AN12868

Camera Interface in LPC55(S)xx

Rev. 3 — 07 September 2021 Application Note

Contents

1 Introduction

This application note introduces a parallel interface for the camera solution for LPC55(S)xx. It includes the introduction of camera interface, features and API routines, and demo.

2 Target application

The camera interface can be used as important part of camera usage as below:

- Object detection
- Gesture recognition
- Color recognition
- QR code scanning, and so on

3 Introduction of camera interfaces

A typical camera interface supports at least one parallel interface, although nowadays many camera interfaces begin to support the MIPI CSI interface.

The camera parallel interface consists of the following lines:

- Data line (D[0:11]):

  These parallel data lines carry pixel data. The data transmitted on these lines change with every Pixel Clock (PCLK).

- Horizontal Sync (HSYNC)

  This is a special signal that goes from the camera sensor. An HSYNC indicates that one line of the frame is transmitted.

- Vertical Sync (VSYNC)

  This signal is transmitted after the entire frame is transferred. This signal is often a way to indicate that one entire frame is transmitted.

- Pixel Clock (PCLK)

  This pixel clock changes on every pixel.

  The application note only focuses on Digital-Video-Port (DVP) interface which is parallel interface.

4 Features of camera interface

- Supported formats (8-bit): RGB565
- Maximum image transfer rate: 30 fps for QVGA(320 × 240). For small RAM parts, reduce the size of image and frame rate.
- Camera module tested: OV7673

- Other camera modules can be supported as long as they provide the same signal timing.

5 Function description

5.1 Camera interface engine

There is a dedicated processor in LPC55S69 which can handle the signals of camera.

It reads the data from camera and stores the data in the RAM which is accessed by Arm core. Before using dedicated processor, some configurations must be made, which includes pin configuration, clock enable, dedicated processor enable, interrupt enable and so on.

5.2 Camera driver library

The instructions of dedicated processor use the type of machine code. The code implements the function of camera interface protocol and is released in lib. Some API routines are provided in this application. User can use API routines to initialize the engine and configure the pins.

5.3 LCD display

The application note uses LCD to display the video stream from camera in real time. The high-speed SPI port is used for LCD driving port. The max speed on the SPI bus is 50 Mbit/s, so it can display the 320 × 240 resolution LCD up to 30 fps.

5.4 System clock

The camera engine shares the system clock with Arm® core. To speed up the processing time, the system clock must be configured to 150 MHz. For those below 150 MHz, such as, 96 MHz system clock part, reduce the frequency of pixel clock.

5.5 Clock source of camera

The camera needs a 50 MHz clock source which is provided by CLKOUT signal from MCU.

5.6 I^2C interface

The camera is configured through I^2C interface which can be connected with I^2C peripheral port in the MCU.

5.7 Memory usage

The 320 × 240 resolution picture requires 150 K bytes RAM space. The solution uses the space address from `0x20010000` to `0x20035800`.

Additionally, the instruction code of camera engine must run in the RAM for high performance. This solution uses the space address from `0x20040000` to `0x20043FFF` to store camera engine code.

5.8 Other supported camera modules

Other camera modules can be supported as long as they provide the same signal timing.

1. The camera module must be configured as RGB565 mode with the timing diagram, as shown in Figure 1.

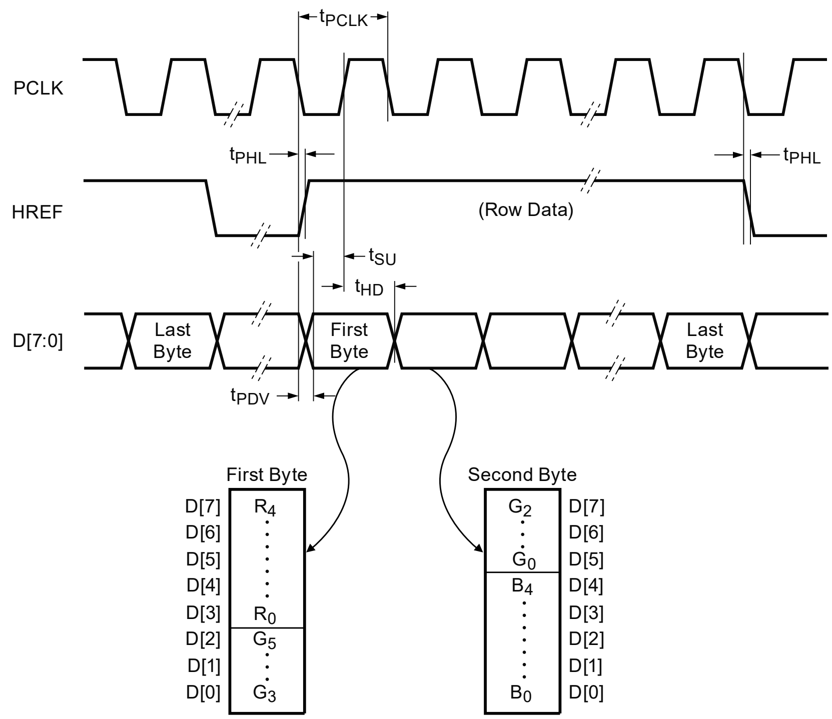

Figure 1. Camera module configuration

2. The resolution should be configured as QVGA (320 × 240).

6 Pin description

6.1 Connection of interface

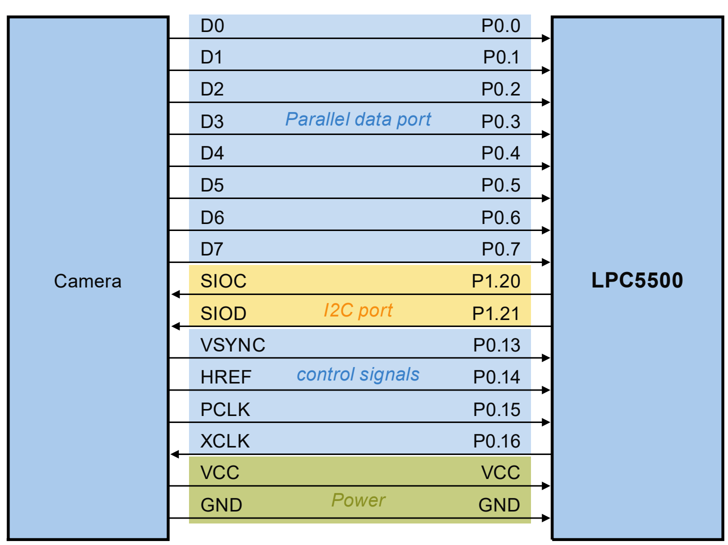

Figure 2. Connection of interface

NOTE

Use P0_16 as a `clkout` pin to provide clock source for the camera.

6.2 Requirement of interface

- The D0-D7 must be connected to P0.0-P0.7 for byte reading the data.
- SIOC and SIOD must be connected to the I^2C interface of MCU for configuration.
- The VSYNC, HREF, PCLK must be connected to pins of **Port0**.
- XCLK must be connected to a clock output pin of MCU.

7 Library and API routine

7.1 Library

The library is named by camera engine lib. It includes the instructions which have to be handled by dedicated processor.

The library can support Keil, MCUXpresso IDE, but not IAR.

7.2 API routine

The main purposes of the API routines include:

- Enable the clock of engine.
- Configure the IO as camera interface function.
- Initial the I^2C interface.
- Enable the interrupt of engine for telling the Arm core data is ready.
- Initialize and start the engine.

7.3 API routine description

Table 1. API routine

Routine	Description
`Reserved46_IRQHandler`	Interrupt routine for camera engine
`OV7670_Init`	Camera module initialization
`Camera_Init();`	Camera engine initialization
`Camera_Start();`	Camera engine start running
`LCD_Init();`	LCD module initialization
`LCD_Refresh();`	LCD refresh

7.4 Code detail description

7.4.1 System clock

Camera engine needs short time to store the data when every pixel edge comes. If the clock frequency of engine is higher, the time cost is shorter. In this solution, the system clock must be set at 150 MHz when engine is running. The code to configure system clock is shown as below:

```
BOARD_BootClockPLL150M();
```

7.4.2 I²C interface

The `flexcomm4` is used as I²C function for initializing the camera before the video starts.

7.4.3 Pin function

Table 2. Pin function

Pin	Function number	Input/output	Description
`P0_0`	15	Input	Camera engine function
`P0_1`	15	Input	Camera engine function
`P0_2`	15	Input	Camera engine function
`P0_3`	15	Input	Camera engine function
`P0_4`	15	Input	Camera engine function
`P0_5`	15	Input	Camera engine function
`P0_6`	15	Input	Camera engine function
`P0_7`	15	Input	Camera engine function
`P0_18`	15	Output	Camera engine function
`p0_13`	0	Input	GPIO as VSYNC input
`P0_15`	0	Input	GPIO as Pixel clock input
`P0_16`	2	Output	CLKOUT
`P1_20`	5	Input/output	`FC4_I2C_SCL`
`P1_21`	5	Output	`FC4_I2C_SDA`
`P1_2`	6	Output	`LSPI_HS_SCK`
`P0_26`	9	Output	`LSPI_HS_MOSI`
`P1_3`	6	Input	`LSPI_HS_MISO`
`P1_1`	5	Output	`LSPI_HS_SSEL1`
p1_11	0	Output	GPIO
`P0_29`	1	Input	`FC0_USART_RX`
`P0_30`	1	Output	`FC0_USART_TX`

From `P0_0` to `P0_7` are the low 8 bits, they can be read by engine at one read instruction which only takes one system clock cycle.

`P0_18` is set as camera engine function. It is operated by engine directly such as set logic high level, clear zero, toggle and so on. `P0_18` will be toggled by engine after every VSYNC edge.

`P0_13` and `P0_15` are input function pins which can receive the VSYNC and pixel signals. The Pixel clock is 1/4 of clock source, it is 12.5 MHz.

As clock output pin, `P0_16` provides 50 MHz clock to camera as its clock source.

The `LSPI_HS` is high-speed SPI interface which is used to drive the LCD TFT screen.

7.4.4 LCD function

LCD is used to display the video of camera in real time. High-speed SPI is used to drive the LCD. The routine named by `LCD_Refresh()` is a very high-efficiency routine, it can refresh a 320 × 240 picture in about 24 ms, up to 40 fps. The bus clock of high-speed SPI can reach up to 50 MHz. The DMA is not used to drive the LCD. Instead, only software is used to drive LCD where the Arm core writes the SPI FIFO data write register once it is empty.

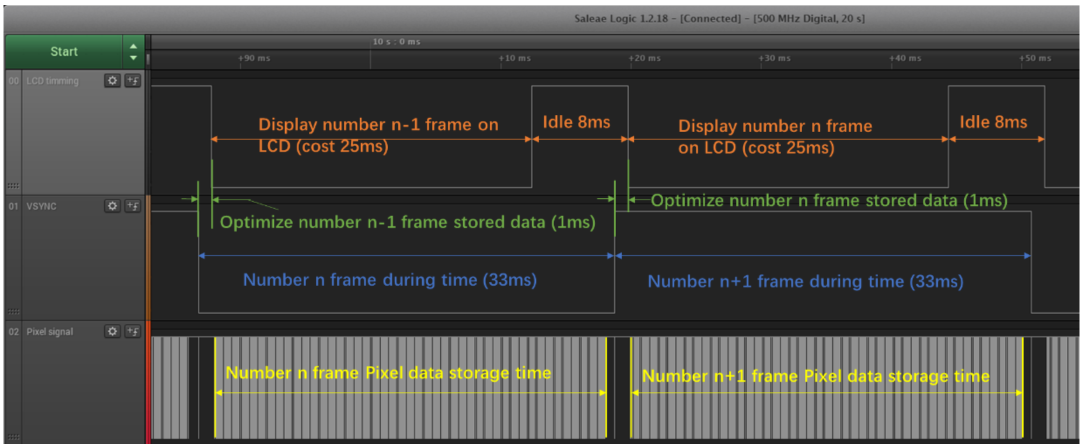

Figure 3. LCD function

7.4.5 OV7670 camera module

The camera module is set as QVGA RGB565 mode. The 16-bit data is received with two pixel edges (high-byte first and low-byte follows).

At the beginning, the MCU initializes the camera through I^2C interface. Then, the video stream data is stored in the RAM by camera engine.

7.4.6 Reserved 46_IRQHandler

Same with other peripheral handler, camera engine handler is implemented by Arm core once engine finishes the storage operation.

In the handler, a flag is set as one. In the `while(1)` routine, the refreshing operation can be allowed when flag turns to logic one.

7.4.7 Data buffer

There needs 150 kB space for one frame of video and LPC55S59 has about 300 kB RAM space. Double buffer is not possible. Only one buffer is used. Because the LCD refresh time (24 ms) is shorted than data storage time (33 ms), so Arm always reads the data for LCD refresh is earlier than storage operation by the engine. Therefore, the media data cannot be lost.

7.4.8 Timing

The LCD always displays the previous frame data from the camera. Before displaying, the data stored must be optimized by dedicated processor for exchanging the high and low bytes of every pixel. Because the speed of LCD module displaying is higher than the speed of camera interface reading data, the single data buffer is used in this application. While the current frame data is stored, the LCD displays the previous frame data.

8 Demonstration

1. Build and compile the project.
2. Plug one side of USB cable in PC USB port and another side in the debug link port in the EVK board, and then download the image in the MCU.
3. Connect camera to MCU by referring to connection of interface in Chapter 4.
4. Connect the LCD panel with Arduino port on the LPCXpresso55s69 EVK board.
5. Connect one side of USB cable to USB power socket and the other to 5 V Power only USB port P5.
6. The LCD displays the video frame from camera as shown in Figure 4.

WARNING

If the code cannot execute after downloaded, one of the reasons is that the MCU goes to ISP mode. The root cause is that camera module generates logic level signal on P0.5 pin during MCU is under reset state. The solutions can be as below:

1. Disconnect the pin before reset and connect it after reset.

2. Power down the camera module before reset and power up after reset.

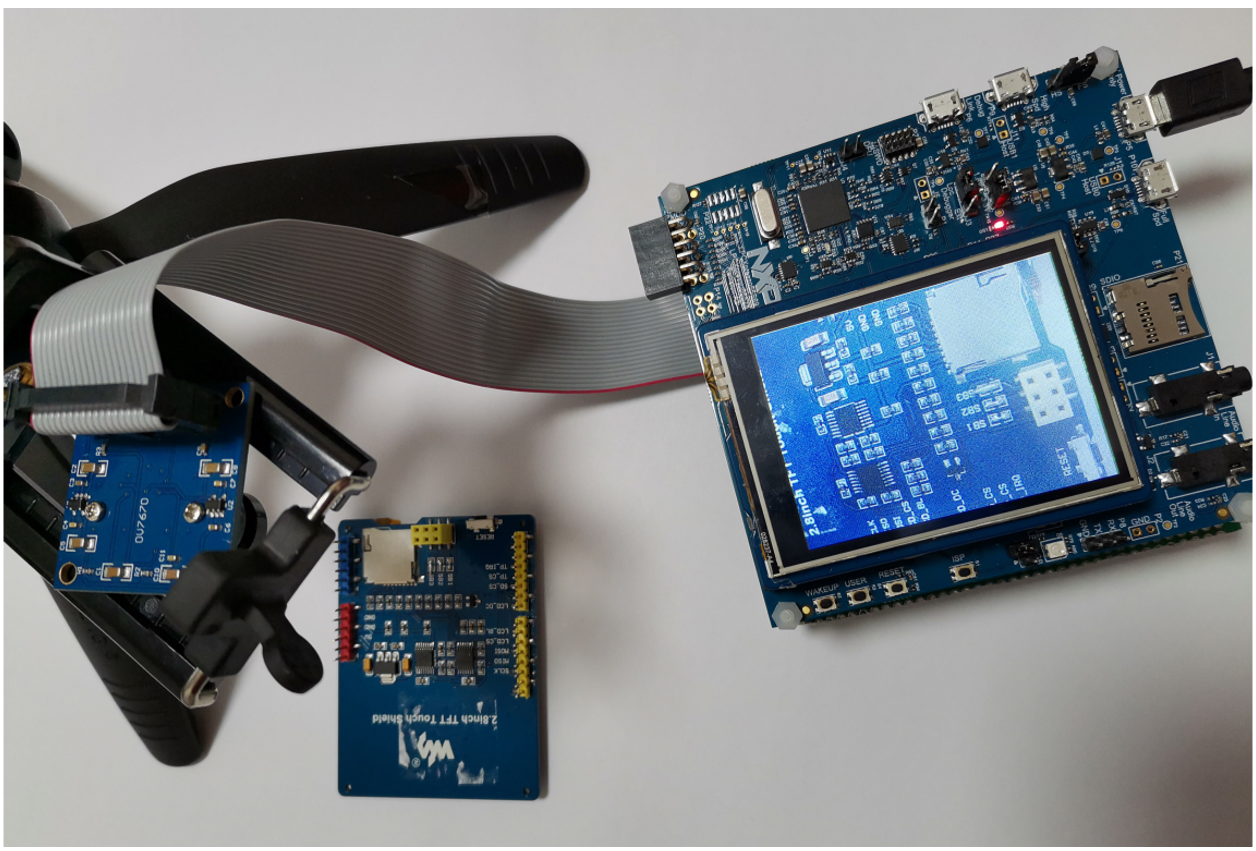

Figure 4. Video frame

9 Revision history

Table 3. Revision history

Rev.	Date	Description
3	07 September 2021	• Added purchase link for camera module in Purchasing LCD module and camera module used in this demo • Changed "coprocessor" to "dedicated processor"
2	07/2021	• Added a warning in Demonstration • Editorial updates
1	10/2020	Replaced LPC55(S)xx for LPC55S69
0	05/2020	Initial release

A Purchasing LCD module and camera module used in this demo

You can purchase the LCD board and camera module from the below links:

1. https://www.waveshare.com/2.8inch-tft-touch-shield.htm
2. http://www.waveshare.net/shop/2.8inch-TFT-Touch-Shield.htm
3. https://www.amazon.com/gp/product/B07S66Y3ZQ/ref=ppx_od_dt_b_asin_title_s00?ie=UTF8&psc=1
4. https://detail.tmall.com/item.htm?spm=a230r.1.14.10.54a37d76CA0csy&id=554248152327&ns=1&abbucket=12

How To Reach Us

Home Page:

nxp.com

Web Support:

nxp.com/support

For more information, please visit: http://www.nxp.com

For sales office addresses, please send an email to: salesaddresses@nxp.com

Date of release: 07 September 2021

Document identifier: AN12868