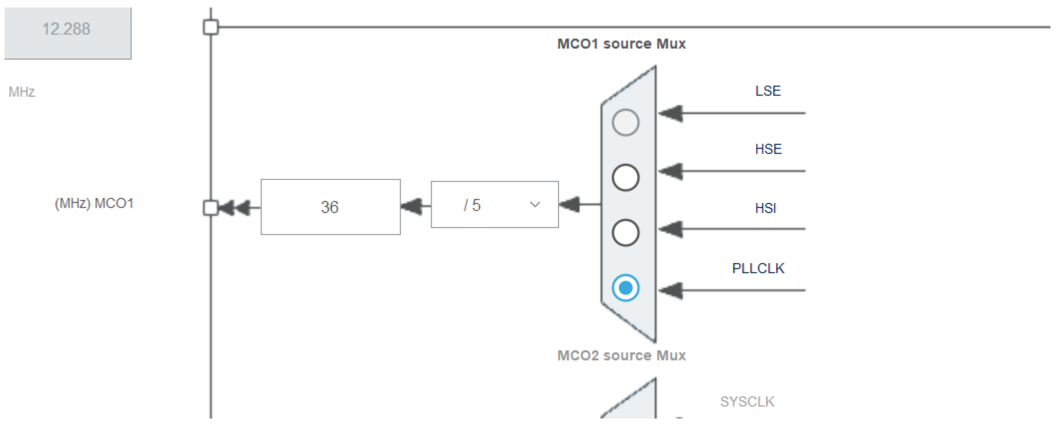


Figure 2.13: *Clock source for camera module using MCO*

2.5 Software

2.5.1 Interface

1. Serial camera control bus (SCCB)

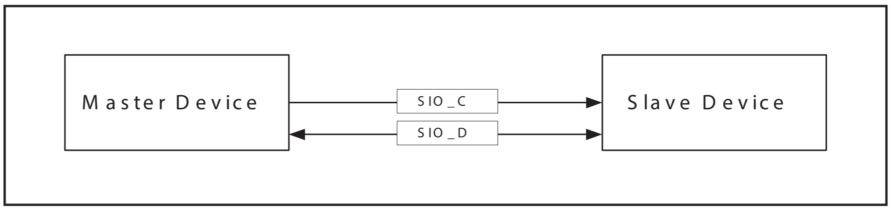


Figure 2.14: *Serial camera control bus connection*

The Serial Camera Control Bus interface controls the Camera Chip sensor operating. The SCCB interface is similar to the I2C interface, but the difference between them is the ACK bit. The ACK bit of the I2C bus is a don't-care bit on the SCCB bus. The waveform of interface is at Figure 2.15

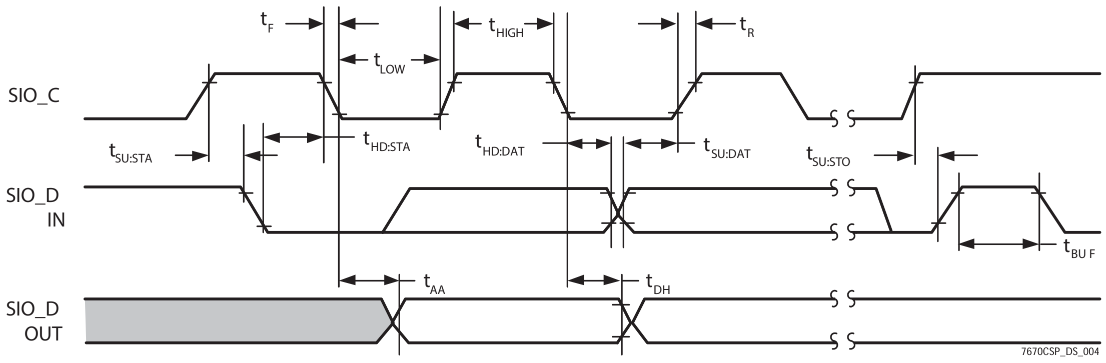


Figure 2.15: *Serial camera control bus waveform*

2. Display data bus
The display data bus has the VSYNC signal, HSYNC (or HREF) signal, and D0-D7 for data. The start of the frame signal is defined as the VSYNC signal. Each line of the frame is controlled by the HSYNC signal.

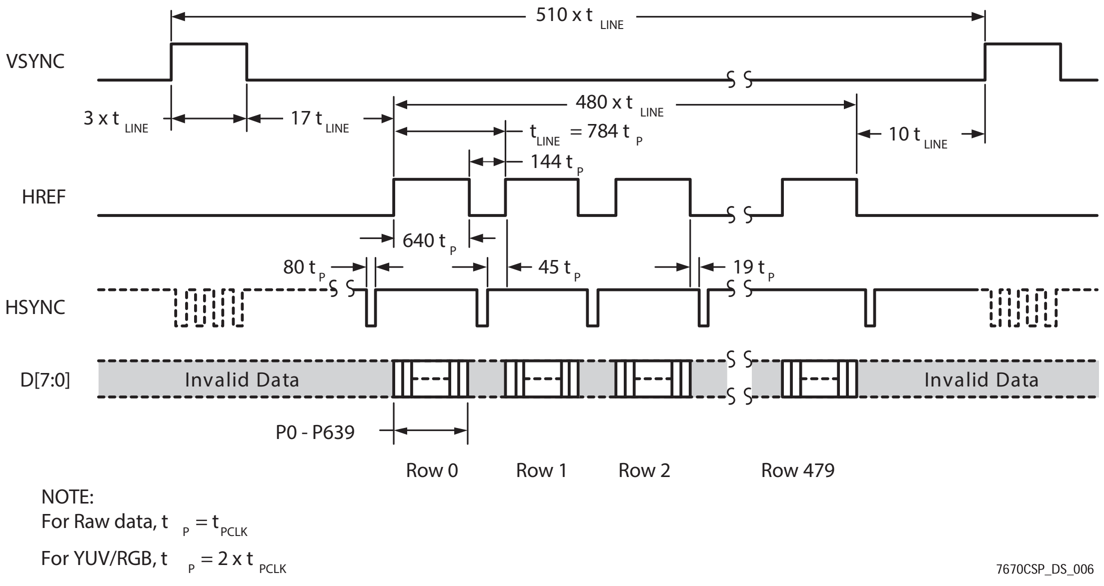


Figure 2.16: *Display data bus waveform*

Besides, The PCLK is the most important that is synchronous the data output on the data bus. The figure 2.17 is showed the waveform of transmission.

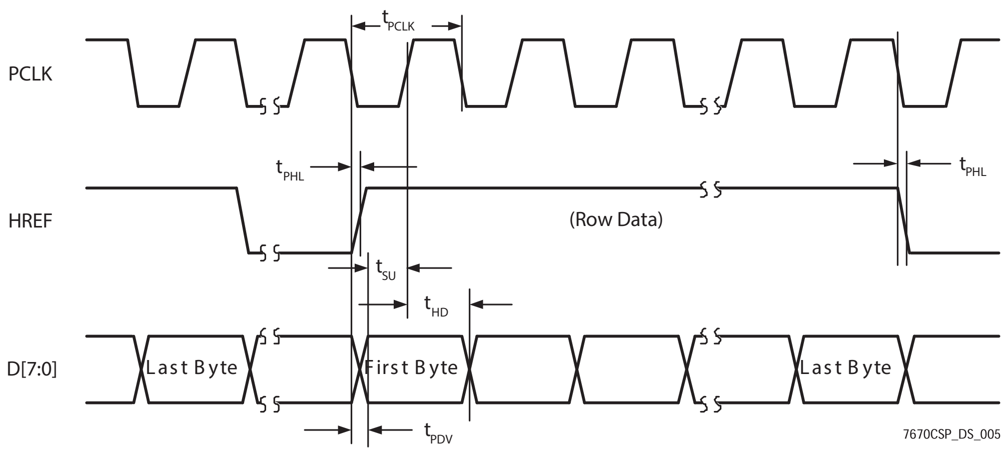


Figure 2.17: *One bytes data transfer*

2.5.2 Memory management

MicroController memory organized

The memory intended for the raw data buffer and the jpeg data is too large. If using global memory define at array form, the uninitialized data will be bigger. The space for that memory is too large. Therefore, using dynamic memory is reasonable.
The heap partition is much larger and can be released when don't use it. The raw data buffer needs 38400 bytes of space. The jpeg data is dynamic that can use one KB for simple images and ten KB for complex images. When the required memory is large than the current memory allocated, the processor will automatically reallocate.

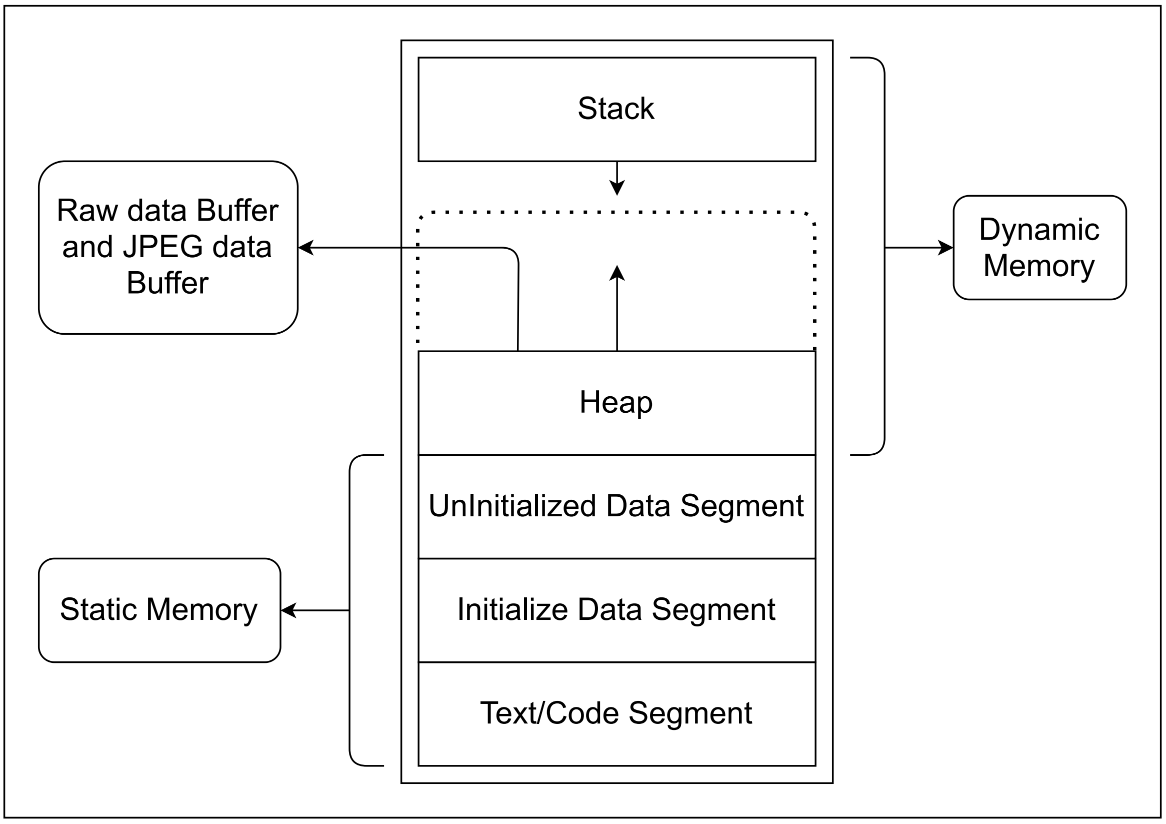


Figure 2.18: *Memory management*

RGB565 and YCrCb colour arrangement

For RGB565, each pixel is encoded to 2 bytes. The Red colour use 5 bits, Green is 6 bits and Blue is 5 bits. Sort order as shown in figure 2.19 below.

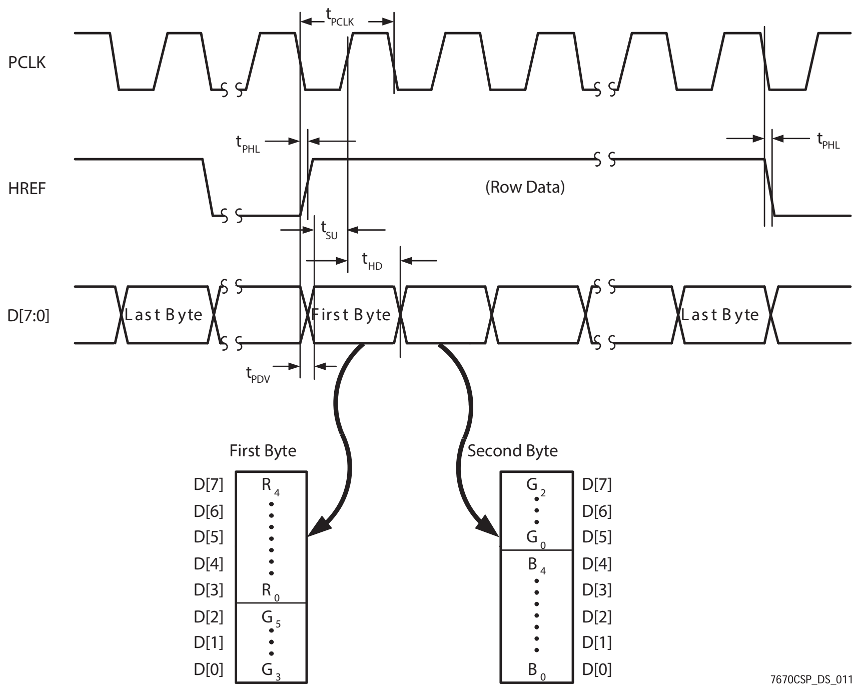
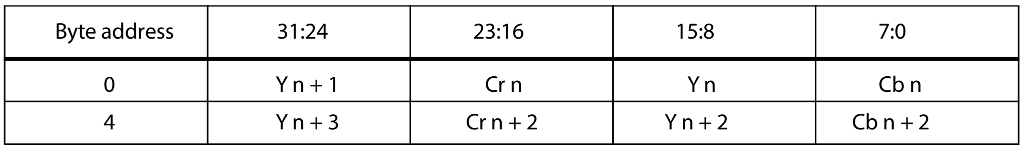


Figure 2.19: *RGB565 output waveform*

For YCbCr, the camera supports the 4:2:2 form. Pixel components are Y (luminance or "luma"), Cb and Cr (chrominance or "chroma" blue and red). Each component is encoded in 8 bits. Luma and chroma are stored together (interleaved) as shown in the figure 2.20 below.



Byte address	31:24	23:16	15:8	7:0
0	Y n + 1	Cr n	Y n	Cb n
4	Y n + 3	Cr n + 2	Y n + 2	Cb n + 2

Figure 2.20: *YCbCr colour format arrange in the memory cell*

2.5.3 Software achitecture

The software architecture is organized with four layers.

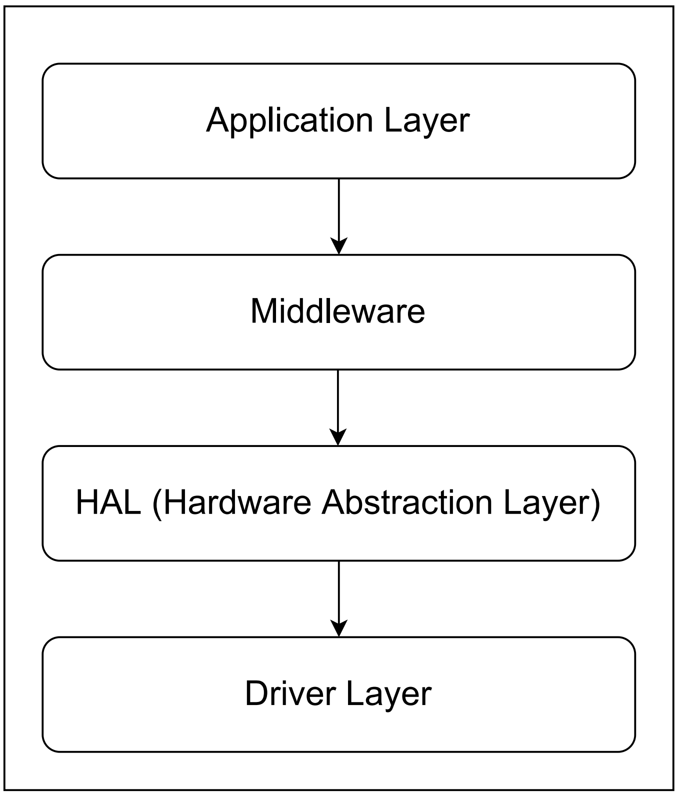


Figure 2.21: *Software architecture*

1. Driver layer: This layer is used to connect hardware with software. Such as GPIO, DCMI, Uart are controlled at the register layer.
2. Hardware abstraction layer: This layer contains STM32HAL and my own HAL [7]. My own HAL is built for the OV camera, functions control the camera as color format, exposure, saturation,...
3. Middleware layer: At this layer can be RTOS or libjpeg. In this system, the libjpeg is used to convert raw data images to standard jpeg images.

[7]Hardware Abstraction Layer

4. Application layer: To control the system easier, the software needs to build some functions. Several functions are capture, stop-capture, change camera characterized. Functional tools will be updated in the following section.

2.5.4 Software implementation

Capture and transmission timing

To optimize in the best way, the stage of one frame process doesn't take place sequentially. It must process continuously. After the system starts to capture the image, the CPU will process line by line the image. The jpeg image will be transmitted when the processing have already done.
After set up DMA for Uart transmission, the software will get started new capturing. Therefor, the frame rate will be increased.

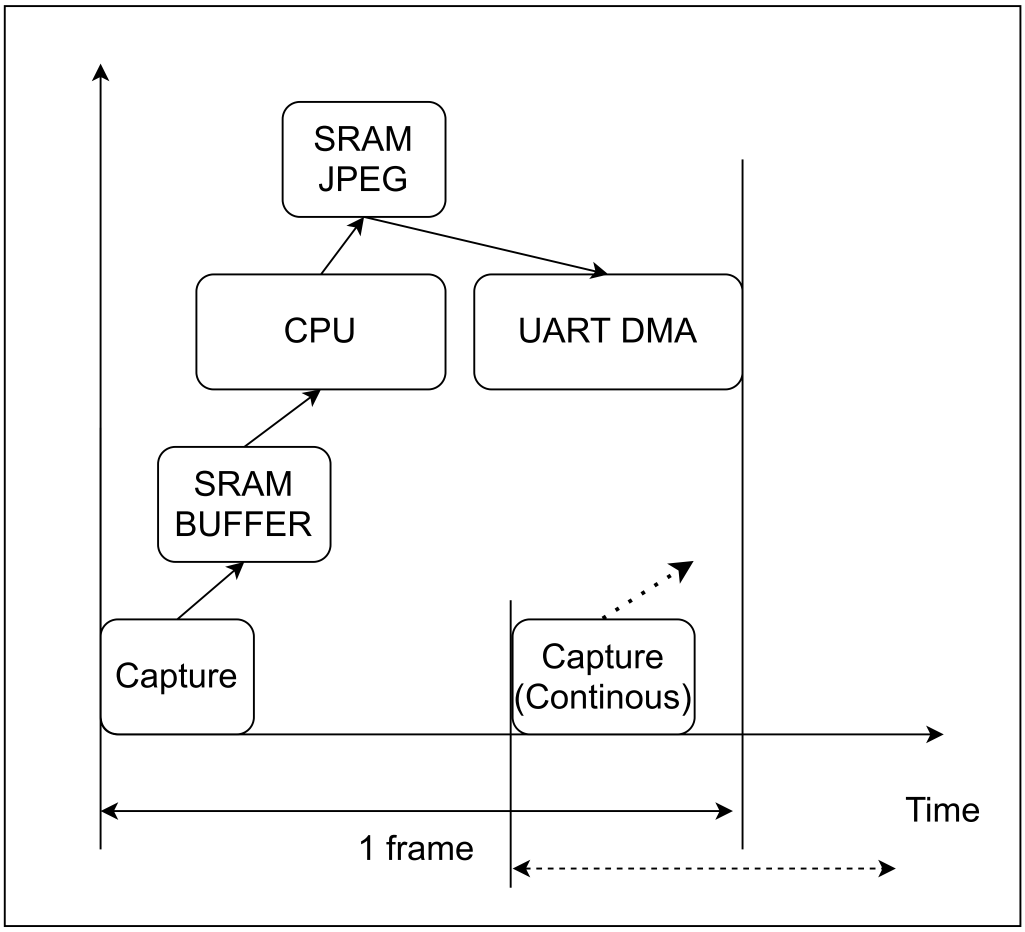


Figure 2.22: *Capture and transmission timing*

Chapter 3

Design Issues

3.1 Constraint issues

3.2 Function issues

- Supported jpeg format files.
- Capture photo or streaming video.
- The serial output data is a common connection.

3.3 Realtime issues

- The system is soft-realtime, but the latency allowed must be smaller than 500ms.

3.4 Concurrent issues

- The system can be active in any environment, but not too hot and humid, e.g the drone system.
- Muti-function: The system can capture images, processing and transmit them at the same time.
- Because of using Uart output connection, the system can easily connect to another system.

3.5 Reactive issues

- Discontinuous interaction: Power on demand.
- Response to external non-periodic events: Can start or stop capture images at any time.

Chapter 4

Conclusion and Plan

4.1 Conclusion

The subject helps us explain much of the knowledge. The jpeg compress on low power processor as Cortex M4. The digital camera interface module, how it works. That subject, streaming video on an embedded system can apply to many projects, for many purposes.

4.2 Plan

In the future, we will design the system for the streaming video from the drone to the server.

Chapter 5

References

Sang Truong Tan *et. al.* [1, 2, 3]

Bibliography

[1] OmniVision, “The ov7670 camerachip image sensor,” 2006, pp. 1–43.

[2] ST, “Stm32f446re datasheet.” ST life.augmented, January 2021.

[3] ——, “Rm0390 reference manual.” ST life.augmented, March 2021.