A Fixed-point 3D Graphics Library with Energy-efficient Cache Architecture for Mobile Multimedia Systems

Min-wuk Lee, Byeong-Gyu Nam, Ju-Ho Sohn, Namjun Cho, Hyejung Kim, Kwanho Kim and Hoi-Jun Yoo
Department of Electrical Engineering and Computer Science
Korea Advanced Institute of Science and Technology (KAIST)
Daejeon, Republic of Korea
leemw@eeinfo.kaist.ac.kr

***Abstract*— A 3D computer graphics (3DCG) library with energy-efficient cache architecture is implemented for mobile multimedia systems. The developed library is based on fixed-point arithmetic for low energy consumption. To achieve high performance, the library is optimized at both of the assembly and the algorithm levels on 3 different Advanced RISC Machines (ARM) processors. In order to enhance energy-efficiency as well as performance furthermore, a series of simulations have been performed on application programs with various cache configurations. We find that 2-way set associative cache consumes low energy with negligible performance degradation. In this cache system, optimized library can achieve 66.1% performance improvement and 25.3% energy saving in average compared with the conventional 4-way set associative cache system. Software and hardware co-optimization achieves 67K polygons/sec with low energy consumption. We verified the graphics library with proposed cache architecture by implementing a mobile graphics LSI.**

I. INTRODUCTION

As the mobile electronics market increases rapidly, 3G multimedia terminals are getting more popular. The applications are already extended to the real-time multimedia such as MP3 audio, MPEG-4 video and even to 3D computer graphics (3DCG) [1]. For mobile systems, low energy consumption is the most important issue because of their limited battery capacities [2]. In spite of these requirements, 3DCG operations demand extensively high computing power and broad memory bandwidth even in PC platform.

To design and analyze 3DCG solution for mobile systems, we developed an OpenGL-ES [3] compatible 3D graphics library, MobileGL. MobileGL is based on fixed-point arithmetic, being optimized to the integer datapath of embedded processors. In the previous work [4], however, the 3DCG library was focused on floating-point arithmetic and simulation platform consisted of CPU and external memory only, without caches.

MobileGL is optimized for fast execution time at both of the assembly and the algorithm levels. Simulation is based on 3 different Advanced RISC Machines (ARM) processors (ARM7, ARM9 and StrongARM), assuming each host processor has the conventional 4-way set associative cache [2]. In cached processors, software optimization results in 66.1% performance improvement in average.

Because energy is the product of power and time, low power architecture as well as fast execution time is necessary. We further optimize the cache architecture of ARM processors for energy-efficiency because caches are the most power consuming part in embedded system [5]. Using MobileGL, the energy and the performance of several cache architectures are compared. And we present 2-way set associative cache as an energy optimal architecture with negligible performance degradation compared with the conventional 4-way set associative cache system.

Software and hardware co-optimization achieved 67K polygons/sec with low energy consumption. We verified the graphics library with proposed cache architecture by implementing a mobile graphics LSI.

II. SIMULATION ENVIRONMENT

Simulation environment consists of target hardware platform, 3DCG library and application programs as shown in Fig. 1.

A. Target hardware platform

The simulated target hardware platform contains ARM processor and cache system. Specifically, ARM7, ARM9 and StrongARM are used for ARM processor. Program execution time is represented by following equations.

$$T_{exe\_total} = T_{exe\_CPU} + Memory\_latency \quad (1)$$

$$T_{exe\_CPU} = \sum_{K=1}^{instruction\_counts} T_{exe\_instruction} + \sum_{K=1}^{cache\_hit\_counts} CPU\_cycle \quad (2)$$

where, T_{exe} is execution time.

With cycle-accurate simulator in ARM software development toolkit [6], we obtained the CPU execution time (2) and memory transaction. Analyzing the memory transaction, the memory latency can be measured with

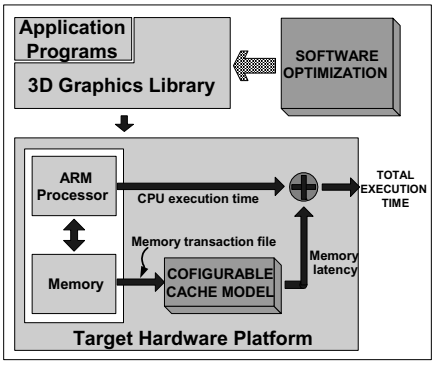


Figure 1. Simulation environment

changing cache configurations. The total execution time (1) is the addition of CPU execution time and memory latency.

B. 3DCG library

The 3DCG library of Fig. 2, MobileGL, is based on OpenGL-ES specification [3]. The library is optimized with the fixed-point arithmetic for non-FPU embedded processors. The graphics pipeline is separated into two parts; 1) LT/LO for lighting and texturing (LT) or lighting only application (LO) and 2) TO for texture-only application. Because texture-only application is widely used, TO part is optimized for reducing unnecessary branch instructions. In geometry stage of TO part, transformation and projection operations can be unified to reduce the matrix-vertex multiplications because lighting process is absent. Disabling perspective correction of texture address is implemented as an extended option which is not specified in OpenGL-ES. After the triangle setup and interpolation, z-comparison for depth test is performed in advance to prevent unnecessary shading and texturing [7]. The implemented library supports QVGA screen size.

C. Application programs

We developed the gaming applications using MobileGL for scene tests. Fig. 3 shows two test images captured from animated image sequences and their benchmarks. Fig. 3(a) was used for texture-only or both texturing and lighting and Fig. 3(b) for lighting only application.

III. SOFTWARE OPTIMIZATION

Fig. 4 shows the performance improvement through the software optimization on three cached ARM processors (ARM7, ARM9 and StrongARM). First, we develop the MobileGL with ANCI C standard (**Step A** and **Step E** in geometry and rendering stage, respectively), then, we further optimize the library step by step.

A. Software optimization in geometry stage

Fig. 4(a) shows the performance improvement in geometry stage with three ARM processors.

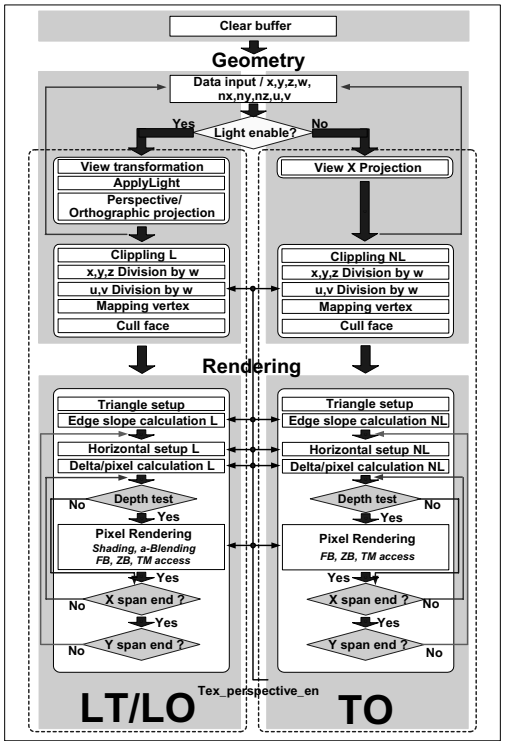


Figure 2. 3DCG library block diagram for mobile multimedia systems

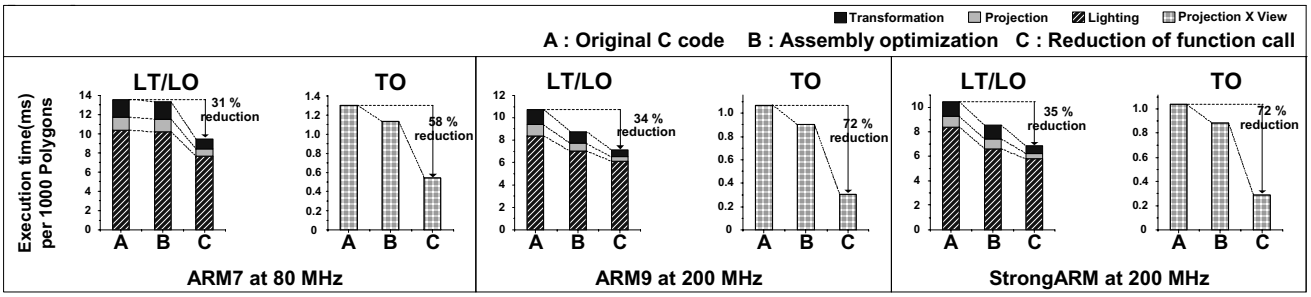
Scene	Screen resolution (Pixels)	Number of polygons	Average pixels per polygon	Texture storage
Tiger	176 X 208	988	8	32KB
Car	176 X 208	1004	12	-

(a) (b) (c)

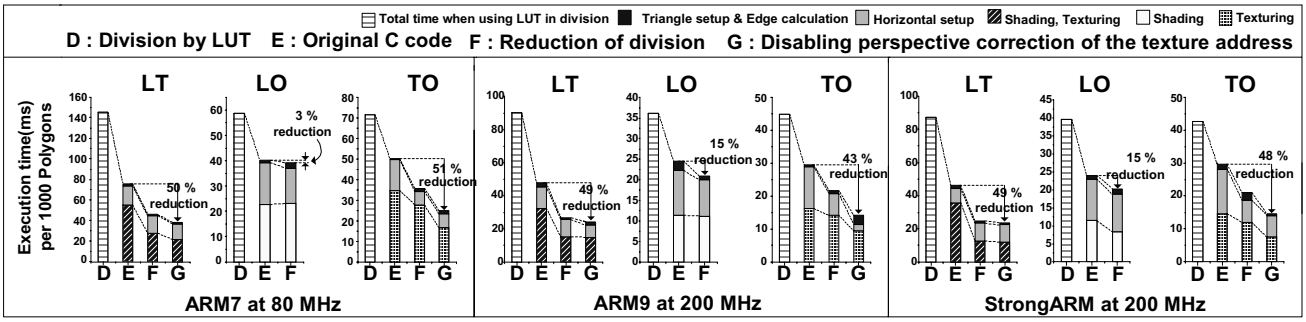
Figure 3. : (a) Tiger scene. (b) Car scene. (c) Test scene benchmarks

Step B: In step B, assembly optimization is performed. Especially, the arithmetic function, division and multiplication are optimized. Because the library adopts the fixed-point arithmetic, geometry stage data are represented by the format Qm.n, where m and n indicate integer precision and decimal fraction, respectively. To insure decimal fraction, n is greater than or equals to m. However, a multiplication may cause an overflow. We implemented special multiply function (SMF) to extend the result to 64-bit in order to avoid overflows. Step B includes the assembly optimization of SMF. Multiple register transfer addressing mode is applied to implement SMF easily and briefly. Division optimization of reducing instruction counts improves performance in lighting stage.

Step C: Step C explains the performance improvement when using ARM's MUL or MLA instruction instead of using SMF in matrix calculation. We can eliminate the function call for SMF with the cost of one more shift by minimizing the coordinate values.



(a) Performance improvement in geometry stage



(b) Performance improvement in rendering stage

Figure 4. Performance improvement through software optimization

B. Software optimization in rendering stage

Fig. 4(b) shows the performance improvement in rendering stage with three ARM processors.

Step D: In rendering stage, most of divisions are reciprocal and their denominators are smaller than 320. In previous work [8], this divider is implemented with look-up table (LUT) in hardware because of its small denominator range, but it is disadvantageous in software implementation with cached processor due to frequent cache misses.

Step F: Because the 3DCG in mobile system requires only a limited screen resolution under QVGA, most edge spans are 1,2,4 or 8 pixel distance. In this case, we used shift operation rather than division because denominators are the power of 2. In texture application, it gives 31% execution time reduction as shown in step F.

Step G: The step G is about disabling perspective correction of the texture address. It also gives performance improvement without critical image quality loss.

By optimizing the software through several steps described above, we achieve 87% performance improvement at both lighting and texturing application, 19.6% at lighting only, and 91.6% at texture-only application in average. It reduces energy consumption because of fast execution time.

Until now, we have focused on reducing energy consumption by optimizing the software library only. From now, we will explain the hardware effort trying to further reduce the energy consumption by proposing the cache architecture.

IV. PROPOSED CACHE ARCHITECTURE

For low energy consumption, low power scheme as well as fast execution is necessary. We optimize the cache architecture for energy-efficiency because cache is responsible for up to 50% of overall on-chip energy consumption in embedded system [5]. In order to select the cache configuration for energy saving, a series of simulations were performed on application programs with changing the cache configurations. The configurable parameters in cache are its total size, line size and set associativity. We focused on the factors of; 1) execution time of application programs, 2) power consumption of cache and 3) energy consumption. Power consumption information was extracted from CACTI 3.1 model [9] and we made the power index with changing cache configurations as shown in Table. 1, which shows consumed power of each configuration in cache hit. In cache miss, the power consumption is from 50 to 200 times bigger than in cache hit [2].

The comparison of normalized energy and execution time is performed on various cache configurations as shown in Fig. 5. We choose a base cache of 8Kbytes having 4-way

TABLE I. POWER INDEX ON VARIOUS CACHE CONFIGURATIONS

Cache size	8KB						
Line size	16B				32B		
Way	8	4	2	1	4	2	1
Cache power	1.62	1.00	0.57	0.38	1.01	0.56	0.38

set associativity and a line size of 16bytes. Fig. 5(a) is related with the unified cache architecture, e.g. that of ARM7 and Fig. 5(b) is about the Harvard cache architecture, e.g. that of ARM9 and StrongARM. Direct mapped cache shows poor hit ratio and hence suffers from poor performance and energy consumption, especially in the unified cache architecture. Although adding associativity increases hit ratio of a cache, the 8-way set associative cache consumes large energy because of the additional power. 2-way set associative cache shows the best energy-efficiency and maximum 5% performance degradation compared with the conventional 4-way set associative cache architecture. Using 2-way set associative cache, we can obtain 25.3% energy saving in average compared with the conventional 4-way set associative cache without critical performance loss.

V. IMPLEMENTATION RESULTS

With software optimization and proposed cache system, 3DCG library achieves 67K polygons/sec, consuming low energy as shown in Fig. 6. The performance is 6.7 times better than that of previous work [10]. The proposed 2- way set associative cache architecture was integrated on a mobile graphics LSI successfully [7], as shown in Fig. 7.

VI. CONCLUSION

The fixed-point 3DCG library was developed for mobile multimedia systems. Software optimization at both of the assembly and the algorithm levels were performed, resulting in 66.1% performance improvement in average. Furthermore, we simulated the performance and energy consumption with various cache configurations for improving energy-efficiency. We found that 2-way set associative cache obtained 25.3% energy saving compared with conventional 4-way set associative one. The library achieved 67K polygons/sec without any hardware accelerators. The graphics library with energy-efficient cache architecture was implemented in a mobile graphics LSI successfully.

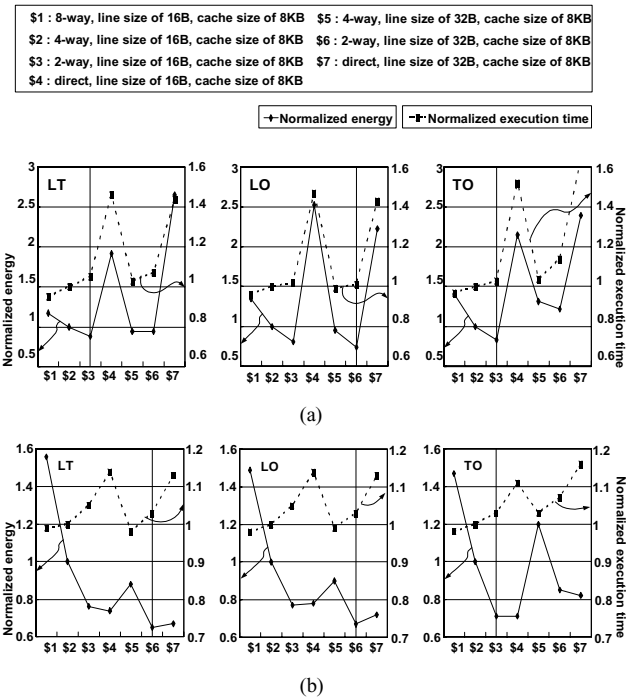


Figure 5. Energy and excution time comparison on the various cache configurations (a) in the unified cache architecture. (b) in the Harvard cache architecture

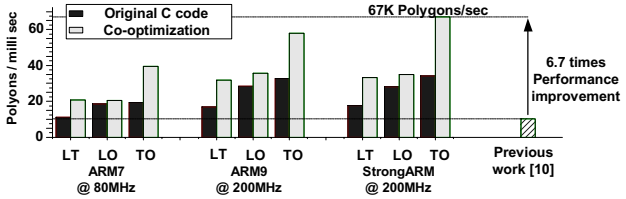


Figure 6. Library performance

Figure 7. Microphotograph of a mobile graphics LSI chip

REFERENCES

[1] Khronos Group, "Bringing 3-D gaming to cell phones," presented at the Game Developers Conf. 2003.

[2] C. Zhang, F. Vahid and W. Najjar, "A highly configurable cache architecture for embedded system," In Proceedings of the 30th ACM/IEEE International Symposium on Computer Architecture, San Diego, CA. 136-146.

[3] OpenGL-ES 1.0 Reference Manual Version 1.0

[4] Ju-ho Sohn, Ramchan Woo and Hoi-Jun Yoo, "Optimization of Portable System Architecture for Real-Time 3D Graphics," ISCAS 2002, Volume: 1 , 26-29 May 2002.

[5] J. Montenaro et al., "160-MHz 32-b 0.5-W CMOS RISC Microprocessor," In Proceedings of International Solid State Circuits Conference, 1996.

[6] ARM Software Development Tookit version 2.50 User Guide, Advanced RISC Machines, Nov, 1998.

[7] Ramchan Woo et al., "A 210mW graphics LSI implementing full 3D pipeline with 264Mtexels/s texturing for mobile multimedia applications," IEEE International Solid-State Circuits Conference, 2003.

[8] Ramchan Woo et al., " A low power 3D rendering engine with two texture units and 29Mb embedded DRAM for 3G multimdedia terminals," European Solid-State Circuits, 2003, Conference on, 16-18 Sept.2003, Pages : 53-56.

[9] CACTI 3.1 :An integrated cache timing, power, and area model

[10] K.Yoshida, T.Sakamoto, T.Hase, " A 3D graphics library for 32-bit microprocessors for embedded systems,", Consumer Electronics, IEEE Transactions on, Volume:44, Issue:3, Aug. 1998.