Multi-Port Register File Design and Implementation for the SIMD Programmable Shader

Kyeong-Seob Kim, Yun-Sub Lee and Sang-Bang Choi

Abstract Characteristically, 3D graphic algorithms have to perform complex calculations on massive amount of stream data. The vertex and pixel shaders have enabled efficient execution of graphic algorithms by hardware, and these graphic processors may seem to have achieved the aim of "hardwarization of software shaders." However, the hardware shaders have hitherto been evolving within the limits of Z-buffer based algorithms. We predict that the ultimate model for future graphic processors will be an algorithm-independent integrated shader which combines the functions of both vertex and pixel shaders. We design the register file model that supports 3-dimensional computer graphic on the programmable unified shader processor. We have verified the accurate calculated value using FPGA Virtex-4(xcvlx200) made by Xilinx for operating binary files made by the implementation progress based on synthesis results.

Keywords Graphic processor · Register file · Shader processor · HDL · FPGA

K.-S. Kim (✉) · Y.-S. Lee · S.-B. Choi
Department of Electronic Engineering, Inha University, 253 Yonghyeon-dong, Nam-gu, Inchon, South Korea
e-mail: st22091108@inha.edu

Y.-S. Lee
e-mail: Skua1204@inha.edu

S.-B. Choi
e-mail: sangbang@inha.ac.kr

K. J. Kim and K.-Y. Chung (eds.), *IT Convergence and Security 2012*,
Lecture Notes in Electrical Engineering 215, DOI: 10.1007/978-94-007-5860-5_32,

1 Introduction

High-performance graphics processor technologies that can efficiently handle massive amounts of data and intricate operation is required in order to support a realistic three-dimensional computer graphics. Existing graphics processors perform geometry and rendering operations with limited functionality. Also most of the rendering to increase a realism have been using a relatively simple algorithm receiving the support of the software. However, programmable shader receiving the support of the hardware is recently being studied so that it can be processed by the graphic processor.

In this paper, we design and implement the register file model that supports three-dimensional computer graphic on the programmable unified shader processor. we have verified the accurate calculated value using FPGA Virtex-4(xcvlx200) made by Xilinx for operating binary files made by the implementation progress based on synthesis results.

2 The Design of SIMD Programmable Shader Processor

2.1 The Architecture of SIMD Programmable Shader Processor

SIMD programmable shader processor presented in this paper was basically designed with the goal to the implementation on the Virtex-4 which is one of the FPGA products of Xilinx. It is a mix of superscalar processor and vector processor architecture with the advantages of parallel processing and pipeline in order to improve performance.

2.2 The Scalable Register File Model

we propose a register file model for the SIMD programmable unified shader processor using the intellectual property (IP) provided by Xilinx FPGA. Input and output signal current of the dual-port block RAM provided by the Xilinx FPGA is present each one. Unfortunately, Input and output operation at the same time cannot be performed on a port of the dual-port block RAM. Therefore, it is difficult to use for the structure of the programmable unified shader processor required reading and writing at the same time on the same register. Thus, In order to support the one read (1R) and two write (2 W) operation simultaneously in a single cycle, we propose an improved 2W1R block RAM structure using the dual-port block RAM provided by the Xilinx FPGA as Fig. 1. The 2W1R operation is as follows.

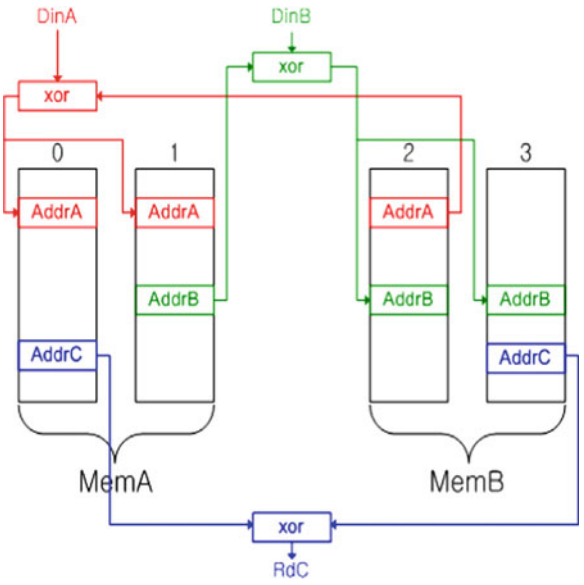

Fig. 1 The register file with 1 read-port and 2 write-port

```
if(WriteEnableMemA)
        MemA[AddrA] : = MemB[AddrA] ⊕ DinA;
endif;
if(WriteEnableMemB)
        MemB[AddrB] : = MemA[AddrB] ⊕ DinB;
endif;
if(ReadEnableMemAddC)
        Rdc = MemA[AddrC] ⊕ MemB[AddrC];
            = DinA;
endif;
```

Four block RAM is used on 2W1R register file structure. Equation 1 means that the number of block RAM required in case of the write port and read port expansion.

$$\begin{aligned} \text{BlockRamNum} = {} & 2 * (\text{WritePortNum}) + (\text{ReadPortNum} - 1) \\ & * \text{WritePortNum} \end{aligned} \tag{1}$$

2.3 The Design of the Register File

The simulation using a benchmark program for a specific procedure frequently used in the algorithms for the graphics was performed in order to determine the structure of a register file for the SIMD programmable unified shader processor. As a result, we have verified that the Dot Product (DP), the Multiply and Add (MAD) operation is repeated in the main procedure. In addition, The Instruction Per Cycle (IPC) is increased due to the increase of the unit which is for the DP and the MAD operation. However, The IPC have not risen compared to hardware used because repeated DP and MAD operation is frequently used in the algorithms increasing more than a certain number of the Arithmetic Logic Unit (ALU). Thus, the Four ALUs was designed for programmable unified shader processor based on result of the simulation using a benchmark program.

2.4 The Time Division Mechanism of the Register File

Due to four ALUs used the pipeline architecture, the ports of register file in the SIMD programmable unified shader processor which consists of the structure to handle a large number of operands with four pixel data is accessed simultaneously. Accordingly, the time division mechanism for the proposed register file is used on the proposed register file as the Fig. 2. The clock signals for the register file and the main processor should be in order to use time division mechanism. Then, the two signals for the register file operation must be synchronized with each other. For this purpose, the buffer circuit for the register file was designed so that it is accessed from four ALUs and generates the two clock signals for the synchronization.

3 The Logic Synthesis and Verification

3.1 The Simulation for the Behavioral Level

By default, the signal validation for all registers is performed to verify the designed register file. Also, normal read and write operations are validated when simultaneously access to the register file through the various ports. The register file in the designed processor is the structure sharing the four ALUs and the ports. To this behavior, time division mechanism and internal signal selector is important. Therefore, the verification for the operation of the register file is performed with the ALUs.

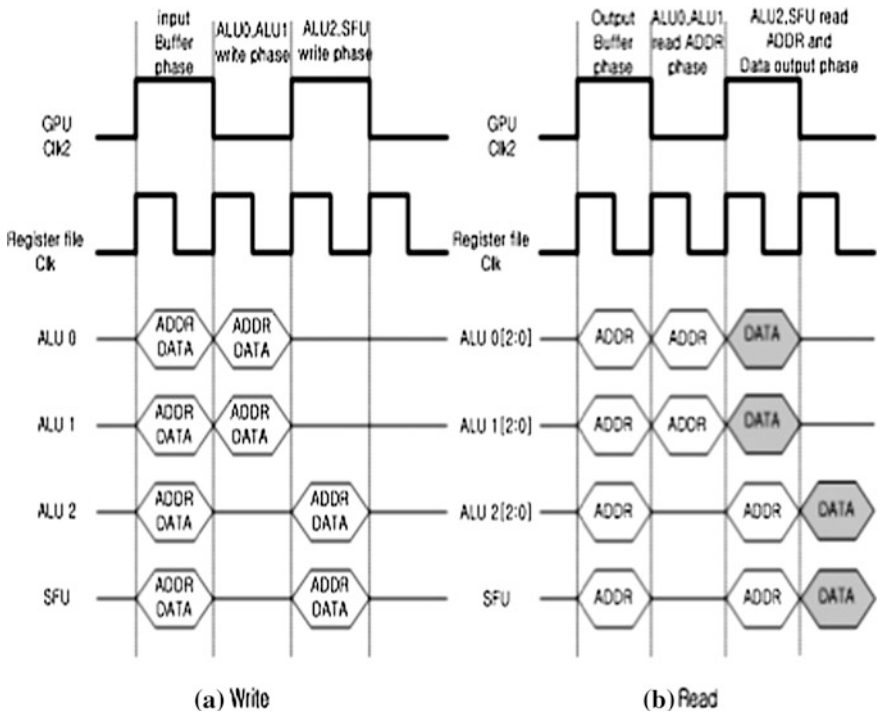

Fig. 2 Timing diagram of reading and writing the register file

Table 1 Synthesis result

Detailed module	Slices	Speed(MHz)
Top of register file	6465	206.695
Frequency divider	2	685.895
Input buffer	555	685.895
Output buffer	1927	685.895
Register file group	5461	288.663
Register file	4274	301.568
Temporary register file	1187	288.663

3.2 HDL Synthesis

The results synthesized using Xilinx ISE 9.2 shown in Table 1. Then, the binary files generated through a process of the implementation were fusing on the logic tile for xc4vlx200 based on the results of the synthesized. In addition, the binary files were targeted to the hardware by connecting the logic tile for xc4vlx200 to the versatile platform baseboard that operates as a host. As a result, we verified that the results of the simulation results with the same output.

4 Conclusions

In this paper, we proposed the multi-port register file that supports three-dimensional computer graphic on the programmable unified shader processor. Designed register file uses improved block RAM configured using dual-port block RAM quickly to process a large number of data by using a relatively small instruction in the Xilinx FPGA performing three read and one write in a single cycle. The further researches will focus on the algorithms related to the efficient management of the register file to minimize the entire system resource usage and the reduction of operating speed even increasing the size of the register file and the ports.

References

1. Hennessy JL, Patterson DA (2006) Computer architecture, a quantitative approach, 4th edn. Elsevier, San Francisco
2. Moya V, Gonzalez C, Roca J, Fernandez A, Espasa R (2005) Shader performance analysis on a modern GPU architecture. In: Proceedings of the 38th annual IEEE/ACM international symposium on microarchitecuture(MICRO‘05), pp 355–364, Nov 2005
3. Atabek B, Kimar A (2008) Implementability of shading models for current game engines, ICCES, pp 427–432
4. Chung K, Yu C, Kim D, Kim L (2008) Tessellation-enabled shader for a bandwidth-limited 3D graphics engine, IEEE CICC, pp 367–370