

Low-Power 3D Graphics Processor using Logarithmic Arithmetic

Byeong-Gyu Nam* and Hoi-Jun Yoo

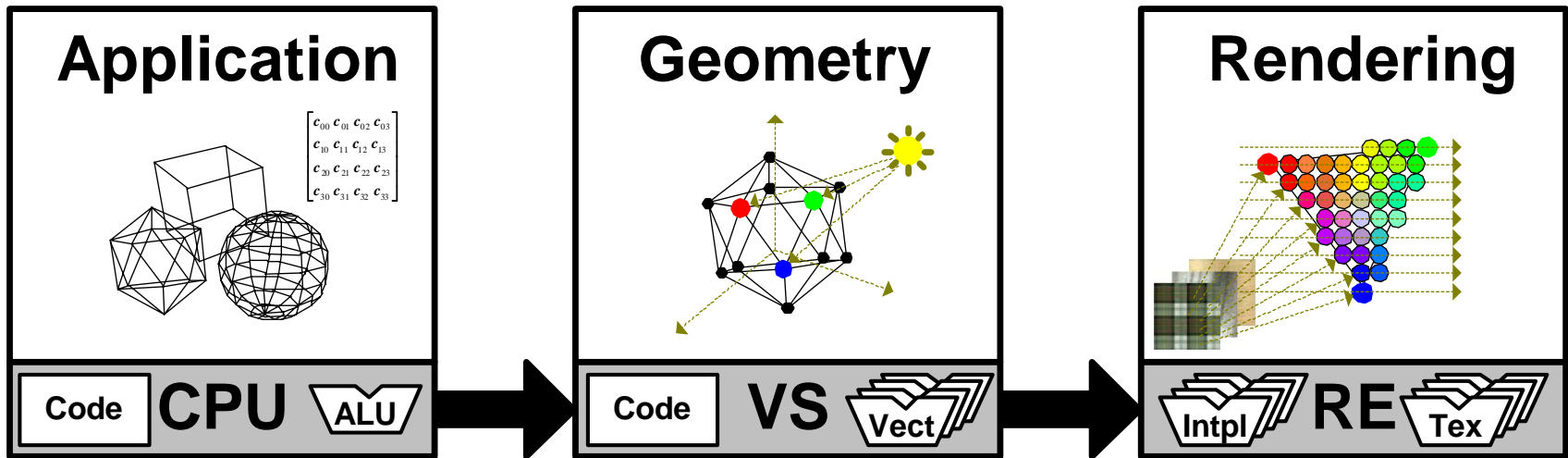
byeonggyu.nam@gmail.com

*Processor Architecture Lab., Samsung Electronics
Semiconductor System Lab., Dept. of EECS, KAIST

Outline

- Backgrounds
- Logarithmic Arithmetic
- Log-based Vector Processor
- Low-Power Graphics Processor (GPU)
- Implementation Results
- Conclusion

3D Graphics Pipeline



- Pipeline Stages

- Application stage: geometry transformation matrix
 - General purpose CPU
- Geometry stage: transformation and lighting (TnL)
 - Numeric vector processor i.e. Vertex Shader
- Rendering stage: interpolation and texture mapping

Logarithmic Arithmetic

- Various Math Operations for 3DG Pipeline

- Matrix-vector multiplication
- Vector operations
- Elementary functions

MAT,
VMUL, VMAD, VDOT, VDIV, VDSQ,
POW, LOG, SIN, COS, ATAN

→ Too complicated for resource limited handsets

- Logarithmic Number System

😊 Reduces arithmetic complexity

☹ Conversion errors

- Handheld 3D Graphics

- Small screen can tolerate a little computation error

→ Logarithmic arithmetic can be used

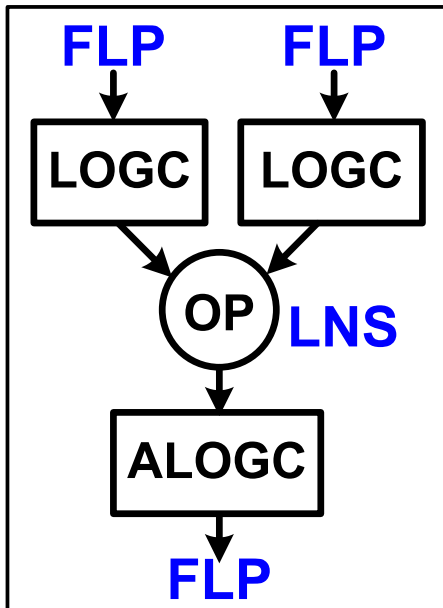
$$x \times y = 2^{(\log_2 x + \log_2 y)} = 2^{X+Y}$$

$$x \div y = 2^{(\log_2 x - \log_2 y)} = 2^{X-Y}$$

$$x^y = 2^{y \times \log_2 x} = 2^{y \times X}$$

Number Converters

- Number System
 - Complex op. in LNS
 - Linear add, sub in FLP
 - Number Converters based on piecewise linear approximation



CSA: carry save adder
CPA: carry propagate adder

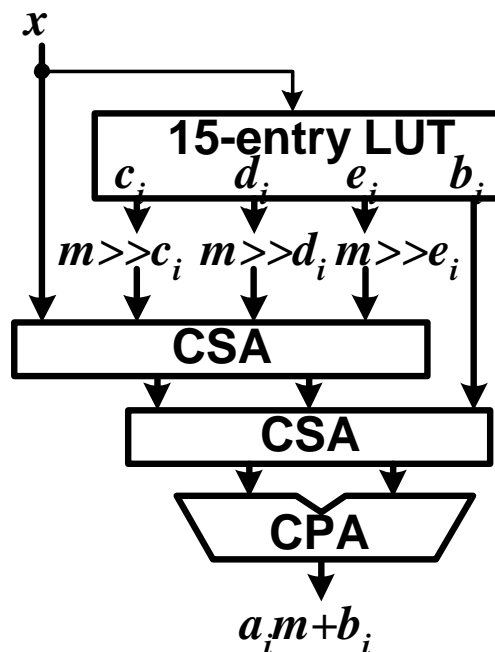
- Logarithmic Converter (LOGC)

$$x = 2^k (1 + m)$$

$$\log_2 x = k + \log_2 (1 + m)$$

$$\log_2 (1 + m) \approx a_i m + b_i$$

- Piecewise linear approx.
- < 0.41% conv. error



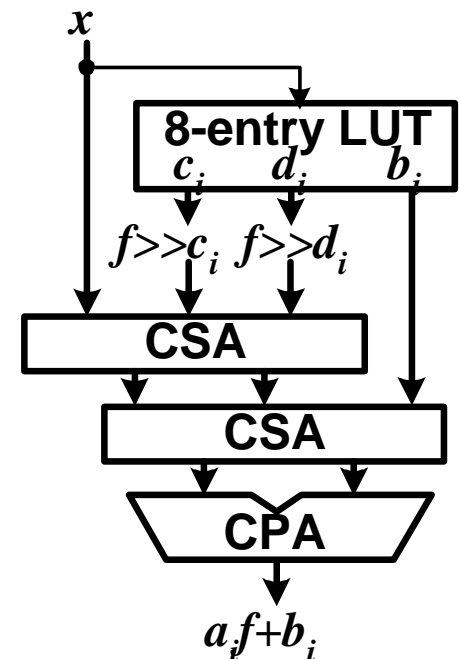
CMOSET08

- Antilogarithmic Converter (ALOGC)

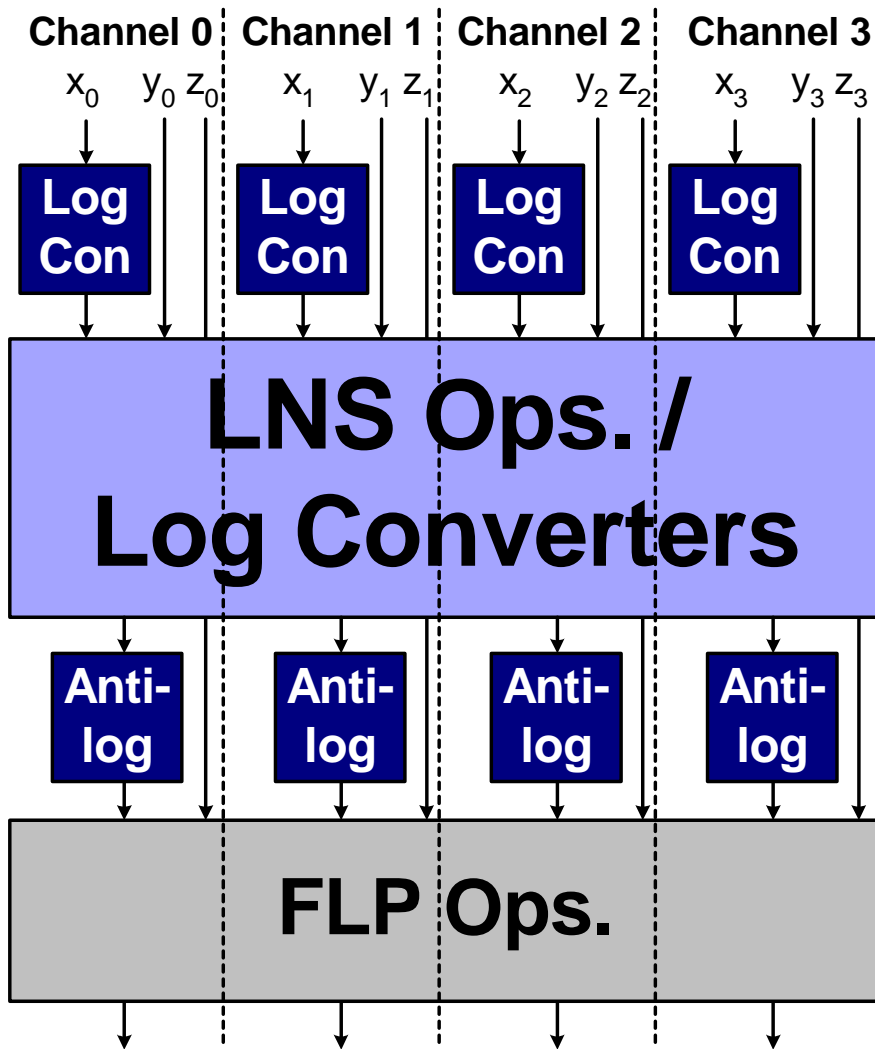
$$2^x = 2^{k+f} = 2^f \ll k$$

$$2^f \approx a_i f + b_i$$

- Piecewise linear approx.
 - < 0.08% conv. error



Multifunction Unit



Operation Set	
Matrix Operation	matrix-vector mul. (MAT)
Vector Operations	add, mul, div, dsq, mad, dot, crs
Elementary Ftn.	pow, log, sin, cos, atan, etc.

- Linear Add / Sub in FLP
- Pipeline Characteristics
 - Throughput of 1-result / 2-cycle, 6-cycle latency for MAT
 - Single-cycle throughput, max. 5-cycle latency

Matrix-Vector Multiplication (1/2)

- Matrix-vector multiplication (MAT)

- Requires 16 multiplications

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

- MAT using log arithmetic

- 20 LOGCs, 16 ADDs, 16 ALOGCs

- Transformation matrix is fixed for a given object

- Matrix coefficients are pre-converted into LNS before processing starts

- 4 LOGCs, 16 ADDs, 16 ALOGCs

- 2-phase implementation → 2 LOGCs, 8 ADDs, 8 ALOGCs per phase

- Implemented as 2-cycle operation on 4-way arithmetic unit

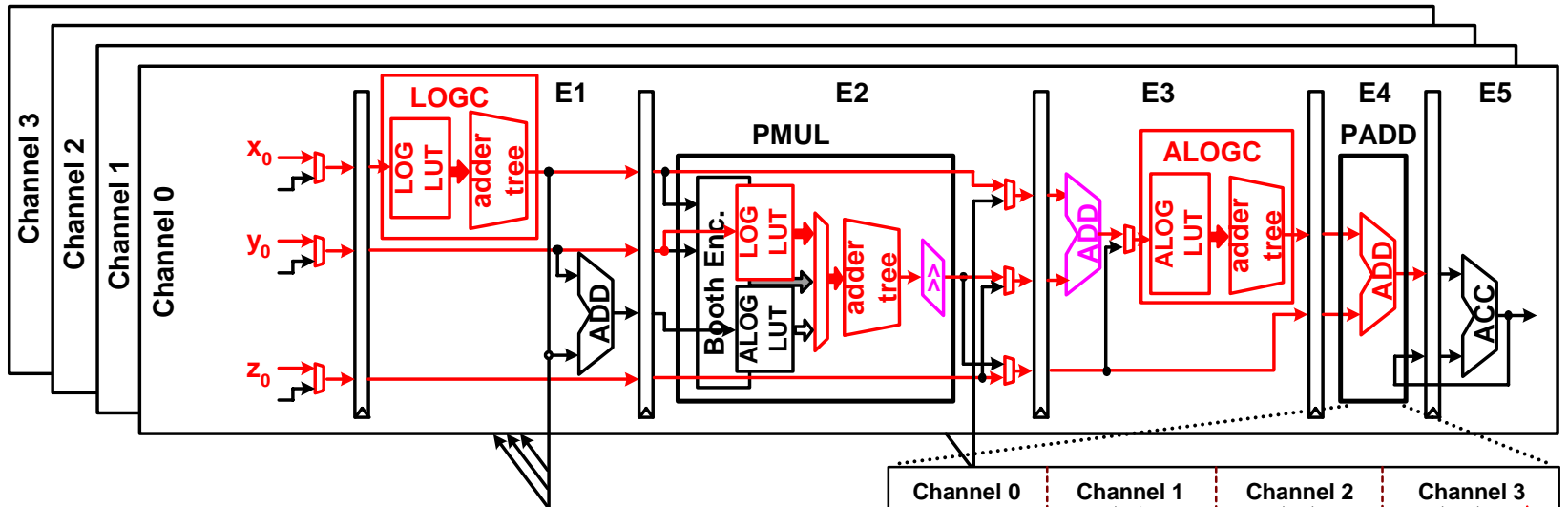
$$\underbrace{2 \left(\begin{bmatrix} \log_2 c_{00} \\ \log_2 c_{10} \\ \log_2 c_{20} \\ \log_2 c_{30} \end{bmatrix} + \log_2 x_0 \right) + 2 \left(\begin{bmatrix} \log_2 c_{01} \\ \log_2 c_{11} \\ \log_2 c_{21} \\ \log_2 c_{31} \end{bmatrix} + \log_2 x_1 \right)}_{\text{MAT 1st phase}} + \underbrace{2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

Vector Operations

- Single Generic Operation for
 - Vector SIMD MUL, DIV, DSQ, MAD
- VECs using log arithmetic
 - Require 2 LOGCs / channel

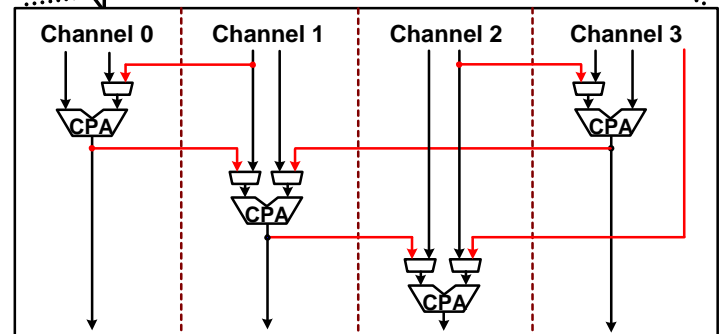
$$\left(x_i \otimes y_i^p \oplus z_i \right)_{i \in \{0,1,2,3\}} = \left(2^{(\log_2 x_i) \oplus (\log_2 y_i \gg q)} \oplus z_i \right)_{i \in \{0,1,2,3\}}$$

where $\otimes \in \{\times, \div\}$, $\oplus \in \{+, -\}$, $p \in \{0.5, 1\}$, $q \in \{0, 1\}$



- Dot Product
 - By programming PADD

$$\sum_{i=0}^3 x_i y_i$$



* PADD: programmable adder

Elementary Functions

- Power Function
- Taylor Series for TRGs
 - SIN, COS, ATAN, ...
 - Single generic operation
- ELMs using log arithmetic
 - Require multiplier in log-domain

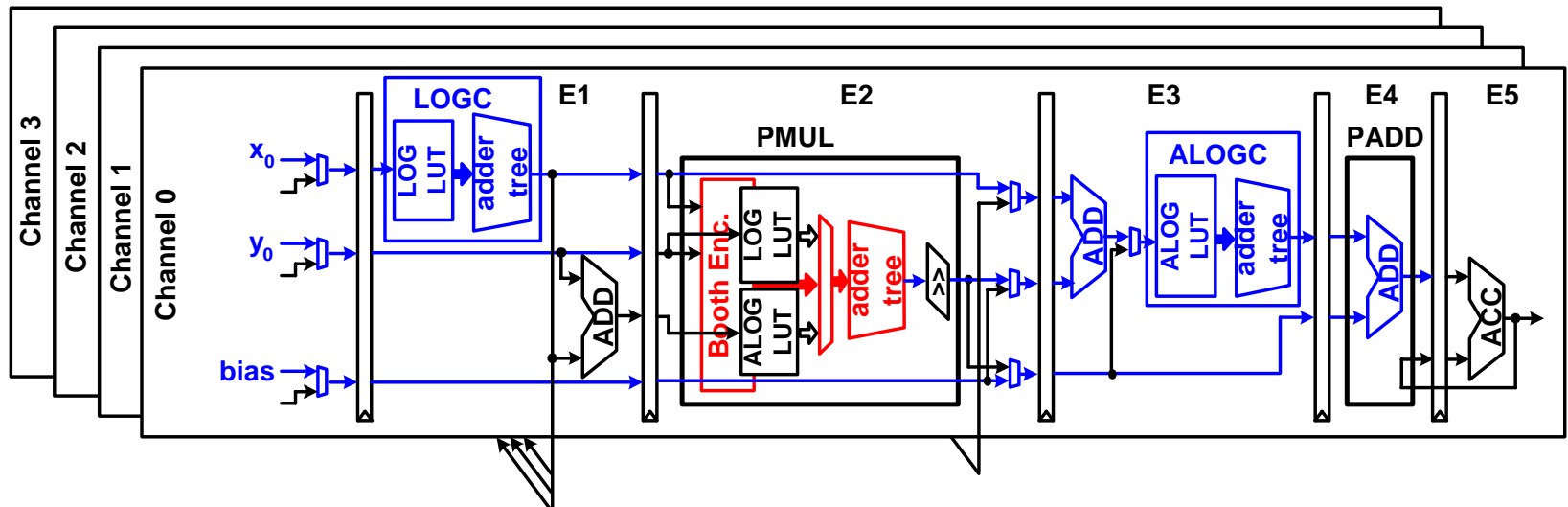
$$x^y = 2^{\log_2 x^y} = 2^{y \times \log_2 x}$$

$$c_0 x^{k_0} \oplus c_1 x^{k_1} \oplus c_2 x^{k_2} \oplus c_3 x^{k_3} \oplus c_4 x^{k_4} =$$

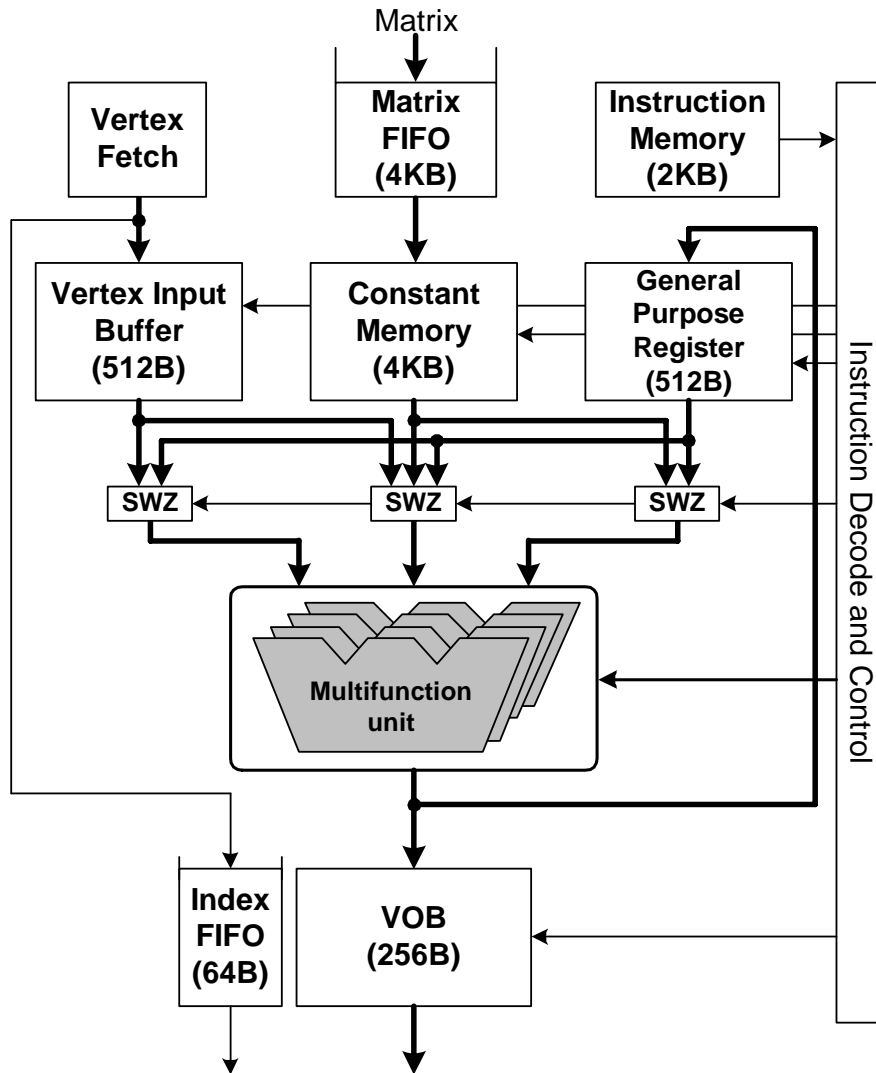
$$c_0 x^{k_0} \oplus [2^{(C_1+k_1 \times \log_2 x)} \oplus 2^{(C_2+k_2 \times \log_2 x)}$$

$$\oplus 2^{(C_3+k_3 \times \log_2 x)} \oplus 2^{(C_4+k_4 \times \log_2 x)}]$$

where $\oplus \in \{+, -\}$, c_i and k_i are coefficients

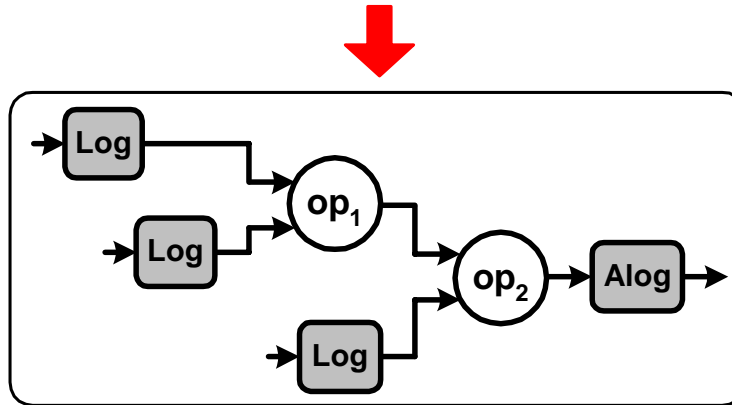
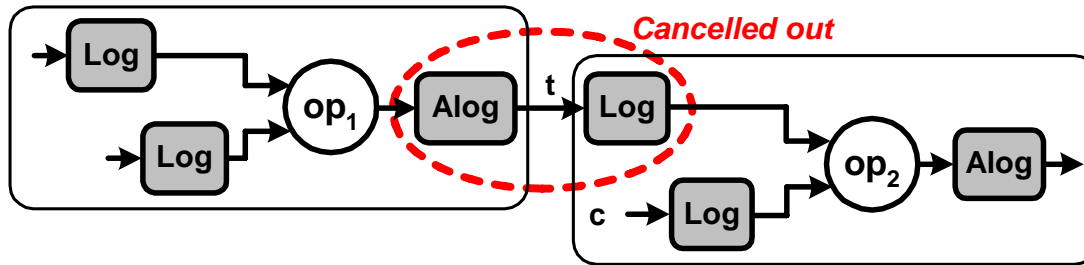


Vector Processor in Log Arithmetic

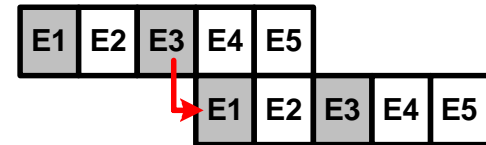


- Vertex Shader
 - Optimized for numeric ops.
 - Used for programmable graphics pipeline
- Multi-function Unit
 - 3D geometry transformation in 2 cycles
 - 4 cycles in conventional
- Operand Forwarding
 - Log-domain forwarding
 - Improves pipeline throughput
 - Reduces computation errors

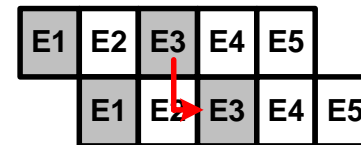
Forwarding in Log-domain



Conventional forwarding

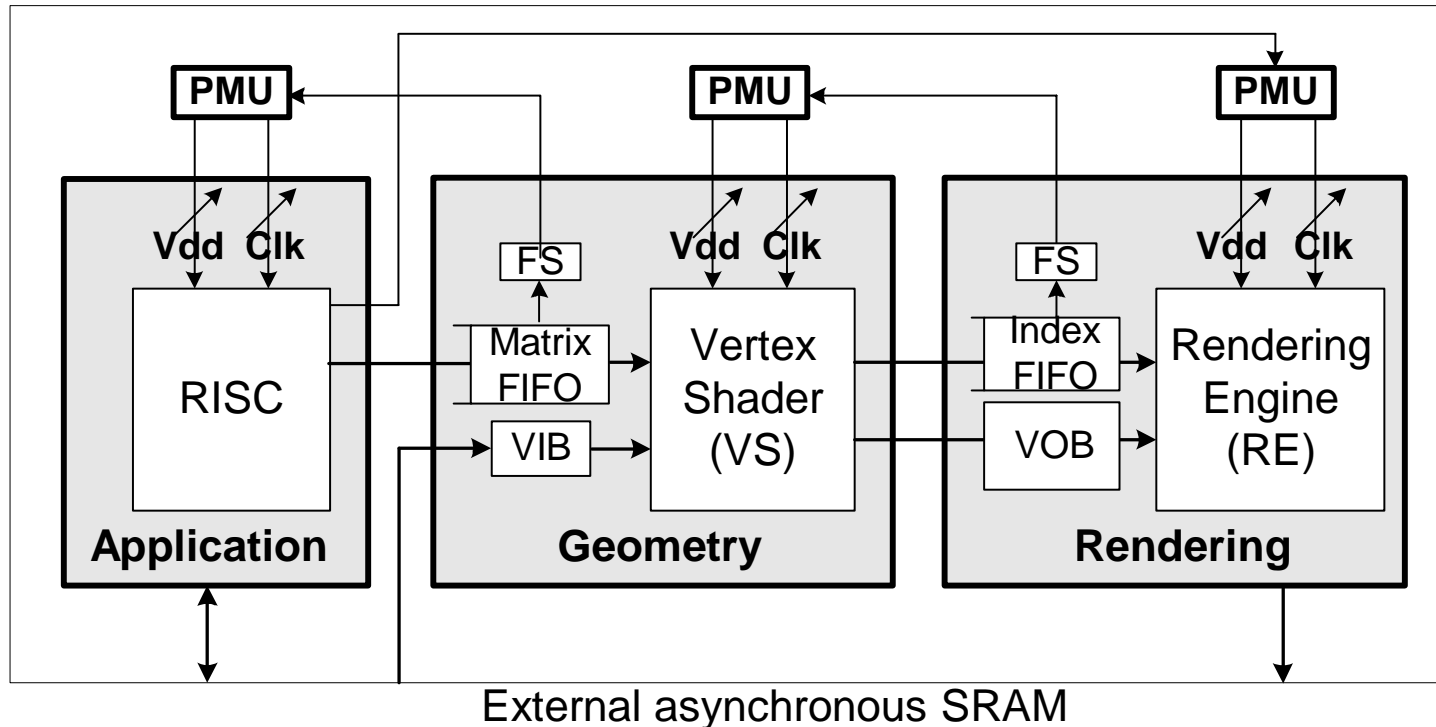


Log-domain forwarding



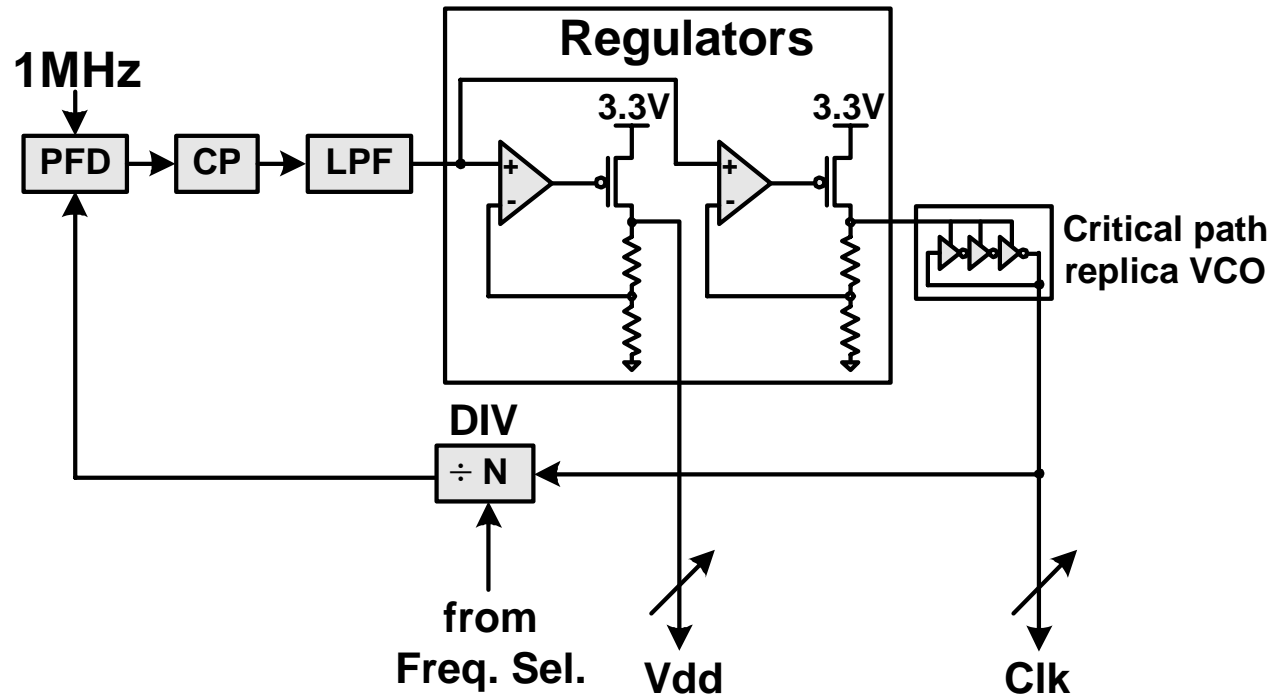
- FLP operations w/o final FLP additions
 - Log and Anti-log conversions cancel each other
 - Forward intermediate log value to next instruction
 - Pipeline latency and computation error reductions

Overall GPU Architecture



- Triple-domain Power Management
 - Different workloads for modules in 3D graphics pipeline
 - Separate domain with DVFS
 - Managed according to different workloads
- Power Management Loop
 - Workload measured from FIFO occupation level is compared with reference point
 - New target frequency is selected and PMU relocks to new freq. and vdd

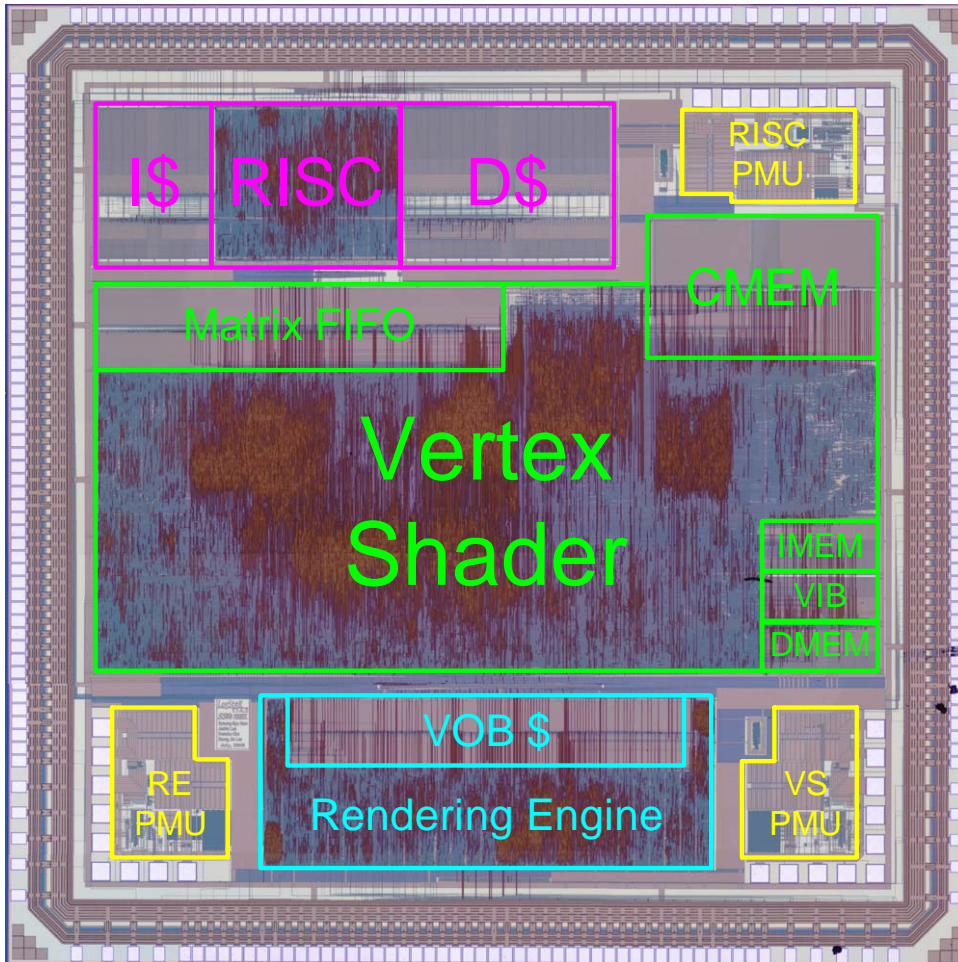
Power Management Unit



- Power Management Unit

- Embeds linear regulator in PLL to sync. the freq. & vdd scaling
- Separate regulators for digital logic and VCO for noise isolation
- VCO models critical path in target domain
- PMU loop: new target freq \rightarrow new ctrl. vtg \rightarrow new vdd \rightarrow new clk

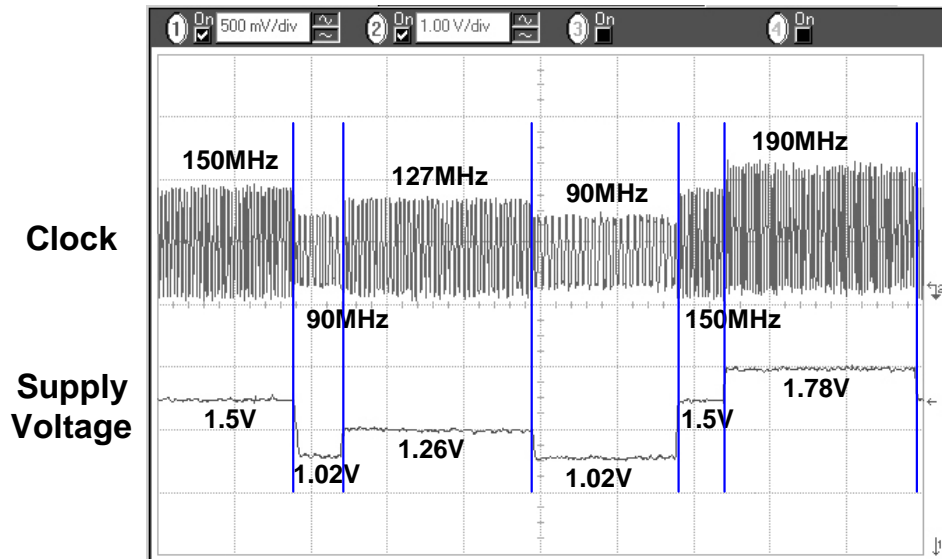
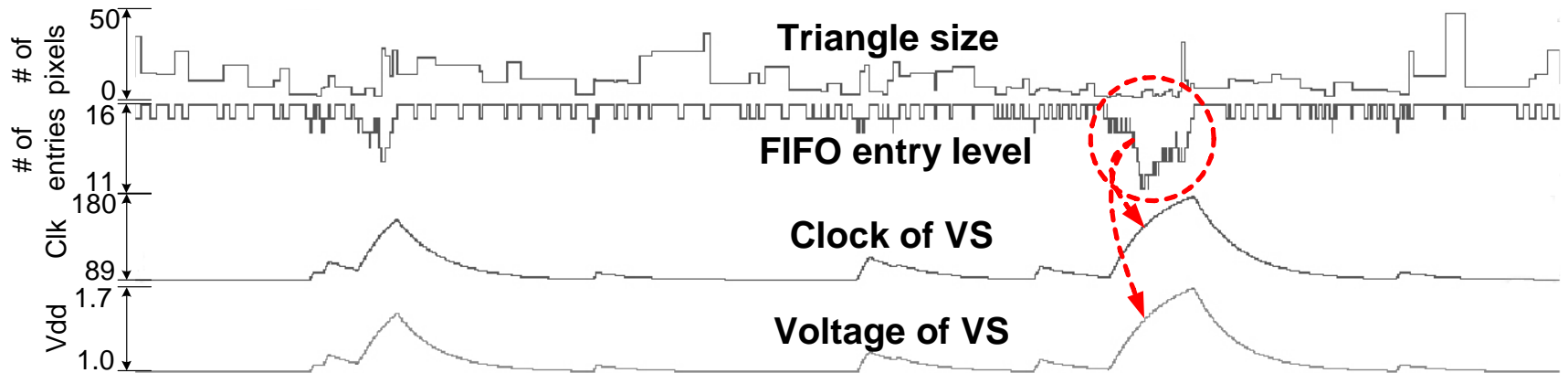
Chip Implementation



Chip Characteristics

Technology	0.18um 1-P 6-M CMOS
Area	17.2mm ²
Power supply	Core: 1.0V - 1.8V IO: 3.3V
Frequency	89MHz – 200MHz
Transistors	1.6M logic, 29KB SRAM
Processing speed	141Mvertices/s
Power consumption	52.4mW @ 60fps 153mW @ full speed

Measurement Results



Comparisons

	Processing speed (vertices/s)	Power consumption (mw@full speed, mw@60fps)	Area (mm ²)	Technology (nm)	Functions
Mitsubishi [GH03]	185K	38 / N.A.	11	180	GE+RE
Hitachi [ISSCC04]	36M	250 / N.A.	N.A.	130	GE only
SONY [HotChips04]	35M	< 500	N.A.	90	GE+RE
C.-H. Yu [ISSCC06]	120M	157 / 106	16	180	GE only
This Work	141M	153 / 52.4	17.2	180	GE+RE

Summary

- **Low-Power GPU for Handheld Devices**
 - 17.5% faster and consumes 50.5% less power than previous works
- **Logarithmic Multifunction Unit**
 - Unifies matrix, vector, and elementary functions on a single arithmetic unit
 - Novel vector processor architecture for vertex shader
 - 3D geometry transformation in every 2 cycles
 - 141Mvertices/s peak processing speed
- **Power Management**
 - 3 separate power domains with DVFS
 - 52.4mW power consumption @ 60fps