Cost-Effective Low-Power Graphics Processing Unit for Handheld Devices

Byeong-Gyu Nam, Jeabin Lee, Kwanho Kim, Seungjin Lee, and Hoi-Jun Yoo,
Korea Advanced Institute of Science and Technology

ABSTRACT

Cost-effective handheld graphics processing units are discussed in the aspects of performance, memory bandwidth, power, and area requirements. The proposed RamP architecture has special features of cost-effective low-power arithmetic units, memory bandwidth reduction, and dynamic power management schemes for handheld GPUs. The detailed design of RamP-VI is explained as an example of the RamP architecture. It adopts logarithmic arithmetic for power and area efficiency, and has a triple-domain power management scheme to minimize power consumption at a given performance level. The proposed GPU shows peak performance of 141 Mvertices/s and 52.4 mW power consumption when it operates at 60 frames/s. It shows 17.5 percent performance improvement and 50.5 percent power reduction compared to the latest work.

INTRODUCTION

As wireless communication terminals like cellular phones and PDAs migrate from text-based terminals to multimedia stations, they are incorporating various multimedia applications such as mobile imaging, MP3 playing, digital multimedia broadcasting (DMB), and real-time 3D graphics gaming. Although advanced very large-scale integrated (VLSI) technology enabled this migration and realization of multimedia systems-on-a-chip (SoCs) for these applications, it is still a challenging issue to implement real-time 3D graphics requiring huge computing power on wireless handheld systems with only limited system resources. To deal with this problem, there have been several studies on hardware graphics accelerators to provide sufficient computing power while reducing power consumption [1–5].

In 3D computer graphics, 3D objects are modeled using a number of triangles, and these triangles are given to the graphics pipeline, which is composed of application, geometry, and rendering stages. In the application stage the geometry transformation matrix for the objects to be drawn is computed based on the processing results of the artificial intelligence and collision detection algorithm. For each vertex of a given triangle, the geometry stage performs geometric transformation using the transformation matrix and lighting according to the properties of the light sources. Taking the triangle from the geometry stage, the rendering stage carries out color interpolation and texture image mapping for each pixel inside the triangle. A depth test is also carried out to determine the visibility of a given pixel.

There have been several graphics application programming interfaces (APIs) released for handheld graphics systems, and OpenGL-ES [6] is the most widely accepted one in the community. Its graphics pipeline is optimized with fixed-point arithmetic and simplified to save power consumption. In its latest release it adopted the programmable 3D graphics pipeline and thereby incorporates a programmable geometry processor, called the *vertex shader*, to provide various graphics effects.

The basic architecture of the vertex shader is a four-way vector SIMD architecture so that it can process graphics primitives of four-element vectors effectively. Using this programmability, the vertex shader can support not only the OpenGL lighting model, but various advanced lighting models to provide more realistic images.

This article describes a handheld graphics processing unit (GPU) that delivers the highest performance with the lowest power consumption compared to the state of the art. It exploits logarithmic arithmetic to achieve higher computation power within limited area and power consumption. It also adopts multiple-domain dynamic voltage and frequency scaling for the lowest power consumption at a given performance level. Several design concerns for the modern handheld GPU are discussed, and the RamP architecture developed from these concerns is described. The architecture and design of RamP-VI, the latest version of the RamP architecture, are explained in detail.

DESIGN CONCERNS FOR HANDHELD GPUS

There are several design constraints for 3D graphics to be realized in resource limited handheld systems. The constraints include perfor-

0163-6804/08/$25.00 © 2008 IEEE

mance, memory bandwidth, power consumption, and area cost. In this section the specific design concerns regarding these constraints are investigated.

Performance Requirements

For QVGA (240 × 320) screen size, the pixel fill rate is required to be more than 23 Mpixels/s for 60 frames/s, assuming an average depth complexity of 5 in a scene. Thus, the rendering stage becomes the first performance bottleneck of a 3D graphics pipeline to be implemented on handheld systems, and its hardware acceleration is essential to meet such a high performance requirement. With the hardware rendering pipeline, the geometry stage computation becomes the next performance bottleneck. The vertex fill rate should be over than 2.3 Mvertices/s because the average pixel count of a triangle is about 10 on a QVGA screen. Therefore, hardware support is also required for geometry stage computations in handheld 3D graphics systems.

Memory Bandwidth

In general, 3D graphics rendering requires a huge amount of memory bandwidth for operations such as texture mapping, depth test, and alpha blending. The recent mobile DDR SDRAM can provide maximum memory bandwidth of 1 Gbyte/s. However, a rendering engine with trilinear MIPMAP filtered texture mapping, which is common for modern GPUs, requires 1.1 Gbyte/s for a rendering performance of 50 Mpixels/s. In addition, the memory bandwidth required for running tje application program and the vertex data transfer to the vertex shader is over 200 Mbytes/s. Therefore, the memory bandwidth reduction also becomes a design concern for handheld graphics systems design.

Power Consumption

One of the most critical design issues for battery-powered handheld devices is reducing power consumption because of their limited power budgets. Since a current Li-ion battery can supply 2000 mWh, system power budget for LCD, CPU, memory, and other peripherals is only about 800 mW for 2∼3-h [7]. Under this situation, only 200∼300 mW can be used for the graphics processor. Thus, low-power design and power management schemes become essential for these kinds of systems.

Die Cost

Even though the above mentioned design concerns can be resolved by adopting finer processes or embedded DRAM technologies, the skyrocketing mask cost prevents them from being widely adopted for cost-effective handheld systems. Therefore, chip fabrication cost should also be taken into account for handheld graphics systems usually targeted at consumer electronics. Moreover, the limited footprint of these devices, the area reduction as well as power reduction, is another major design issue.

RamP Architecture

The RamP (RAM + Processor) architecture is proposed to deal with the design concerns investigated in the previous section. The RamP started as a realization of a handheld GPU focused on reducing memory bandwidth and power consumption [1]. It was targeted at the multimedia application processor in modern cell phone architectures. To achieve this goal, the embedded DRAM approach was used in the early generations to reduce external memory bandwidth and power consumption. The RamP architecture also adopted various low-power design techniques at the algorithm, architecture, and circuit design levels. It evolved from a simple shading engine with Gouraud shading and alpha blending capabilities to a full 3D graphics pipeline including texture units and a programmable vertex shader to support OpenGL-ES2.0-like functions, the latest standard 3D graphics API for embedded 3D graphics systems. In this section the low-power arithmetic, memory bandwidth reduction, and power management schemes proposed in the RamP architecture are discussed.

Since 3D graphics requires intensive arithmetic operations, the efficient design of arithmetic units is the most important factor and directly affects the area and power overheads of the GPU.

Cost-Effective Arithmetic Units

Since 3D graphics requires intensive arithmetic operations, the efficient design of arithmetic units is the most important factor and directly affects the area and power overheads of the GPU. Thus, power- and area-efficient arithmetic units for graphics processing are investigated in the RamP architecture. Most of all, the fixed-point number system (FXP) uses simple integer arithmetic circuits, so the FXP arithmetic units can operate at a higher operating frequency and consume less power than the floating-point one. Therefore, these FXP units have been widely used for the rendering engines that operate in the screen coordinates, which have limited dynamic ranges. Their bit width and precision are optimized carefully for specific use in the rendering engine of the RamP architecture [1–4].

For the programmable GPU, FXP with programmable precision is used for the vertex shader in [4]. Since the dynamic range of FXP is quite limited, the precision of FXP is made programmable from Q32.0 to Q1.31 to cope with the different dynamic range requirements of the various shader programs. This has enabled the precision optimization of the shader programs for the required dynamic range and results in low-power design for vertex shaders. The vertex shader includes a four-way vector SIMD multiply-accumulate (MAC) unit and a special function unit for reciprocal (RCP) and reciprocal square root (RSQ) based on fixed-point numbers with programmable precision. This FXP SIMD unit showed 30 percent higher operating frequency and 17 percent lower power consumption than a floating-point one [4].

The newly defined standard 3D graphics API for embedded systems, OpenGL-ES, requires more than 24-bit floating-point number system (FLP) precision for vertex shaders [6] to better support wider dynamic range. Thus, the power- and area-efficient implementation of the FLP vertex shader is investigated in our work.

It is known that the logarithmic number system (LNS) can simplify various arithmetic operations such as multiplication, division, and square root into addition, subtraction, and right shift, respectively. However, addition and subtraction in the LNS become more complex operations, which

The GPU is divided into triple power domains and clock frequency and supply voltage of each domain are controlled separately according to its own workload conditions. This can minimize the power consumption of the GPU at a given performance level.

require nonlinear function evaluations. Therefore, the hybrid approach of the FLP and the LNS is adopted. In this approach, the addition and the subtraction are done in the FLP while other complicated operations are performed in the LNS. Based on this, a power- and area-efficient multifunction unit is proposed that unifies matrix, vector, and elementary functions such as matrix-vector multiplication, vectored multiplication, division, division-by-square-root, multiply-add, dot-product, power, logarithm, and several trigonometric functions into a single four-way arithmetic unit. A detailed description of this unit is covered later.

Memory Bandwidth Reduction

In order to reduce the memory bandwidth required for handheld GPUs, there have been several approaches proposed in the RamP architecture [1–4]. High memory bandwidth can be obtained if the memory is integrated with the processing logic on a single chip. Therefore, embedded DRAM technology was used for the implementation of the previous RamP series [1–3]. The specific design of DRAM macros for 3D graphics operations and their integration into a single chip with the processing units resolved the memory bandwidth problem. However, the increased die cost due to the process complexity of the embedded DRAM technology prevented this approach from being widely adopted for handheld GPUs. Pure DRAM technology was used in [3] to integrate the DRAM macros with reduced fabrication cost. It used the peripheral transistors to implement the required processing logic. However, this approach still suffered from inefficient use of silicon area because the embedded memory is only allocated to the 3D graphics applications, and separate off-chip system memory is required for the other applications.

To solve this problem, the graphics cache system is adopted in [4], where the 26-kbyte graphics cache contains frame, depth, and texture caches. The frame and depth caches have direct-mapped configurations, and the texture cache is configured as a two-way set-associative cache. These caches with 32 bytes cache line show over 90 percent hit rates for the small screens in handheld systems [4]. With this graphics cache system, the frame, depth, and texture memory has been moved to external system memory, which is shared with other applications.

Dynamic Power Management

The RamP architecture also includes various dynamic power management schemes. Clock gating is fully exploited through the RamP architecture for every pipeline stage to reduce unnecessary switching power consumption. In particular, the depth test is moved into an earlier stage of the graphics pipeline to disable the pipeline for invisible pixels from the viewpoint. Thus, the clock to the texture and blending units is gated according to the depth comparison result, and the dynamic power consumption and external memory bandwidth are reduced for invisible pixels [1–4].

Dynamic frequency scaling is adopted in [3, 4] to reduce the dynamic power consumption for the required performance level. Three frequency combinations for the RISC processor and rendering engine are used for FAST, NORMAL, and SLOW modes in [3]. The frequency changes continuously and adaptively according to the performance level in [4].

Since dynamic power consumption is quadratically proportional to supply voltage, dynamic voltage scaling combined with frequency scaling can drastically reduce system power consumption. Therefore, dynamic voltage and frequency scaling (DVFS) is applied for the multiple power domains in our approach. The GPU is divided into three power domains, and the clock frequency and supply voltage of each domain are controlled separately according to its own workload conditions. This can minimize the power consumption of the GPU at a given performance level. The power management scheme is explained in detail in the following section.

RAMP-VI

In this section the latest version of the RamP architecture, RamP-VI [8], is presented in detail including its vertex shader, rendering engine, and power management approach. Several design schemes made for its cost-effective implementation are covered.

Overall Architecture

Figure 1 shows the architecture of the proposed GPU. It consists of three major modules: an ARM10-compatible RISC processor, a vertex shader (VS), and a rendering engine (RE). The RISC processor is included for main system control, artificial intelligence, and collision detection for 3D gaming applications. The VS and RE are designed based on the benefits of logarithmic arithmetic, in which arithmetic complexities are significantly reduced. Thus, a power- and area-efficient design is developed using this property. In addition, the GPU has three power domains to manage the power consumption of each module independently to get the lowest power consumption at the given performance level. First-in first-out buffers (FIFOs) are used across the power domains for buffering the transferred data between modules in spite of the response delay of the power management units (PMUs). Three PMUs are included for the triple-domain DVFS, and each of them changes their clock frequency and supply voltage adaptively according to the workload information obtained from the FIFO entry level.

Vertex Shader

The proposed vertex shader is shown in Fig. 2. It includes floating-point vector register files, a floating-point multifunction unit, and a vertex cache. The floating-point operands are read from the vertex input register file (VIR), general-purpose register file (GPR), or constant memory (CMEM), and the result is written back to the GPR or vertex output register file (VOR).

The multifunction unit is based on the hybrid number system (HNS) of FLP, and the LNS unifies matrix, vector, and elementary functions in a single four-way arithmetic unit. It is organized with four channels and five pipeline stages, as

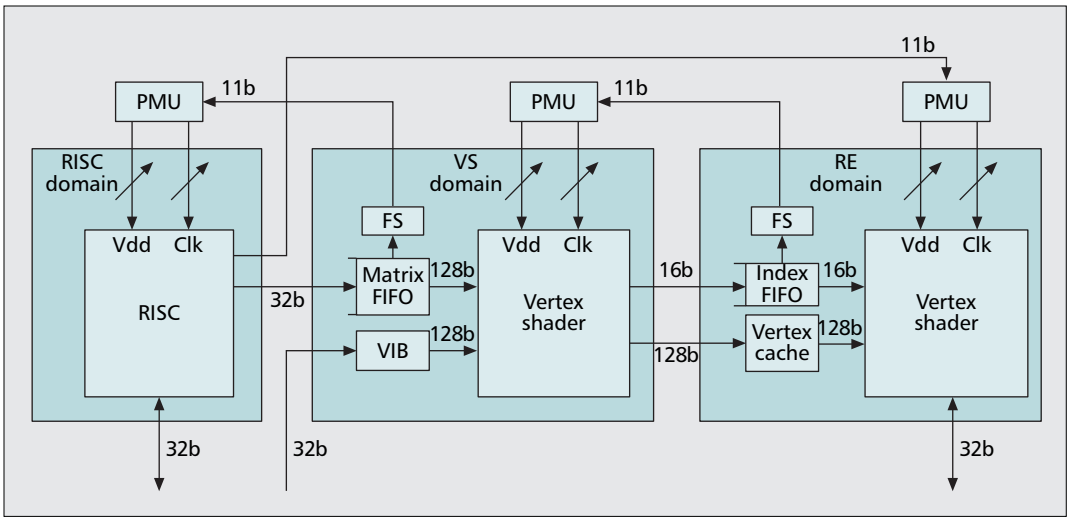

Figure 1. *Proposed handheld GPU.*

shown in Fig. 2. Its fully pipelined architecture achieves single-cycle throughput and maximum five-cycle latency for vector and elementary functions, and two-cycle throughput and six-cycle latency for matrix-vector multiplication.

Logarithmic converters (LOGCs) and antilogarithmic converters (ALOGCs) are used for the number conversion between FLP and LNS. They uses the piecewise linear interpolation scheme for the approximation of nonlinear term evaluations as proposed in [9]. Four of the input 32-bit FLP operands are converted into logarithmic numbers through the four LOGCs in E1. E2 includes a programmable multiplier (PMUL), which can be used as the Booth multiplier for elementary functions (ELM), four LOGCs for vector operations (VEC), or four ALOGCs for matrix-vector multiplication (MAT) by just adding 15-entry 64B LOG and 8-entry 56B ALOG lookup tables to the Booth multiplier and sharing the CSA tree and a CPA. In this way, the number of LOGCs in E1 is reduced to 4 from 8 in [9]. In E3 four adders in the logarithmic domain are provided for the VECs, and the resulting values are converted into the FLP numbers through the four ALOGCs. The programmable adder (PADD) in E4 can be programmed into a single five-input adder tree or four-way two-input SIMD adders for target operations as proposed in [9]. E5 provides an SIMD accumulator for the final accumulation required for the MAT.

Matrix-Vector Multiplication — The geometry transformation in 3D graphics can be computed by the multiplication of a 4×4 matrix with a 4-element vector, which requires 16 multiplications and 12 additions. This can be converted into the HNS as in Eq. 1 requiring 20 LOGCs, 16 adders, 16 ALOGCs, and 12 FLP adders. Since the coefficients of a geometry transformation matrix are fixed during processing of a 3D object, these can be preconverted into the logarithmic domain and used as constants during the processing. Thus, the MAT only requires four LOGCs for vector element conversion, 16 adders in the logarithmic domain, 16 ALOGCs, and 12 FLP adders. This can be implemented in two phases on this four-way arithmetic unit, as illustrated in Eq. 1. In this scheme eight adders in the logarithmic domain and eight ALOGCs are required per phase, and the eight ALOGCs in the first phase are obtained from four ALOGCs in E2 by programming the PMUL into four ALOGCs together with the four ALOGCs in E3. The CPAs in E1 and E3 are used for the eight adders in the logarithmic domain. The four multiplication results from the ALOGCs in E2 and the other four from E3 are added in E4 by programming the PADD into a four-way SIMD FLP adder to get the first phase result. With the same process repeated, the accumulation with the first phase result in E5 completes the MAT. Thus, the MAT is implemented with two-cycle throughput on this four-way arithmetic unit, where it was implemented with 4-cycle throughput in conventional way [4, 9].

$$\begin{bmatrix} c_{00} & c_{01} & c_{02} & c_{03} \\ c_{10} & c_{11} & c_{12} & c_{13} \\ c_{20} & c_{21} & c_{22} & c_{23} \\ c_{30} & c_{31} & c_{32} & c_{33} \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} c_{00} \\ c_{10} \\ c_{20} \\ c_{30} \end{bmatrix} x_0 + \begin{bmatrix} c_{01} \\ c_{11} \\ c_{21} \\ c_{31} \end{bmatrix} x_1 + \begin{bmatrix} c_{02} \\ c_{12} \\ c_{22} \\ c_{32} \end{bmatrix} x_2 + \begin{bmatrix} c_{03} \\ c_{13} \\ c_{23} \\ c_{33} \end{bmatrix} x_3$$

$$= \underbrace{2^{\left(\left\lceil \begin{matrix} \log_2 c_{00} \\ \log_2 c_{10} \\ \log_2 c_{20} \\ \log_2 c_{30} \end{matrix} \right\rceil + \log_2 x_0\right)} + 2^{\left(\left\lceil \begin{matrix} \log_2 c_{01} \\ \log_2 c_{11} \\ \log_2 c_{21} \\ \log_2 c_{31} \end{matrix} \right\rceil + \log_2 x_1\right)}}_{\text{MAT 1st phase}} + \underbrace{2^{\left(\left\lceil \begin{matrix} \log_2 c_{02} \\ \log_2 c_{12} \\ \log_2 c_{22} \\ \log_2 c_{32} \end{matrix} \right\rceil + \log_2 x_2\right)} + 2^{\left(\left\lceil \begin{matrix} \log_2 c_{03} \\ \log_2 c_{13} \\ \log_2 c_{23} \\ \log_2 c_{33} \end{matrix} \right\rceil + \log_2 x_3\right)}}_{\text{MAT 2nd phase}} \quad (1)$$

Vector Operations — The vector multiplication, division, square root, and multiply-add (MAD) can be represented by a single generic operation and is converted into HNS by Eq. 2. For example, operations like $x \times y$, $x \div \sqrt{y}$, $x \times y + z$ can be represented with this generic operation.

$$(x_i \otimes y_i^s \oplus z_i)_{i\in\{0,1,2,3\}} = (2^{(\log_2 x_i)} \oplus {}^{(s\times\log_2 y_i)} \oplus z_i)_{i\in\{0,1,2,3\}}$$

where

$$\otimes \in \{\times, \div\}, \oplus \in \{+, -\}, s \in \{0.5, 1\} \qquad (2)$$

Since Eq. 2 requires two LOGCs for two operands per channel, the PMUL is programmed into four LOGCs to make the eight LOGCs for four channels together with the four LOGCs in E1. The vector MAD and dot product (DOT) require vector element multiplication and final summation. The PADD for the final summation is programmed into a four-way two-input SIMD adder for the vector MAD and a single five-input adder tree for DOT. The *bias* port is used

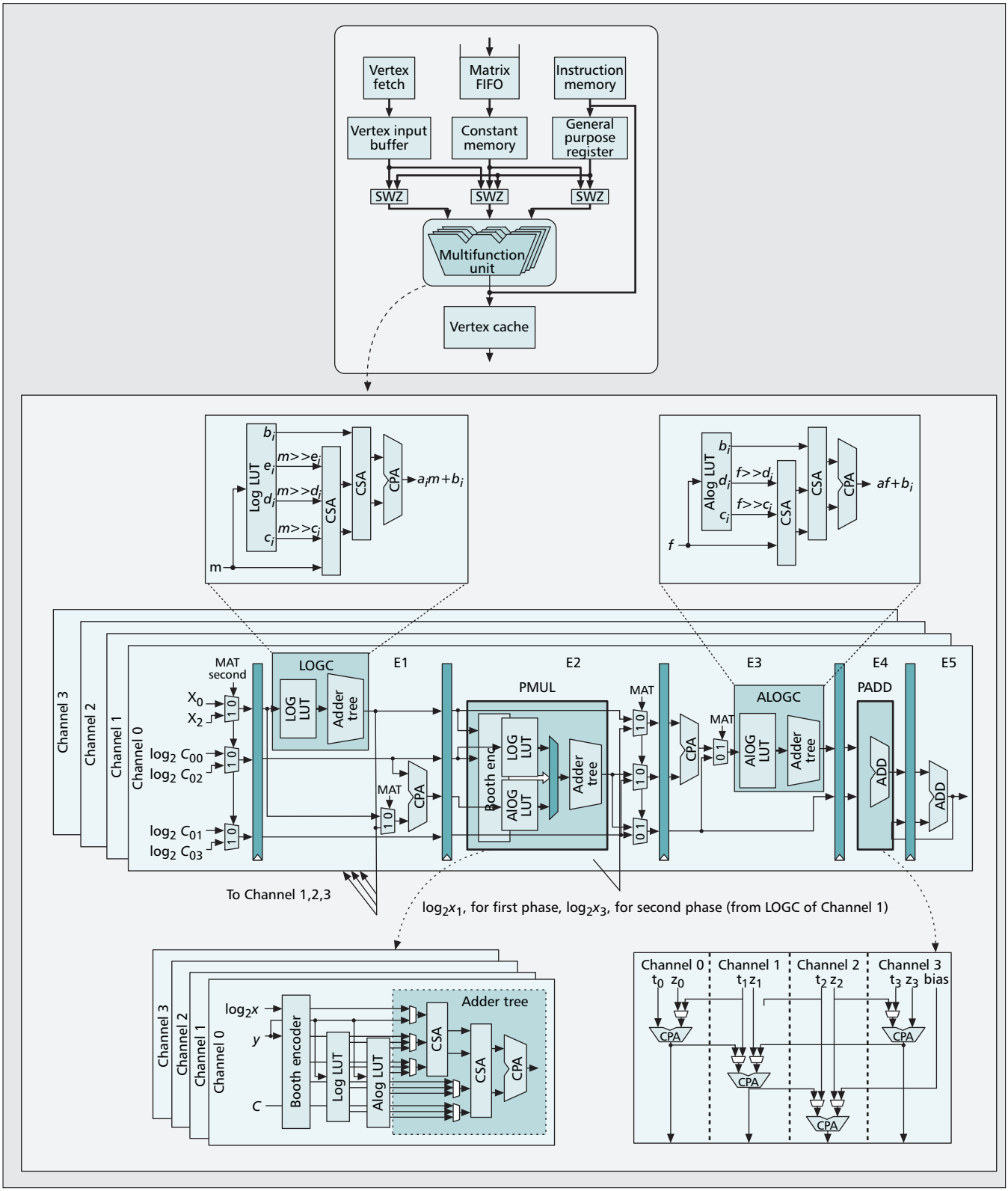

Figure 2. *Vertex shader.*

for TRGs.

Elementary Functions — The trigonometric functions are unified with matrix and vector operations using the Taylor series expansion. For the first five terms of Taylor series computation, a new generic operation is defined and converted into the HNS:

$$c_0x^{k_0} \oplus c_1x^{k_1} \oplus c_2x^{k_2} \oplus c_3x^{k_3} \oplus c_4x^{k_4}) = c_0x^{k_0} \oplus 2^{\log_2 c_1 + k_1 \times \log_2 x} \oplus 2^{\log_2 c_2 + k_2 \times \log_2 x} \oplus 2^{\log_2 c_3 + k_3 \times \log_2 x} \oplus 2^{\log_2 c_4 + k_4 \times \log_2 x}$$

where $\oplus \in \{+, -\}$, and c_i, $\log_2 c_i$, and k_i are coefficients. (3)

For example, an operation like $\sin x = x - x^3/3! + x^5/5! - x^7/7! + x^9/9!$ can be represented with this expression. Since the power is converted into multiplication in the logarithmic domain and k_i is the small integer, this power series is implemented by a four-way 32b × 6b multiplier in the logarithmic domain and final summation of these terms. The first term does not need to be converted into the logarithmic domain since the first term of the Taylor series is usually a constant or input x, which can be directly fed into the bias port. Thus, the power series can be implemented by programming the PMUL into a four-way 32b × 6b BMUL and the PADD into a single five-input summation tree.

The vertex cache is included in the vertex shader to reuse the previously processed vertices. It consists of 16 VORs and shows a 58 percent hit rate. With the twp-cycle throughput matrix-vector multiplication and vertex cache, the vertex shader achieves a peak performance of 141 Mvertices/s at 200 MHz operating frequency.

Rendering Engine

Figure 3 shows the block diagram of the rendering engine. It consists of a triangle setup engine (TSE), rasterizer, and pixel pipeline. In these stages all dividers are implemented using logarithmic arithmetic to avoid conventional power consuming dividers. The TSE and rasterizer compute the vector interpolation (i.e., $x_0 + (\Delta x/\Delta y) \times y$) to set up the triangle parameters. The sequence of division and multiplication required for the interpolation are converted into a sequence of subtraction and addition in the logarithmic domain. The texture unit in the pixel pipeline requires four-way division (i.e., $(s, t, u, v)/w$) for the perspective correct texture address calculation. This is also implemented with subtractors in the logarithmic domain.

The texture unit implements bilinear rather than trilinear MIPMAP texture filtering for power-efficient texture mapping and memory bandwidth reduction. To avoid a power consuming large cache memory, the texture unit contains a cache system composed of a 16-entry texture cache and 8-entry depth and pixel caches. With bilinear MIPMAP texture filtering and a small cache system, the external memory bandwidth is reduced to 550 Mbytes/s, which current mobile DDR SDRAM can provide.

Multiple-Domain Power Management

In this GPU three power domains with separate frequencies and supply voltages are tuned by tracking the workloads. This GPU uses two levels of a hierarchical power management scheme with interframe and intraframe levels. Since the 3D graphics scenes are displayed at a given frame rate, the RE should draw only a finite amount of pixels within the time slot of a single frame. Therefore, at the interframe level, the target frequency and supply voltage for the RE are determined. At every completion of drawing a scene, the software library running on the RISC measures the time elapsed for the drawing and sets a new target frequency and supply voltage adaptively for the next scene processing.

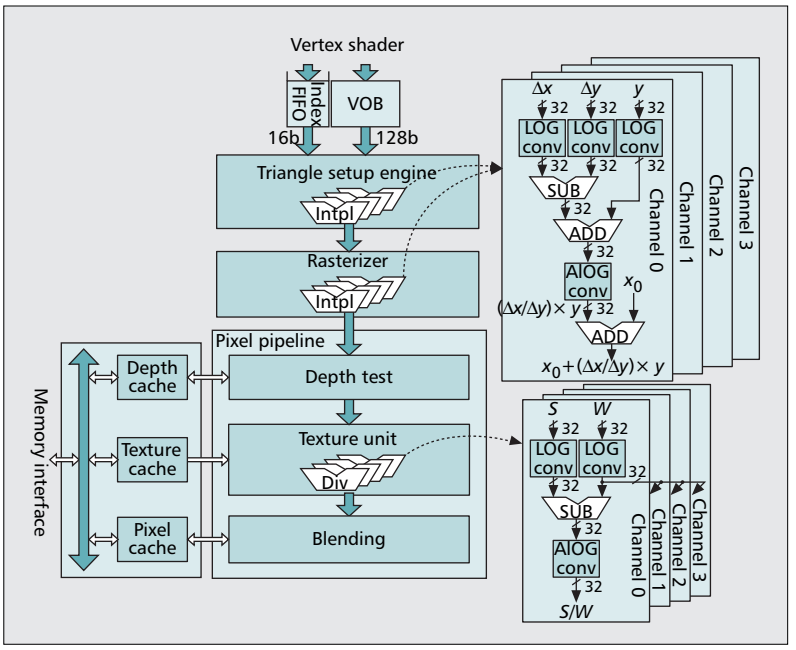

Figure 3. *Rendering engine.*

Since the objects in a scene are composed of a number of triangles, and these are composed of a number of pixels, the workloads of the RISC, VS, and RE, which operate per object, per triangle, and per pixel, respectively, can be completely different. Therefore, at the intraframe level of power management, the RISC, VS, and RE are divided into different power domains, and their frequencies and supply voltages are separately controlled according to their workloads. Figure 4 illustrates the proposed power management scheme.

The workload is obtained from measuring the occupation level of the FIFO. The occupation level is compared to its reference level to give the error value so that the power management unit determines the new clock frequency and supply voltage.

Evaluation Results

Implementation

RamP-VI is fabricated using 0.18 μm 6-metal complementary metal oxide semiconductor (CMOS) technology. Figure 5 shows the chip micrograph, and each power domain is outined with a different color. The core size is 17.2 mm^2, and has 1.57M transistors and 29 kbytes SRAM. The graphics performance of this chip is 141 Mvertices/s for peak geometry transformation

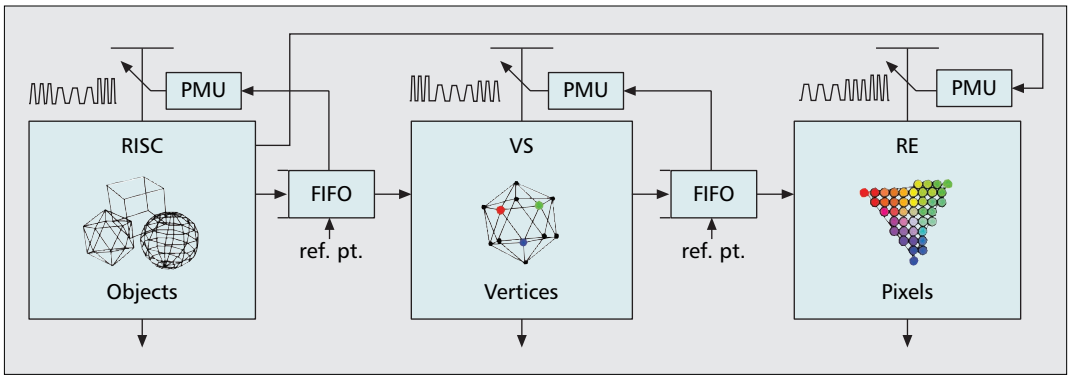

Figure 4. *Triple-domain DVFS scheme.*

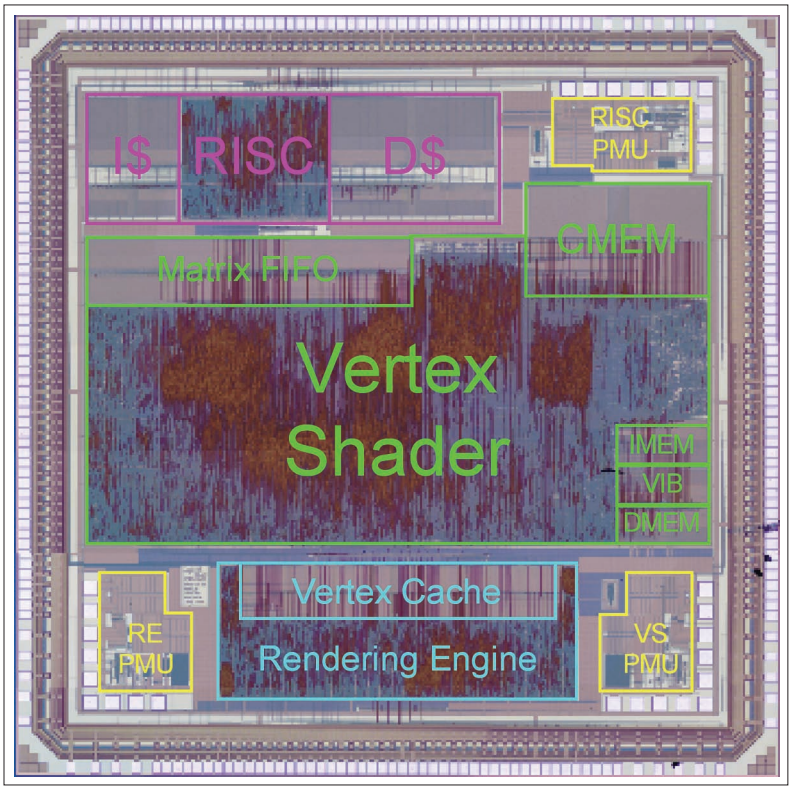

Figure 5. *RamP-VI micrograph.*

and 50 Mpixels/s for rendering operation with texture image mapping. It integrates the RISC, VS, and RE into a small area of 17.2mm^2 and achieves outstanding graphics performance of 141 Mvertices/s by using logarithmic arithmetic in implementing the graphics pipeline. With the multiple-domain power management scheme, this chip only dissipates 52.4 mW when the scenes are drawn at the rate of 60 frames/s, which is a 51 percent reduction from previous work [5].

Comparison

The comparison with other handheld GPUs is listed in Table 1. Since area and power consumption as well as performance are important factors in handheld devices, all these factors are compared in this table. We also compare the normalized performance values regarding power consumption.

The comparison results show that our work achieves the highest performance compared to other handheld GPUs. Mitsubishi's Z3D shows the lowest power consumption, but its performance is severely limited by its low operating frequency. The previous work from our group shows lower performance than this in spite of its similar integration level and process technology. The work in [5] shows similar performance, area, and power consumption to ours even though it integrates the vertex shader only, without the RISC processor and rendering engine.

Conclusion

A cost-effective handheld GPU is investigated in the aspects of performance, memory bandwidth, power, and area requirements. The RamP architecture has the special features of cost-effective low-power arithmetic units, memory bandwidth reduction, and dynamic power management schemes for handheld GPUs. The RamP-VI integrates a full 3D graphics pipeline with an ARM-10 compatible RISC processor, a vertex shader, a rendering engine, and three power management units. The vertex shader includes a 4-way 32-bit floating-point multifunction unit, which exploits logarithmic arithmetic and carries out 3D geometry transformation every two cycles. The two-cycle geometry transformation and 58 percent hit rate vertex cache achieve a peak geometry processing rate of 141 Mvertices/s at 200 MHz clock frequency. The GPU includes three separate power domains, and each domain adopts DVFS for 52.4 mW power consumption when scenes are drawn at 60 frames/s. The comparison results show that the proposed GPU achieves 17.5 percent performance improvement and 50.5 percent power reduction from the latest work.

References

[1] Y.-H. Park *et al.*, "A 7.1GB/s Low Power Rendering Engine in 2D Array Embedded Memory Logic CMOS for Portable Multimedia System," *IEEE J. Solid-State Circuits*, vol. 36, no. 6, 2001, pp. 944–55.
[2] C.-W. Yoon *et al.*, "A 80/20MHz 160mW Multimedia Processor Integrated with Embedded DRAM, MPEG4 and 3D Rendering Engine for Mobile Applications," *IEEE J. Solid-State Circuits*, vol. 36, no. 11, 2001, pp. 1758–67.
[3] R. Woo *et al.*, "A 210mW Graphics LSI Implementing Full 3D Pipeline with 264Mtexels/s Texturing for Mobile Multimedia Applications," *IEEE J. Solid-State Circuits*,

Architecture	Performance	Operating frequency	Power	Area	Function	kvertices/ mW
Mitsubishi Z3D [10]	185 kvertices/s 5.1 Mpixels	30 MHz	38 mW	11 mm2	GE+RE	4.87
KAIST RamP-V [4]	50 Mvertices/s 50 Mpixels/s	200 MHz	155 mW	23 mm2	GE+RE	322.58
C.-H. Yu [5]	120 Mvertices/s	100 MHz	157mW 106 mW @ 60 fps	16 mm2	GE	764.33
KAIST RamP-VI [8]	141 Mvertices/s 50 Mpixels/s	200 MHz	153 mW 52.4 mW @ 60 fps	17.2 mm2	GE+RE	921.57

Table 1. *Comparison of handheld GPUs.*

The RamP architecture has the special features of cost-effective low-power arithmetic units, memory bandwidth reduction, and dynamic power management schemes for the handheld GPUs.

vol. 39, no. 2, 2004, pp. 358–67.
[4] J.-H. Sohn *et al.*, "A 155-mW 50-Mvertices/s Graphics Processor with Fixed-Point Programmable Vertex Shader for Mobile Applications," *IEEE J. Solid-State Circuits*, vol. 41, no. 5, 2006, pp. 1081–91.
[5] C.-H. Yu *et al.*, "A 120Mvertices/s Multi-threaded VLIW Vertex Processor for Mobile Multimedia Applications," *IEEE Int'l. Solid-State Circuits Conf. Dig. Tech. Papers*, Feb. 2006, pp. 408–09.
[6] Khronos Group, OpenGL-ES 2.0, http://www.khronos.org
[7] W.R. Hamburgen *et al.*, "Itsy: Stretching the Bounds of Mobile Computing," *IEEE Computers*, vol. 34, issue 4, Apr. 2001, pp. 28–36.
[8] B.-G. Nam *et al.*, "A 52.4mW 3D Graphics Processor with 141Mvertices/s Vertex Shader and 3 Power Domains of Dynamic Voltage and Frequency Scaling," *IEEE Int'l. Solid-State Circuits Conf. Dig. Tech. Papers*, Feb. 2007, pp. 278-279.
[9] B.-G. Nam *et al.*, "A Low-Power Unified Arithmetic Unit for Programmable Handheld 3-D Graphics Systems," *Proc. IEEE Custom Integrated Circuits Conf.*, Sept. 2006, pp. 535–38.
[10] M. Kameyama *et al.*, "3D Graphics LSI Core for Mobile Phone-Z3D," *Proc. SIGGRAPH/Eurographics Conf. Graphics Hardware 2003*, Aug. 2003, pp. 60–67.

Biographies

Byeong-Gyu Nam (byeonggyu.nam@gmail.com) received a B.S. degree (summa cum laude) in computer engineering from Kyungbook National University, Daegu, Korea, in 1999, and an M.S. degree in computer science from Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Korea, in 2001. From 2001 to 2002 he was with the Electronics and Telecommunication Research Institute (ETRI), Daejeon, Korea. He received a Ph.D. degree in electrical engineering from KAIST in 2007. He is currently working for Samsung Electronics, Giheung, Korea. His research interests include low-power arithmetic units, 3D graphics processors, and microprocessor design.

Jeabin Lee (jeabin@eeinfo.kaist.ac.kr) received a B.S. degree (summa cum laude) in electrical engineering from KAIST in 2006, and an M.S. degree in electrical engineering from KAIST in 2008. He is currently working for Samsung Electronics, Suwon, Korea. His research interest include low-power mixed-mode circuit design and power management of SoCs.

Kwanho Kim (kkh82@eeinfo.kaist.ac.kr) received B.S. and M.S degrees in electrical engineering from KAIST in 2004 and 2006, respectively. He is currently working toward a Ph.D. degree in electrical engineering at KAIST. In 2004 he joined the Semiconductor System Laboratory (SSL) at KAIST as a research assistant. His research interests include VLSI design for object recognition, and architecture and implementation of NoC-based SoCs.

Seungjin Lee received B.S. and M.S. degrees in electrical engineering and computer science from KAIST in 2006 and 2008, respectively. He is currently working toward a Ph.D. degree in electrical engineering and computer science from KAIST. His research interests include low-power digital signal processors for digital hearing aids and body area communication. Currently, he is investigating parallel architectures for computer vision processing.

Hoi-Jun Yoo (hjyoo@ee.kaist.ac.kr) graduated from the Electronic Department of Seoul National University, Korea, in 1983, and received M.S. and Ph.D. degrees in electrical engineering from KAIST in 1985 and 1988, respectively. His Ph.D. work concerned the fabrication process for GaAs vertical optoelectronic integrated circuits. From 1988 to 1990 he was with Bell Communications Research, Red Bank, New Jersey, where he invented the 2D phase-locked VCSEL array, the front-surface-emitting laser, and the high-speed lateral HBT. In 1991 he became manager of a DRAM design group at Hyundai Electronics, and designed a family of fast-1 MDRAMs and synchronous DRAMs, including 256-Mbyte SDRAM. From 1995 to 1997 he was a faculty member with Kangwon National University. In 1998 he joined the faculty of the Department of Electrical Engineering at KAIST. In 2001 he founded a national research center, System Integration and IP Authoring Research Center (SIPAC), funded by the Korean government to promote worldwide IP authoring and its SOC application. From 2003 to 2005 he was the project manager for SoC in the Korea Ministry of Information and Communication. His current interests are SOC design, IP authoring, high-speed and low-power memory circuits and architectures, design of embedded memory logic, optoelectronic integrated circuits, and novel devices and circuits. He is the author of the books *DRAM Design* (Hongleung, 1996; in Korean) and *High Performance DRAM* (Sigma, 1999; in Korean). He received the Electronic Industrial Association of Korea Award for his contribution to DRAM technology in 1994 and the Korea Semiconductor Industry Association Award in 2002.