Development and Evaluation of Raytracing Accelerating Engine with Bit Serial Arithmetic Units

Tomoyuki Kawamoto, Kazuya Tanigawa, Tetsuo Hironaka and Yuhki Yamabe
Graduate School of Information Sciences, Hiroshima City University
3-4-1, Ozuka-higashi, Asaminami-ku, Hiroshima, 731-3194, Japan
E-mail : kawamoto@csys.ce.hiroshima-cu.ac.jp

Abstract: In recent years, development of various rendering algorithm provide photorealistic image. However, requirement of quality for rendering causes significant increase of processing time. Raytracing method can generate high quality images. But, it needs much time to get images. Several methods of parallel computing for high speed processing for raytracing with hardware are discussed. But, the chip of conventional raytracing hardware needs huge area because of parallel arithmetic units on it , which result in difficulty on inserting more processing elements(PEs) into the chip. In this paper, we present hardware design of a raytracing accelerator engine with bit serial arithmetic unit to improve performance. Area of bit serial arithmetic unit is small, and operation performance per the unit area is high. We adopt a dot mode for parallel processing because it can make maximum use of the characteristic of bit serial(BS) arithmetic unit. As the first stage, we design prototype hardware which calculate only the process of highest load ratio in PE, and evaluate it. As a result, prototype hardware is about 5.09 times faster on processing time.

1. Introduction

Raytracing generates high quality images by consideration of reflection and refraction of ray. But, it takes extremely long calculation time to generate the high quality images because it requires a huge amount of calculations caused by the large number of the ray to process. So, to accelerate the calculation, many researches are proposed. As one of these researches, parallel processing method for accelerating the calculation with hardware circuits has been proposed[1]. But, it has problem that hardware area to implement is large, so it is difficult to implement many PEs on a chip to improve the performance[2]. To solve the problem, we propose the raytracing accelerator engine that adopts BS arithmetic units [3]. The advantages of the BS arithmetic unit is to achieve high performance in a small chip area. In this paper, we describe about how to adopt the BS arithmetic unit in the raytracing engine and present its efficiency by evaluation.

2. Analysis of Raytracing

2.1 Advantage of bit serial arithmetic unit

Many hardware resources is required when parallel degree is increased. Parallel arithmetic unit requires a large area, so it is not good for implementing highly parallel raytracing engine. On the other hand, BS arithmetic unit has three points which is good for high parallel computing.

- The size of data is handled in one clock is small.
- The chip area to implement it is small.
- The processing speed per area is fast.

The number of wires can be reduced when applications are designed with the BS arithmetic unit, because it handles small bit width data in one clock. And, BS arithmetic unit are several times faster in processing speed per area when compared to the parallel processing units [3].

2.2 Parallel processing method

The raytracing is possible to make in use of various levels of parallelism, fine-grain to coarse grain, such as pixel, object, ray and space level. In this paper, the proposal engine adopts the most fine-grained parallel processing level and pixel parallel processing level because these level can achieve the highest performance of all levels under the consideration that there are no limits on the number of parallel processing units[4]. Pixel parallel processing uses the features that raytracing can calculate a pixel color without any dependency with other pixel color. And, it has the following three modes.

- Block mode
  This way, at first it divides a picture into rectangular areas, and each of the rectangular areas are allocated to each PEs. It is easy division method. It is an easy division method. But it is not a good method for load balancing, because some of the divided area may not have any ray hit with the object when the objects location is unbalanced.
- Scanline mode
  This way divides a picture into scan lines. It improves the load balancing than block mode which is an easy division method.
- Dot mode
  PEs calculate on different point at equally spaced, and make a picture. Dot mode has the best load balancing in these three division methods. But, it is difficult to use similarity of ray and brightness value in adjacent pixel.

The architecture with BS arithmetic unit is expected to be implemented in small chip area. It can layout more PEs on same chip size than designing architecture with parallel processing unit. Therefore, if the chip area size is equal, the architecture with the BS arithmetic units is possible to increase the number of the arithmetic units, compared with the architecture with parallel processing units, which mean we can decrease the workload of each PEs. And the processing time may increase when the placement of objects are unbalanced. This is because the process speed of one bit serial arithmetic unit is extremely slow compared to the parallel arithmetic unit. To avoid this problem, we adopt Dot mode for the proposed en-

gine. Dot mode gives us good load balancing between each PEs when the number of PEs is large.

2.3 Accelerating by application specific circuits

Designing the whole raytracing accelerator engine as a single application specific integrated circuit (ASIC) is difficult, because of the required chip size for implementation. So we analyzed the raytracing process and found out the hot spot of the process. The conditions of the analysis are the following; the number of rendered object where set to 2 through 1000, and the number of reflection and refraction is 1, 10 and 100. These results are presented in Figure1, Figure2, Figure3. Each notation on the graph stands for the processes name used on simulation, and their contents are the following.

- RayCross:Judging intersection of ray vector from view point and objects.
- LightCross:Judging intersection of ray vector from light source and objects.
- UnitVector:Calculating unit vector of input vector.
- LightColor:Calculating color form ray refraction.

As a result, the hot spot of the raytracing was RayCross and LightCross process. Which reached 99% of the total execution time. So the intersection calculation which are included in both modules, RayCross and LighCross process were selected as the hardware accelerating module.

2.4 Hardware structure of the intersection calculation module

The intersection calculation module consists of following processing.

(1) Normal vector calculation : This module calculates normal vector of the polygon by the following equations.

$$Ns = (P_1 - P_0) \cdot (P_2 - P_0) \tag{1}$$

$$N = \frac{N}{|Ns|} \tag{2}$$

where P0; P1; P2 are the coordinates of the polygon, and N is the normal vector.

(2) Distance Calculation : This module calculates distance from view point to P which is a point at the intersection of ray vector with polygon.

$$to = \frac{P_0 - V_0 \cdot N}{V \cdot N} \tag{3}$$

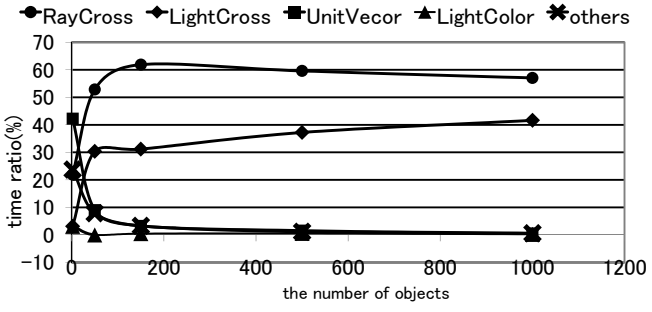

Figure 1. the number of reflection and refraction is 1.

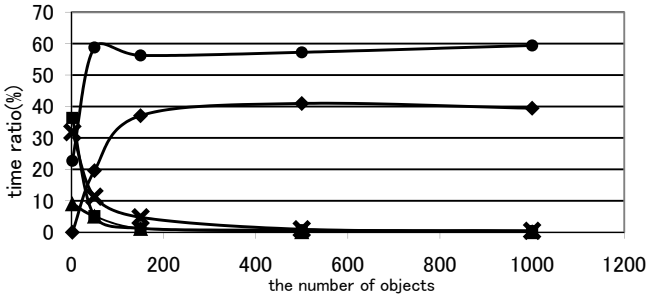

Figure 2. the number of reflection and refraction is 10.

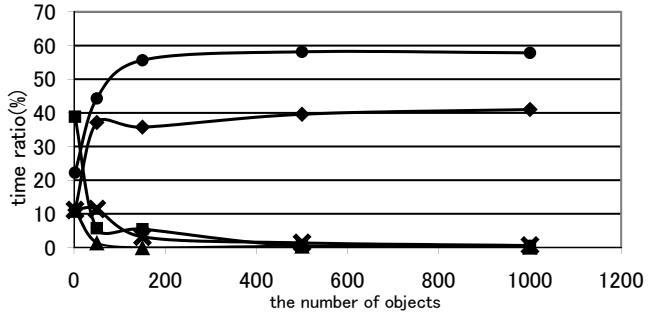

Figure 3. the number of reflection and refraction is 100.

(3) Judging of Intersection and Updating : This module judges intersect polygon with ray vector and validity of distance data calculated in Distance Calculation module.

(4) Result output

On designing the intersection calculation hardware, the total chip area for implementing above five processing module is too large. So we evaluated the number of arithmetic units included in each modules, to select an module that is the possible size to implement as a hardware circuits. Table 1 shows the details of the number of the used arithmetic units in each processing module. From the results shown in Table 1, the largest processing module, Judging of Intersection and Updating, ware selected as accelerated engine.

3. Design of Judging of Intersection and Updating

Figure 4 shows the block diagram of the hardware engine. The hardware engine consists of the following modules.

(1) Calculating point of intersection (Figure 5) : The intersection point is calculated by the following equation.

Table 1. Details of used arithmetic units

Processing	(1)	(2)	(3)	(4)
Adder	2	4	9	0
Subtractor	9	4	29	1
Multiplier	12	6	30	0
Divider	1	1	0	0
Square root unit	1	0	0	0

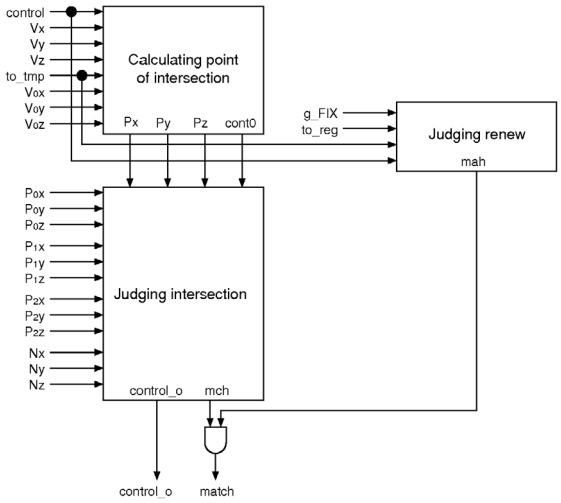

Figure 4. The hardware engine for judging of intersection and updating.

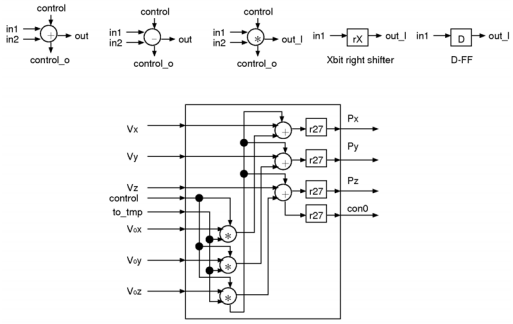

Figure 5. Calculating point of intersection.

$$P = V_0 + to\_tmp \cdot V \tag{4}$$

In this equation, V_0 is viewpoint or tracing point, V is tracing vector, and to_tmp is distance data.

(2) Judging renew (Figure 6) : This module judges which data, to_tmp and to_reg, is near from the view point. Further, it judges whether to_tmp is a effective data or not by using g_fix.

(3) Judging intersection (Figure 7) : This module judges whether a polygon includes a intersection with normal vector by the following equations.

$\{(P_1 - P_0) \times (P - P_0)\} \cdot N \geq 0$
$\{(P_2 - P_1) \times (P - P_1)\} \cdot N \geq 0$
$\{(P_0 - P_2) \times (P - P_2)\} \cdot N \geq 0$

4. Evaluation results

The hardware engine with BS arithmetic unit is designed by Verilog-HDL. And the hardware engine with Parallel arithmetic unit is designed for comparison. The technology used for the design is 5-metal-layer Rohm 0.18μm CMOS technology, The data format used for raytracing is fix point integer format with 64 bits. In this evaluation, the number of

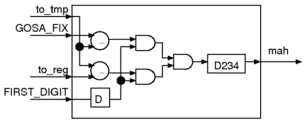

Figure 6. Judging renew.

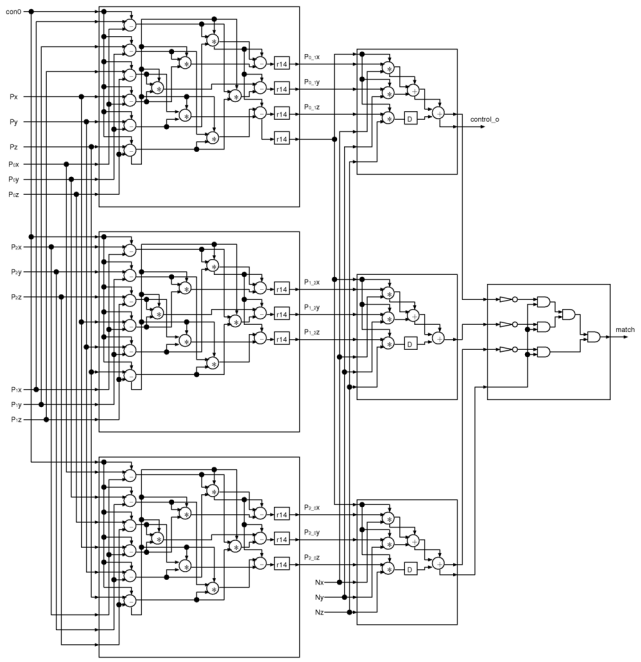

Figure 7. Judging intersection.

cycles required to process one data was estimated. Table 2 shows the evaluation results. Here, P engine is designed with parallel arithmetic unit, and BS engine is designed with BS arithmetic unit. From the results, the area of BS engine is small as 11.1% compared with the P engine. Here, sum of area of only arithmetic unit, are shown in Table 3. It shows that almost area of each engine is area of arithmetic unit because it sum of area size in Table 3 is almost equal to the area size in Table 2. In addition, when we look at area size on Table 3, the area size of BS engine is 10.8% of the P engine. It show that difference of area size between BS engine and P engine comes from the area of arithmetic unit, and it show the significant influence of the small area size of BS arithmetic unit. From our evaluation, we can see that high performance was achieved by implementing several hardware engines in a chip, to make it possible to make in use of the parallelism. Next, we compared the performance and the power consumption in a condition that these hardware engines have the same chip size. On the evaluation of the performance and the power consumption, we evaluated it by on the point of view of performance per area and power per area by following equation.

$$\text{performance per area} = \frac{\text{clock frequency}}{\text{required number of cycle}} \times \frac{1}{\text{area}} \tag{5}$$

$$\text{power per area} = \frac{\text{power}}{\text{area}} \tag{6}$$

Table 2. Evaluation results

	P engine	BS engine
area($\times 10^6 \mu m^2$)	15.858	1.758
power(W)	0.692	1.642
clock frequency(MHz)	54.1	757.6
required number of cycle	12	298

Table 3. The area of the arithmetic unit and sum of them used in each engine.

kind of arithmetic unit	BS engine ($\times 10^6 \mu m^2$)	P engine ($\times 10^6 \mu m^2$)
Adder	$5.03\times10^{-4}\times9$	$1.88\times10^{-2}\times9$
Subtractor	$5.06\times10^{-4}\times29$	$2.96\times10^{-2}\times29$
Multiplier	$5.51\times10^{-2}\times30$	$4.82\times10^{-1}\times30$
Sum of area	1.67	15.5

The BS engine achieves 5.09 times higher performance per area and 21.4 times higher power consumption than the P engine.

The reason for the high performance of the BS engine is that the BS arithmetic unit is significantly small. And, the number of register which adjust data input/output timing is small because BS engine send serial data. so it can exploits many parallelism found in the processing module effectively. The reason for the high power consumption of the BS engine is that the BS arithmetic unit works at higher clock frequency. Figure 8 shows the relations between clock frequency and power consumption. To analyze the relations between clock frequency and power consumption in the BS engine, the following data are considered,

BS(1) Max clock frequency of the Figure 8.

BS(2) The clock frequency which power consumption is same as that of the P engine.

BS(3) The clock frequency when the performance per power is same between the BS engine and P engine.

Table 4 shows the evaluation result. As a result, BS engines, BS(1), BS(2) and BS(3) consumes more power than the P engine to achieve the same performance.

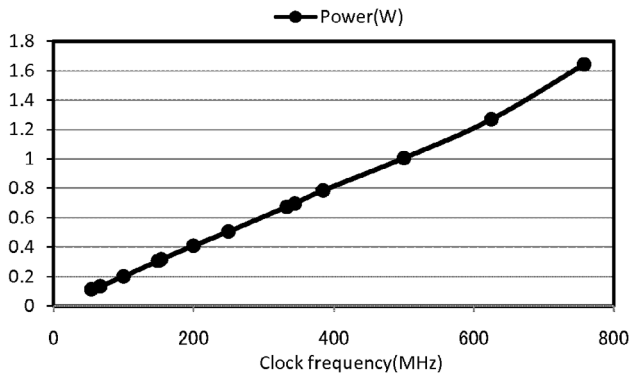

Figure 8. Clock frequency and power consumption of BS engine.

Table 4. Power consumption according to the various clock frequency

	P engine	BS(1)	BS(2)	BS(3)
area($\times 10^6 \mu m^2$)	15.858	1.758	1.754	1.754
clock frequency (MHz)	54.1	757.6	344.8	305.8
power(W)	0.692	1.642	0.695	0.305
processing speed per power($\times 10^6$/W)	6.5	1.55	1.66	1.63

From these results, BS engines achieves higher performance, but the power consumption is higher, compared with the P engine.

5. Conclusion and future work

In this paper, to accelerate the Judge of Intersection and Update processing in raytracing, hardware engines with bit serial arithmetic units are developed. As a result, the BS hardware engine achieves higher performance per area. We didn't compare the performance of BS engine with the raytracing hardware's developed in the past, because it is only a part of raytracing hardware. But, it is realized that the raytracing engine which is designed with BS arithmetic unit may have the higher process speed. But, the BS hardware engine consumes more power, so we plan to decrease the power consumption of the hardware engine in the future.

Acknowledgement

This work is supported by the Japanese Grant-in-Aid for Scientific Research (B) (No. 18300016), and VLSI Design and Education Center(VDEC), the University of Tokyo in collaboration with Cadence Design Systems, Inc and Synopsys, Inc.

References

[1] T.Abe and Y.Ohno, "A Ray Tracing Algorithms for Parallel Computers," *The Journal of the Institute of Television Engineers of Japan*, vol. 49, no. 10, pp.1266-1271, 1995.

[2] Jorg Schmittler, Ingo Wald, and Philipp Slusallek, "SaarCOR - A Hardware Architekture for Ray Tracing," *Proceedings of the ACM SIGGRAPH / Eurographics Conference on Graphics Hardware*, isbn 1-58113-580-7, pp.27-36, 2002.

[3] K.Suzuki, M.Asao, K.Nakanishi, K.Tanigawa and T.Hironaka, "Evaluation of the n Bit-Serial Arithmetic Units in Consideration of Trade-off between Area and Performance," *ITC-CSCC 2005*, vol. 3, pp.943-944, 2005.

[4] M.Nakajima, M.Mizutani and H.Takahashi, "A Study on Parallel Raytracing Algorithms for MIMD Distributed Parallel Computers," *The Journal of the Institute of Television Engineers of Japan*, vol. 49, no. 10, pp.1266-1271, 1995.