AMD

Entering the Golden Age of Heterogeneous Computing

Michael Mantor
Senior GPU Compute Architect / Fellow
AMD Graphics Product Group

michael.mantor@amd.com

The 4 Pillars of massively parallel compute offload

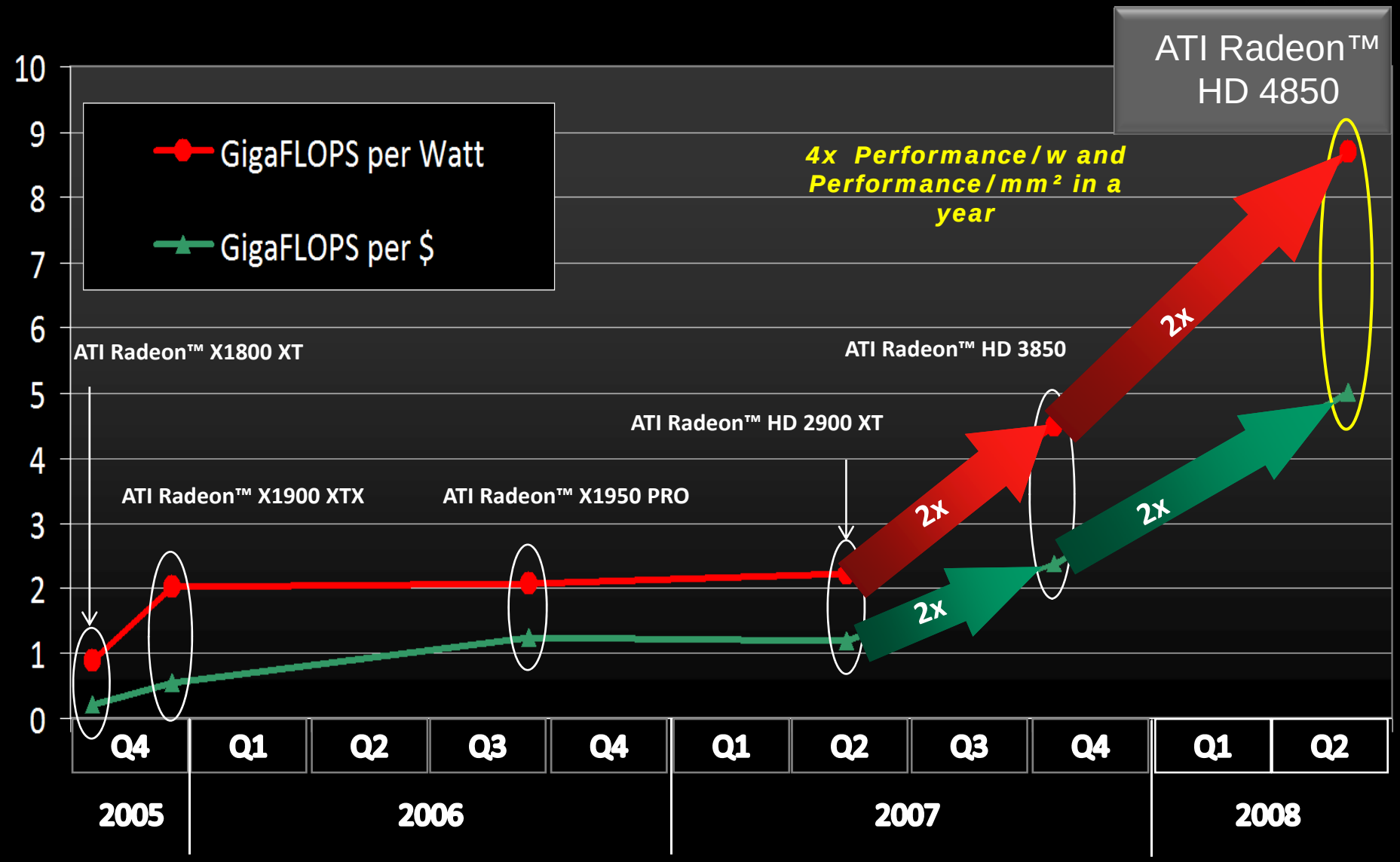
AMD
Smarter Choice

- Performance
- Power
- Price
- Programming Models

Moore's Law ➔ 2x < 18 Months
Frequency\Power\Complexity Wall
Parallel ➔ Opportunity for growth

GPU is the first successful massively parallel COMMODITY architecture with a programming model that managed to tame 1000's of parallel threads in hardware to perform useful work efficiently

Quick recap of where we are – Perf, Power, Price

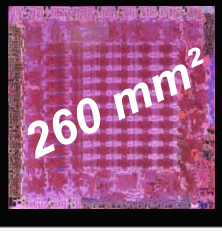


Source of GigaFLOPS per watt: maximum theoretical performance divided by maximum board power.
Source of GigaFLOPS per $: maximum theoretical performance divided by price as reported on www.buy.com as of 9/24/08

ATI Radeon™HD 4850

AMD Smarter Choice

Designed to Perform in Single Slot



SP Compute Power	1.0 T-FLOPS
DP Compute Power	200 G-FLOPS
Core Clock Speed	625 Mhz
Stream Processors	800
Memory Type	GDDR3
Memory Capacity	512 MB
Max Board Power	110W
Memory Bandwidth	64 GB/Sec

ATI Radeon™HD 4870

AMD Smarter Choice

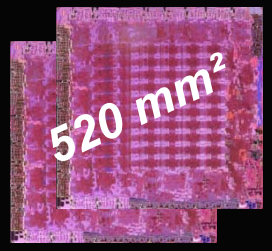
First Graphics with GDDR5

SP Compute Power	1.2 T-FLOPS
DP Compute Power	240 G-FLOPS
Core Clock Speed	750 Mhz
Stream Processors	800
Memory Type	GDDR5 3.6Gbps
Memory Capacity	512 MB
Max Board Power	160 W
Memory Bandwidth	115.2 GB/Sec

ATI Radeon™HD 4870 X2

AMD Smarter Choice

Incredible Balance of Performance, Power, Price



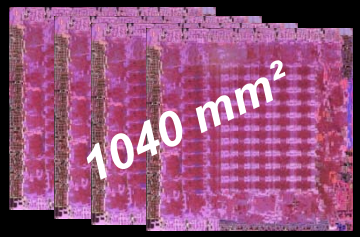
Compute Power	2.4 TFLOPS
DP Compute Power	240 G-FLOPS
Core Clock Speed	750 Mhz
Stream Processors	1600
Memory Type	GDDR5 3.6Gbps
Memory Capacity	2 GB
Max Board Power	285W
Memory Bandwidth	230 GB/Sec

AMD FireStream™ 9250

AMD
Smarter Choice

- AMD's Second Generation Stream Computing Product
- Single PCI Slot
- Computational Power
  - One T-FLOPS Single Precision Float
  - 200 GFLOPS Double Precision
- 1 GB GDDR3
- 150 Watts → 8GFLOPS/Watt
- Familiar 32 and 64 bit Linux® and Windows® Environment
- Stream software supports multiple GPUs per system
- Brook+ (Open Source C-level language & Compiler)
  - GPU Shader Analyzer
  - AMD Code Analyst
- AMD's Compute Abstraction Layer (CAL)

Crossfire FragBox QuadFire @ Falcon Northwest



- QuadFire 2x ATI Radeon™ 4870X2 2 GB
- 3200 Single Precision Stream Processer
- Or 160 DP units with 160 SP units

AMD
Smarter Choice

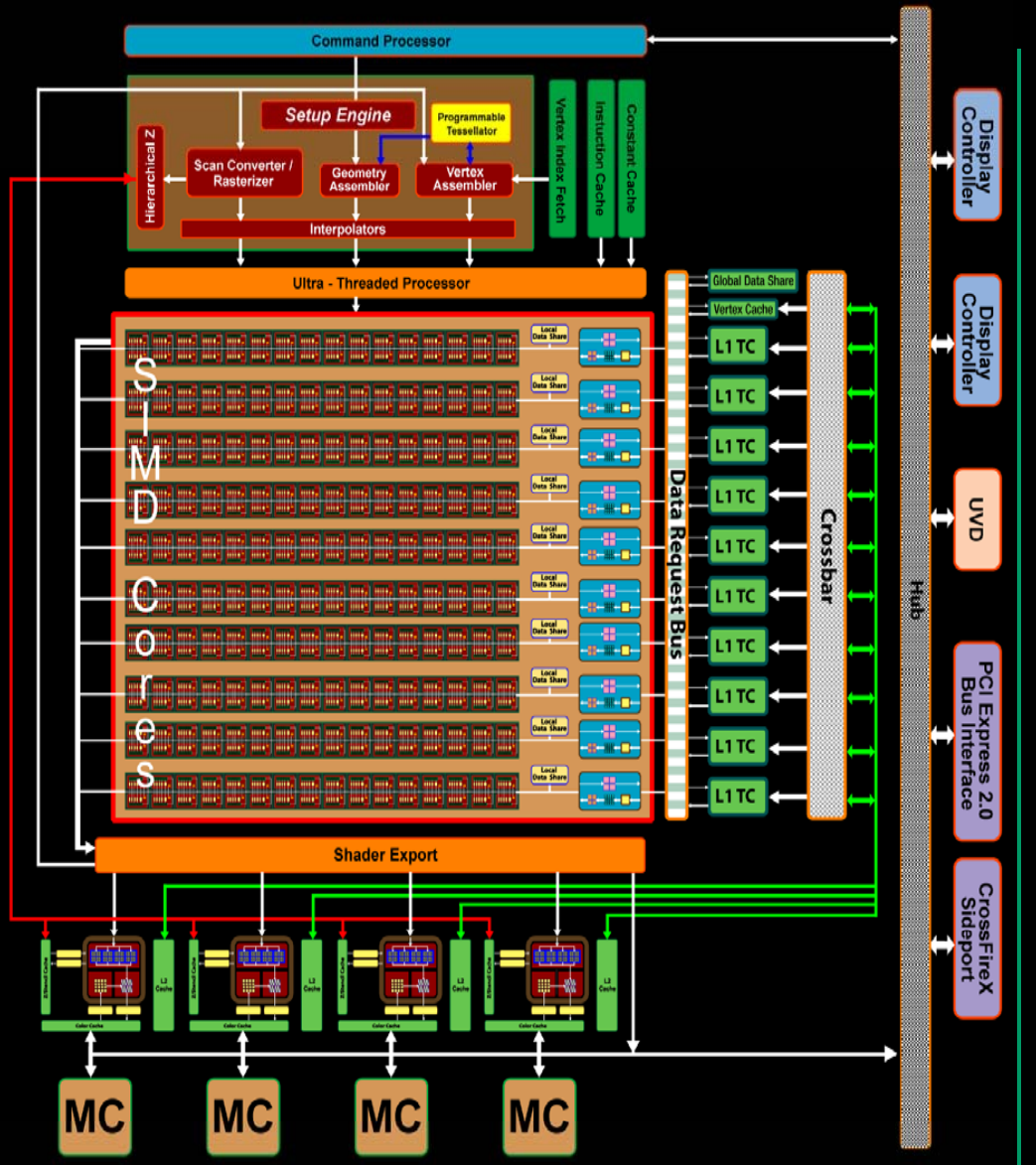
Rapid Advances for Compute Today !!

Meet the ATI Radeon™ HD48xx Architecture

Terascale Unified Graphics Engine

AMD Smarter Choice

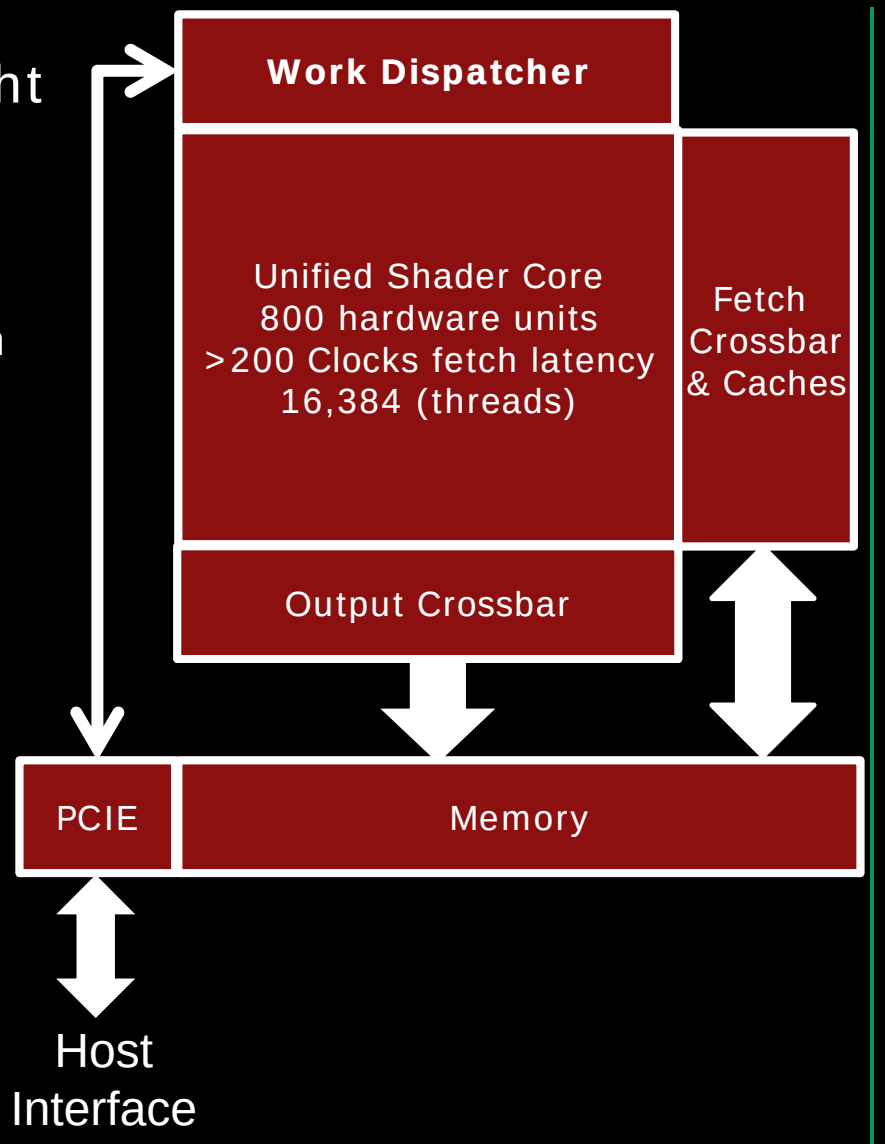
- 800 highly optimized stream processing units
- New Unified SIMD core layout
- Optimized texture units
- New texture cache design
- New memory architecture
- Optimized render back-ends for fast anti-aliasing performance
- Enhanced geometry shader & tessellator performance



Simple View of a Unified Data Parallel Throughput Machine

AMD Smarter Choice

- Highly threaded to hide latency for light ALU loads and I/O constrained Apps
  - Hardware has support for ~16,000 shader program invocations
  - Hardware has register & resources for each invocation
- I/O bound with low ALU count
  - 16 arrive per clock,
  - 16 leave per clock
  - 40 threads issue fetch per clock
  - Fetch latency
    - 300 clks latency ➔ 4,800 threads
    - 1 dependant fetch ➔ 9,600 threads
    - 2 dependant fetch ➔ 14,400 threads

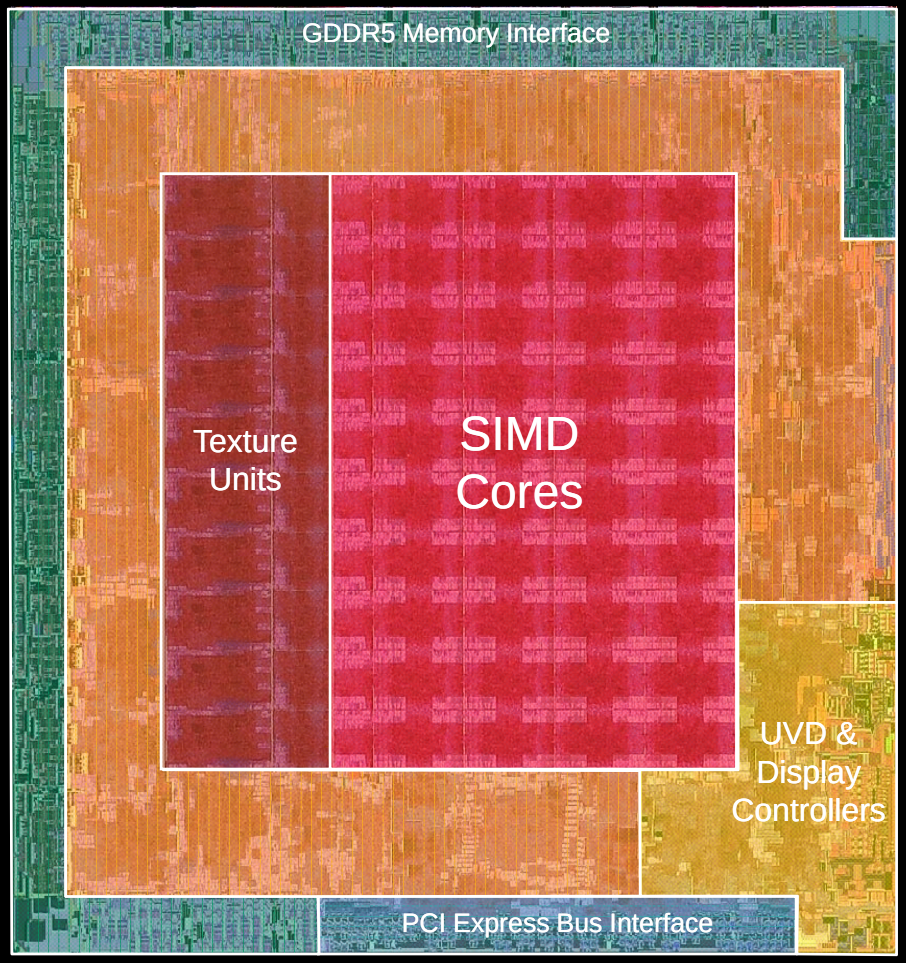


ATI Radeon™ HD 4800 Series Architecture

AMD Smarter Choice

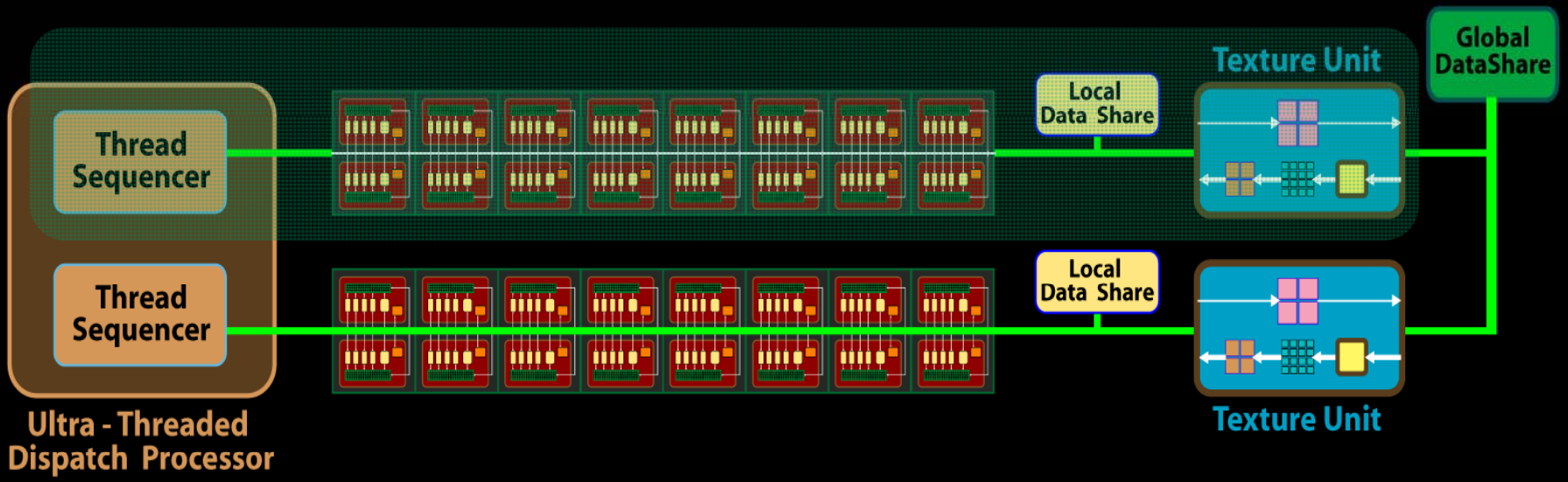
- 10 SIMD cores
  - Each with 80 32-bit Stream Processing Units (800 total)
  - Dual Identity – 16 64-bit & 16 32-bit Stream Processing Units
- 40 Texture Units
- 115+ GB/sec GDDR5 memory interface

	ATI Radeon™ HD 3870	ATI Radeon™ HD 4870	Difference
Die Size	190 mm^2	260 mm^2	1.4x
Memory	72 GB/sec	115 GB/sec	1.6x
AA Resolve	32	64	2x
Z/Stencil	32	64	2x
Texture	16	40	2.5x
Shader	320	800	2.5x



SIMD Cores

- Each core:
  - Includes 80 scalar stream processing units in total + 16KB Local Data Share
  - Has its own control logic and runs from a shared set of threads
  - Has 4 dedicated texture units + L1 cache
  - Communicates with other SIMD cores via 16KB global data share
- New design allows texture fetch capability to scale efficiently with shader power, maintaining 4:1 ALU:TEX ratio

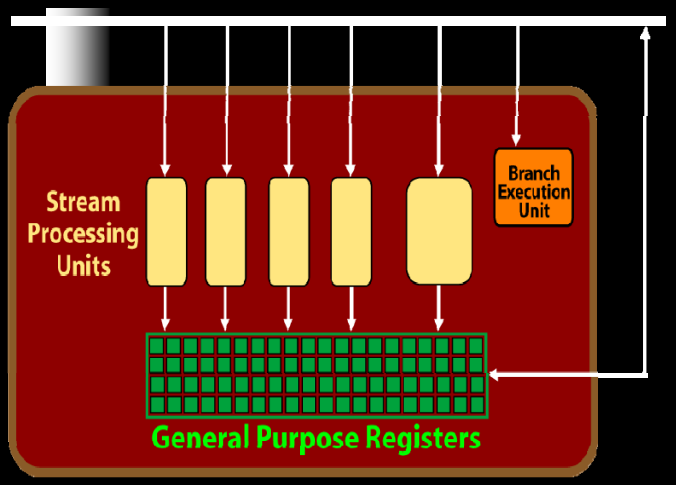


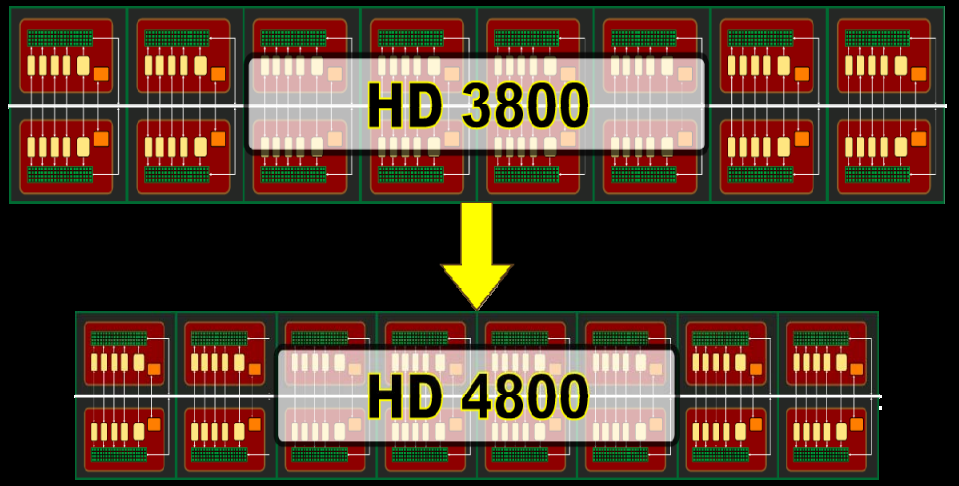
Stream Processing Units

AMD
Smarter Choice

- 40% increase in performance per mm²*
- More aggressive clock gating for improved Performance per Watt
- Fast double precision processing (240 GigaFLOPS)
- Integer bit shift operations for all units

  (12.5x improvement *)

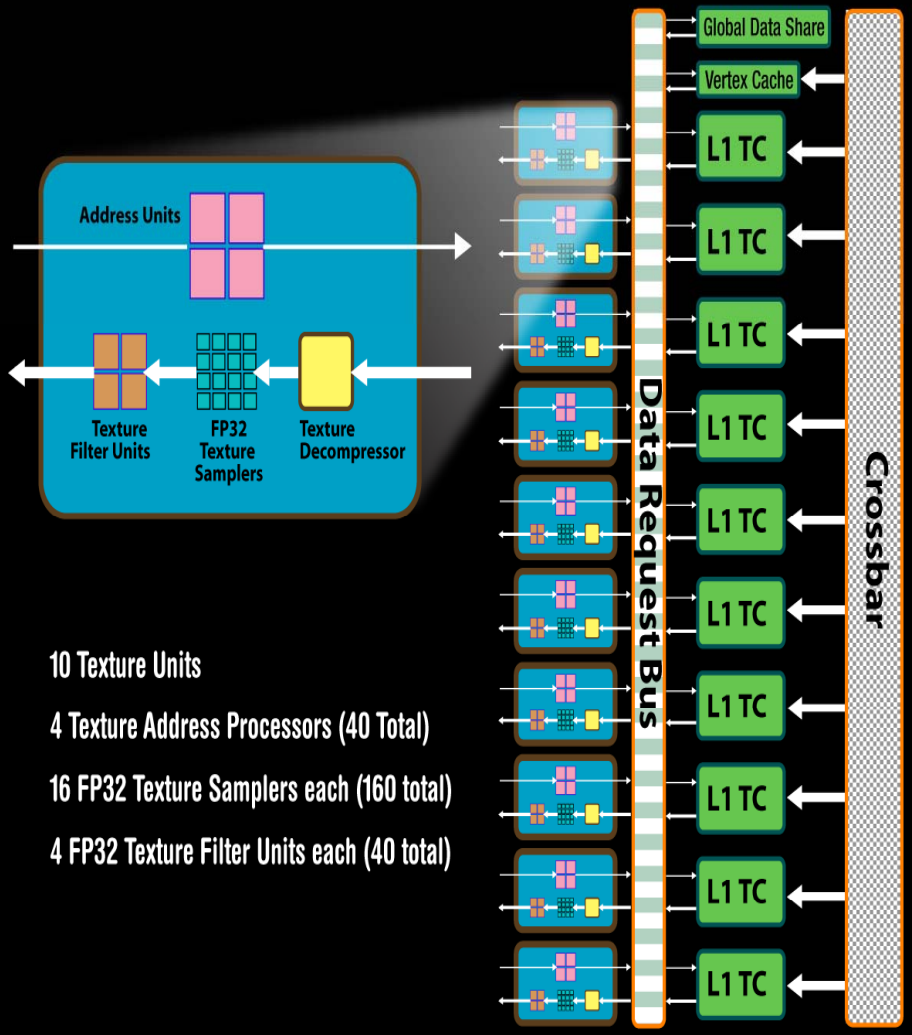


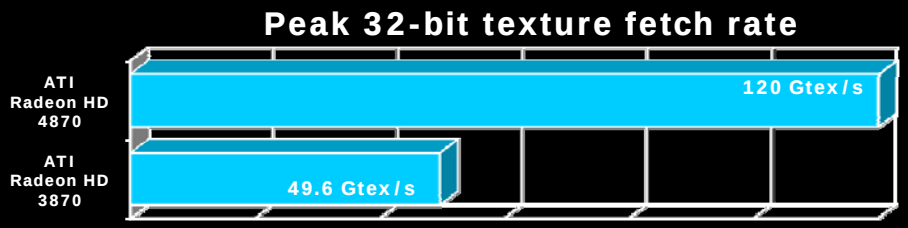


Source of performance per mm²: maximum theoretical performance divided by surface area of aSIMD
Source of 12.5x improvement: for ATI Radeon™ HD 3800 Series 4 SIMDs x 16 shift operations per SIMD = 64 shift operations.
For ATI Radeon 4800 Series 10 SIMDs x 80 shift operations = 800 shift operations.

Texture Units

- Streamlined design
  - 70% increase in performance/mm^2 *
- More performance
  - Double the texture cache bandwidth of the ATI Radeon™ HD 3800 SERIES
  - 2.5x increase in 32-bit filter rate
  - 1.25x increase in 64-bit filter rate
  - Up to 160 fetches per clock





Source of performance per mm^2: maximum theoretical performance divided by surface area of the texture units

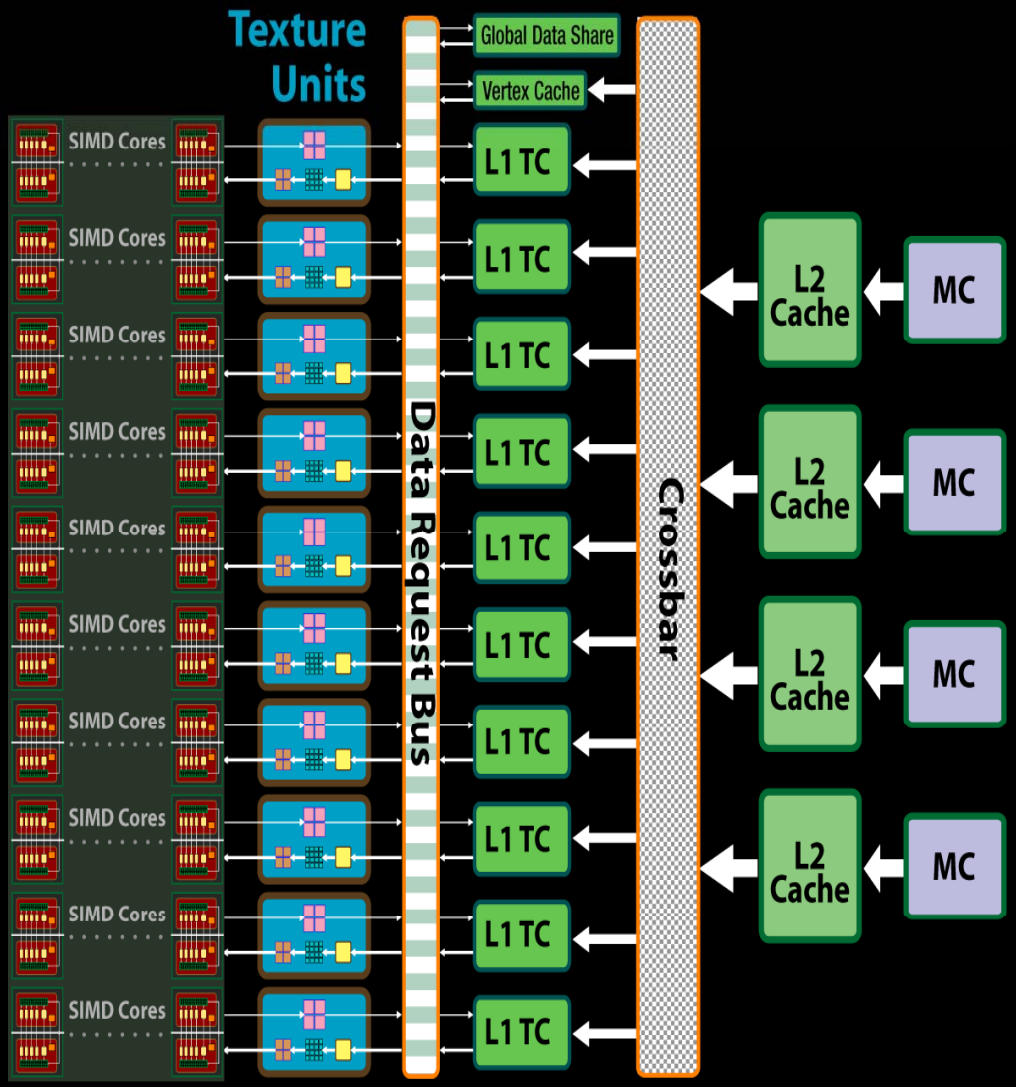
Texture Units

- New cache design
  - L2s aligned with memory channels
  - L1s store unique data per SIMD

    2.5x increase aggregate L1*
  - Separate vertex cache
  - Increased bandwidth

    Up to 480 GB/sec of L1 texture fetch bandwidth

    Up to 384 GB/sec between L1 & L2

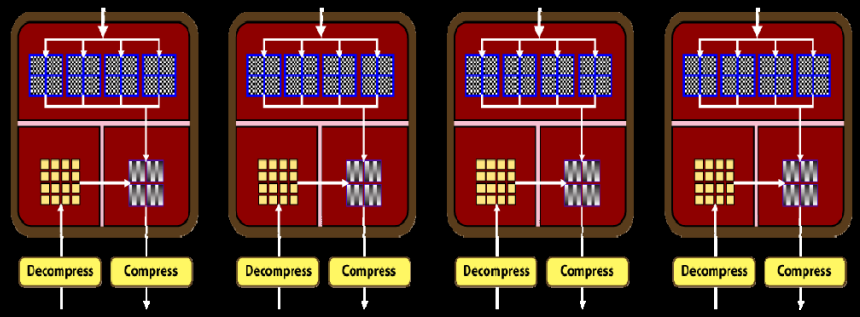


* Comparing ATI Radeon™ HD 4800 series and ATI Radeon™ HD 3800 series

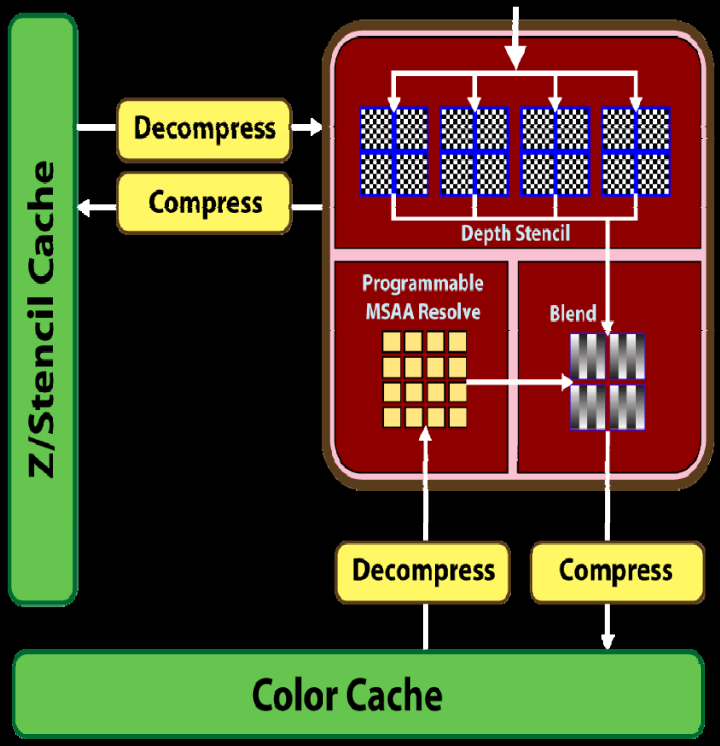
Render Back-Ends

AMD Smarter Choice

- Focus on improving AA performance per mm^2
  - Doubled peak rate for depth/stencil ops to 64 per clock
  - Doubled AA peak fill rate for 32-bit & 64-bit color *
  - Doubled non-AA peak fill rate for 64-bit color
- Supports both fixed function (MSAA) and programmable (CFAA) modes



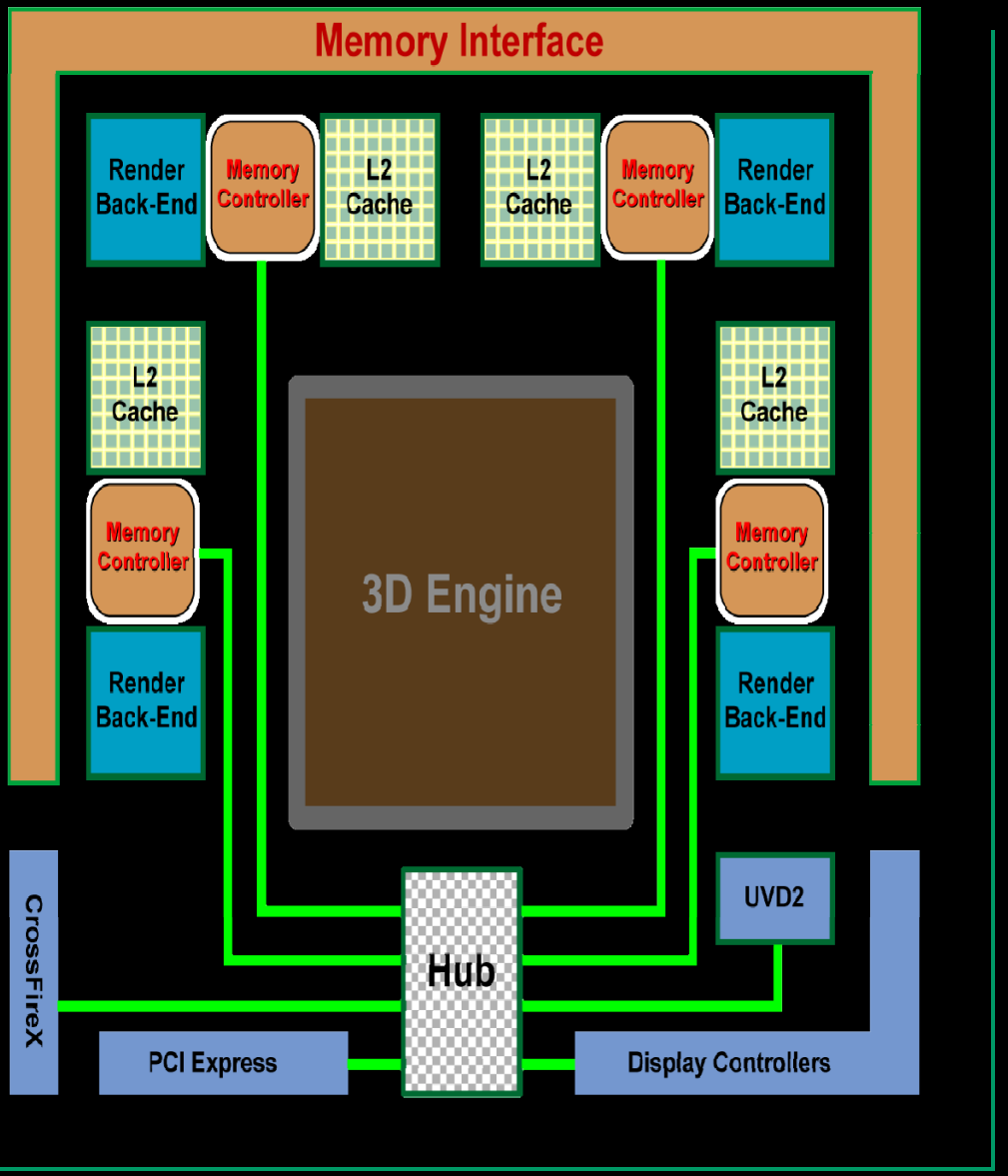
	Color	ATI Radeon™ HD 3800 series	ATI Radeon™ HD 4800 series	Difference
No MSAA	32-bit	16 pix/clk	16 pix/clk	1x
2x / 4x MSAA		8 pix/clk	16 pix/clk	2x
8x MSAA		4 pix/clk	8 pix/clk	2x
No MSAA	64-bit	8 pix/clk	16 pix/clk	2x
2x / 4x MSAA		8 pix/clk	16 pix/clk	2x
8x MSAA		4 pix/clk	8 pix/clk	2x
Depth / stencil only		32 pix/clk	64 pix/clk	2x



* Comparing ATI Radeon™ HD 4800 series and ATI Radeon™ HD 3800 series

Memory Controller Architecture

- New distributed design with hub
- Controllers distributed around periphery of chip, adjacent to primary bandwidth consumers
- Memory tiling & 256-bit interface allows reduced latency, silicon area, and power consumption
- Hub handles relatively low bandwidth traffic
  - PCI Express®, CrossFireX™ interconnect, UVD2, display controllers, intercommunication)



ATI Radeon™ HD 4870 Computation Highlights

AMD Smarter Choice

- >100 GB/s memory bandwidth
  - 256b GDDR5 interface
- Targeted for handling thousands of simultaneous lightweight threads
- 800 (160x5) stream processors
  - 640 (160x4) basic units (FMAC, ADD/SUB, CMP, etc.)
    - ~1.2 TFlops theoretical peak
  - 160 enhanced transcendental units (adds COS, LOG, EXP, RSQ, etc.)
  - Support for INT/UINT in all units (ADD/SUB, AND, XOR, NOT, OR, etc.)
  - 64-bit double precision FP support
    - 1/5 single precision rate (~240GFlops theoretical performance)

4 SIMDs -> 10 SIMDs

- 2.5X peak theoretical performance increase over ATI Radeon™ 3870
- ~1.2 TFlops FP32 theoretical peak
- ~240 GFlops FP64 theoretical peak

Scratch-pad memories

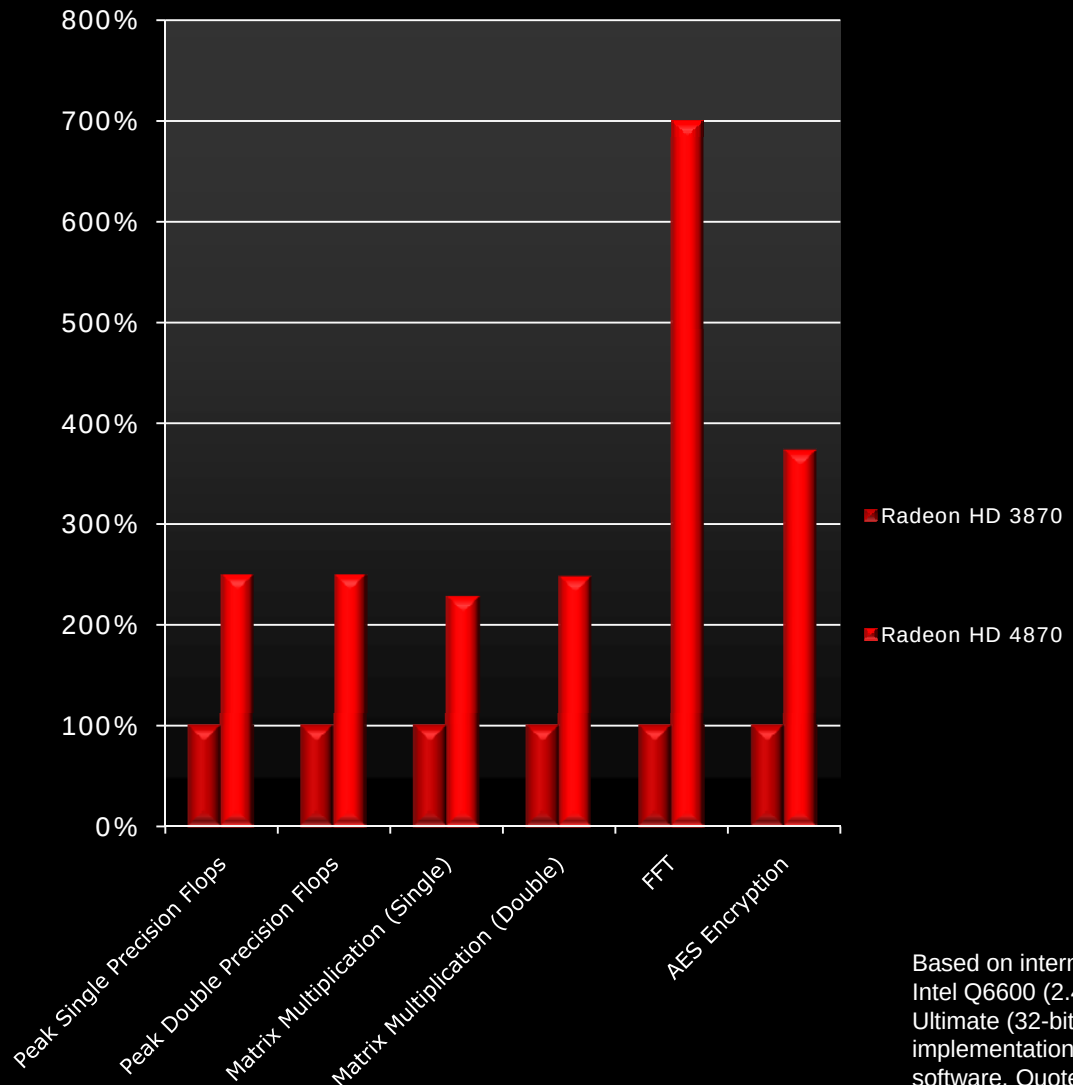
- 16KB per SIMD (LDS)
- 16KB across SIMDs (GDS)

Synchronization capabilities

Compute Shader

- Launch work without rasterization
- "Linear" scheduling
- Faster thread launch

Performance



- AMD Core Math Library (3870)

  SGEMM - 300 GFLOPS

  DGEMM - 137 GFLOPS

- FFT

  305 GFLOPS (Mercury Computer Systems, Inc.)

Based on internal testing at AMD Performance Labs on reference platform. **Configuration**: Intel Q6600 (2.4 GHz), 2GB DDR3, ASUS P5E3 Deluxe motherboard, Windows Vista® Ultimate (32-bit). **For ATI Radeon™ GPUs**: CAL 1.01, Matrix multiply (CAL optimized implementation, available in SDK), FFT (CAL optimized implementation), ATI Catalyst™ 8.6 software. Quoted peaks are based on manufacturer claims. All results normalized to ATI Radeon™ HD 3870 GPU.

ATI Radeon™ HD 4800 Series Stream Architecture

AMD Smarter Choice

- Several enhancements done for stream computing
  - Fast compute dispatch
  - Local /Global data shares
  - Increased Integer Processing Abilities
  - Fast Mem import/export
- Significant increases in performance on many important stream processing workloads

Agenda

AMD
Smarter Choice

- **Stream SDK strategy update**
- **Stream implication for professional and consumer markets**
- **Brook+**
- AMD's Compute Abstraction Layer (CAL)
- DirectX® 11.0 **Compute Shader Introduction**
- **OpenCL Introduction**
- **Stream processing demos**

Stream SDK Momentum

AMD
Smarter Choice

- AMD was the first company to offer a freely downloadable, open set of programming tools for GPGPU programming
- Adoption of Stream SDK, launched in 2006, continues to grow
- Hundreds of topics have been posted and discussed on the AMD Stream Developer Forum, making it the most active developer forum at AMD
- http://developer.amd.com/devforum

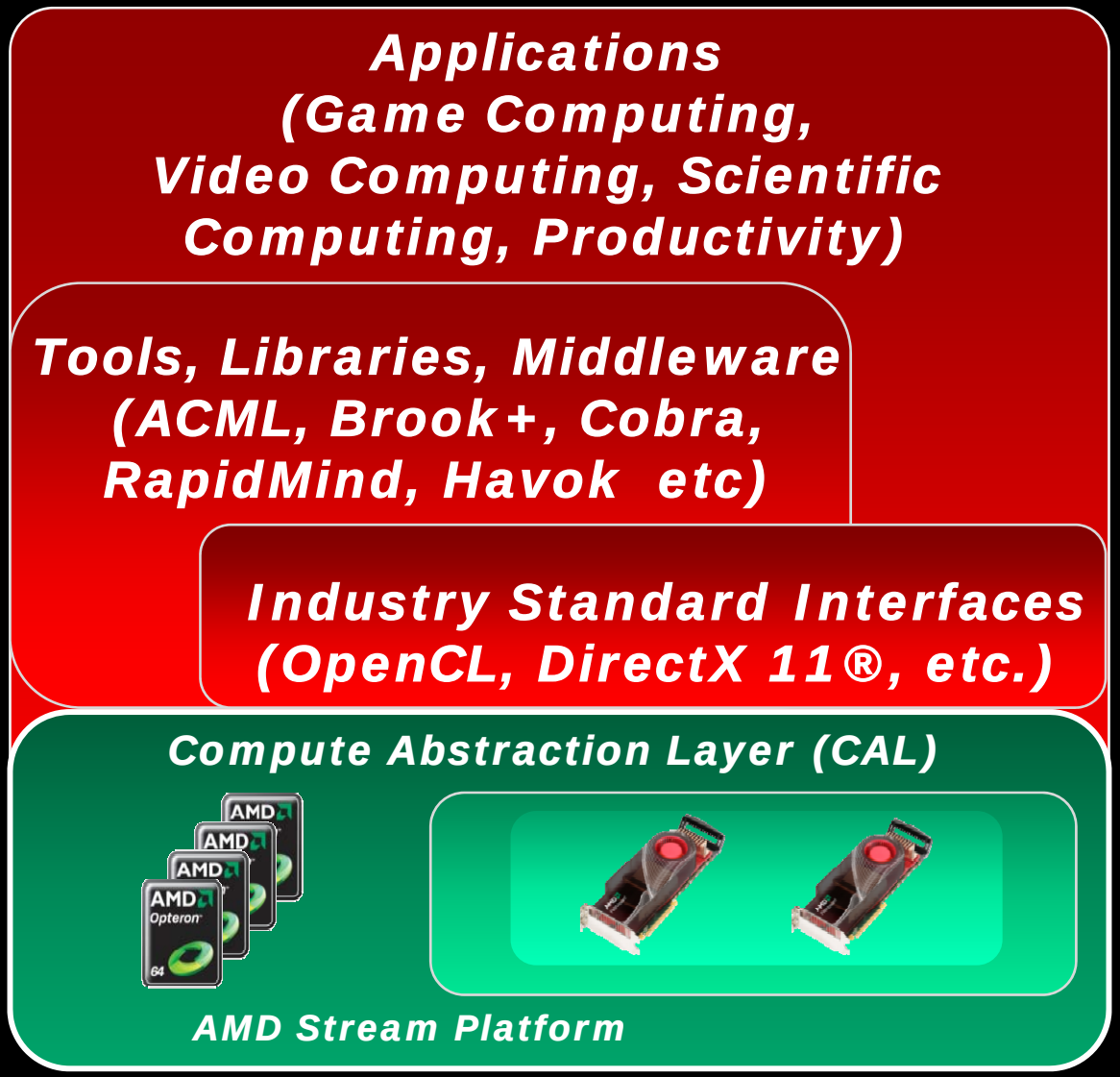
Key imperatives for the growth of an application eco-system

AMD
Smarter Choice

- Industry standard Programming models
- Tools, Libraries, Middleware
- Services
- Research community and early adopters

AMD Stream Processing Strategy

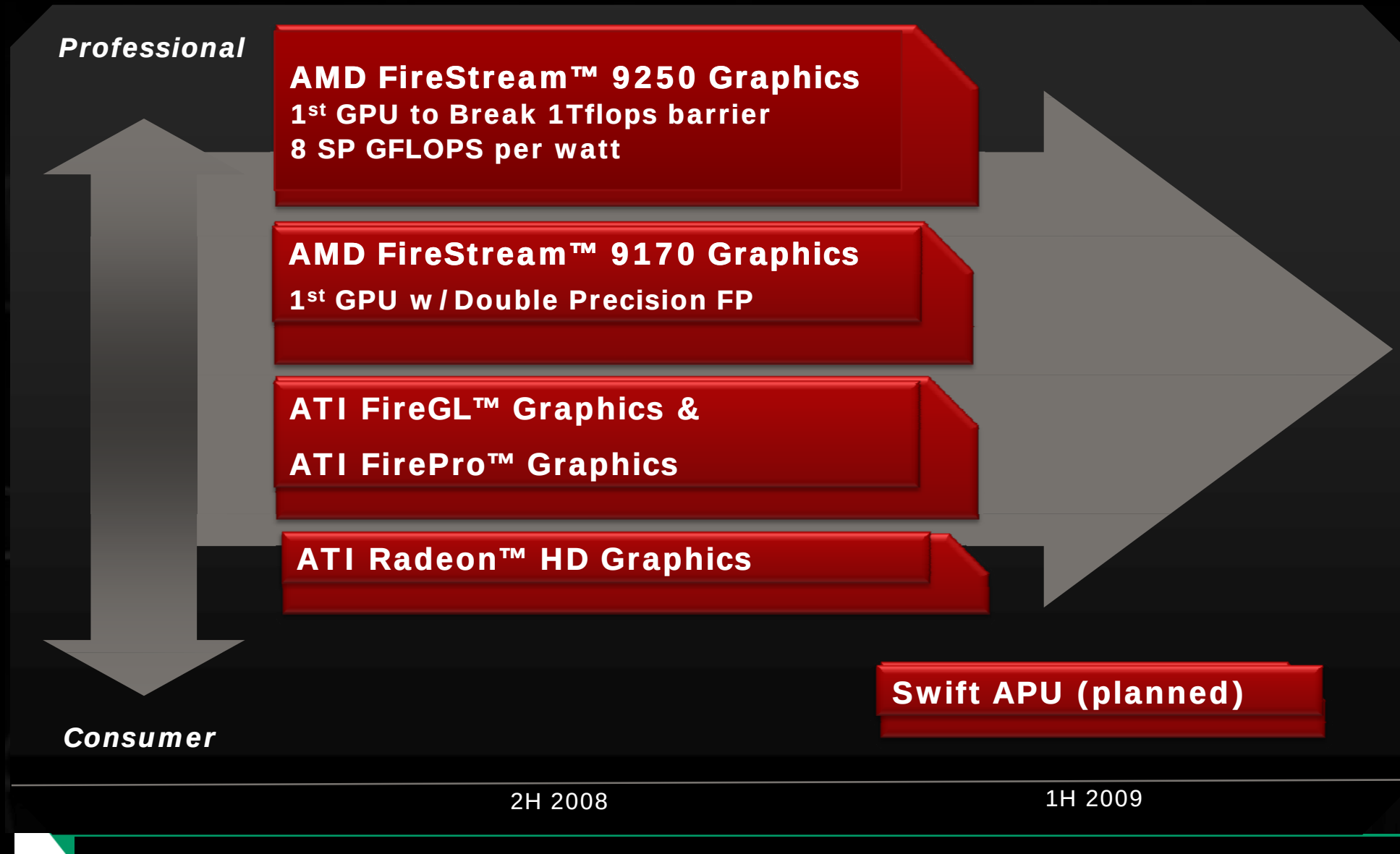
AMD
Smarter Choice



Single Programming Environment

Top to bottom support of stream applications on AMD GPUs

AMD Smarter Choice



http://ati.amd.com/products/streamprocessor/specs.html

AMD
Smarter Choice

Brook+

Computing Language

What is Brook+?

AMD
Smarter Choice

Brook is an extension to the C-language for stream programming originally developed by Stanford University

Brook+ is an implementation by AMD of the Brook GPU spec on AMD's compute abstraction layer with some enhancements

Asynchronous CPU->GPU transfers (GPU->CPU still synchronous)

Linux®, Windows Vista®, Microsoft® Windows® XP 32 & 64-bit

Extension Mechanism

- Allow ASIC specific features to be exposed without ‘sullying’ core language

Simple Example

AMD Smarter Choice

```
kernel void sum(float a<>, float b<>, out float c<>)
{
    c = a + b;
}

int main(int argc, char** argv)
{
    int i, j;
    float a<10, 10>;
    float b<10, 10>;
    float c<10, 10>;

    float input_a[10][10];
    float input_b[10][10];
    float input_c[10][10];

    for(i=0; i<10; i++) {
        for(j=0; j<10; j++) {
            input_a[i][j] = (float) i;
            input_b[i][j] = (float) j;
        }
    }
    streamRead(a, input_a);
    streamRead(b, input_b);

    sum(a, b, c);

    streamWrite(c, input_c);
    ...
}
```

Kernels – Program functions that operate on streams

Streams – collection of data elements of the same type which can be operated on in parallel.

Brook+ memory access functions

Brook+ kernels

AMD Smarter Choice

```
kernel void sum(float a<>, float b<>, out float c<>)
{
    c = a + b;
}

kernel void sum(float a[], float b[], out float c<>)
{
    int idx = indexof(c);
    c = a[idx] + b[idx];
}

kernel void sum(float a<>, float b<>, out float c[])
{
    int idx = indexof(c);
    c[idx] = a + b;
}
```

a[0]	a[1]	a[2]	a[3]	a[4]	a[5]	a[6]	a[7]
+	+	+	+	+	+	+	+
b[0]	b[1]	b[2]	b[3]	b[4]	b[5]	b[6]	b[7]
=	=	=	=	=	=	=	=
c[0]	c[1]	c[2]	c[3]	c[4]	c[5]	c[6]	c[7]

```
}
```

Standard Streams - implicit and predictable access pattern

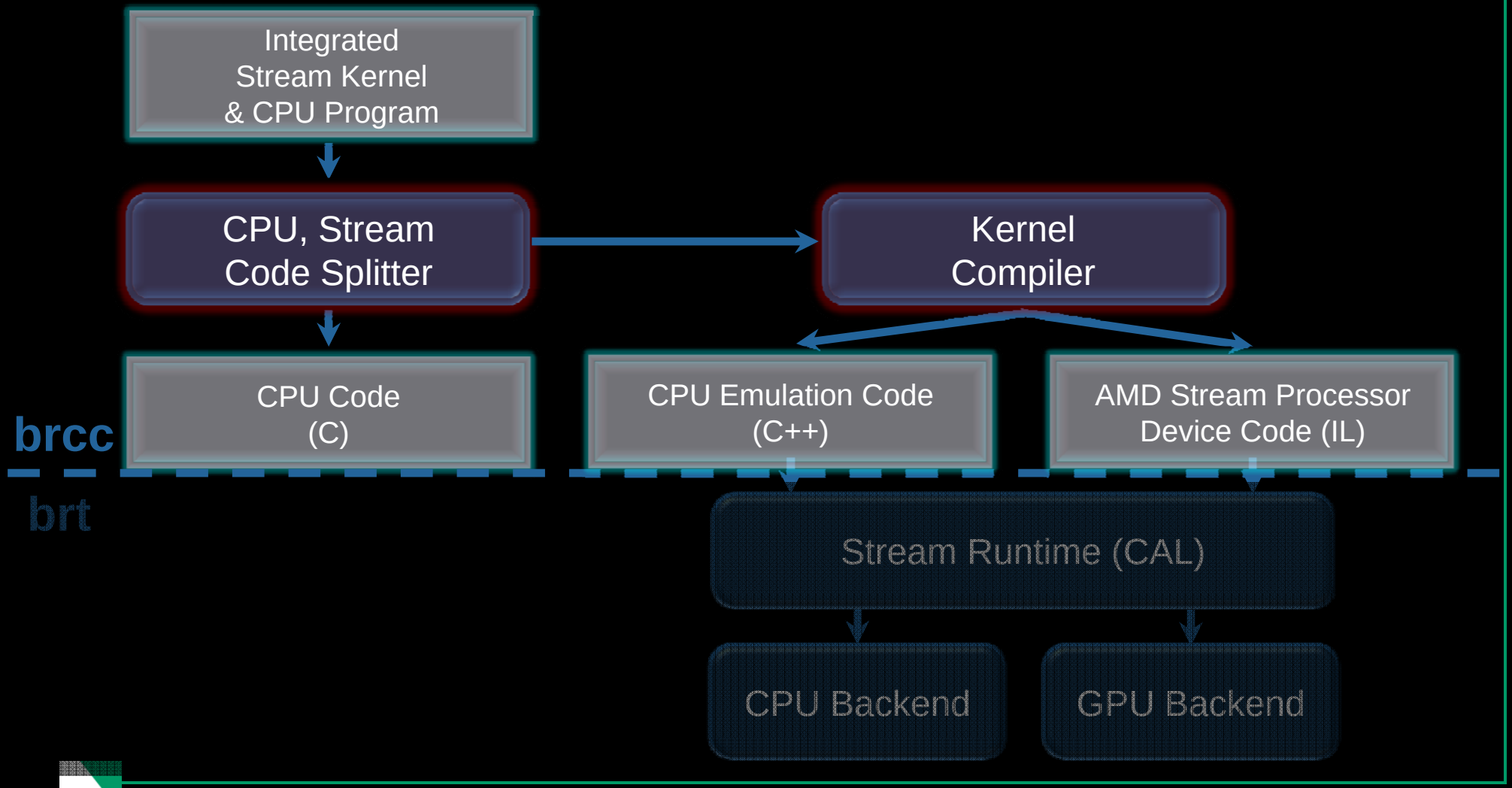
Gather Streams - dynamic read access pattern

Scatter Stream - dynamic write access pattern

Brook+ Compiler

AMD
Smarter Choice

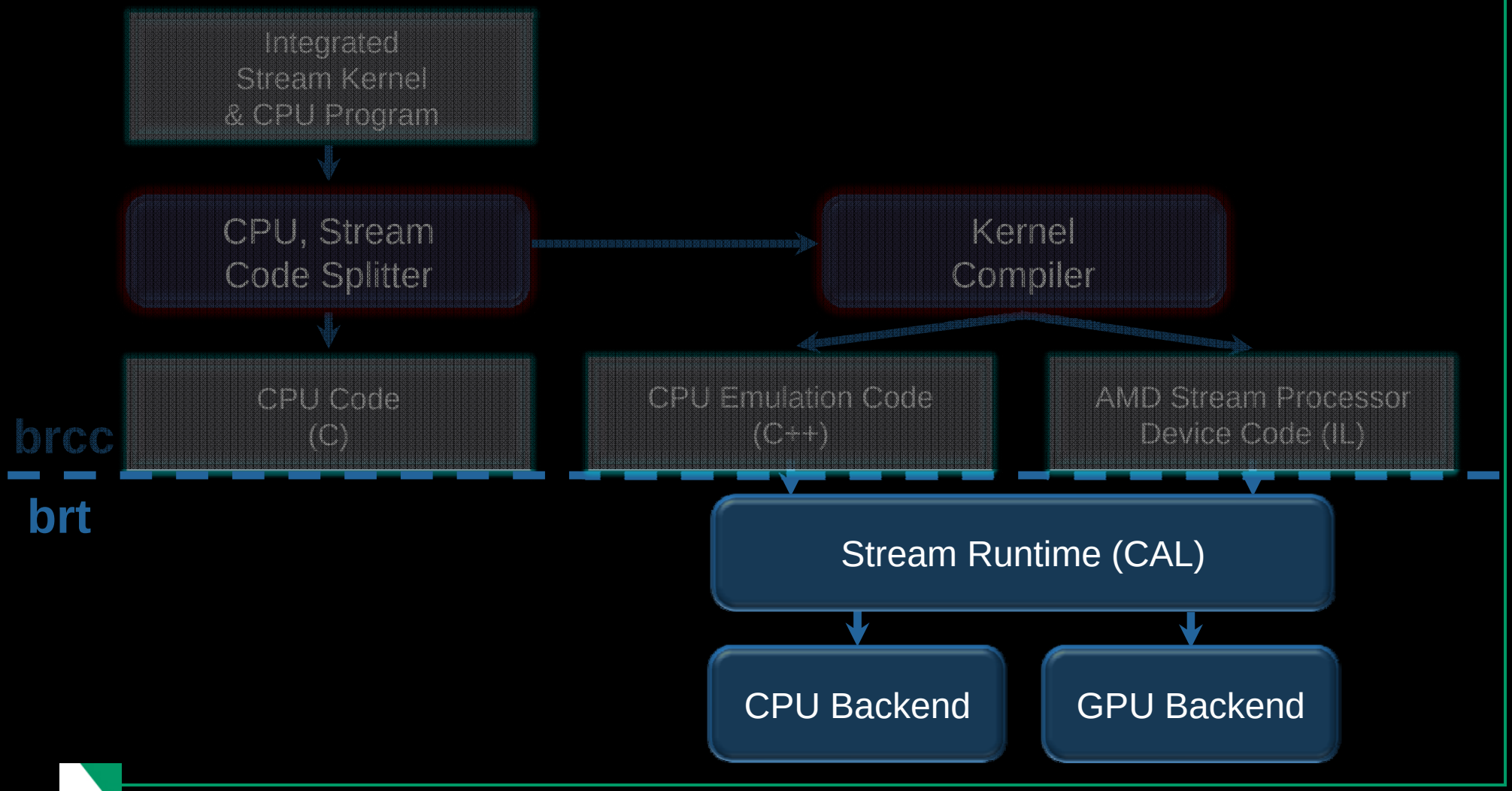
Converts Brook+ files into C++ code. Kernels, written in C, are compiled to AMD's IL code for the GPU or C code for the CPU.



Brook+ Compiler

AMD Smarter Choice

IL code is executed on the GPU. The backend is written in CAL.



Improving and Evolving Brook+

AMD
Smarter Choice

- Improve and evolve Brook+ programming language, compiler and runtime, including:
  - improved error handling
  - data transfer optimizations
  - C++ runtime API
  - access to CAL-level functionality

Aims to provide a stable, high-performance platform for accelerating applications today, paving migration path towards
OpenCL/DirectX®
Compute

AMD
Smarter Choice

CAL

Compute Abstraction Layer

Compute Abstraction Layer (CAL) goals

AMD
Smarter Choice

Expose relevant compute aspects of the GPU

- Command Processor
- Data Parallel Processor(s)
- Memory Controller

Hide all other graphics-specific features

Provide direct & asynchronous communication to device

Eliminate driver implemented procedural API

- Push policy decisions back to application
- Remove constraints imposed by graphics APIs

35

CAL Highlights

AMD
Smarter Choice

Memory managed

- Don’t have to manually maintain offsets, etc
- Asynchronous DMA: CPU→GPU, GPU→GPU, GPU→CPU
- Multiple GPUs can share the same “system” memory

Core CAL API is device agnostic

Enables multi-device optimizations

- e.g. Multiple GPUs working together concurrently
- Multiple GPUs show up as multiple CAL devices

Extensions to CAL provide opportunities for device specific optimization

Commitment to Industry Standards – AMD Believes:

AMD
Smarter Choice

Industry standards and open collaboration is key to driving development of stream applications and ensuring these applications work on the broadest range of hardware

Proprietary Solutions

- Helps to drive experimentation of new technology
- Appropriate when no standards in place

Industry Standards

- As hardware and software evolves, key to making it accessible in a unified way
- Stream processing has evolved to a point where proprietary solutions are not helping to drive broader technology acceptance

AMD
Smarter Choice

DirectX® 11

Compute Shaders

An evolving processing model for GPUs

DirectX® 11.0 Siggraph2008 slides courtesy of Chas Boyd Architect @ Microsoft Windows Desktop and Graphics Technology

Introducing: the Compute Shader

- A new processing model for GPUs
  - Data–parallel programming for mass market client apps
- Integrated with Direct3D
  - For efficient inter-op with graphics in client scenarios
- Supports more general constructs than before
  - Cross thread data sharing
  - Un-ordered access I/O operations
- Enables more general data structures
  - Irregular arrays, trees, etc.
- Enables more general algorithms
  - Far beyond shading

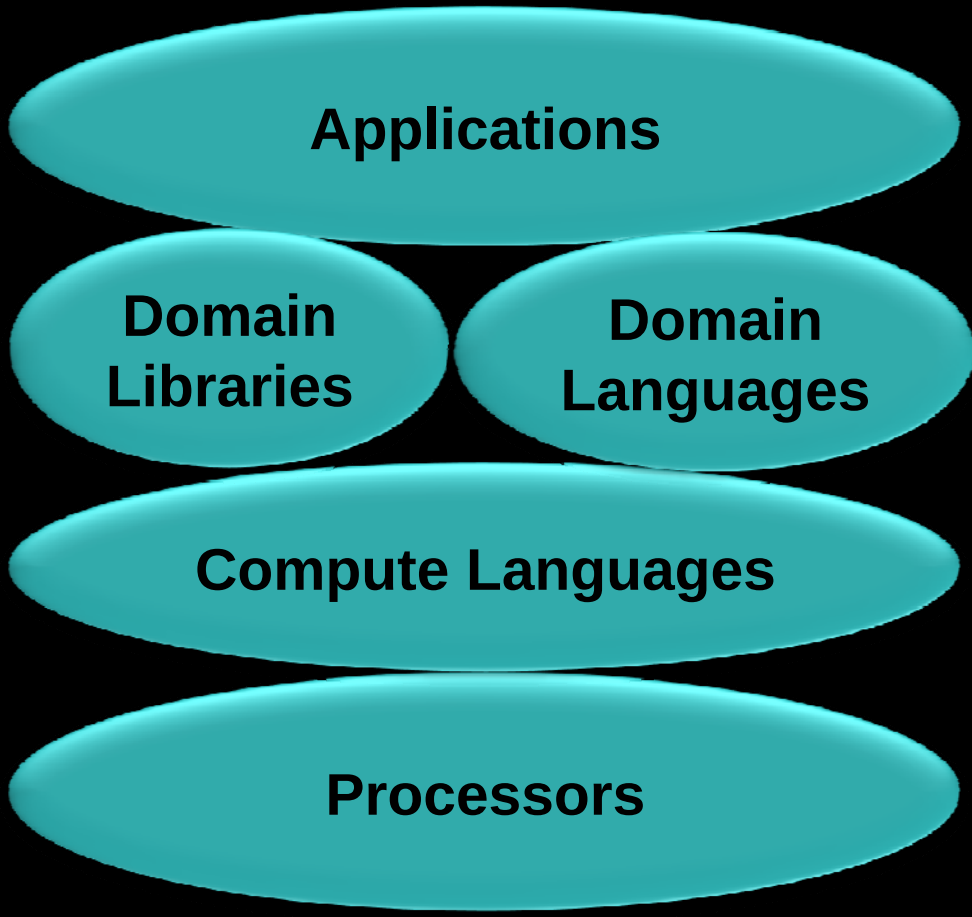
Target: Interactive Graphics/Games

- Image/Post processing:
  - Image Reduction, Histogram, Convolution, FFT
- Effect physics
  - Particles, smoke, water, cloth, etc.
- Advanced renderers:
  - A-Buffer/OIT, Reyes, Ray-tracing, radiosity, etc.
- Game play physics, AI, etc.
- Production pipelines

Target: Media Processing

- Video:
  - Transcode, Super Resolution, etc.
- Photo/imaging:
  - Consumer applications
- Non-client scenarios:
  - HPC, server workloads, etc.

Component Relationships



Media playback or processing, media UI, recognition, etc.

Accelerator, Brook+, Rapidmind, Ct
MKL, ACML, cuFFT, D3DX, etc.

DirectX® 11.0 **Compute,**
CUDA, CAL, OpenCL,
LRB Native, etc.

CPU, GPU, Larrabee
nVidia, Intel, AMD, S3, etc.

DirectX® 11.0 Siggraph2008 slides courtesy of Chas Boyd Architect @ Microsoft Windows Desktop and Graphics Technology

Compute Shader Features

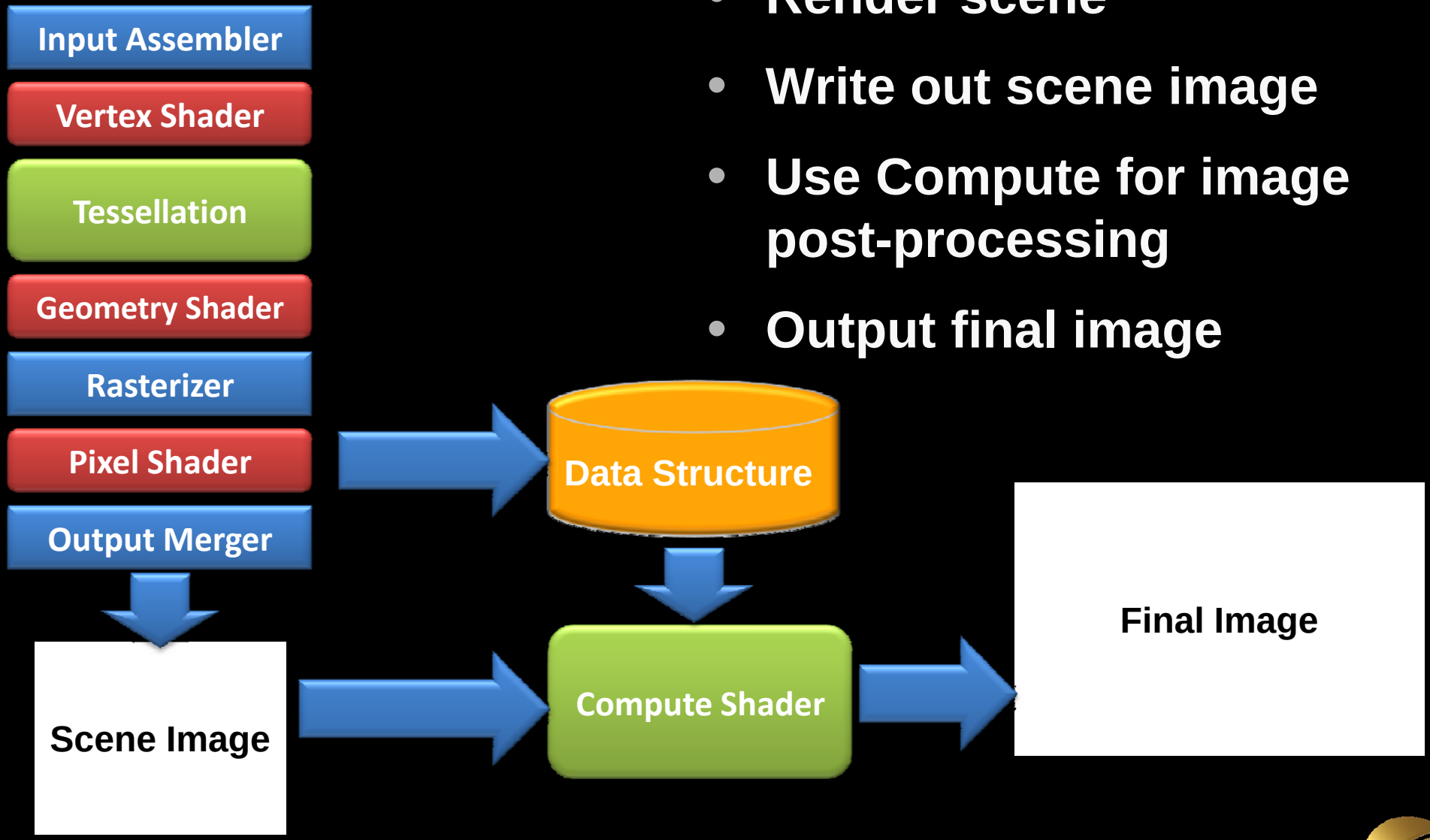
- Predictable Thread Invocation
  - Regular arrays of threads: 1-D, 2-D, 3-D
  - Don’t have to ‘draw a quad’ anymore
- Shared registers between threads
  - Reduces register pressure
  - Can eliminate redundant compute and i/o
- Scattered Writes
  - Can read/write arbitrary data structures
  - Enables new classes of algorithms
  - Integrates with Direct3D resources

Integrated with Direct3D

- Fully supports all Direct3D resources
- Targets graphics/media data types
- Evolution of DirectX® HLSL
- Graphics pipeline updated to emit general data structures via addressable writes
- Which can then be manipulated by compute shader
- And then rendered by Direct3D again

Integration with Graphics Pipeline

- Render scene
- Write out scene image
- Use Compute for image post-processing
- Output final image



Memory Objects

- DXGI Resources
  - Used for textures, images, vertices, hulls, etc.
  - Enables out-of-bounds memory checking
    - Returns 0 on reads
    - Writes are No-Ops
  - Improves security, reliability of shipped code
- Exposed as HLSL ‘Resource Variables’
  - Declared in the language as data objects

Optimized I/O Intrinsics

- **Textures & Buffers**
  - RWTexture2D, RWBuffer
  - Act just like existing types
- **Structured I/O**
  - RWStructuredBuffer
  - StructuredBuffer (read-only)
  - Template type can be any struct definition
- **Fast Structured I/O**
  - AppendStructuredBuffer, ConsumeStructuredBuffer
  - Work like streams
  - Do not preserve ordering

Atomic Operator Intrinsics

Enable basic operations w/o lock/contention:

```
InterlockedAdd( rVar, val );
InterlockedMin( rVar, val );
InterlockedMax( rVar, val );
InterlockedOr(  rVar, val );
InterlockedXOr( rVar, val );
InterlockedCompareWrite( rVar, val );
InterlockedCompareExchange( rVar, val );
```

Reduction Compute Code

```
Buffer<uint> Values;
OutputBuffer<uint> Result;

ImageAverage()
{
   groupshared uint Total;                    // Total so far
   groupshared uint Count;                    // Count added

   float3 vPixel = load( sampler, sv_ThreadID );
   float fLuminance = dot( vPixel, LUM_VECTOR );
   uint value = fLuminance*65536;

   InterlockedAdd( Count, 1 );
   InterlockedAdd( Total, value );

   GroupMemoryBarrier(); // Let all threads in group complete

   . . . .
```

Summary

- DirectX® 11.0 Compute Shader expected to deliver the performance of 3-D games to new applications
- Tight integration between computation and rendering
- Scalable parallel processing model
  - Code should scale for several generations

AMD
Smarter Choice

OpenCL

Open Computing Language

Siggraph2008 slides courtesy of Aaftab Munshi
Architect @ Apple & Khronos OpenCL Workgroup Member

Update on OpenCL

AMD
Smarter Choice

- Khronos OpenCL working group making aggressive progress (www.khronos.org)
- Contributing Industry leaders of task/data parallel processors
- Standardize framework and language for multiple heterogeneous processors
- PC developers are expected to have early version in Q1 2009
- Based on a proposal by Apple
  - OpenCL Parallel Computing on GPU and CPU (Munshi @ SigGraph 2008)
  - Developed in collaboration with industry leaders

OpenCL – A Sneak Preview

Design Goals of OpenCL

- Use all computational resources in system
  - GPUs and CPUs as peers
  - Data- and task- parallel compute model
- Efficient parallel programming model
  - Based on C
  - Abstract the specifics of underlying hardware
- Specify accuracy of floating-point computations
  - IEEE 754 compliant rounding behavior
  - Define maximum allowable error of math functions
- Drive future hardware requirements

OpenCL Software Stack

- Platform Layer
  - query and select compute devices in the system
  - initialize a compute device(s)
  - create compute contexts and work-queues
- Runtime
  - resource management
  - execute compute kernels
- Compiler
  - A subset of ISO C99 with appropriate language additions
  - Compile and build compute program executables
  - online or offline

OpenCL Execution Model

- Compute Kernel
  - Basic unit of executable code — similar to a C function
  - Data-parallel or task-parallel
- Compute Program
  - Collection of compute kernels and internal functions
  - Analogous to a dynamic library
- Applications queue compute kernel execution instances
  - Queued in-order
  - Executed in-order or out-of-order
  - Events are used to implement appropriate synchronization of execution instances

OpenCL Data-Parallel Execution Model

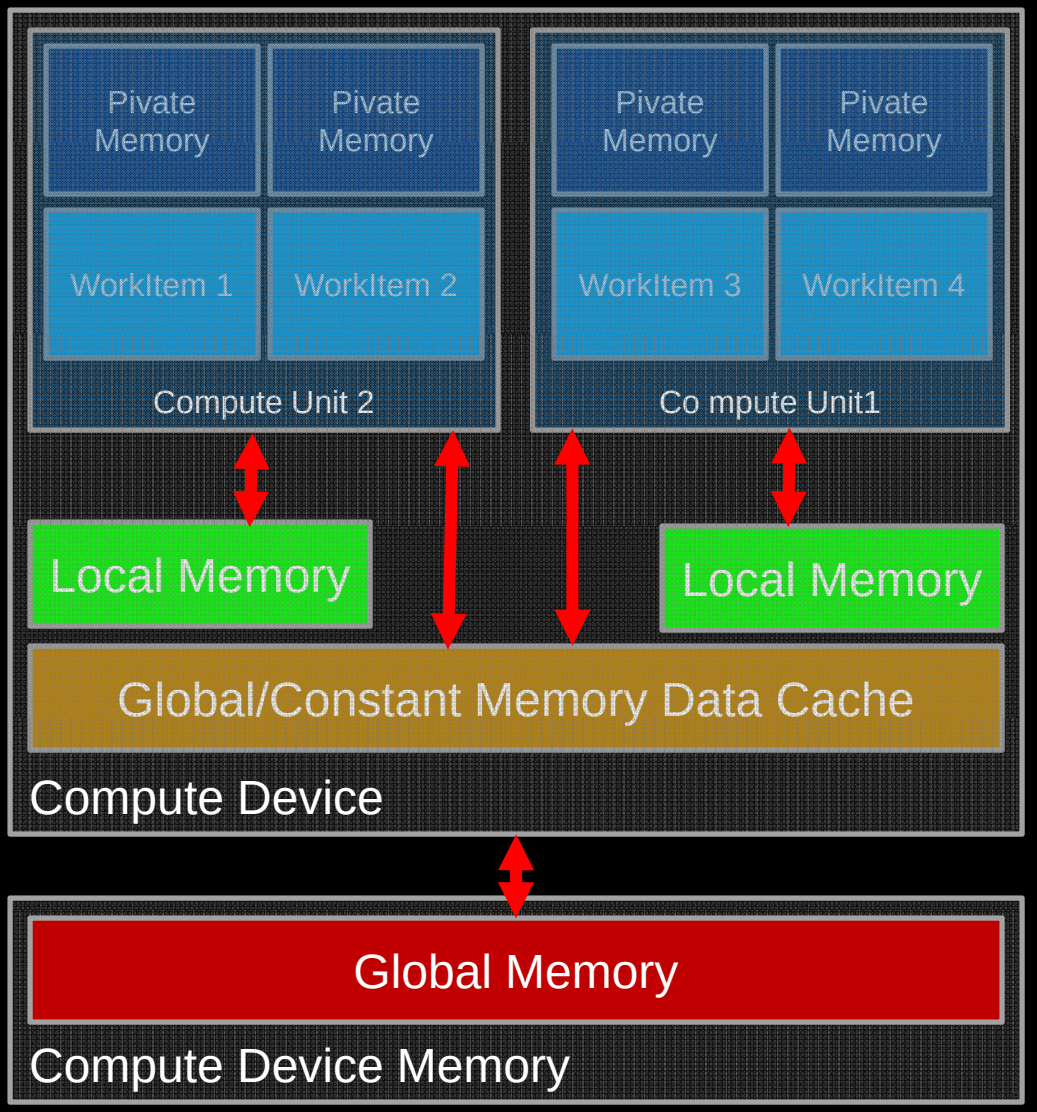
- Define N-Dimensional computation domain
  - Each independent element of execution in N-D domain is called a work-item
  - The N-D domain defines the total number of work-items that execute in parallel — global work size.
- Work-items can be grouped together — work-group
  - Work-items in group can communicate with each other
  - Can synchronize execution among work-items in group to coordinate memory access
- Execute multiple work-groups in parallel
- Mapping of global work size to work-groups
  - implicit or explicit

OpenCL Task Parallel Execution Model

- Data-parallel execution model must be implemented by all OpenCL compute devices
- Some compute devices such as CPUs can also execute task parallel compute kernels
  - Executes as a single work-item
  - A compute kernel written in OpenCL
  - A native C / C++ function

OpenCL Memory Model

- Implements a relaxed consistency, shared memory model
- Multiple distinct address spaces
  - Address spaces can be collapsed depending on the device's memory subsystem
  - Address Qualifiers
  - __private
  - __local
  - __constant and __global
  - Example:
    - __global float4 *p;



Language for writing compute kernals

- Derived from ISO C99
- A few restrictions
  - Recursion, function pointers, functions in C99 standard headers ...
- Preprocessing directives defined by C99 are supported
- Built-in Data Types
  - Scalar and vector data types
  - Pointers
  - Data-type conversion functions
  - convert_*type*<_*sat*><_*roundingmode*>
  - Image types
  - image2d_t, image3d_t and sampler_t

Language for writing compute kernels

- Built-in Functions — Required
  - work-item functions
  - math.h
  - read and write image
  - relational
  - geometric functions
  - synchronization functions
- Built-in Functions — Optional
  - double precision
  - atomics to global and local memory
  - selection of rounding mode
  - writes to image3d_t surface

Summary

- • A new compute language that works across GPUs and CPUs
  - C99 with extensions
  - Familiar to developers
  - Includes a rich set of built-in functions
  - Makes it easy to develop data- and task- parallel compute programs
- Defines hardware and numerical precision requirements
- Open standard for heterogeneous parallel computing

Summarizing AMD's Industry Standards Efforts

AMD
Smarter Choice

- Easing cross-platform development with major enhancements for stream software
- Single development environment for open, flexible software development and support for a broad range of GPU solutions
- A clear path to OpenCL and DirectX® 11

AMD FireStream™ Go-to-Market: Commercial Verticals Focus

AMD
Smarter Choice

Defense

Tele-Presence

Search

Oil & gas

Financial modeling and risk assessment

Facial image recognition

Climate research and forecasting

Life sciences

Building the Ecosystem: AMD Stream Application Successes

AMD Smarter Choice

Telanetix

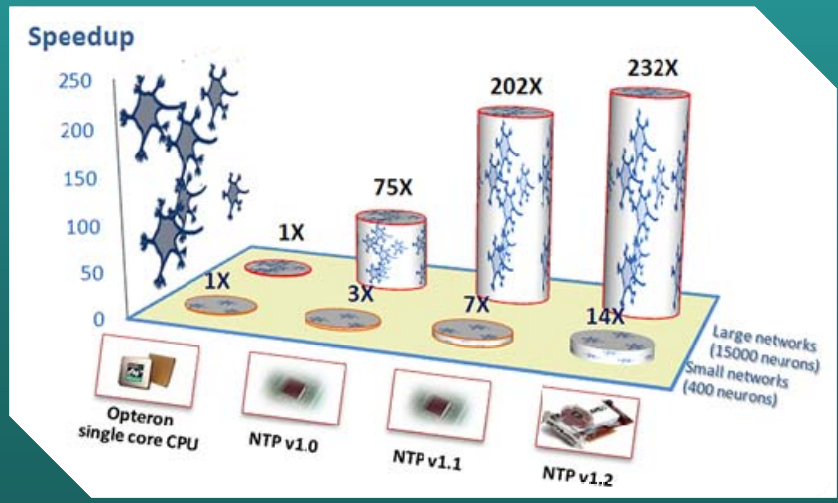
The Telanetix Digital Presence product line enables fully immersive and interactive meeting environments that incorporate voice, video and data from multiple locations into a single environment

Stream computing drives optimal performance, maximum flexibility for future enhancements

neurala

Developing Neurala Technology Platform for advanced brain-based machine learning applications – applicable to finance, image processing, etc.

Achieving 10-200x speedups* on biologically inspired neural models



•AMD FireStream™ 9150 versus dual AMD Opteron™ 248 processor (using only a single processor for comparison) w/ 2GB SDRAM DDR 400 ECC dual channel and SUSE Linux 10 (custom kernel)

•Performance information supplied by customer(s). AMD has not independently verified these results

Building the Ecosystem: Stream Research Breakthroughs

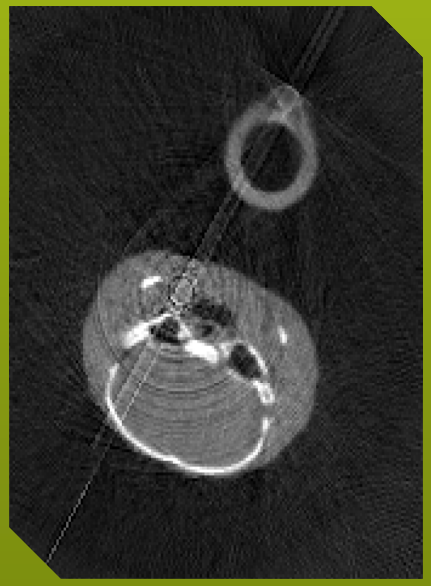
AMD Smarter Choice

Centre de Physique des Particules de Marseille

Tomographic Reconstruction:
Feldkamp, Davis, Krell Algorithm. Done by Alain Bonissent.

Widely used for tomographic reconstruction of X-Ray CT scanner data

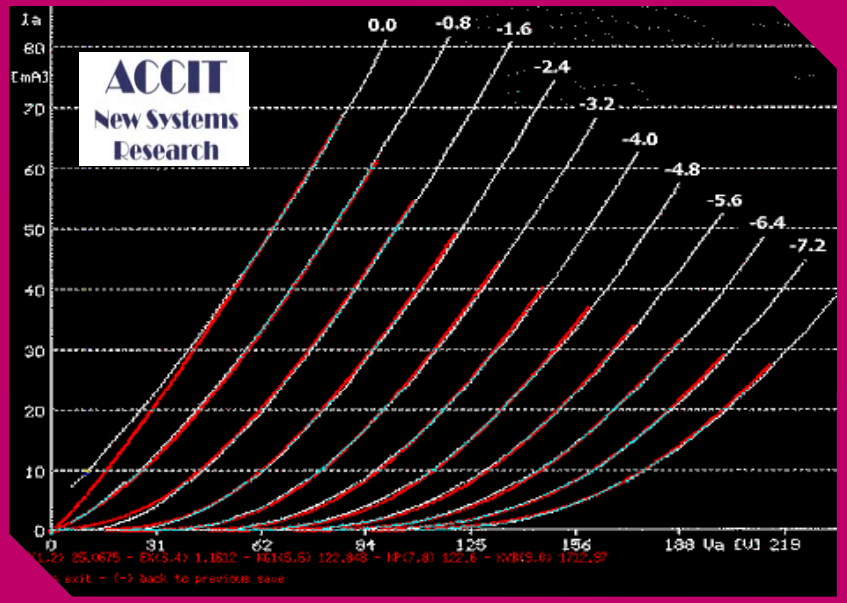
- Achieving 42-60x* speedups
- This image: 7 minutes in optimized C++; 10 seconds in Brook+



ACCIT

Parallel solutions for computationally-intensive EDA simulation engines

Currently beta testing applications demonstrating >10x speedup**



* AMD FireStream™ 9150 versus Intel® Core™ 2 Duo E6550 2.33 MHz running Windows XP 32-bit

** AMD FireStream™ 9150 versus quad-core AMD Phenom™ 9500 processor (2 GHz) w/ 8GB ECC DDR2 running at 667 MHz and AMD 790FX motherboard running Windows XP 64-bit

Performance information supplied by customer(s). AMD has not independently verified these results

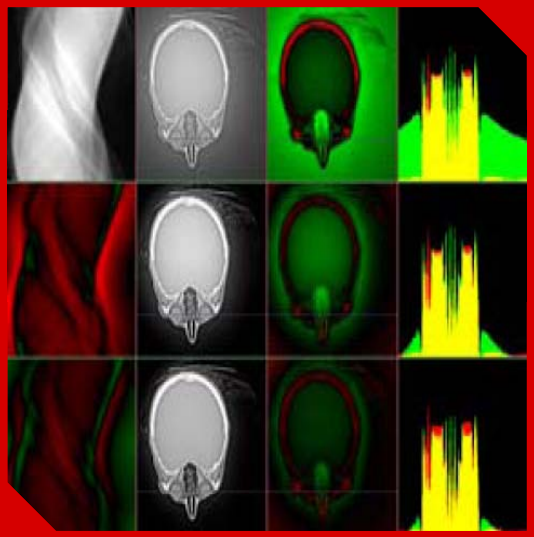
Building the Ecosystem: Stream Development Tools Successes

AMD Smarter Choice

RAPIDMIND

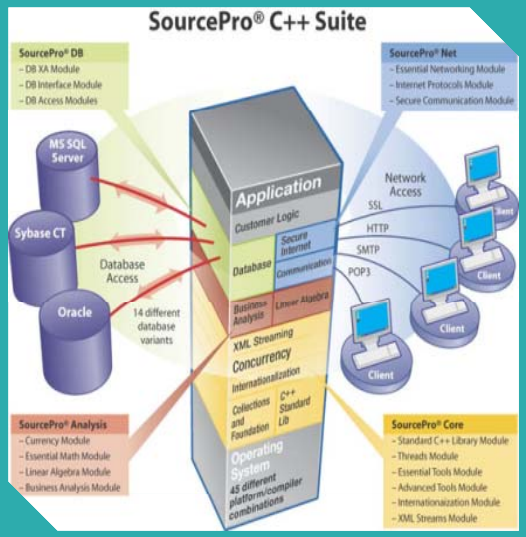
Development Environment for multi-core CPUs and GPUs

Demonstrating 55x speedup* on binomial options pricing calculator



ROGUE WAVE SOFTWARE

Facilitating parallel computing for financial applications



CAPS

HMPP Toolkit for development on accelerators; industry solutions for oil and gas, defense, finance, life sciences

•AMD FireStream™ 9150 versus Quantlib running solely on single core of a dual-processor AMD Opteron™ 2352 B3 processor on Tyan S2915 w/ Windows XP 32-bit (Palomar Workstation from Colfax)

•Performance information supplied by customer(s). AMD has not independently verified these results

Building the Ecosystem: Stream Development Services

AMD Smarter Choice

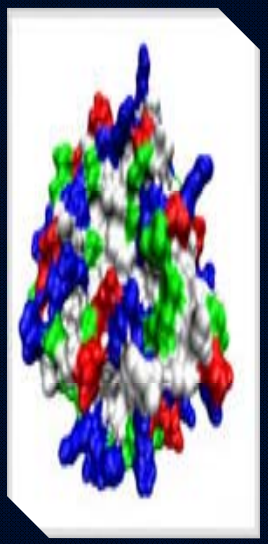
Computer Systems, Inc. MERCURY

Provides specialized, high-performance computing systems and software designed for complex HPC and embedded applications

Achieving 305 GFLOPS on large 1D complex single-precision FFTs

VIZEXPERTS

"Provide Stream Computing consulting, development and integration services specializing in the following industries -
- Oil & Gas
- Medical Imaging
- Financial Services"



Mercury benchmark system details: 2x Opteron quad-core 2.3GHz processors, 4GB, ATI Radeon 4870 GPU

Expansion of Stream Strategy into Consumer Applications

AMD
Smarter Choice

Science

Gaming

Productivity

Entertainment

Summary & Questions

AMD
Smarter Choice

Easing cross-platform development with major enhancements for stream software strategy

Aggressively expanding stream strategy to consumer segment

Contact: AMD Stream Computing SDK
http://ati.amd.com/technology/streamcomputing/

Disclaimer and Attribution

AMD
Smarter Choice

DISCLAIMER

The information presented in this document is for informational purposes only and may contain technical inaccuracies, omissions and typographical errors.

AMD MAKES NO REPRESENTATIONS OR WARRANTIES WITH RESPECT TO THE CONTENTS HEREOF AND ASSUMES NO RESPONSIBILITY FOR ANY INACCURACIES, ERRORS OR OMISSIONS THAT MAY APPEAR IN THIS INFORMATION.

AMD SPECIFICALLY DISCLAIMS ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR ANY PARTICULAR PURPOSE. IN NO EVENT WILL AMD BE LIABLE TO ANY PERSON FOR ANY DIRECT, INDIRECT, SPECIAL OR OTHER CONSEQUENTIAL DAMAGES ARISING FROM THE USE OF ANY INFORMATION CONTAINED HEREIN, EVEN IF AMD IS EXPRESSLY ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

ATTRIBUTION