Design and Test of Fixed-point Multimedia Co-processor for Mobile Applications

Ju-Ho Sohn, Jeong-Ho Woo, Jerald Yoo and Hoi-Jun Yoo

Semiconductor System Laboratory, Department of EECS
Korea Advanced Institute of Science and Technology, Daejeon, Korea.
(sohnjuho@eeinfo.kaist.ac.kr)

Abstract:

In this research, a fixed-point multimedia co-processor is designed and tested into an ARM-10 based mobile graphics processor for portable 2-D and 3-D multimedia applications. The fixed-point co-processor architecture with dual operations realizes advanced 3-D graphics algorithms and various streaming multimedia functions in a single hardware while consuming low power. The instruction-wise clock gating on fixed-point SIMD datapath allows fine-grained power control in application-specific manner. The co-processor takes 10.2mm^2 in 0.18μm 6-metal standard CMOS logic process and achieves 50Mvertices/s graphics performance with 75.4mW power consumption. The implemented chip is successfully demonstrated on the development board equipped with software graphics library and evaluation environment.

1. Introduction

The increasing popularity of handheld terminals such as PDAs and smart cell-phones requires more multimedia functions to be integrated in a single hardware. Especially, their applications are already moving to various 2-D multimedia functions and even to the realtime 3-D graphics. Recently, several researches on multimedia architectures have tried to increase the mobile graphics capabilities in handheld terminals. In the previous works, they did not integrate all required hardware blocks like the vertex shader [1] and showed lack of processing parallelism for streaming graphics data [2]. Or, they provided the fixed functionalities and were revealed to show low performance. Also, the conventional floating-point architectures intentionally decreased the performance by lowering the operating frequency to meet the limited power budget [3][4].

Since users watch the 3-D graphics images on a small screen of mobile devices very close to their eyes, every pixel in mobile applications should be drawn with higher quality by advanced graphics algorithms than that in a PC system [5]. These advanced graphics algorithms require the programmability such as DirectX and OpenGL shading language extensions in graphics hardware. Moreover, the programmability in mobile terminals can allow the various multimedia applications to be optimized through software in a single compact and fast hardware.

In this research [6], we designed and tested a fixed-point multimedia co-processor with programmable SIMD vertex shader for the ARM-10 based mobile graphics processor [7]. The ARM-10 co-processor architecture with dual operations is utilized for implementation and multimedia extensions. The fixed-point SIMD datapath performs 3-D vertex shading programs and various multimedia operations in compact and low power hardware, because we can use only simple integer arithmetic circuits. The implemented chip is successfully demonstrated on the development board equipped with software graphics library and evaluation environment, while satisfying the requirements of the battery lifetime and the system resources in mobile devices.

2. Design Concepts

The ARM architecture defines the co-processor as a general mechanism for extensions of instruction set architecture (Figure 1) [8]. The multimedia co-processor is a 128-bit 4-way SIMD co-processor of the ARM-10 processor and can perform general integer and fixed-point SIMD arithmetic operations and 3-D graphics functions such as geometry transformation and lighting calculation. We used ARM-10 co-processor architecture for multimedia extensions because of the following reasons.

- Pipeline-wise locking operations of co-processor avoids complex synchronizations and provides a single programmer's view of control, which enabling easy integrations with efficient programmability.
- Direct signal path of co-processor interface doesn't need the bus arbitrations, which reducing the unwanted stalls of hardware accelerators.
- Data cache of main processor can be shared with co-processor during multimedia operations.

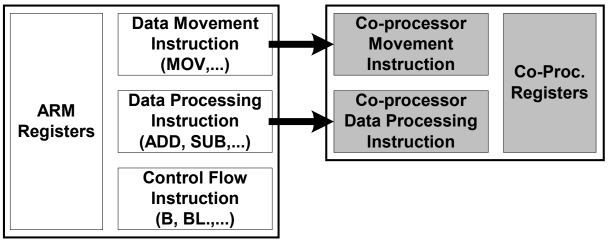
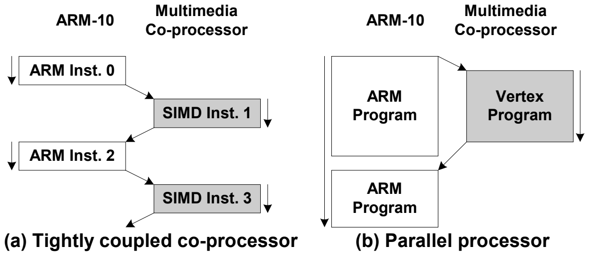

ARM RISC CPU
Co-processor
ARM Registers
Data Movement Instruction (MOV,...)
Data Processing Instruction (ADD, SUB,...)
Control Flow Instruction (B, BL.,...)
Co-processor Movement Instruction
Co-processor Data Processing Instruction
Co-Proc. Registers

Figure 1: ARM Co-processor Architecture

3-9810801-0-6/DATE06 © 2006 EDAA

However, in order to increase parallelism further in multimedia processing, the proposed multimedia co-processor has dual operating states as shown in Figure 2.

- Tightly coupled co-processor (TCC): In this state, the multimedia co-processor is a normal ARM-10 co-processor. The instructions of the co-processor are issued in the instruction stream of the main processor as extended co-processor instructions, and they are executed in lock step with pipeline of the main processor. The TCC state implements general integer and fixed-point SIMD data processing instructions and all instructions can be executed conditionally like other ARM instructions.
- Parallel Processor (PP) [9]: In this state, the co-processor is an independent processor and can operate without control of ARM-10 processor. In the programmer's view, the PP state instructions are the subset of the TCC state instructions with graphics extensions such as source swizzling and write-masks. Moreover, in the PP state, there are more options for input and output operands than the TCC state. The co-processor executes the independent vertex program codes while ARM-10 processor performs main application program or even enters into cache miss state. Various user-defined vertex processing such as geometry transformation and lighting calculation can be performed for current vertex input during next vertex fetch of ARM-10 processor.



(a) Tightly coupled co-processor (b) Parallel processor

Figure 2: Dual Operations

3. System Architecture

Figure 3 shows the block diagram of the multimedia co-processor. It consists of two parts – control and datapath. In the control part, there is a 2kB code memory that stores vertex program codes of graphics instructions. Vertex program control unit (VPCTRL) issues the graphics instructions without control of the ARM-10 processor. The general SIMD instructions are transferred through the co-processor interface and the contents of control register determine its operating state. The two operating states – the TCC state and the PP state – share all of the hardware blocks except instruction fetch units. To maintain the communication protocol of ARM-10 co-processor interface, the multimedia co-processor drives the co-processor busy signal (CPbusy) in the PP state so that next co-processor instruction from the main processor stands by for synchronization.

In the datapath part, there is a fixed-point vector unit that is responsible for all SIMD arithmetic operations such as addition and multiplication. Special function unit (SFU) is responsible for reciprocal (RCP) and reciprocal square root (RSQ) operations. Most of the operations are performed as 32-bit fixed-point numbers, and achieve a single cycle throughput. For streaming graphics processing, the vertex shader contains multiple register files – input vertex registers (VIR), output vertex registers (VOR) and general SIMD registers (VGR). The VIR, used to hold the vertex attributes such as position and normal vector, is fed into the fixed-point SIMD datapath. The VGR is used to store temporary results during vertex program execution. The shaded vertex output is transformed into one of the VORs. There are three VORs for caching of vertex data in the primitive assembly and only one of them is accessible in the vertex program. We have the display list buffer, implemented as 32kB synchronous SRAM, to store graphics primitives, reducing the traffic on external memory I/O. Also, the display list buffer can be shared to hold vertex constants at the same time for design simplicity of hardware. To enhance the efficiency of addressing and to avoid the conflicts when accessing display list buffer, we have two integer address registers for indexed display list buffer reads. In addtion, the display list buffer has the following two features.

- Auto increment and decrement addressing modes for stream management
- 8 bit / 16 bit unpack with shuffling of vector components for geometry compression

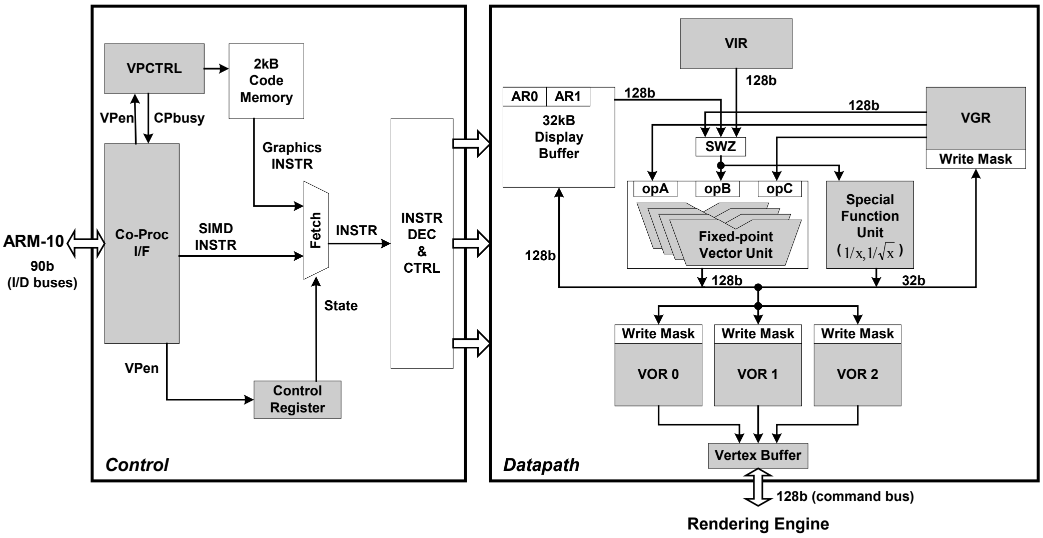


Figure 3: Block Diagram of the Multimedia Co-processor

In order to maximally save the power consumption, the clock gating is performed as instruction-by-instruction basis as shown in Figure 4. By the definition of the ARM-10 co-processor interface, ARM-10 processor must drive co-processor instruction valid (CPINSTV) signal to the co-processor only when the current instruction issued from ARM-10 processor is the valid co-processor instruction. Using CPINSTV, the clock signals of SIMD register files can be gated off when the write operations of the register files are not required. CPINSTV also reduces power dissipated in the datapath of SIMD arithmetic units by eliminating the unnecessary signal transitions.

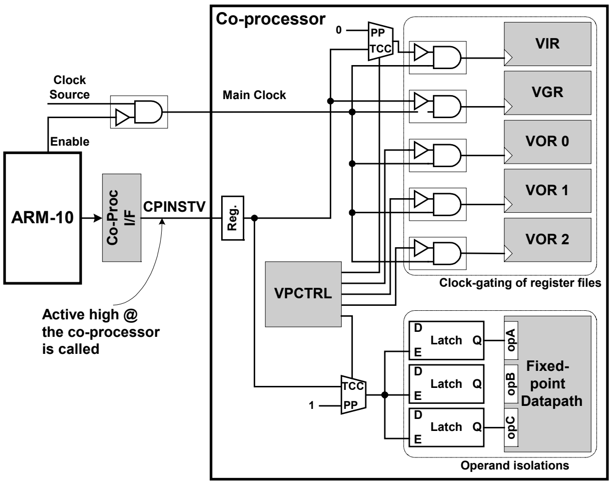


Figure 4: Instruction-wise Clock gating

4. Fixed-point SIMD datapath

Most of 2-D and 3-D multimedia applications require real number representation to support various algorithms. Simple integer datapath of fixed-point unit can achieve higher clock frequency while consuming less power than floating-point unit, yielding total energy reduction. For typical 3-D matrix transformation, gate level simulation of 4-stage pipelined 32-bit fixed-point multiplier showed 30% higher maximum operating frequency than 6-stage pipelined single-precision floating-point multiplier. In addition, the fixed-point multiplier consumed only 83% power of the floating-point multiplier at the same operating frequency. Consequently, when the fixed-point arithmetic is applied to graphics applications, 36% of total energy consumption can be saved on average. All datapath elements of the multimedia co-processor are designed to perform fixed-point arithmetic operations efficiently by using only simple integer arithmetic circuits.

Figure 5 shows the block diagram of the 32-bit fixed-point multiplier in the SIMD multiply unit. Two stage 32-by-16 integer multipliers with integer shifters for fixed-point conversion achieve the single cycle throughput for fixed-point multiply and accumulate (MAC) operation. In addition, fast 4-cycle matrix transformation (TRFM) is implemented in the SIMD multiply unit. By broadcasting vector elements of input vertex, TRFM can be calculated by the first MUL and the following three MAC operations. However, fixed-point MUL and MAC operations require two cycle integer multiplications and two cycle integer additions, leading latency to be 4-cycle. To resolve data dependency between these MUL and MAC operations, it is allowed that the intermediate value of the integer multipliers can be bypassed to accumulate input of the integer adder. By this scheme, the proposed co-processor shows 50Mvertices/s peak graphics performance for parallel projection at 200MHz operating frequency.

The SIMD ALU in the fixed-point datapath can calculate all of the arithmetic and logic operations including byte shuffle, data packing and operand alignment (Figure 6). Since 32-bit fixed-point number is represented in a typical 32-bit integer type, integer adder and shifter circuits are used for calculation of the fixed-point numbers.

Although fixed-point arithmetic provides robust performance in mobile multimedia processing, various multimedia applications such as physical calculations still require enhancement of dynamic range in real number representation. In the co-processor, two special instructions – controlled ADD/SUB (CAS) and controlled logical shift (CLS) are added for efficient software floating-point emulation as shown in Figure 6(b). In order to enhance SIMD parallelism in software programming of floating-point routines, the CAS and the

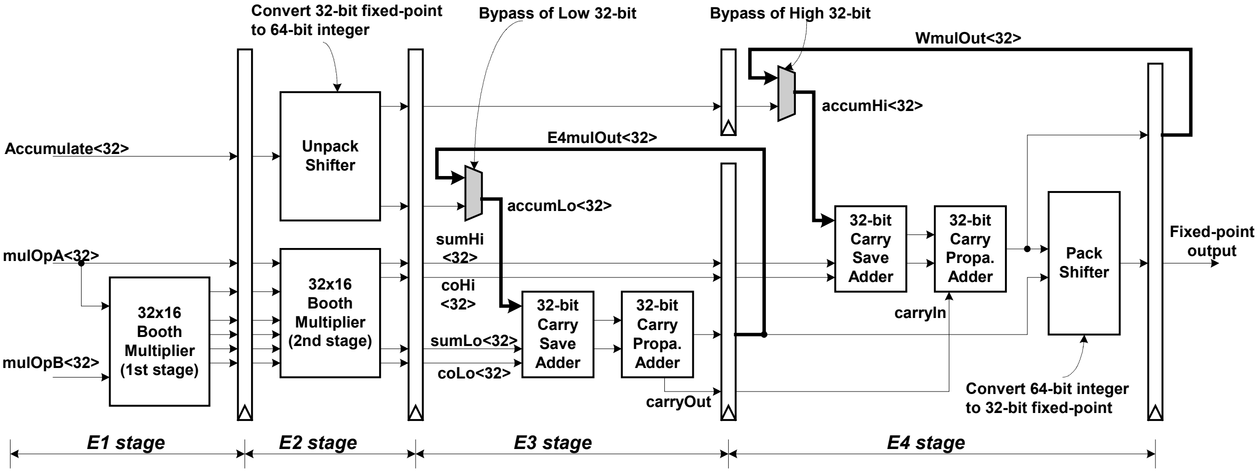
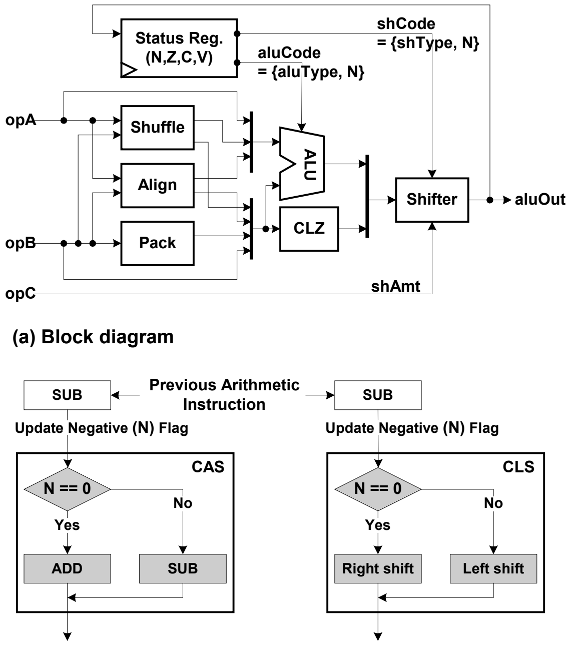


Figure 5: Fixed-point Multiplier Unit

CLS instructions change the control flow instructions to single cycle SIMD arithmetic operations. With floating-point emulation, the proposed co-processor shows 80Mflops peak floating-point performance at 200MHz operating frequency.



(a) Block diagram

(b) Two instructions (CAS, CLS) for floating-point emulations

Figure 6: SIMD ALU

5. Evaluation Platform

The proposed multimedia co-processor was fabricated in 0.18μm 6-metal standard CMOS logic process and integrated into the ARM-10 based mobile graphics processor. The die photograph is shown in Figure 7. It takes 10.2 mm^2 and consumes 75.4mW in the continuous calculations of full 3-D geometry operations. The peak graphics performance is 50Mvertiecs/s for parallel projection at 200MHz operating frequency.

The evaluation platform (Figure 8) was developed to evaluate and demonstrate mobile 3D graphics using a flexible topology and protocol [10]. It incorporates Intel's PXA255 host system since the prototype chip doesn't implement subsidiary hardware blocks such as memory management unit and an LCD controller. The host system is used for displaying and accessing the target system while varying the configuration parameters such as external memory capacity and bus protocols. The hardware layer of the evaluation platform contains the target system equipped with the fabricated chip and an FPGA system controller.

The mobile graphics library, MobileGL, was implemented in the software layer to simplify development of applications. MobileGL is an OpenGL-ES compatible graphics library optimized with hand-written assembly language to improve performance of an ARM-based mobile 3D graphics system. MobileGL consists of a fixed-point math library, vertex shader invocation routines, rendering engine invocation routines, primitive assembly, and state variables with vertex array capability. The native platform interface (NPI) provides intrinsic functions of hardware-dependent programmer's model in assembly and a high-level language for the core of the MobileGL. MobileGL can be ported to various hardware configurations without major architecture modifications by using NPI. The cycle-accurate software emulator of target hardware and the performance profiler were implemented in the evaluation platform for performance evaluations and future derivative development.

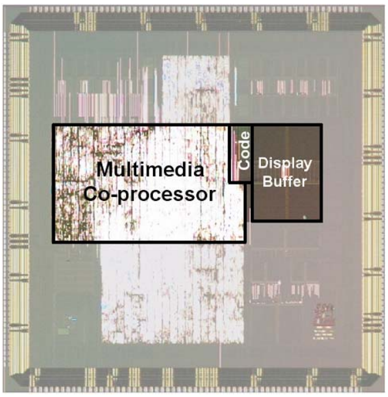
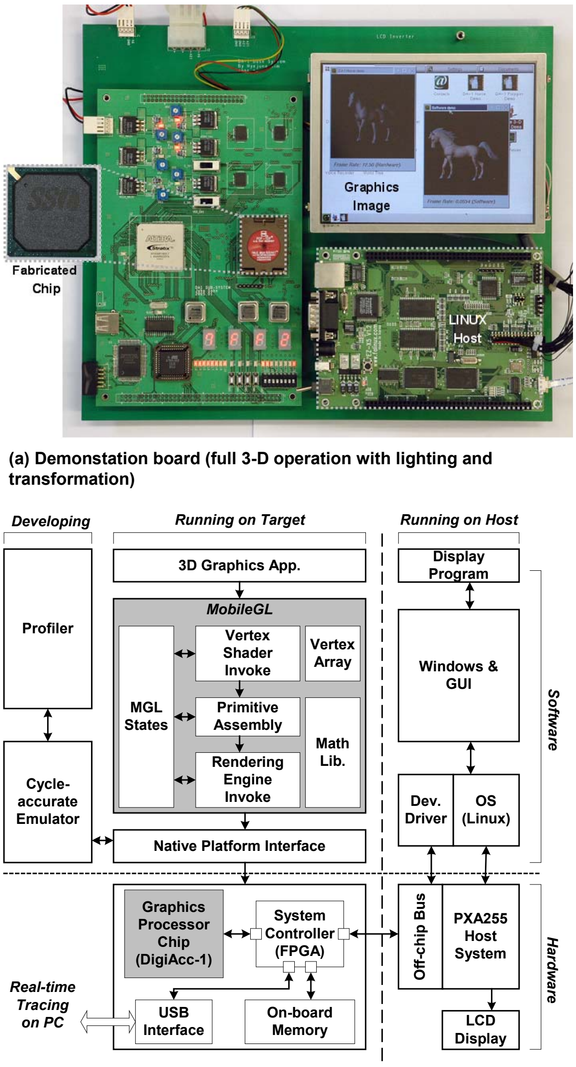


Figure 7: Die Photograph



(a) Demonstation board (full 3-D operation with lighting and transformation)

(b) Block diagram

Figure 8: Evaluation Platform

6. Conclusion and Future Works

We have presented the design and test of fixed-point multimedia co-processor for mobile applications. Most multimedia architectures for mobile applications have mainly focused on design of specific hardware accelerators in conventional bus-based system-on-a-chip. Especially, the 3-D graphics architectures have been investigated with comprehensive considerations of rasterization and texture mapping functions. In order to balance 3D graphics pipeline within the limited system resources and provide a single hardware solution for various multimedia applications, we used simple and efficient programmable architecture instead of using dedicated hardware engine with complex functions. Since main purpose of the proposed design is to provide high performance with low power consumption, we used the co-processor architecture composed of fixed-point SIMD datapath for maximum throughput and easy programmability. Moreover, the dual operations enhanced from conventional co-processor architecture allow parallel operations in streaming multimedia processing. And the instruction-wise clock gating makes it possible to save the power consumption maximally in various adaptations of the co-processor. The multimedia co-processor was fabricated in 0.18μm 6-metal standard CMOS logic process and integrated into the ARM-10 based mobile graphics processor. The peak graphics performance is 50Mvertiecs/s for parallel projection at 200MHz operating frequency. We also designed the development platform with software graphics library and evaluation environment, and successfully demonstrated the implemented chip running realtime 3-D graphics applications.

The data flow and interface between computing elements of multimedia engine and external memory is the crucial concern in designing multimedia hardware. This interface problem strictly limits the whole system performance in most cases. Therefore, as for the future works, efficient memory streaming system for the proposed multimedia co-processor will be investigated to boost overall performance with high-sustained throughput. Scalable data stream engine including direct memory access (DMA) and multi-layer bus architecture will be focused with consideration of software library such as vertex array and vertex buffers.

References:

[1] Chi-Weon Yoon, et al, “An 80/20MHz 160mW Multimedia Processor Integrated With Embedded DRAM, MPEG-4 Accelerator, and 3D Rendering Engine for Mobile Applications,” ISSCC, pp.142-143, 2001

[2] Ramchan Woo, et al, “A 210mW Graphics LSI Implementing Full 3D Pipeline with 264Mtexels/s Texturing for Mobile Multimedia Applications,” ISSCC, pp. 44-45, 2003

[3] Masatoshi Imai, et al, “A 109.5mW 1.2V 600Mtexels/s 3-D graphics engine,” ISSCC, pp. 332-333, 2004

[4] Masatoshi Kameyama, et al, “3-D LSI core for mobile phones – Z3D,” Graphics Hardware, pp.60-67, 2003

[5] Tomas Akenine-Moller, et al, “Graphics for the masses: A hardware rasterization architecture for mobile phones,” SIGGRAPH, pp.801-808, 2003

[6] Ju-Ho Sohn, et al, “A Fixed-point multimedia co-processor with 50Mvertices/s programmable SIMD vertex shader for mobile applications,” ESSCIRC, pp. 207-210, 2005

[7] Ju-Ho Sohn, et al, “A 50Mverties/s graphics processor with fixed-point programmable vertex shader for mobile applications,” ISSCC, pp. 192-193, 2005

[8] Steve Furber, “ARM: System-on-chip Architecture”, 2nd edition, Addison-Wesley Press, 2000

[9] Prashant P. Gandhi, “SA-1500: A 300MHz RISC CPU with Attached Media Processor,” HotChips 10, 1998

[10] Ju-Ho Sohn, et al, “Low power 3D graphics processors for mobile terminals,” *To be published in* IEEE Communication Magazine, Dec. 2005