Improvement of Display Unit in Generic Verification Environment for Video Image Processing Hardware on FPGA

Gen Nishizawa*, Kazunari Yoshikawa, Akira Yamawaki

Kyushu Institute of Technology, Kitakyushu, 804-8550 Japan

*Corresponding Author: yama@ecs.kyutech.ac.jp

Abstract

Video image processing is an essential feature in embedded devices. In order to facilitate the development of video image processing hardware, we are developing the generic verification environment. This is able to install in any of the FPGA and has the standard circuits which interface to the cameras, memories, displays and a host PC around the video image processing hardware. The current verification environment has the single frame buffer for the display. If the video image processing hardware in the verification environment makes a large processing load, flickering on the display is preventing visual confirmation. In this paper, we improve the display unit by introducing a double buffer in the verification environment. In the experiment, the effect of the double buffer is evaluated by compared to the single buffer. The result shows that the double buffer is able to eliminate the flicker of the video image. Resource utilization of the verification environment didn't change significantly. That is, the circuit scale of the verification environment is maintained even though the several circuits arbitrating the double buffer between the video processing hardware and the display controller are introduced.

Keywords: video image processing, double buffer, FPGA.

1. Introduction

The system-on-chips (SoCs) in embedded products have been used in various fields. The SoCs require high performance in order to implement the complicated processing. Video image processing is an essential feature in the embedded products. For example, we are able to speed control and inform situation and an object in the car. The hardware modules are required to perform these

difficult processes with high-speed and power saving more. This is because the video image processing in software shows the lower performance and higher power than hardware implementation.

In developing hardware, the operation verification performs a logic simulation on the programs written in a hardware description language (HDL). However, the logic simulation for the video image processing needs a huge test pattern. It is insufficient to verify because verification for simulation time is long. Furthermore, visual confirmation accompanied by actual operations, for example object tracking, is difficult. Therefore, the actual verification using an FPGA is conventionally performed (1-4). However, in addition to the development of the video image processing hardware, many peripherals like a camera, an image memory, an interface to the host PC, and so on have also to be developed. These jobs may be a large burden for developers.

Therefore, we are developing a generic verification environment on FPGA for the developers of the video processing hardware (5). This generic verification environment can be installed in any FPGA. The FPGA independent standardized peripheral circuits that can be connected to the video image processing hardware are provided. From this fact, the design and development of hardware is simplified. In other words, developer can build a verification environment on the developer's own FPGA board. Developer can concentrate only on the design and development of hardware.

However, the current verification environment has the single frame buffer for the display. If the video image processing hardware in the verification environment makes a large processing load, flickering on the display is preventing visual confirmation. In this paper, we introduce the double buffer synchronizing between a camera and a

display whose frame rates are different in order to reduce flicker on the display.

2. Double Buffering

In the verification environment, the video image processing hardware stores the processed image into the VRAM while the contents of VRAM is displayed on the display. When the hardware consumes the large time to process the image, the updating content of VRAM may be displayed. Therefore, flickering on the display occurs. This is caused by overlapping. Overlapping is a phenomenon in which a new display image and the previous display are mixed.

An overview of overlapping is shown in Figure 1. If overlapping happens, we cannot watch the latest processed video image correctly. That is, visual verification of the video image becomes difficult. At present, verification environment is a single buffer. The introduction of the double buffer is to improve the flicker of the display. We prepare two VRAMs. When one VRAM is written by the hardware, the content of this VRAM isn't displayed. The VRAM that has the processed image is displayed. We switch between these VRAMs alternately. As a result, the previous image and the new image are not mixed on the display. Therefore, it can improve the flicker.

An overview of the execution snapshot of double buffer is shown in Figure 2. Arrows mean the data flow. First, as shown in Figure 2. (1), the user hardware writes the processed image data from the camera into VRAM0. Then, as shown in Figure 2. (2), the user hardware stores the next processed data into VRAM1 while the content of VRAM1 that has the last processed image is displayed. Last, as shown in Figure 2. (3), By turning the VRAM0 and VRAM1 among the user hardware and the display, the previous image and the latest image are being displayed without overlapping.

Fig. 1. Overview of Overlapping.

Fig. 2. Overview of the Flow at the Time of Double Buffer.

3. Verification Environment

3.1 Verification Environment

An Overview of Verification Environment is shown in Figure 3. Interface of the camera module (camera settings for the IF and camera data IF), interface with the host PC (serial communication IF), a user of the video image processing hardware (video image processing HW), the video image processing hardware control and state of register group (MB), the memory of the video image display portion to display (VRAM), interface with the display (VGAC_IF). These constitute a verification environment. Setting of the camera module is carried out by IF for the camera settings. The captured image is acquired by the camera data IF. Communication with the host PC is performed by serial communication. Video image processing hardware is the user design hardware. In order to correspond to the verification environment, the user rewrites the hardware. MB is used for communication with the host PC. VRAM can be set to any size, depending on the standard of the FPGA to be used. VRAM data can be displayed on a display connected to the FPGA. If you use

Fig. 3. Overview of Verification Environment.

Fig. 4. Overview of Verification flow for Video Image Processing Hardware.

an external memory, VRAM is connected to an external memory interface (SRAM_IF and DDRA_IF).

An Overview of Verification flow for Video Image Processing Hardware is shown in Figure 4. We verify the video image processing hardware in the order from (1) to (4). In the flow of (1), the camera module is set up by the host PC (PIN arrangement, etc.). In the flow of (2), the captured image which is obtained by the camera module is sent to the video processing hardware. In the flow of (3), we write the commands and instructions to the specified register in the MB. Therefore, we can control the hardware. In the flow of (4), the image which is the processing finish is stored in the VRAM. VRAM is read by VGAC_IF, the image is displayed on the display. You can select the flow of (5) and (6) as necessary. In the flow of (5), this is a case where it is an object of power consumption control, the high-speed operation. Via an external memory interface, it is possible to use a SRAM and SDRAM. In the flow of (6), this is a case where it is intended to read the image data of one frame from the VRAM. Via a serial communication, you can read the image data.

3.2 Design of Double Buffer

An Overview of the Double Buffer that we have designed is shown in Figure 5. The VRAM_ARB is an arbiter between the user hardware and the display, which have different resolutions and frame rates. The UHW_WE is the signal indicating that the user hardware outputs the processed pixel to the VRAM. The UHW_REQ is the signal meaning that the user hardware requests VRAM at first frame to the arbiter. The UHW_REL is the signal indicating that the user hardware releases the VRAM used. The DISP_REQ is the signal indicating that the display requests to read the VRAM to be displayed on. The DISP_REL means that the display finishes reading the

Fig. 5. Overview of Double Buffer.

whole frame in the VRAM and releases the VRAM.

In the generic verification environment, any camera and any display may be connected to the FPGA board. Each camera and each display have different frame rate. If the camera and display used are connected to the FPGA board simply, video image may not be displayed correctly due to different frame rates. In order to adjust frame rate of camera and frame rate of display, we have designed an arbiter synchronizing the different frame rates. The arbiter manages the state of each VRAM. By the VRAM state, the arbiter selects the dedicated VRAM and grants the usage of the VRAM selected to the user hardware or the display.

Each VRAM has four states; Empty, Latest, Displayed, and Busy. Empty means that the VRAM has no data. Latest means that the VRAM has the image processed by the hardware. Displayed indicates that the display have finished reading the VRAM. Busy means that the VRAM being used by the hardware or the display.

3.3 Controller on Arbiter

The Arbiter State Transition Diagram is shown in Figure 6. UHW_REQ and DISP_REQ see 3.2. UHW_FIN is the signal indicating that the user hardware inform the

Fig. 6. The Arbiter State Transition Diagram

arbiter of the VRAM finished using. UHW_FIN is the signal meaning that the display inform the arbiter of the VRAM finished using. IDLE is the state indicating that the arbiter waits the user hardware request or the display request. When the arbiter is requested the user hardware, the arbiter changes UHW_RESP which responds to the request. Otherwise, when the arbiter is requested the display, the arbiter changes DISP_RESP which responds to the request. When either the user hardware or the display informs the arbiter of those used the VRAM, the VRAM state renews. Therefore, the arbiter state changes IDLE and the arbiter arbitrates again.

UHW_RESP is the state indicating that the arbiter responds arbitration results to the user hardware. As shown on Table 1, by the VRAM state, the arbiter replies usable VRAM number to the user hardware and lets the Busy = 1 at the same time. Table 1 has high priority towards the top. The “-” mark in the Table 1 is any VRAM state. When the display informs the arbiter of the display used the VRAM, the VRAM state renews. Therefore, the arbiter state changes IDLE and the arbiter arbitrates again. If display of the VRAM is delay, the image processed don’t display and the VRAM may have new processed image.

DISP_RESP is the state indicating that the arbiter responds arbitration result to the display. As shown on Table 2, by the VRAM state, the arbiter replies usable VRAM number to the user hardware and lets the Busy = 1

Table 1. UHW_RESP Response

VRAM Number	VRAM0	VRAM1
0	Empty	-
1	Not Empty	Empty
0	Displayed	-
1	-	Displayed
0	-	Busy
1	Busy	-

Table 2. UHW_RESP Response

VRAM Number	VRAM0	VRAM1
0	Latest	-
1	-	Latest
0	Displayed	Busy
1	Busy	-
-1	Others	

at the same time. Table 2 also has high priority towards the top. The “-” mark in the Table 2 is as same the Table 1. VRAM number -1 means that displayable VRAM don’t exist. At that time, skipping the display of 1 frame. When the user hardware informs the arbiter of the user hardware used the VRAM, the VRAM state renews. Therefore, the arbiter state changes IDLE and the arbiter arbitrates again. When process of the VRAM is delay, the Displayed VRAM display again.

4. Experiment

We have implemented a verification environment to Zynq-7000 that is installed in the Xilinx Corp. ZYBO board. We used Vivado of Xilinx, Inc. in logic synthesis of HDL description. The size of the display is 256×128 . We moved a pen in front of the camera mounted on the verification environment. The camera module is the OV9655. The OV9655 frame rate is 7.5 fps. The display is the E152FPC. The E152FPC frame rate is 60Hz. Serial communication module is the FT232RL.

Table 3 shows the utilization of a single buffer and double buffer resources. With the exception of BRAM, there is no significant change in utilization.

Table 3. Utilization of a Single Buffer and Double Buffer Resources

	Single Buffer	Double Buffer
FF	1.57%	1.72%
LUT	2.74%	3.16%
I/O	35.00%	35.00%
BRAM	55.00%	81.67%
BUFG	6.25%	6.25%
PLL	50.00%	50.00%

Fig. 7. Video Image of Each Single Buffer and Double Buffer.

A video image of each single buffer and double buffer is shown in Figure 7. Flicker occurs in a single buffer. Flicker has not occurred in the double buffer.

5. Conclusion

We have improved a display portion of the video image processing hardware general-purpose verification environment. In a conventional verification environment, if the heavy processing load is performed, flicker occurs in display was difficult to visually confirm. Improve the flicker of the display by introducing the double buffer has become possible to visually check. In the future, we introduce a triple buffer, a comparison of the double buffer during Installation of the display and the triple buffer during Installation of the display.

References

- (1) Toshinori Sueyoshi, Masahide Ouchi, Koichiro Tanaka, An Environment for Rapid System Prototyping using Multiple FPGAs Information Processing Society of Japan 95(307), 79-86(1995)
- (2) Keita Inoue, Xing Weijie, Shinji Kimura, Acceleration of Prototyping Design Verification Using Circuit Modification, IPSJ SIG Technical Reports 2007(27), 113-118(2007)
- (3) Masami Nakajima, Hiroyuki Kondo, Hirokazu Takata, etc., Development of the high-performance microcomputer "M32R" by prototyping method with FPGA 102(165), 61-66, (2002)
- (4) Gabe Moretti(USA) GABEonEDA, Inc., <http://ednjournal.com/content/issue/2006/06/content06.html> (2006)
- (5) Takayoshi Abe, Seiichi Serikawa, Akira Yamawaki, Suggestion of User Hardware Switching mechanism in Generic Verification Environment for Video Image Processing Hardware on FPGA (2013)