ATLANTIS PRESS

Advances in Engineering Research, volume 149

2nd International Conference on Mechanical, Electronic, Control and Automation Engineering (MECAE 2018)

Design of Image transmission and Display System Based on ZYNQ

Ran He [a], Zili Chen [b]

Department of UAV Engineering, Army Engineering University, Shijiazhuang 050003, China.

[a]17731019962@163.com, [b]chenzili_jxxy@163.com

Keywords: ZYNQ, Embedded Linux, QT, FFMPEG, IP Network Camera.

Abstract. Aiming at the problems of high cost, large volume of the PC image transmission and display system, poor backend portability, transmission delay of the image transmission and display system using network camera, an image transmission and display system of IP network camera based on SOC was designed. The system used ZYNQ-7000 series processors to build an embedded Linux system in PS to capture, decode and store network camera image data. In PL, the image data stored in DDR3 was transmitted to the HDMI interface to drive the display. The experimental results showed that the image transmission delay acquired by the system was low and its real-time performance was good. The design system meets the needs of subsequent image processing and pattern recognition.

1. Introduction

In recent years, the image processing technology had developed rapidly, and had been widely used in aerospace, medical, security, military and other fields. The quality of the image directly determined the effect of advanced processing [1]. The traditional image transmission and processing system obtained the image information through the image acquisition card and then processes it through a PC. Although this way had the advantages of fast speed and short development period, it had the disadvantages of high cost and poor portability [2]. Many image transmission and display systems used serial MIPI, parallel DVP interface camera and USB interface camera. However, there are shortcomings such as limited transmission distance and bandwidth, and signal interference. At present, the network IP camera which combines the traditional camera technology and network technology had been widely used by digitizing the original analog video signal, using H.264 compression coding and finally transmitting the code stream to the network according to the network protocol. Network cameras had the advantages of high stability, high image quality, long-distance transmission, large transmission bandwidth and networked. However, due to the complexity of the video coding and decoding mechanism, it had delay in transmission and poor backend portability [3]. With the development of electronic technology, there were more and more image transmission processing platforms based on embedded systems. It had the advantages of low cost, small size and low power consumption [4]. Xilinx's ZYNQ-7000 All-programmable SOC (System on Chip) processor integrated ARM Cortex-A9 Dual-core processor (PS, Processing System) and high-performance FPGA (PL, Programmable Logic) [5]. Information was passed through the AXI bus between PL and PS. The design used the custom IP core to convert the video stream information stored in DDR3 to TMDS signals through VDMA then driving the HDMI display in PL. It reduced the PS data transmission tasks and improved the efficiency. Meanwhile, the design used the embedded development tool (Peta Linux) to customize, build, deploy Linux system in PS. The image information was acquired by got H264 stream from network camera and decoded it by using FFMPEG. The design had the advantages of compatibility and portability.

2. System Design

Image transmission and display system diagram was shown in Figure 1. The system mainly consisted of ZYNQ-7000 processor, IP network camera, HDMI display, DDR3, SD Card and other components.PS run custom operating system and obtained code stream from IP network camera by

using TCP/IP protocol. Then upper application decoded it and stored in DDR3. PL used VDMA to send video stream information into IP core for format conversion to drive the display

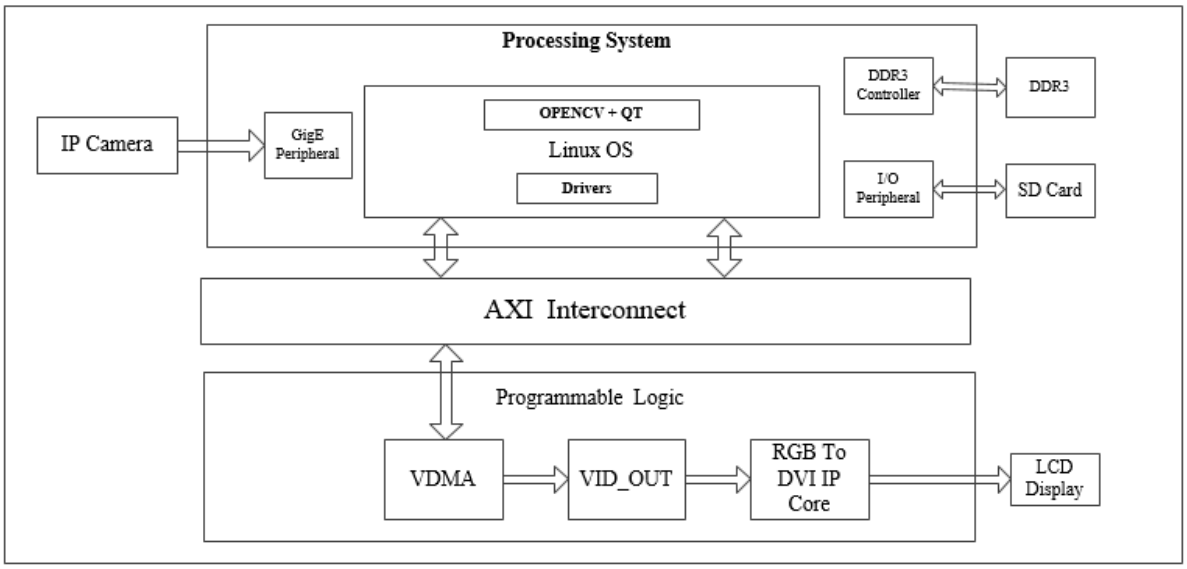

Fig. 1 System diagram

3. Hardware System Design

3.1Custom ZYNQ7 Processor

According to the design requirements, the PS peripherals were respectively enabled and configured, such as AXI_HP, ENET, SD, UART, etc. The ENET interface was used to control the network interface to read the data from the network camera. The AXI_HP interface was provided High-speed data transmission channel for PS and PL [6], so that the communications between PS and PL were high throughput while had low latency between PL and DDR.

3.2 Data Transfer Between PS and PL

AXI bus interconnect IP core and VDMA IP core deployment and configuration, in which AXI bus interconnect IP core provided arbitration for data exchange across multiple devices, and each channel had a burst transmission mechanism. VDMA IP core was used to convert AXI Stream format data to AXI4 Memory Map format for data communication with DDR3 through the AXI_HP interface. AXI4 Stream interface was divided into independent AXI4 MM2S read channel and AXI4 S2MM write channel. Since this design read from the DDR3 data into PL, so only enable its read channel. Because there was no simultaneous read and write process, in order to speed up data transmission, Frame Buffers was set to single-frame cache. Figure 2 showed the timing diagram of the MM2S read channel. After mm2s_fsync received the signal, the m_axi_mm2s_arvalid signal was sent five times to accept five lines of vertical size. At the same time, the data being read was stored in the buffer and then sent to the video stream interface.

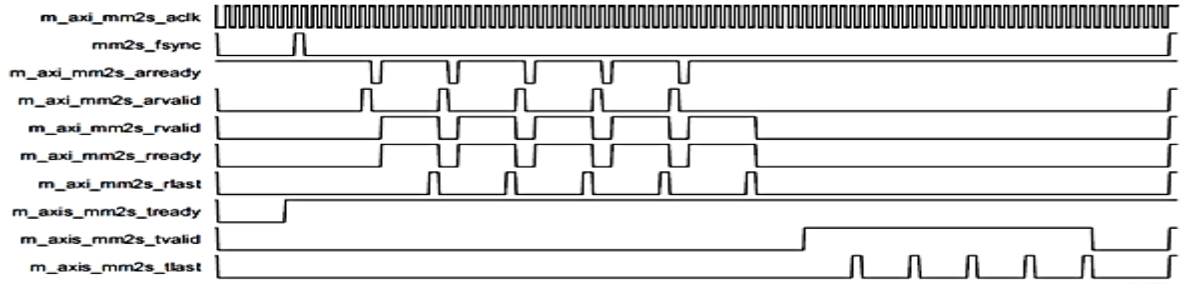

Fig. 2 MM2S read channel timing diagram

3.3 Data Format Conversion in PL

AXI4-Stream to Video Out (VID_OUT) IP Core and RGB to DVI (RTD) IP Core Deployment and Configuration. The VID_OUT IP core converted the AXI4 Stream signal into an RGB format video signal. The RTD IP core encoded the RGB video data into a Minimized Transmission Differential

Signal (TMDS) to drive the display. In the DVI interface, the clock channel had a character-rate frequency reference, which required a five-fold fast serial clock using a pixel clock frequency and a PLL unit was designed to generate the serial clock.

4. Software System Design

System software was mainly composed of embedded Linux system, driver module and upper application. The design used PetaLinux development tools to customize, build and deploy embedded Linux system in ZYNQ. BOOTROM initialized SD and basic peripheral controller and loaded FSBL (First Stage Boot Loader). After configuring PS and PL, System loaded U-BOOT to finish the rest of initialization then loaded kernel, device tree and root file system into memory [7]. For the boot process, the design used Petalinux to build the system. Flow chart was shown in Figure 3. First, the design set the boot image storage and root file system type to SD card in the top configuration and added the RTD IP video encoding driver to the Linux kernel source before configuring the kernel, then modify the corresponding device tree, C/C++ standard library was compiled into the root file system. Finally, U-BOOT, kernel, device tree and root file system were compiled, the generated FSBL, U-BOOT and hardware design of the bit stream file were combined into BOOT.BIN in which used to boot system in the FAT partition.

Aiming at the problem of poor portability and lacking compatibility of back-end software in different IP network cameras, this paper designed system with Open Source Decoding Library (FFMPEG), OPENCV and QT. The design compiled source code of the OPENCV and QT with Xilinx cross-compiling tool on the host computer, and then transplanted the compiled library to EXT4 format root file system in SD Card [7]. Application design usually adopted the mode of "host/target".

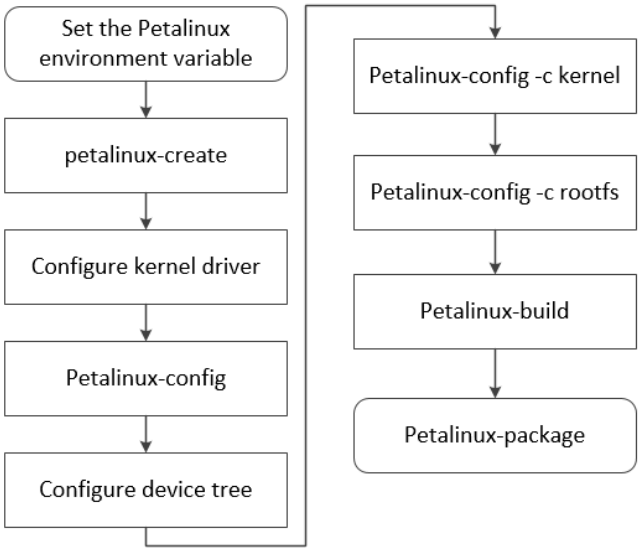

Fig. 3 Petalinux build flow chart

4.1Image Acquisition

The principle of network IP camera was that the image information was collected by CMOS/CCD sensor and used the H.264 algorithm to compress, then transported through the network-based network protocols such as IPv4 / IPv6, HTTP, FTP, RTSP, TCP, UDP, etc. The network IP camera used in this design supported the RTSP protocol. RTSP protocol was the application layer protocol in TCP/IP. It used RTP protocol to complete data stream transmission. RTP was based on UDP protocol. UDP belonged to connectionless protocol, which has high transmission speed and high efficiency. The image acquisition program read the code stream of the network IP camera through the RSTP protocol, decoded the compressed code stream through the FFMPEG, and then accessed the OPENCV to save. Acquisition program flow chart was shown in Figure 4.

4.2Image Display

With good portability and rich control resources, QT/Embedded can run on Linux with a variety of architecture processor deployments. Its libraries were C ++ based and had a lower memory footprint than X11 Framebuffer as the underlying graphics interface, ideal for embedded system GUI development. This design mainly displayed the image on the QWidget object and draws the video image on the QImage through the QPainter. Image display program design flow chart was shown in Figure 5. The PaintEvent overloaded function updated the video frame and periodically triggered the video frame function through the QTimer so as to realize the real-time display of the video image.

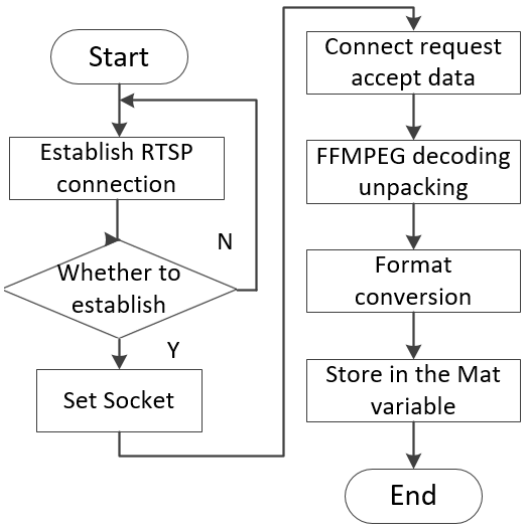

Fig. 4 Acquisition program flow chart

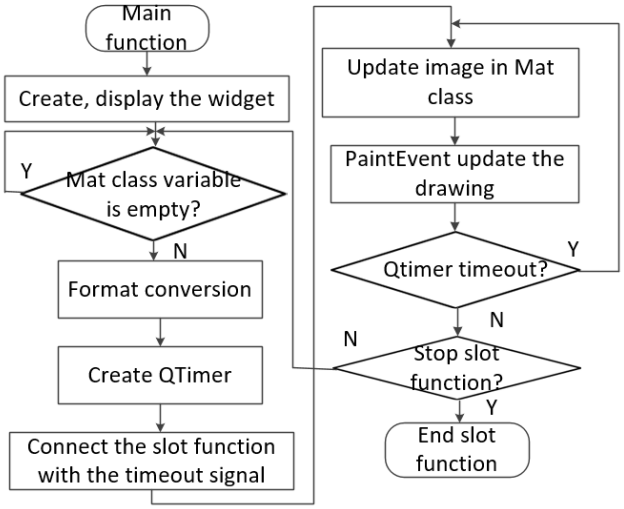

Fig. 5 Image display design flow chart

5. Experiments

The experiment system used the ZYNQ development board (the chip was XC7Z020, hardware interfaces included HDMI port, Ethernet port, JTAG and so on), Hikvision IP camera, display. Boot mode was set to SD Card that was divided into FAT32 and EXT4 partitions. System boot files were stored in the first partition and the root file system was mounted on the second partition. Network IP camera was set to sub stream, the output image format was VGA (640*480). After the system started up, the development board connected to the host computer through the serial port and used the PUTTY serial port tool to monitor the system start-up information. In order to be able to get the camera code stream, the experiment need to make the IP address of host PC and camera IP address in the same network segment before system startup. After the application was running, the monitor displayed the video information in low latency. The experimental results were shown in Figure 6.

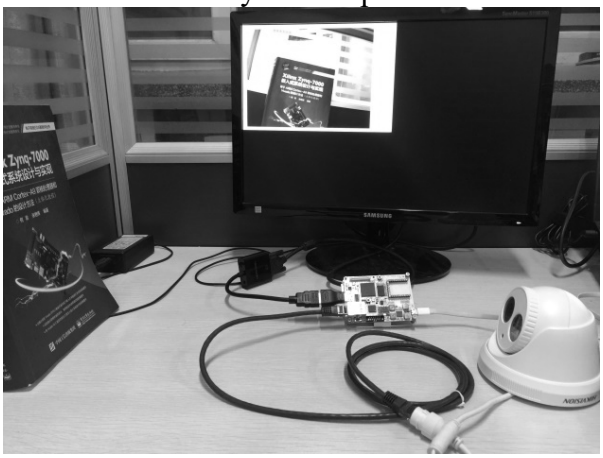

Fig. 6 Experimental results

In order to analyze the transmission delay of the designed image transmission and display system, experiments were conducted in the following four environments:

1.FFMPEG library on PC 2.VLC software on PC 3. Hikvision official SDK component library 4. The experimental system in this paper

First, using the network camera to monitor the timer, then using another camera to capture the timer and the timer on the display at the same time, and finally comparing the collected data between the data on the timer and the data on the display. The delay in four experimental environments was compared by the time difference between the above data. Respectively, taking 3,6,15,27,32,56 frame data for comparison. Experiments 1 and 3 were implemented on the PC with the QtCreator development platform by using C ++ and OPENCV. Experiment 2 was implemented in the video media player software on PC. Experiment 4 was implemented in the designed system in this paper. The computer's operating system was ubuntu16.04, the CPU was Inter Core i5, memory size was 8.00GB. Experimental results were shown in Figure 7.

The experimental results showed that the actual delay of image transmission and display system designed in this paper was about 0.28s, while the open source decoding library on PC was about 0.33s. As the official did not provide support for ARM's SDK component library and source code, the delay was about 0.22s by using the official SDK component library on the PC. By comparing the experimental results, it can be seen that the delay of the system designed in this paper is somewhat lower than results of Experiments 1 and 2. Compared with the delay of the official SDK

Component library, it was also close and had a low network latency. In order to further analyze the performance of this design system, the average time taken to acquire and display one frame of image in the actual experimental system was calculated. The experiment selected the first 50 frames for calculation. The experimental results showed that getting and displaying an image with a resolution of 640*480 was about 23ms and the frame rate was about 43 frames per second(fps), meeting the requirements of real-time display.

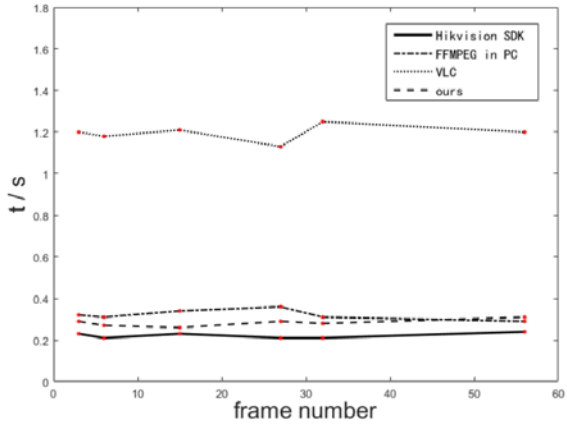

Fig. 7 Compared results

6. Conclusion

The design built the Linux system in PS and transplanted open source decoding library to get network IP camera video stream. This design improved system compatibility. At the same time the task load of the PS was reduced by using the hardware IP core in the PL to transmit video and drive display in order to reduce the image display delay. The experimental results showed that the video stream captured by this system not only had low latency, real-time performance and high image quality, which meet the needs of subsequent image processing and pattern recognition, but also had the advantages of low cost, low power consumption and small size. At the same time, it provided solutions for different types of network IP camera. The design provided engineering applications and reference value for industrial control, video surveillance, UAV, smart home and other fields.

References

[1]. Hou Guangqi, Wei Ping, Bai Tingzhu. Study of multi-channel high-speed embedded image capture and storage system [J]. Chinese Journal of Scientific Instrument. Vol. 32 (2011) No. 11, p. 2543-2546.

[2]. ZHU Yi-dan, FANG Yi-bing. Image acquisition and VGA display system based on FPGA [J]. Journal of Computer Applications. Vol. 5 (2011) No. 9, p. 1258-1261.

[3]. Li Song: Implement of Video Network RTP / RTCP [D], Heilongjiang University, China 2015. p. 54-76.

[4]. LI Hui-min, FAN Ji-ming, YANG Xiao. Design of image acquisition and display system based on STM32 and OV7670 [J]. Transducer and Microsystem Technologies. Vol. 35 (2016) No. 9, p. 114-117.

[5]. Zhang Yanhui, Guo Mingyu, He Bin. Construction and implementation of Vivado HLS embedded real-time image [J]. Application of Electronic Technique. Vol. 42 (2016) No. 9, p. 115-117.

[6]. He Bin, Zhang Yanhui. Xilinx Zynq-7000 Design and Implementation of Embedded System [M], Electronics Industry, 2016, p. 231-236.

[7]. Yang Shanbin: Detection and tracking of moving target based on SOC system [D], Harbin Institute of Technology, China, 2015, p. 32-65.