HPG 2011

HIGH PERFORMANCE GRAPHICS HOT 3D

AMD GRAPHIC CORE NEXT

Low Power High Performance Graphics & Parallel Compute

Michael Mantor
AMD Senior Fellow Architect
Michael.mantor@amd.com

Mike Houston
AMD Fellow Architect
michael.houston@amd.com

At the heart of every AMD APU/GPU is a power aware high performance set of compute units that have been advancing to bring users new levels of programmability, precision and performance.

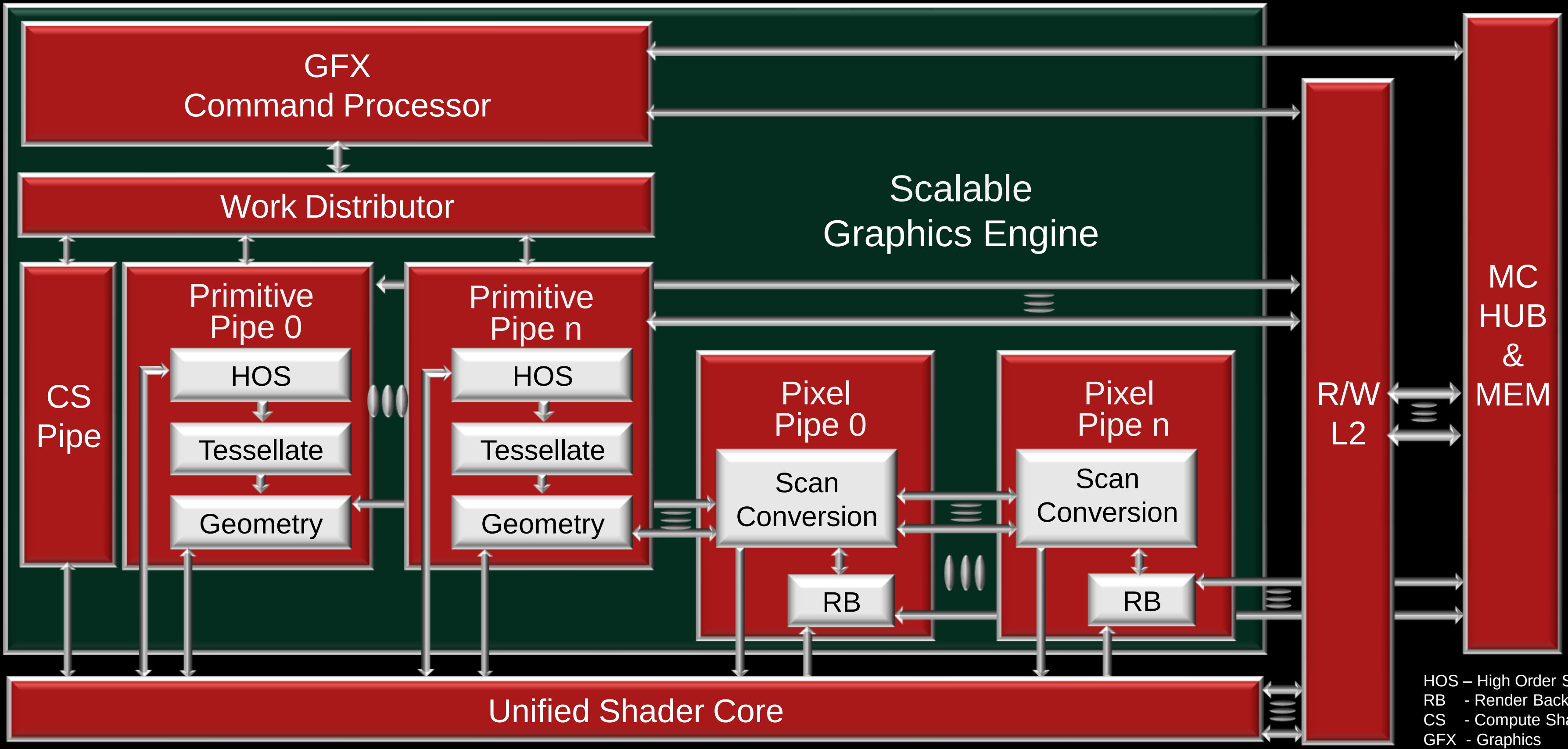
AGENDA → AMD Graphic Core Next Architecture

- Unified Scalable Graphic Processing Unit (GPU) optimized for Graphics and Compute
  - Multiple Engine Architecture with Multi-Task Capabilities
  - Compute Unit Architecture
  - Multi-Level R/W Cache Architecture

- What will not be discussed
  - Roadmaps/Schedules
  - New Product Configurations
  - Feature Rollout

- Visit AMD Fusion Developers Summit online for Fusion System Architecture details
  - http://developer.amd.com/afds/pages/session.aspx

SCALABLE MULTI-TASK GRAPHICS ENGINE



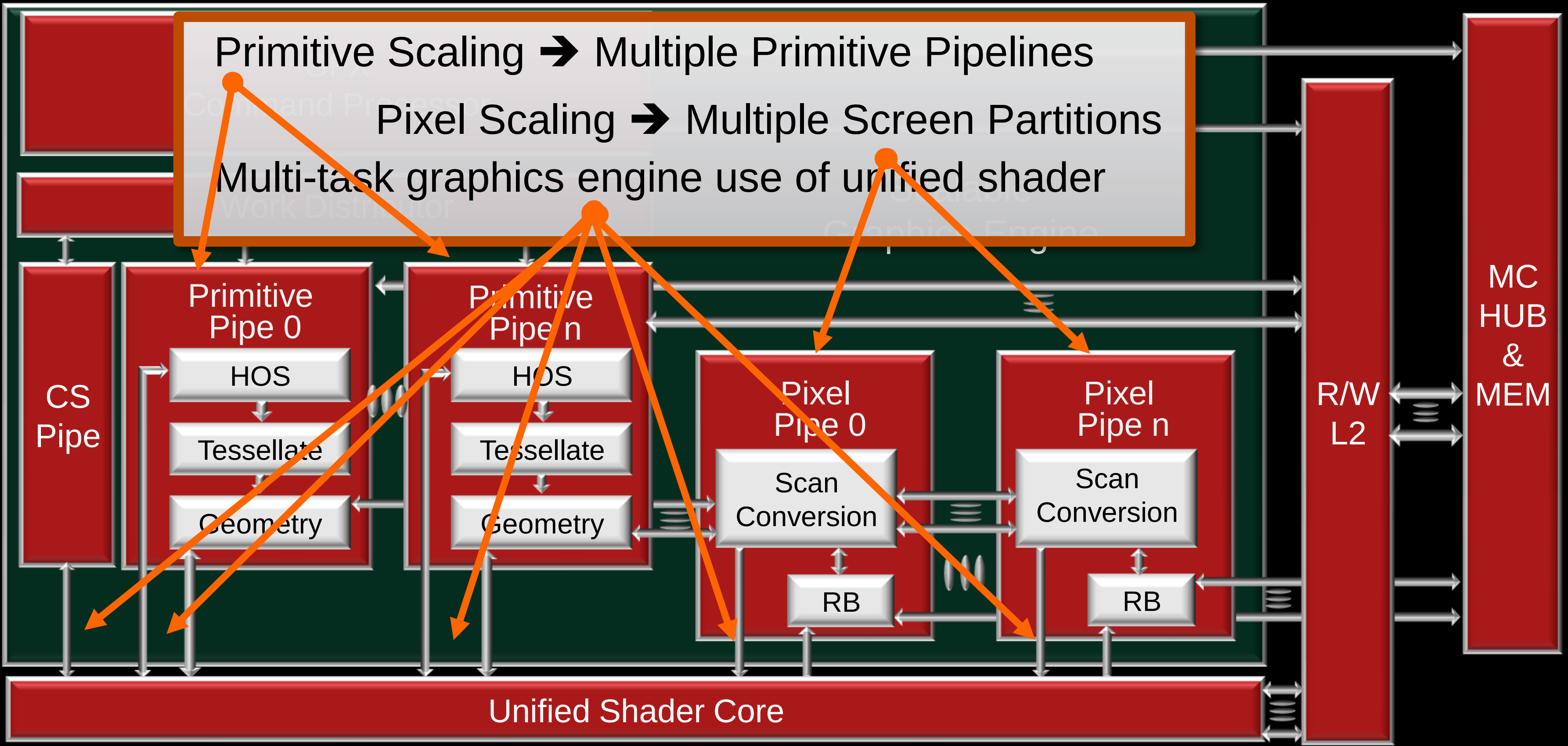
HOS	– High Order Surface
RB	- Render Backend
CS	- Compute Shader
GFX	- Graphics

SCALABLE MULTI-TASK GRAPHICS ENGINE

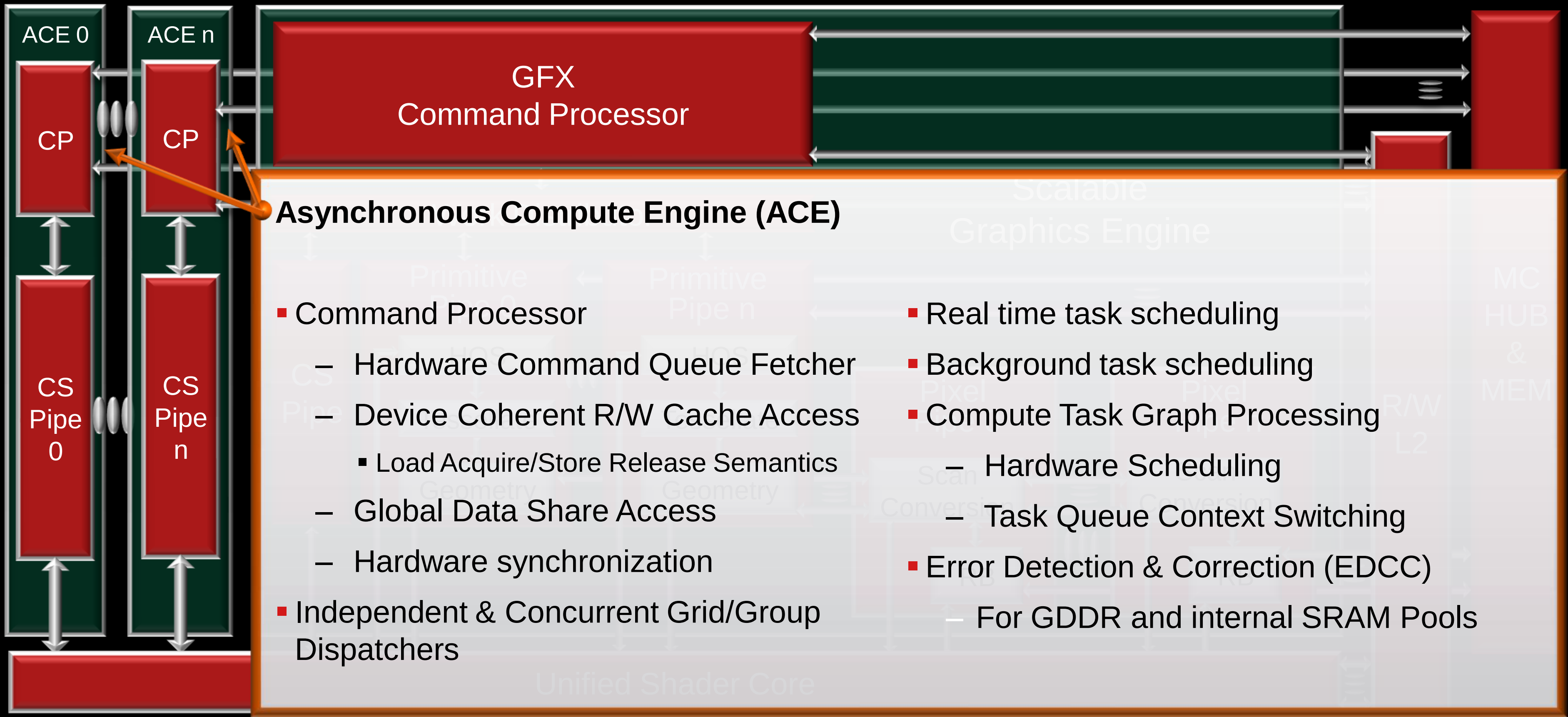
Primitive Scaling ➔ Multiple Primitive Pipelines

Pixel Scaling ➔ Multiple Screen Partitions

Multi-task graphics engine use of unified shader



MULTI-ENGINE UNIFIED COMPUTING GPU



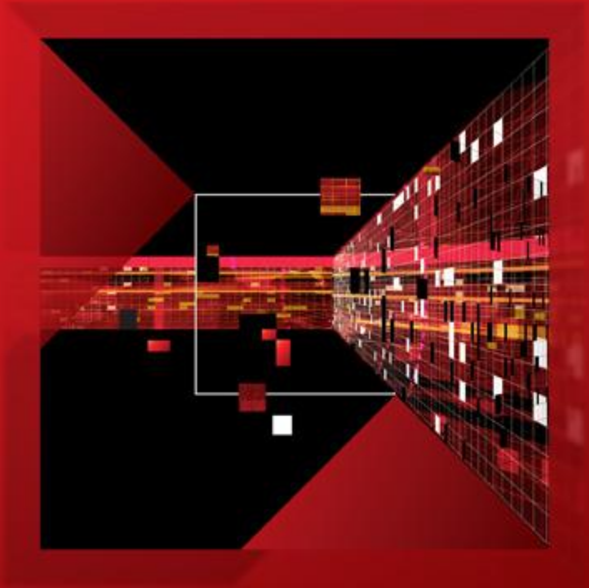
Asynchronous Compute Engine (ACE)

- Command Processor
  - Hardware Command Queue Fetcher
  - Device Coherent R/W Cache Access
    - Load Acquire/Store Release Semantics
  - Global Data Share Access
  - Hardware synchronization
- Independent & Concurrent Grid/Group Dispatchers

- Real time task scheduling
- Background task scheduling
- Compute Task Graph Processing
  - Hardware Scheduling
  - Task Queue Context Switching
- Error Detection & Correction (EDCC)
  - For GDDR and internal SRAM Pools

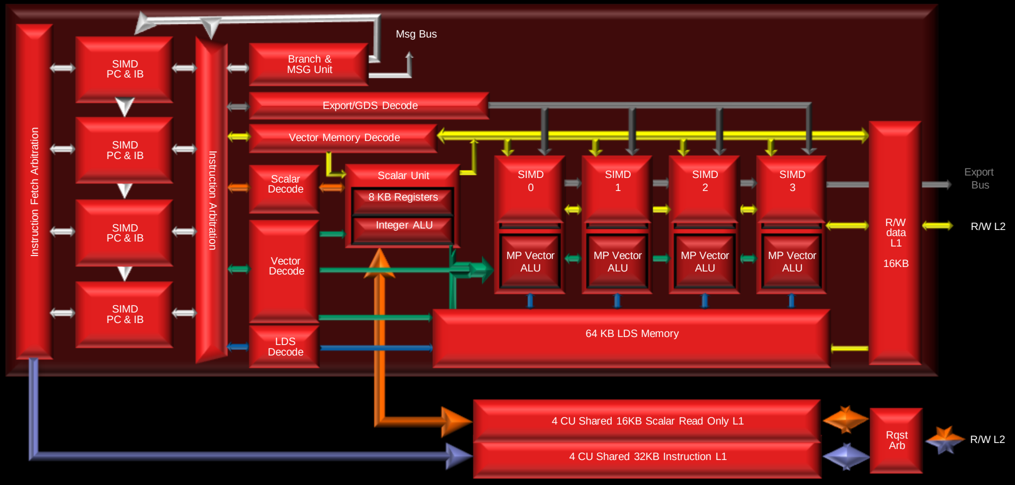
AMD GRAPHIC CORE NEXT

COMPUTE UNIT ARCHITECTURE

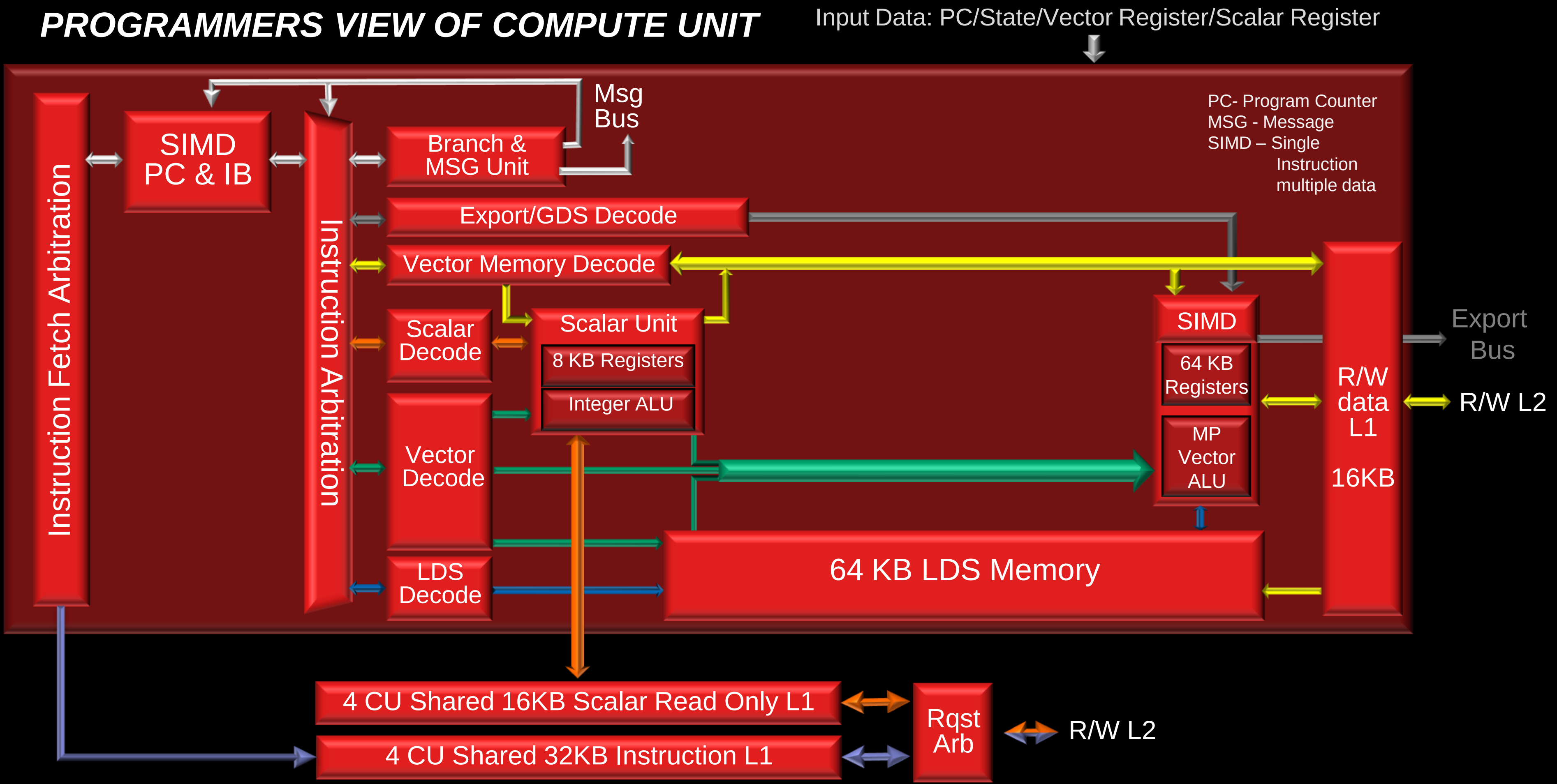


NON-VLIW ISA WITH SCALAR + VECTOR UNITS

- Simpler ISA compared to previous generation
  - No clauses
  - No VLIW packing
  - Control flow more directly programmed
- Advanced language feature support
  - Exception support
  - Function calls
  - Recursion
- Enhanced extended ALU operations
  - Media ops
  - Integer ops
  - Floating point atomics (min, max, cmpxchg)
- Improved debug support
  - HW functionality to improve debug support



PROGRAMMERS VIEW OF COMPUTE UNIT



SOME CODE EXAMPLES (1)

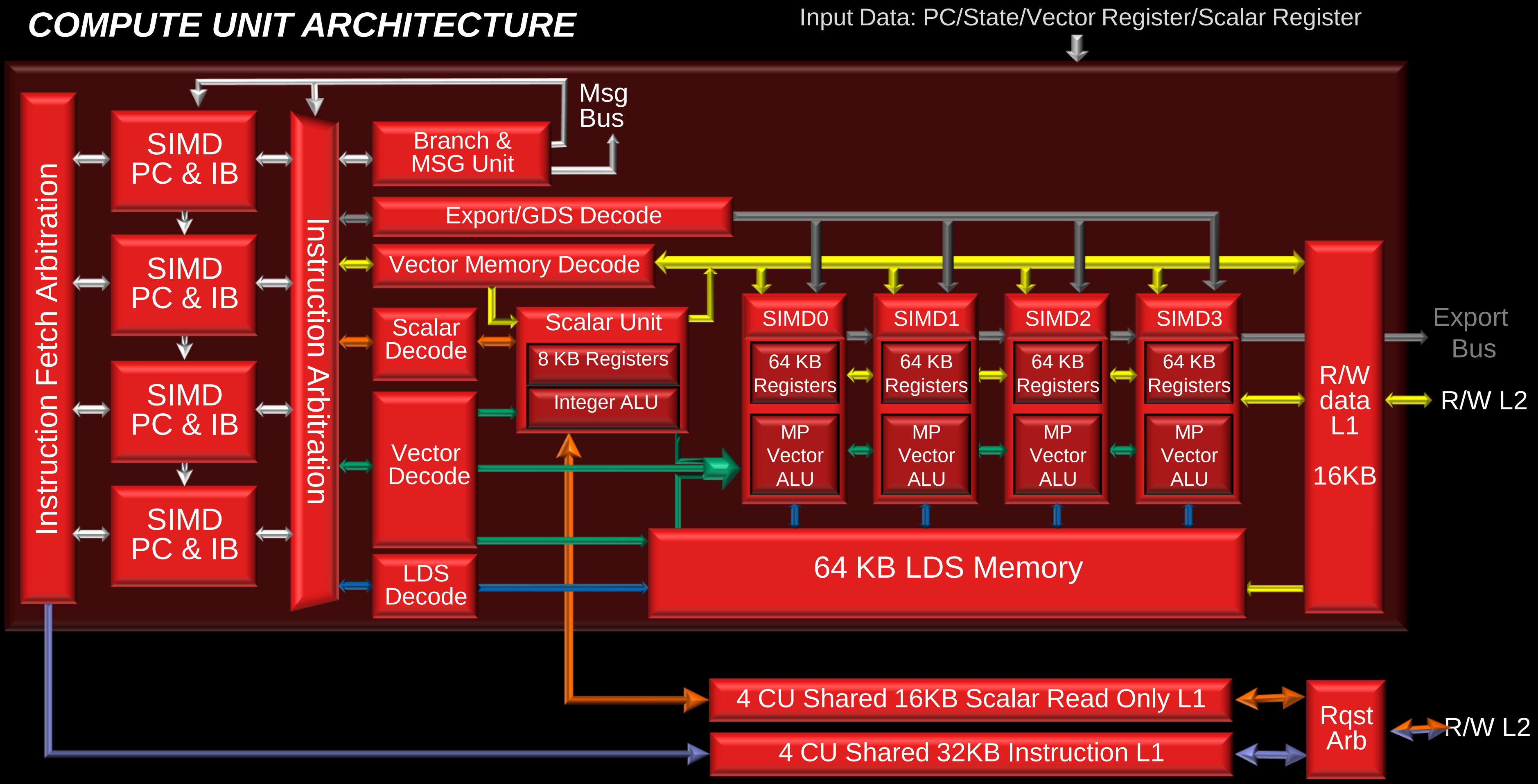
```
float fn0(float a,float b)
{
   if(a>b)
      return((a-b)*a);
   else
      return((b-a)*b
```

```
//Registers r0 contains “a”, r1 contains “b”
//Value is returned in r2

   v_cmp_gt_f32      r0,r1            //a > b, establish VCC
   s_mov_b64         s0,exec          //Save current exec mask
   s_and_b64         exec,vcc,exec    //Do “if”
   s_cbranch_vccz    label0           //Branch if all lanes fail
   v_sub_f32         r2,r0,r1         //result = a - b
   v_mul_f32         r2,r2,r0         //result=result * a

label0:
   s_andn2_b64       exec,s0,exec     //Do “else”(s0 & !exec)
   s_cbranch_execz   label1           //Branch if all lanes fail
   v_sub_f32         r2,r1,r0         //result = b - a
   v_mul_f32         r2,r2,r1         //result = result * b
label1:
   s_mov_b64         exec,s0          //Restore exec mask
```

Optional:
Use based on the number of instruction in conditional section.

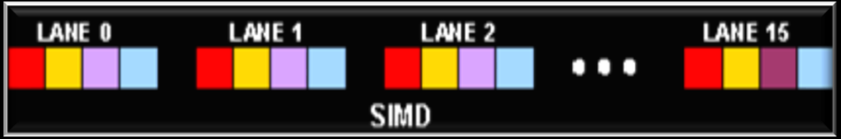
- Executed in branch unit

COMPUTE UNIT ARCHITECTURE

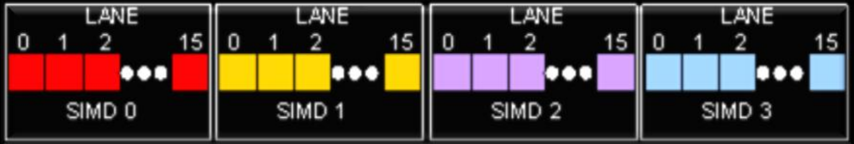


NON-VERY LONG INSTRUCTION WORD (VLIW) VECTOR ENGINES

4 WAY VLIW SIMD



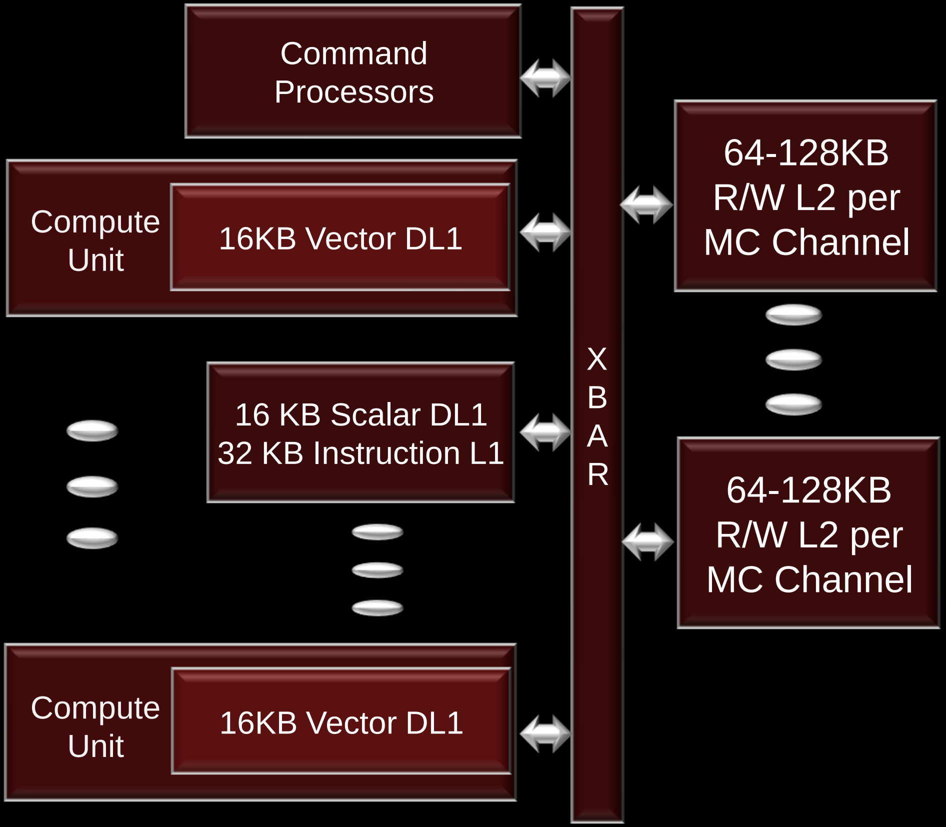
4 Non-VLIW SIMD



4 Way VLIW SIMD	4 SIMD non-VLIW
64 Single Precision MAC	64 Single Precision MAC
VGPR ➔ 64 * 4 * 256-32bit ➔ 256KB	VGPR ➔ 4 * 64 * 256-32bit ➔ 256KB
1 VLIW Instruction * 4 Ops ➔ Dependencies limitations	4SIMD * 1 ALU Operation ➔ Occupancy limitations
3 SRC GPRs, 1 Vector Destination	3 SRC GPRs, 1 Vector\1Scalar Register Destination
Compiler manage VGPR port conflicts	No VGPR port conflicts
VALU Instruction Bandwidth ➔ 1-7 dwords(~2 dwords/clk)	VALU Instruction Bandwidth ➔ 1-2 dwords/cycle
Interleaved wavefront instruction required	Vector back-to-back wavefront instruction issue
Specialized complicated compiler scheduling	Standard compiler scheduling & optimizations
Difficult assembly creation, analysis, & debug	Simplified assembly creation, analysis, & debug
Complicated tool chain support	Simplified tool chain development and support
Less predictive results and performance	Stable and predictive results and performance

R/W CACHE

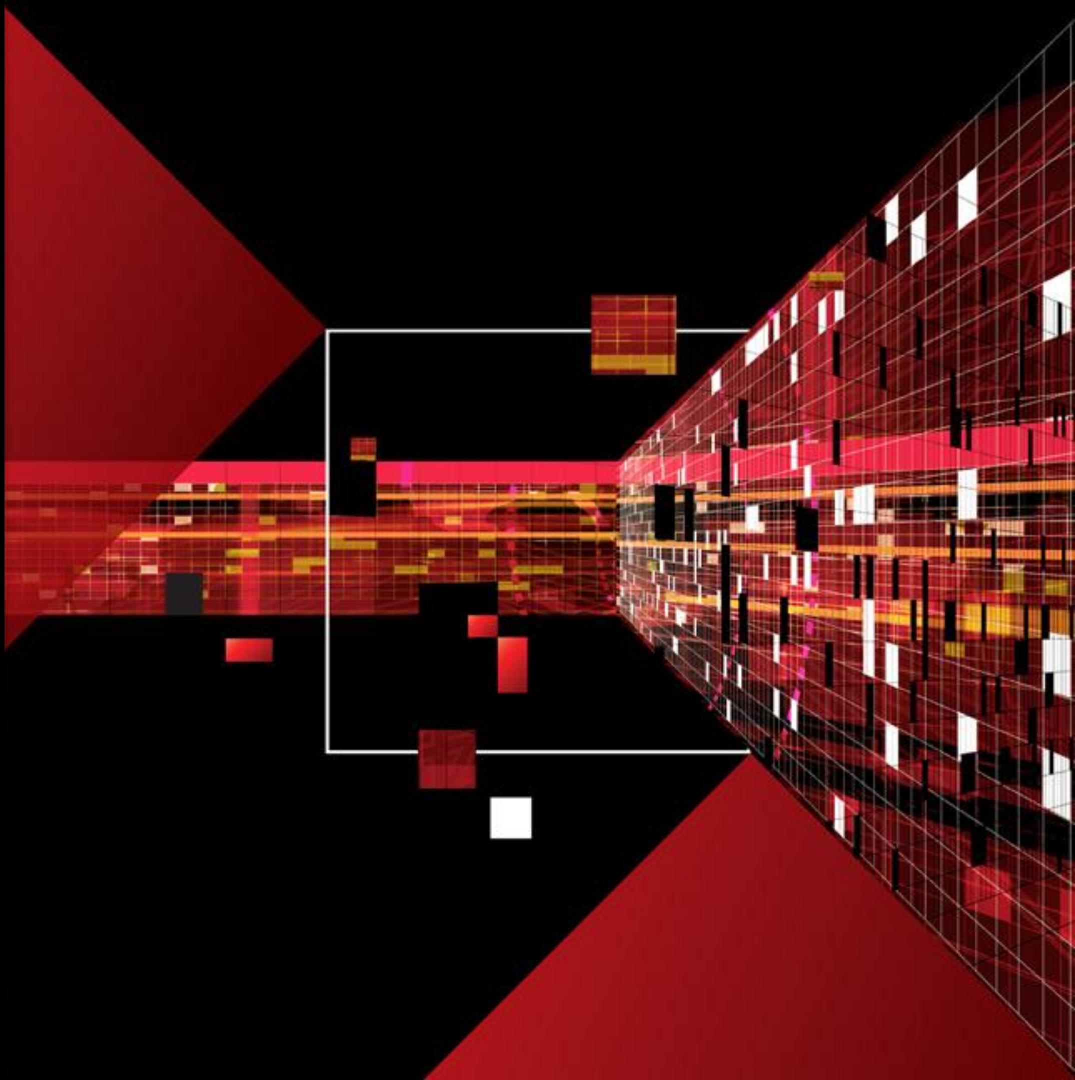
- Read / Write Data cached
  - Bandwidth amplification
  - Improved behavior on more memory access patterns
  - Improved write to read reuse performance
  - L1 Write-through / L2 write-back caches
- Relaxed memory model
  - Consistency controls available for locality of load/store/atomic
- GPU Coherent
  - Acquire / Release semantics control data visibility across the machine
  - L2 coherent = all CUs can have the same view of data
- Remote Global atomics
  - Performed in L2 cache



AMD Graphic Core Next Compute Unit Architecture Summary

- A heavily multi-threaded Compute Unit (CU) architected for throughput
  - Efficiently balanced for graphics and general compute
  - Simplified coding for performance, debug and analysis
  - Simplified machine view for tool chain development
  - Low latency flexible control flow operations
  - Read/Write Cache Hierarchy improves I/O characteristics
  - Flexible vector load, store, and remote atomic operations
  - Load acquire / Store release consistency controls

QUESTIONS ?



Disclaimer & Attribution