Camera Module
AN5640
User Manual

ALINX

Table of Contents

Part 1: Camera Module AN5640 General Description

The camera module AN5640, use CMOS chip image sensor OV5640 from OmniVision Corporation of the United States, which supporting auto focus function. The CMOS OV5640 chip supports DVP and MIPI interfaces. On the OV5640 module, images are transmitted through the DVP interface and the FPGA connection. Figure 1-1 detailed the AN5640 module product photo.

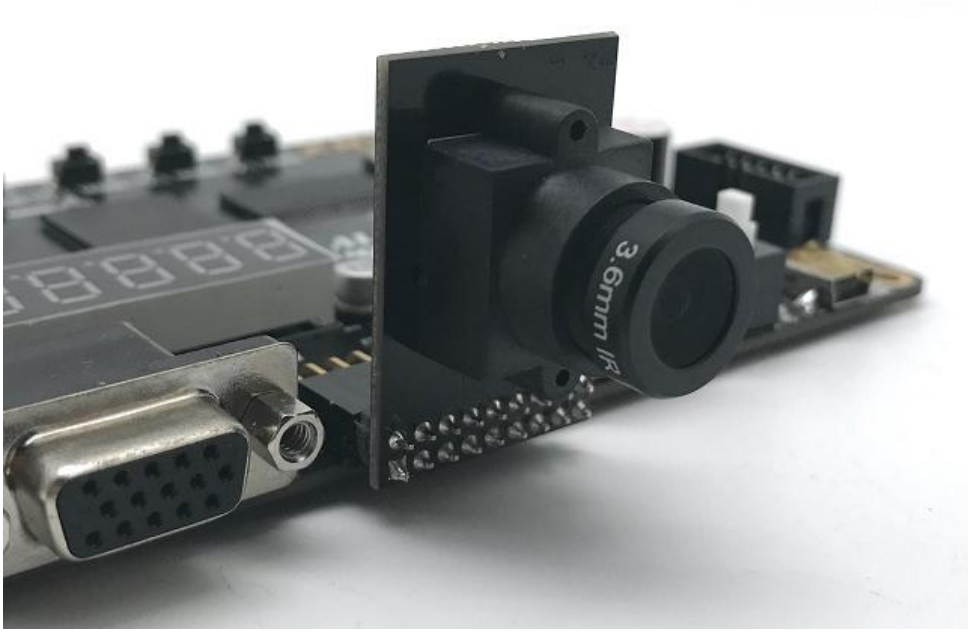


Figure 1-1: AN5640 module product photo

Part 1.1：AN5640 Camera Module Detail Parameter

- Module Interface: DVP interface
- support for images sizes: 5 megapixel
- Photosensitive chip: OV5640
- optical size of 1/4"
- Module content: OV5640 power supply circuit, flash drive circuit
- automatic image control functions: Manual focus, automatic exposure control (AEC), automatic white balance (AWB)
- support for output formats: RAW RGB, RGB565/555/444, CCIR656, YUV422/420, YCbCr422, and compression

- maximum image transfer rate:

  QSXGA (2592x1944): 15 fps

  1080 30 fps

  1280x960: 45 fps

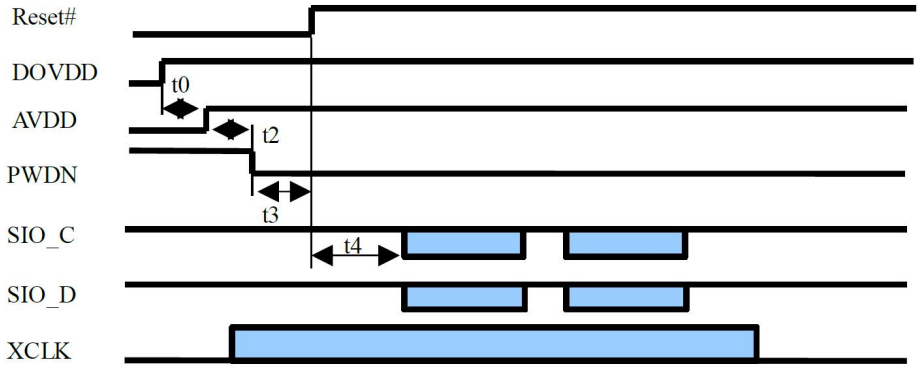
  720p: 60 fps

  VGA (640x480): 90 fps

  QVGA (320x240): 120 fps

- Working temperature: -30~70°C, stable working temperature is 0~50°C

Part 1.2：Chip OV5640 power-on requirements



t0:>=0ms. Delay from DOVDD stable to AVDD stable

t2:>=5ms. Delay from AVDD stable to sensor power up stable

t3:>=1ms. Delay from sensor power up stable to Reset# pull high

t4:>=20ms. Delay from Reset pull high to SCCB initialization

The power-on steps of the chip OV5640 are as follows:

- Step 1: ResetB is pulled low, reset AN5640, PWDN is pulled high
- Step 2: Both DOVDD and AVDD are powered up simultaneously, which is implemented in the power supply design of the module.
- Step 3: After 5ms of AVDD reaching stable, pull PWDN to low.
- Step 4: After 1ms of PWDN go low, pull high ResetB
- Step 5: After 20ms, initialize OV5640 by SCCB initialization:

Part 1.3：Register Configuration of CMOS Chip OV5640

The register configuration of the OV5640 is configured through the I2C interface of the FPGA (or other CPU). The user needs to configure the correct register value to let the OV5640 output the image format we need. In our example, we configure the OV5640 as the image format of the 720P (1280x720) video output image and the RGB565 frame rate of 30fps.

To facilitate debugging, the user can configure registers to enable internal test images of the OV5640, such as color bars and color squares.

Color bar

write_i2c(0x503d, 0x80);

write_i2c(0x4741, 0x00);

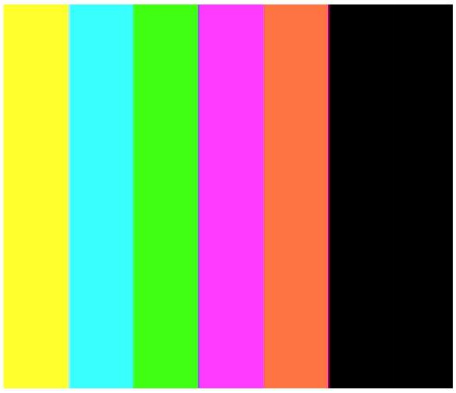


Figure 1-2: Color Bar

Color square

write_i2c(0x503d, 0x82);

write_i2c(0x4741, 0x0);

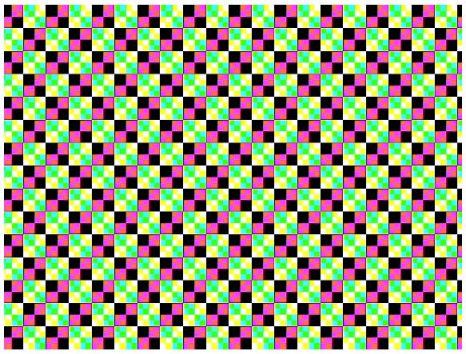


Figure 1-3: Color Square

The data format of the OV5640's camera output is configured in the following 0x4300 registers. In our example, the OV5640 is configured as an RGB565 output format.

0x4300	FORMAT CONTROL 00	0xF8	RW

Format Control 00
Bit[7:4]: Output format of formatter module
0x0: RAW
Bit[3:0]: Output sequence
0x0: BGBG... / GRGR...
0x1: GBGB... / RGRG...
0x2: GRGR... / BGBG...
0x3: RGRG... / GBGB...
0x4~0xF: Not allowed
0x1: Y8
Bit[3:0]: Does not matter
0x2: YUV444/RGB888 (not available for full resolution)
Bit[3:0]: Output sequence
0x0: YUVYUV..., or GBRGBR...
0x1: YVUYVU..., or GRBGRB...
0x2: UYVUYV..., or BGRBGR...
0x3: VYUVYU..., or RGBRGB...
0x4: UVYUVY..., or BRGBRG...
0x5: VUYVUY..., or RBGRBG...
0x6~0xE: Not allowed
0xF: UYVUYV..., or BGRBGR...
0x3: YUV422
Bit[3:0]: Output sequence
0x0: YUYV...
0x1: YVYU...
0x2: UYVY...
0x3: VYUY...
0x4~0xE: Not allowed
0xF: UYVY...
0x4: YUV420

There are many more registers on the OV5640, but many register users do not need to understand, the configuration of the registers can be configured according to the OV5640 application guide. If you want to know more about the register information, you can refer to the register description in the OV5640 datasheet.

Part 2：Hardware Connection

The following ALINX AX301 FPGA development board is an example to introduce the hardware connection of the camera module AN5640 and ALINX serial FPGA development kit. The 18-pin female headers of camera module AN5640 detailed in future 2-1.

Figure 2-1: 8-pin Female header of Camera Module AN5640

Pin	Pin Name	Pin	Pin Name
Pin1	3.3V	Pin2	GND
Pin3	CMOS_SCLK	Pin4	CMOS_SDAT
Pin5	CMOS_PCLK	Pin6	CMOS_VSYNC

Pin7	CMOS_D3	Pin8	CMOS_D2
Pin11	CMOS_XCLK	Pin12	CMOS_HREF
Pin13	CMOS_D0	Pin14	CMOS_D4
Pin15	CMOS_D5	Pin16	CMOS_D1
Pin17	CMOS_RESET	Pin18	CMOS_PWDN

On the AX301 FPGA development board, there is a 16-pin CAMERA interface (J5). When the OV5640 module is connected to the development board, simply insert the module into the CAMERA interface on the development board. Pin1 of the module is aligned with Pin1 of the CAMERA interface on the development board (square pad is 1 pin). Figure 2-2 detailed the camera module connected with AX301 FPGA board.

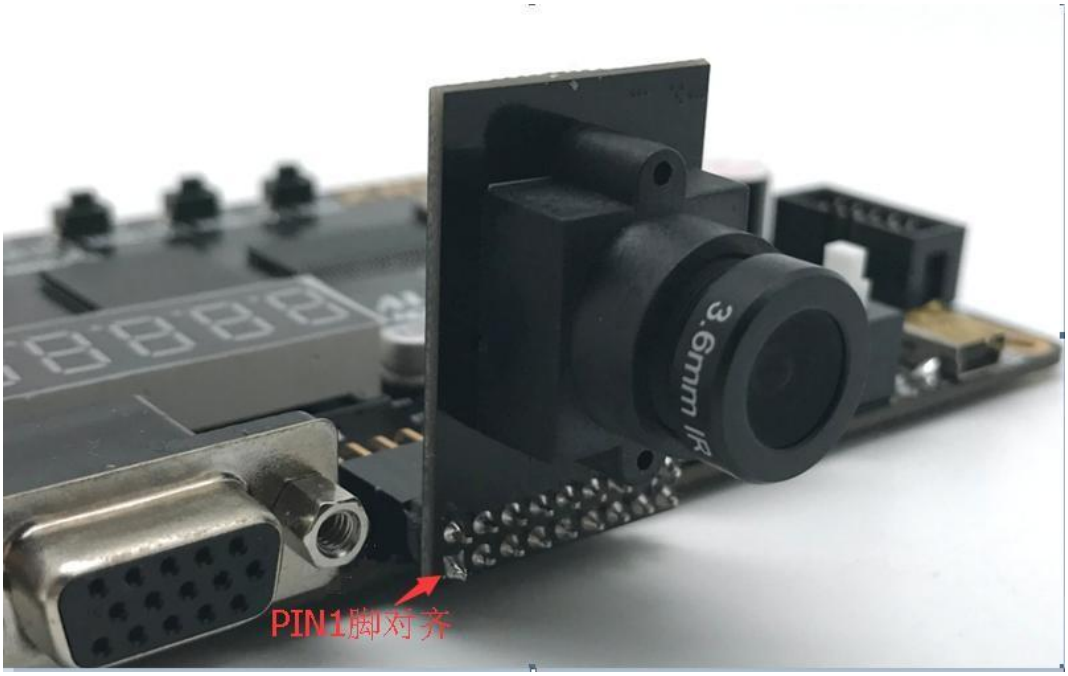


Figure 2-2: The Camera Module Connected with AX301 FPGA Board

After the connection, the pin correspondence between the OV5640 and the FPGA of the development board AX301 is as follows:

OV5640 Signal Name	Camera Interface Pin	FPGA of AX301
CMOS_SCLK	PIN3	F1
CMOS_SDAT	PIN4	F3

CMOS_PCLK	PIN5	G1
CMOS_VSYNC	PIN6	F2
CMOS_D3	PIN7	M1
CMOS_D2	PIN8	G2
CMOS_D7	PIN9	J2
CMOS_D6	PIN10	J1
CMOS_XCLK	PIN11	K2
CMOS_HREF	PIN12	K1
CMOS_D0	PIN13	L2
CMOS_D4	PIN14	L1
CMOS_D5	PIN15	N5
CMOS_D1	PIN16	M6
CMOS_RESET	PIN17	N6
CMOS_PWDN	PIN18	M7

If it is the development board of AX515, AX530 or other ALINX serial FPGA Board, because there is no camera interface on the development board, we need to connect the camera module with our adapter board and insert the adapter board into the expansion port of the development board (AX515 and AX530 are expansion port J3; AX822 is the expansion port J15), then insert the camera module into the expansion board. Figure 2-3 detailed the camera module connected with AX515 FPGA board.

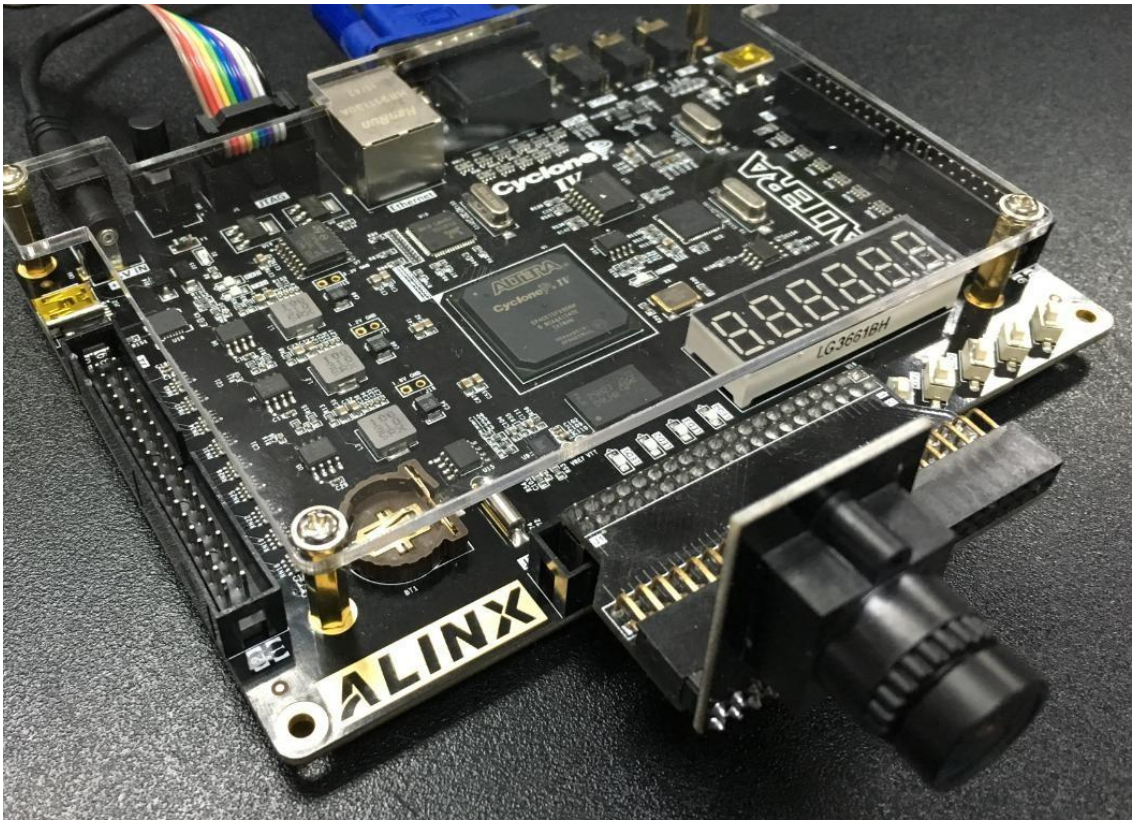


Figure 2-3: The Camera Module Connected with AX515 FPGA Board

Part 3: AN5640 VGA Display Experiment

In this experiment, the AX301 development board is taken as an example. The 1024*720 video image of OV5640 is output to the VGA display. After the program is powered on, the register of the OV5640 is set, and the image of the acquisition camera is stored in the SDRAM. The image data is then taken out of the SDRAM and displayed on the VGA display.

Part 3.1 Programming

In the program, the SDRAM memory space is divided into two areas (Bank0 and Bank3), and the SDRAM is read and written in different Bank spaces. When Bank0 is writing the image captured by the camera, VGA reads the data display of Bank3; after the Bank0 finish to write an image, the space of reading and writing is exchanged, Bank3 starts to write the image captured by the camera, and Bank0 is the image for reading out the VGA output.

In addition, the image obtained from the OV5640 camera is 1024 x 720 image size, and the VGA display can only display 1024x768 image data. We need to add black background data to the extra 48 lines in the VGA display program.

The OV5640 VGA display routine consists of a top-level module sdram_ov5640_vga.v, a power-on waiting program power_on_delay.v, a register configuration program reg_config.v, a camera data acquisition program CMOS_Capture.v, a VGA display and SDRAM control program sdram_vga_top. v, a system control module system_ctrl.v.

The completed engineering architecture is shown in the figure below:

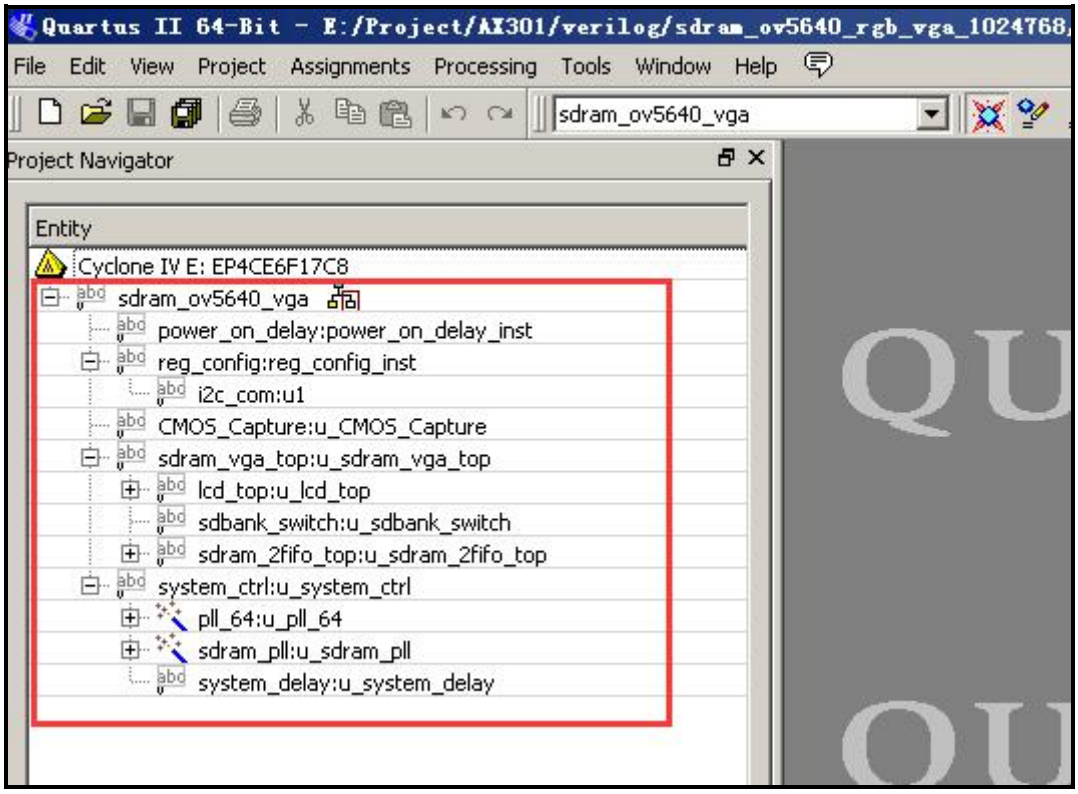


Figure 3-1: The Engineering Architecture

The function description of each module, detailed as below:

1). Power-on waiting program：power_on_delay.v

Because the OV5640 chip has power-on timing requirements, this program is waiting for a period of time after the FPGA is powered on and then enabling

the OV5640 register to meet the timing requirements of the OV5640.

2). OV5640 register configuration module：reg_config.v

After power-on the FPGA, the register configuration program of the OV5640 calls the I2C communication program to set the parameters of the register of OV5640 chip. Here the images output by the OV5640 chip is set to RGB565 format, and the numbers of image is 1280*720.

3). Camera Image Acquisition Program：COMS_Capture.v

The camera image acquisition program converts the 8-bit image from the OV5640 module into a 16-bit data width and generates a write signal for the SDRAM. In addition, the program generates a frame_valid signal to indicate that an image acquisition is completed, and informs the read and write space exchange of Bank0 and Bank3 of SDRAM.

4). Sdram Control and Read/Write Program: sdram_top.v

The sdram_top module and three submodules (sdram_ctrl.v, sdram_cmd.v, sdram_wr_data.v) implement sdram initialization, user interface read and write command parsing, sdram burst read and write, self-refresh and pre-charge operation control and etc.

The sdram_ctrl module implements SDRAM initialization, 60ms self-refresh, user read and write request command parsing, and uses status machines and counters to generate status bits for different SDRAM operations.

The sdram_cmd module is based on the state machine init_state and work_state generated in the sdram_ctrl module, to generate various SDRAM control or burst read and write commands.

Sdram_wr_data module is SDRAM read and write bidirectional data control module, When writing SDRAM, the data is transferred to the SDRAM data bus, and when the SDRAM is read, the data on the SDRAM bus is transferred to the user interface.

5). FIFO Control Program Module: dcfifo_ctrl.v

The Dcfifo_ctrl.v module is used to control the read FIFO, write FIFO, SDRAM read and write commands and read and write addresses. In this experiment, the data written to the SDRAM is first stored in the write FIFO, and the data read from the SDRAM is first stored in the read FIFO.

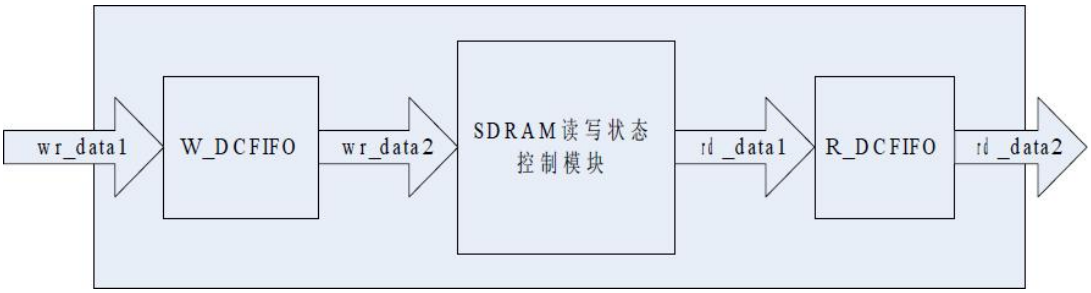


Figure 3-2: FIFO Control Program Module

A DDR burst write request is generated when the amount of data in the write FIFO is greater than the set write burst length (128)

```
158     end
159     else if(ddr_init_done == 1'b1)
160       begin                          //写入优先，带宽内防止数据丢失
161       if(wrf_use >= wr_length & !wr_flag) //写入FIFO的数据数量超过burst长度,写DDR开始
162         begin                        //wrfifo满突发长度
163         ddr_wr_req <= 1;             //写ddr使能
164         wr_flag <= 1;
165         end
166       else if(ddr_wr_finish)//读FIFO里的数据数量小于burst长度,读DDR开始
167         begin                //rdfifo满突发长度
168         ddr_wr_req <= 0;     //写ddr空闲
169         wr_flag <= 0;
```

Figure 3-3: Generate DDR Burst Write Request program

The Sdram read request is generated when the amount of data in the read FIFO is less than the set read burst length (128).

```
191     else if(ddr_init_done == 1'b1)
192       begin                          //写入优先，带宽内防止数据丢失
193       if(rdf_use <= rd_length &  !rd_flag)//读FIFO里的数据数量小于burst长度,读DDR开始
194         begin                //rdfifo满突发长度
195         ddr_rd_req <= 1;     //读ddrm使能
196         rd_flag <= 1;
197         end
198       else if(ddr_rd_finish)
```

Figure 3-4: Generate Sdram Read Request program

6). VGA Display Program: lcd_driver.v

The vga_disp.v module implements image display of a VGA display, and generates line synchronization, column synchronization, and timing of image data signals according to the VGA timing standard. VGA clock frequency: Take 1024x768@59.94MHz (60Hz) as an example. Each field corresponds to 806 line periods, of which 768 is the display line. Each display line includes 1344 points of clock, of which 1024 points are valid display areas. It can be seen that the clock frequency of VGA is required: 806*1344*60 is about 65MHz. Figure 3-5 and figured 3-6 detailed the timing diagram of VGA:

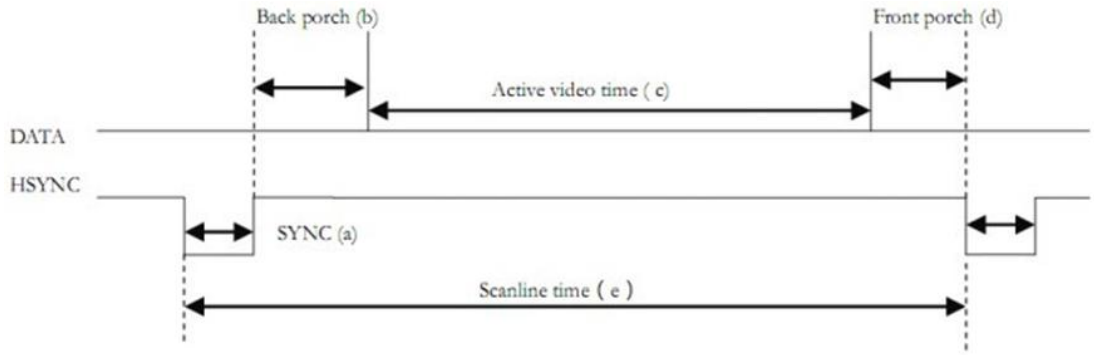


Figure 3-5: VGA Time Timing

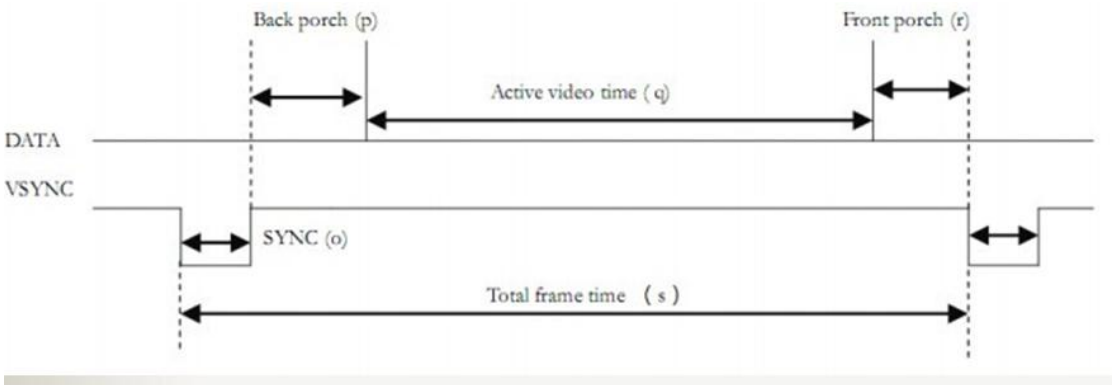


Figure 3-6: VGA Filed Timing

In addition, because the video image output by OV5640 is 1024x720 image size, but the VGA display is 1024x768 image, in order to make the video image in the middle of the VGA display, black image data has been inserted in the first

24 lines and the last 24 lines of the VGA image

```
assign   lcd_en      =  (hcnt >= `H_SYNC + `H_BACK  && hcnt < `H_SYNC + `H_BACK + `H_DISP) &&
                        (vcnt >= `V_SYNC + `V_BACK + 24  && vcnt < `V_SYNC + `V_BACK + `V_DISP - 24)
                        ? 1'b1 : 1'b0;
assign   lcd_rgb  =  lcd_en ? lcd_data : 16'd0;
assign   lcd_framesync = lcd_vs;


//------------------------------------------
//ahead a clock
assign   lcd_request =  (hcnt >= `H_SYNC + `H_BACK - 1'd1 && hcnt < `H_SYNC + `H_BACK + `H_DISP - 1'd1) &&
                        (vcnt >= `V_SYNC + `V_BACK + 24 && vcnt < `V_SYNC + `V_BACK + `V_DISP - 24)
                        ? 1'b1 : 1'b0;
assign   lcd_xpos =  lcd_request ? (hcnt - (`H_SYNC + `H_BACK - 1'b1)) : 11'd0;
assign   lcd_ypos =  lcd_request ? (vcnt - (`V_SYNC + `V_BACK - 1'b1)) : 11'd0;
```

Figure 3-7: Video Image in the Middle of the VGA display

7). SDRAMBank exchange program: sdbank_switch.

The Bank exchange program implements SDRAM read and SDRAM write in different Bank operations. When Bank0 writes the image captured by the camera, VGA reads Bank3 data display; after Bank0 writes an image finished, SDRAM reads and writes space exchange, Bank3 starts to write the image captured by the camera, and Bank0 reads the image output by the VGA.

8). System Control Program: system_ctrl.v

Generate SDRAM clock (100Mhz) and VGA image clock (65Mhz), and the program also generates a reset signal for the entire system.

Part 3.2. OV5640 VGA display experiment

After writing the program, assign the pin of the FPGA and recompile it to start the OV5640 VGA display experiment. The development board connect with the camera module and VGA interface to connect VGA display, then download sof file to FPGA, you can see 1024x768 video image on VGA monitor. Figure 3-8 and Figure 3-9 detailed the video image effect.

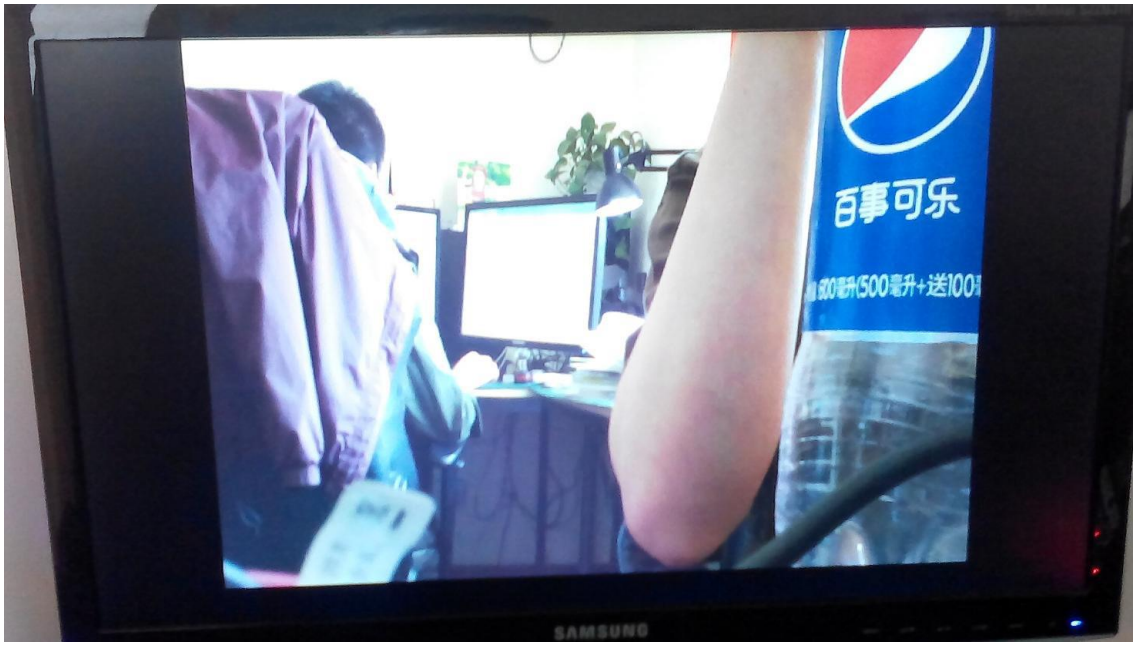


Figure 3-8: OV5640 Video Image Display Effect 1 (close-up view):

Figure 3-9: Video Image Display Effect 2 (distant view)

Part 4: LCD Display Experiment

This experiment demonstrates the video image of the OV5640 with an ALINX 7-inch LCD display, which is displayed on a 7-inch LCD screen. In the experiment, taking the AX301 development board as an example, the OV5640's 800*480 image size video image is output to the LCD for display. After the program is powered on, the OV5640 register is set first, and then the image of

the acquisition camera is stored in the SDRAM. The image data is taken out from the SDRAM and displayed on the LCD display.

Part 4.1: Programming

The programming ideas and methods are basically the same as the OV5640 VGA display example. Just explain the difference between the OV5640 LCD display and the OV5640 VGA display experiment.

1). LCD driver program: lcd_driver.v

The 7-inch LCD display has an image size of 800x480 and an image refresh rate of 60hz. The LCD driver displays the image of the LCD screen and generates line synchronization, column synchronization, DE and image data signal timing according to the data sheet of the LCD screen. The display timing requirements of the LCD screen are as follows:

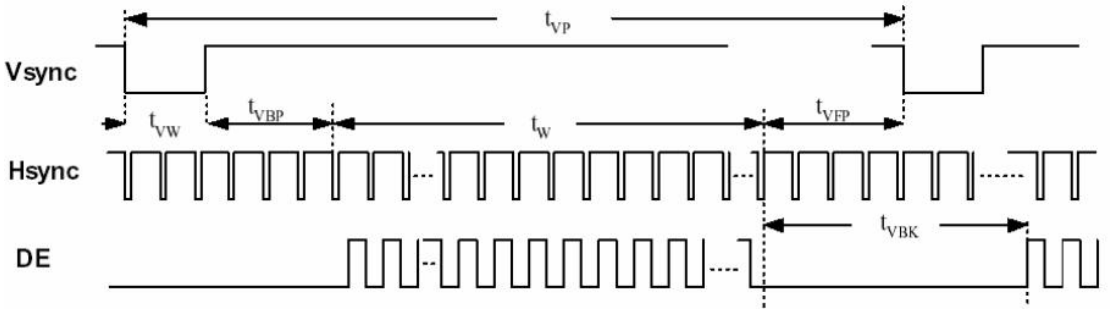


Figure 4-1: Input Vertical Timing

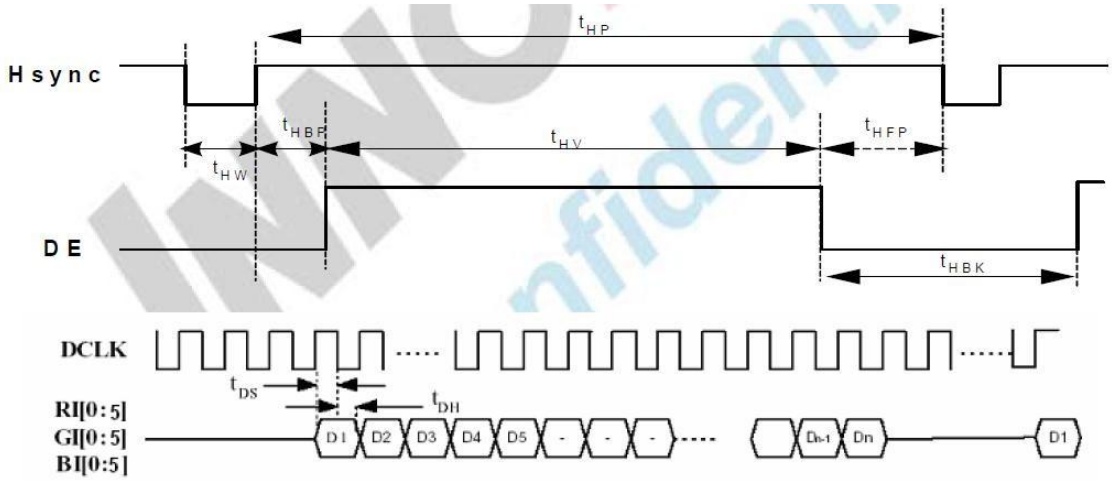


Figure 4-2: Input Horizontal Timing

Please refer to the LCD datasheet for specific timing requirements.

2). System Control Program: system_ctrl.v

7-inch LCD clock frequency is 25Mhz

3). OV5640 Register Configurator Program: reg_config.v

Configure the image output of the OV5640 to be 800x480 pixel size

Part 4.2: OV5640LCD Display Experiment

After writing the program, assign the pin of the FPGA, and after recompilation, we can start the OV5640 LCD display experiment. The FPGA development board connected with the camera module OV5640 and 7” LCD screen with LCD interface (P3), then after downloading the sof file to the FPGA, we can see the 800x480 video image on the LCD screen.

Figure 4-3: OV5640 Video Image LCD Display Effect