GPU Computing: Past, Present and Future
with ATI Stream Technology

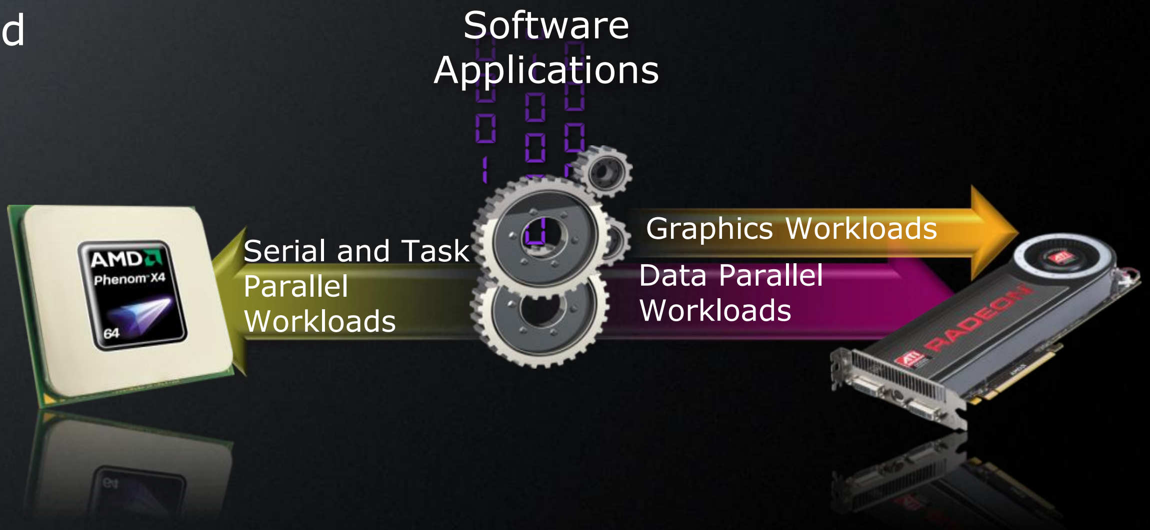
Michael Monkang Chu

Product Manager, ATI Stream Computing Software
michael.chu@amd.com

March 9, 2010

Harnessing the Computational Power of GPUs

- GPU architecture increasingly emphasizes programmable shaders instead of fixed function logic
- Enormous computational capability for data parallel workloads
- New math for datacenters: enables high performance/watt and performance/$

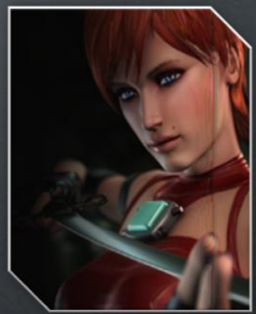


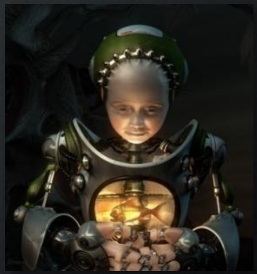
ATI Stream Technology is…

Heterogeneous: Developers leverage AMD GPUs and CPUs for optimal application performance and user experience

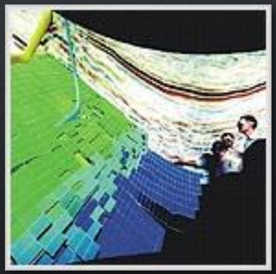
Industry Standards: OpenCL™ and DirectCompute 11 enable cross-platform development

High performance: Massively parallel, programmable GPU architecture enables superior performance and power efficiency

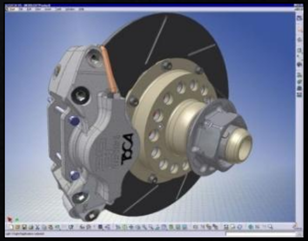

Gaming


Digital Content Creation

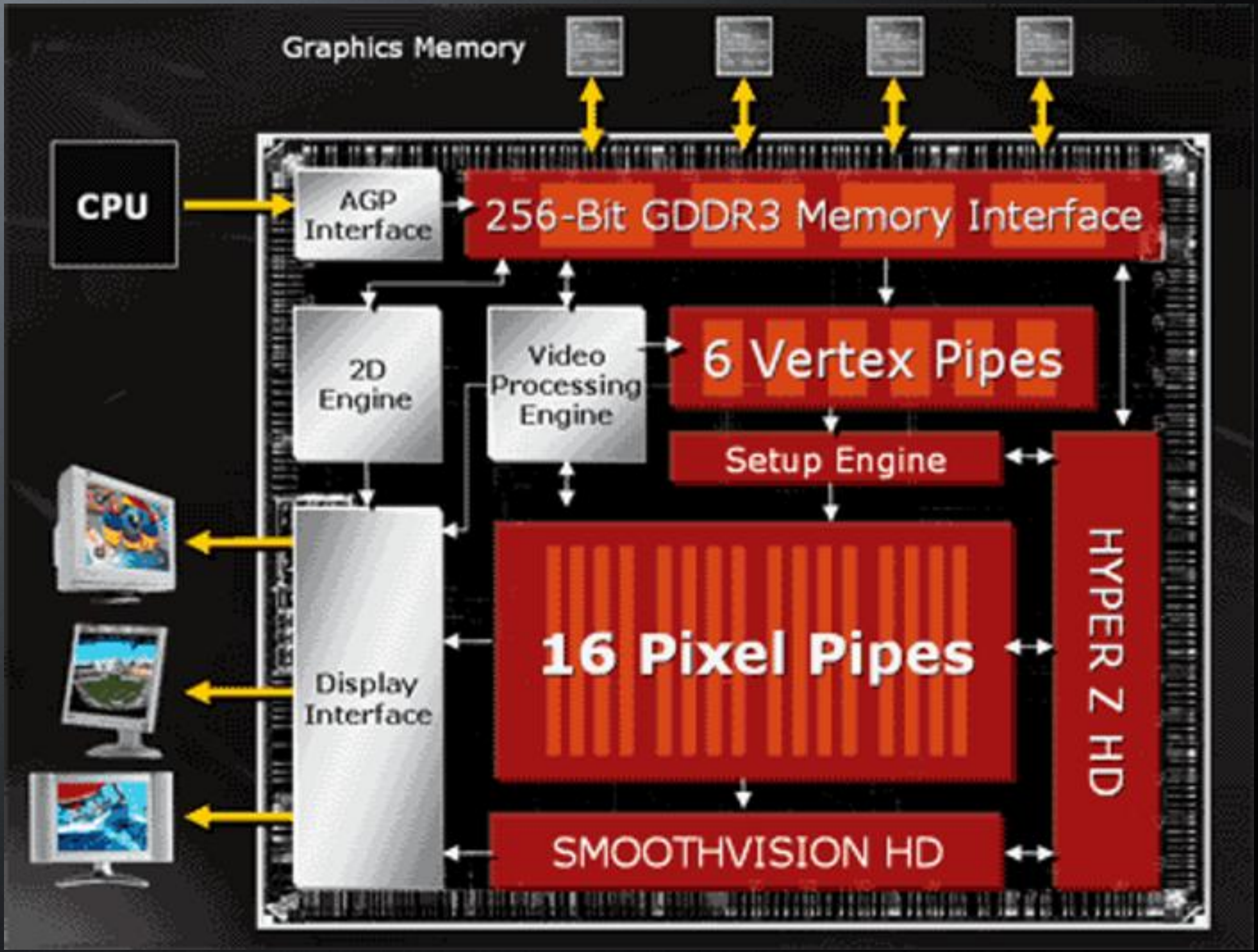

Productivity


Sciences

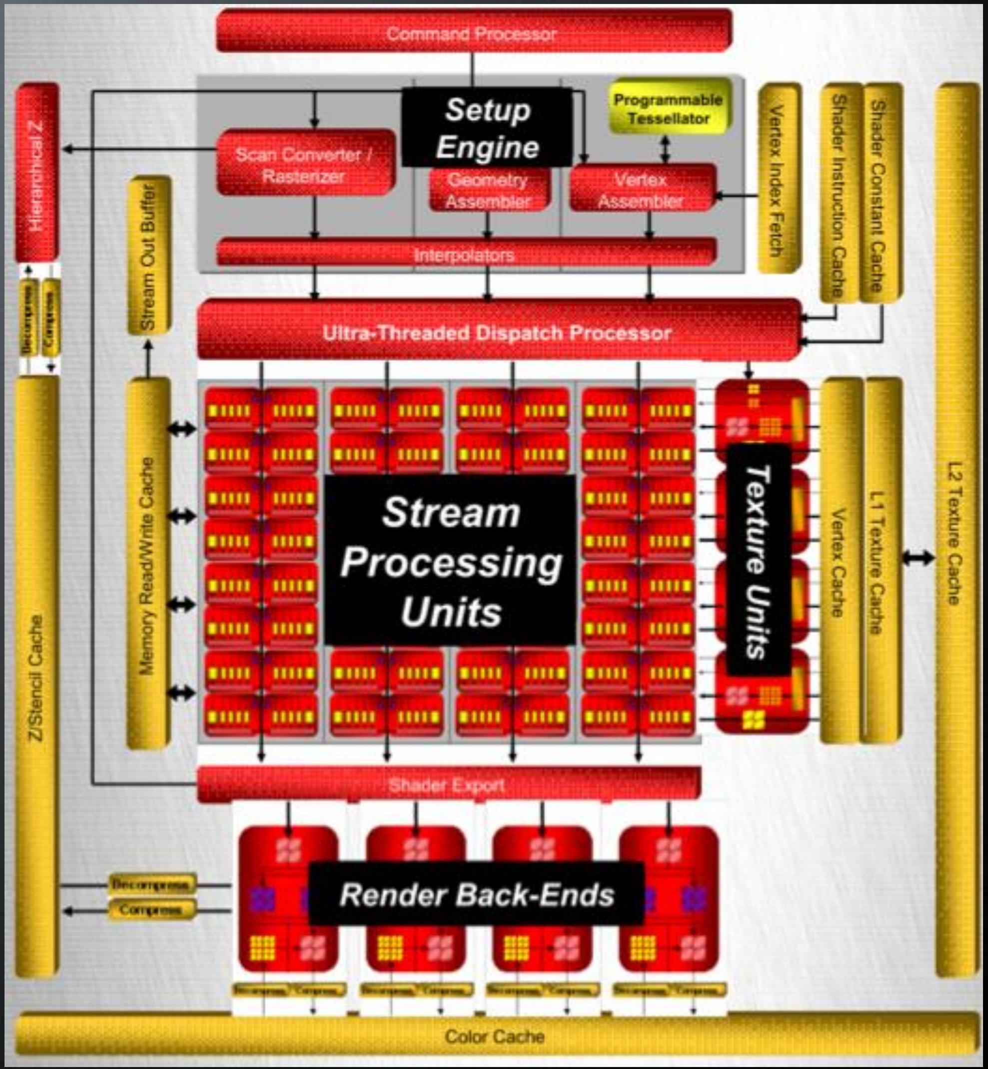

Government


Engineering

ATI Radeon™ X800 Architecture (2004)



ATI Radeon™ HD 3870 Architecture (2007)



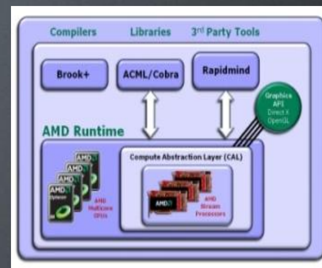
GPU Compute Timeline

Folding @Home

Proof of concept achieving >30x speedup over CPUs

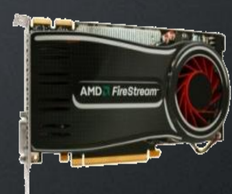
Stream Computing Development Platform

CTM for data parallel programming



ATI Stream SDK v0.9

Open systems approach to drive broad customer adoption (Brook+ & CAL/IL)



AMD FireStream™ 9170 GPU Compute Accelerator

First GPU Stream processor with double-precision floating point

ATI Stream SDK v1.0

Enhancements to improve computation performance

AMD FireStream™ 9250 GPU Compute Accelerator

Breaks the 1 TFLOPS barrier
Up to 8 GFOPS/watt

ATI Stream SDK v2.0

First production version of OpenCL™ for both x86 CPUs and AMD GPUs

ATI Radeon™ HD 5870 GPU

2.72 TFLOPS - SP

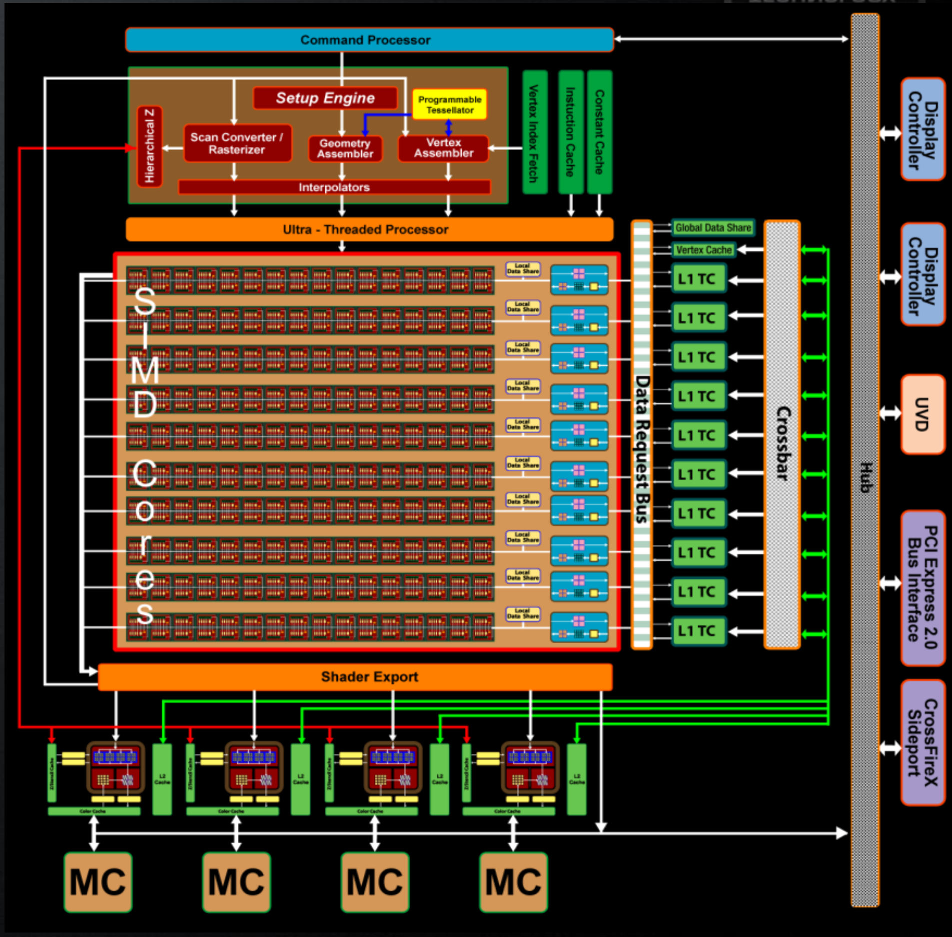
544 GFLOPS - DP

2006 | **2007** | **2008** | **2009**

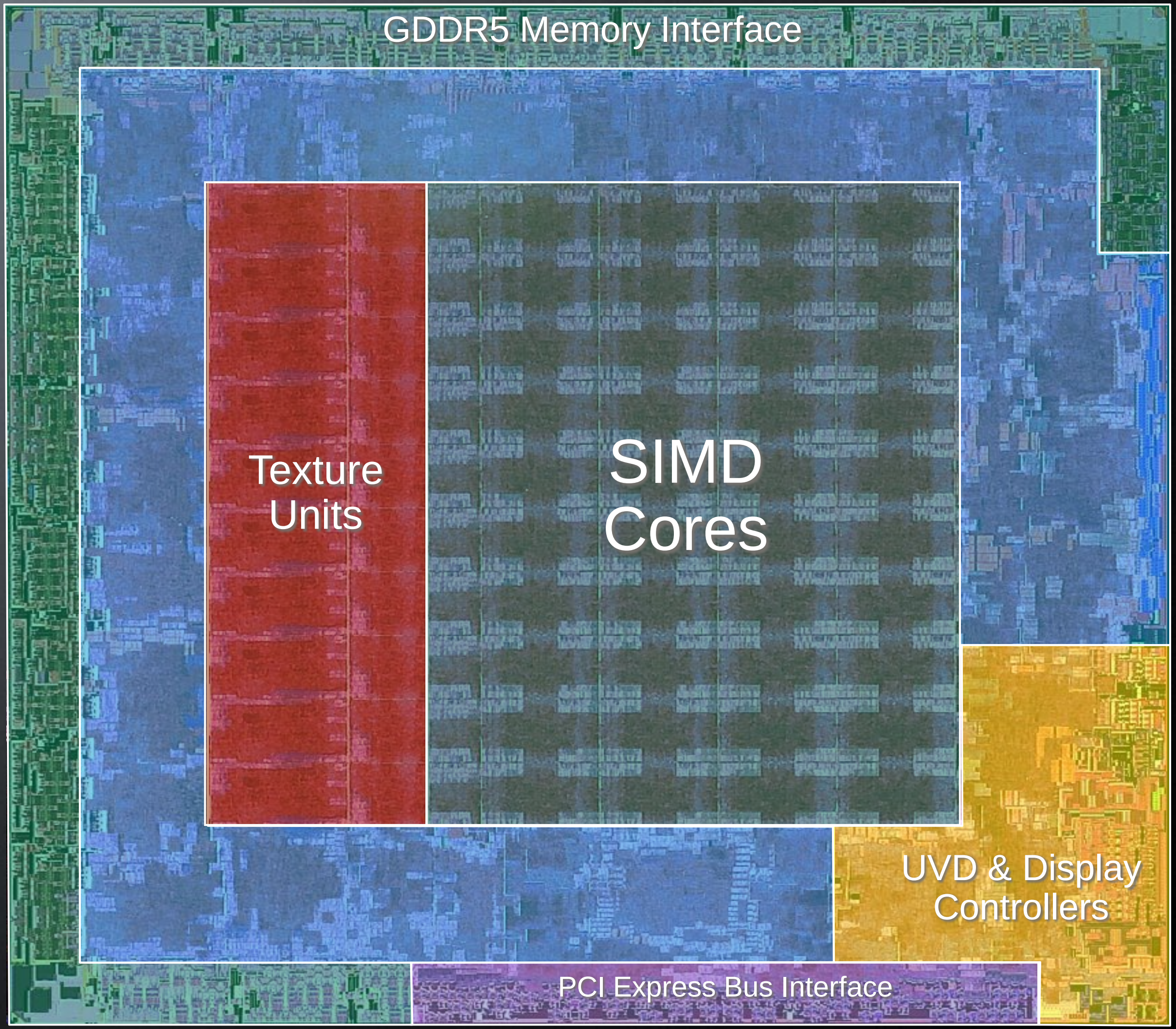
Enhancing GPUs for Computation

ATI Radeon™ HD 4870 Architecture (2008)

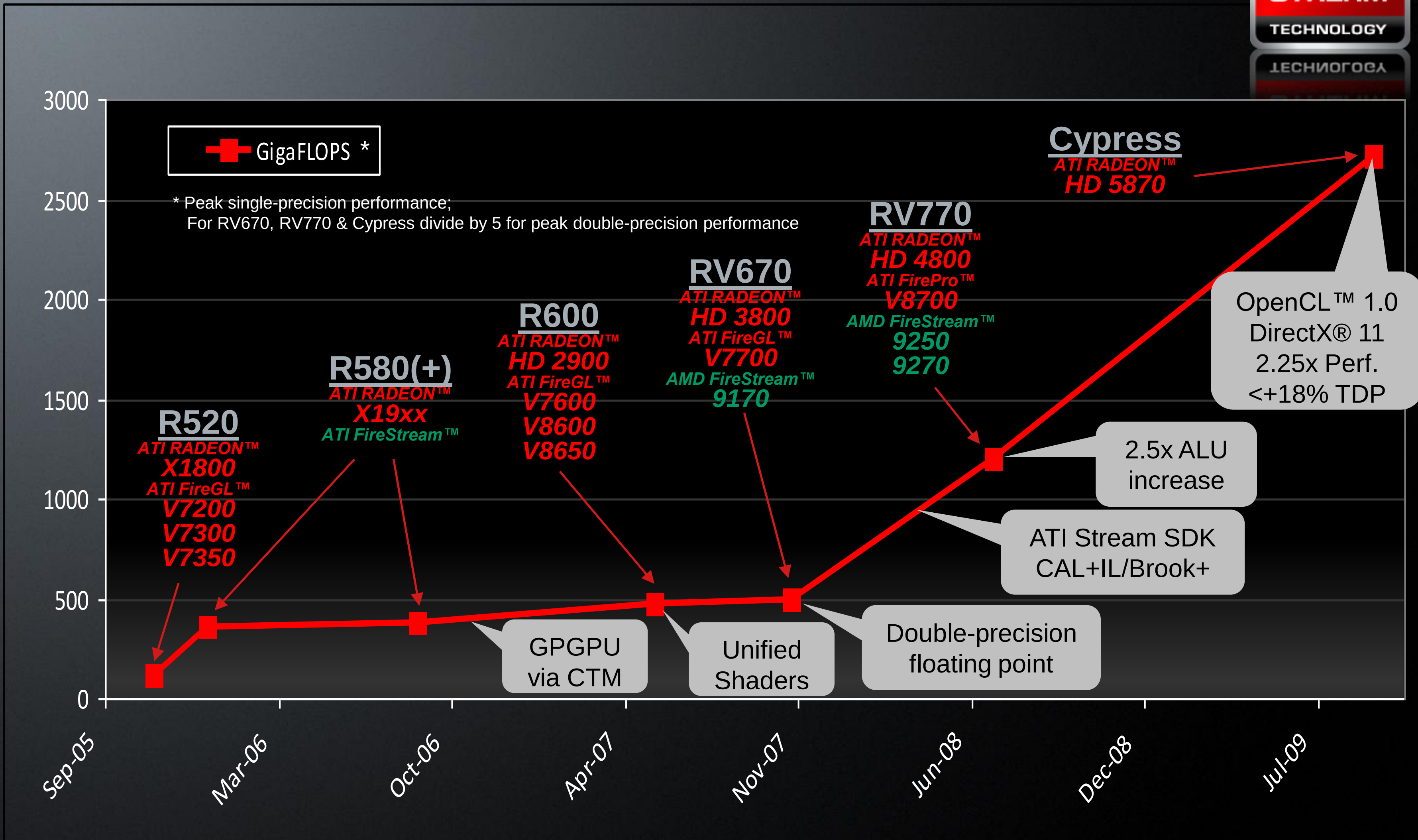
- 800 stream processing units arranged in 10 SIMD cores
- Up to 1.2 TFLOPS peak single precision floating point performance; Fast double-precision processing w/ up to 240 GFLOPS
- 115 GB/sec GDDR5 memory interface
- Up to 480 GB/s L1 & 384 GB/s L2 cache bandwidth
- Data sharing between threads
- Improved scatter/gather operations for improved GPGPU memory performance
- Integer bit shift operations for all units – useful for crypto, compression, video processing
- More aggressive clock gating for improved performance per watt



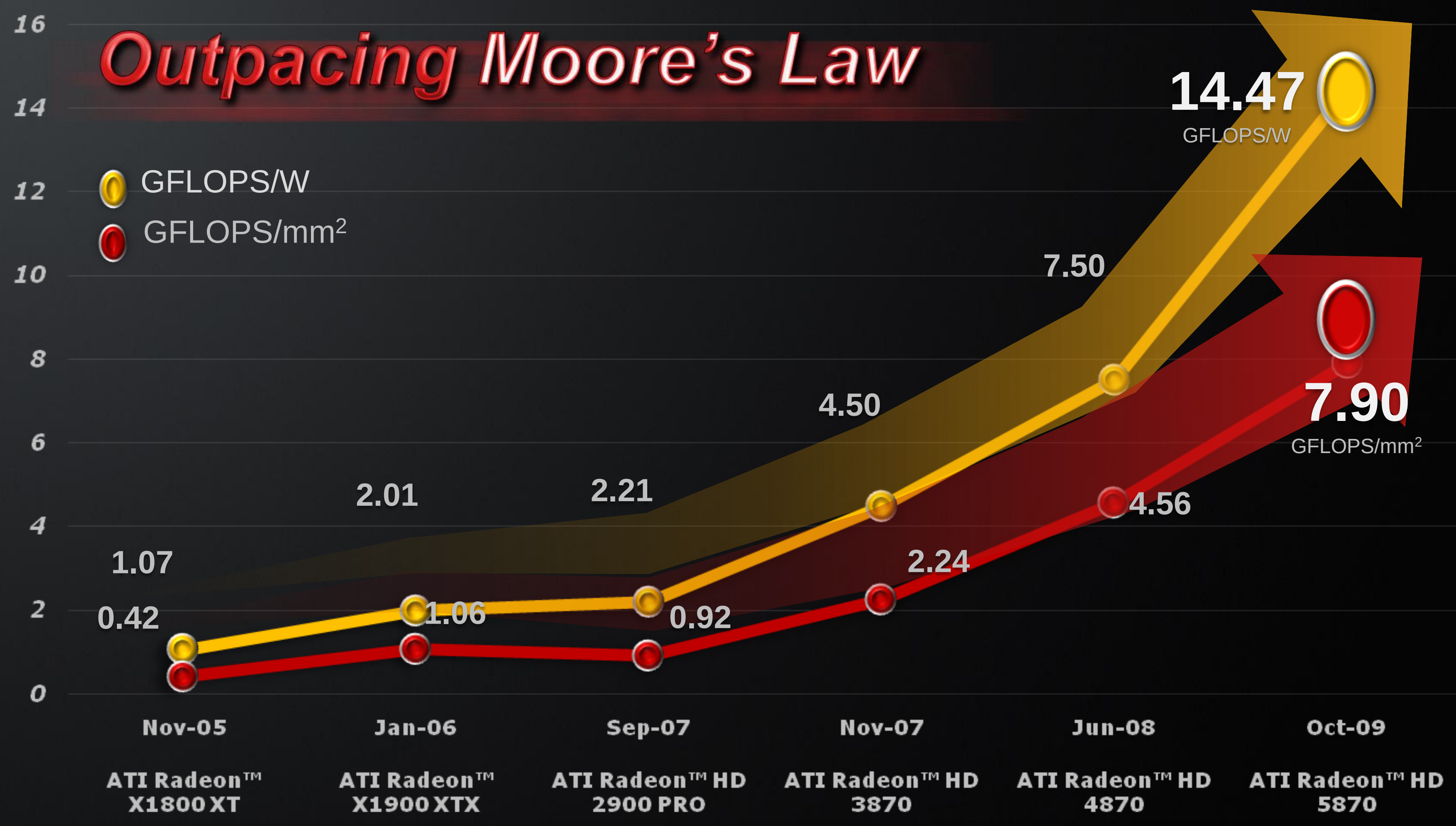
ATI Radeon™ HD 4870 Architecture



GPU Compute Processing Power Trend



The World's Most Efficient GPU*



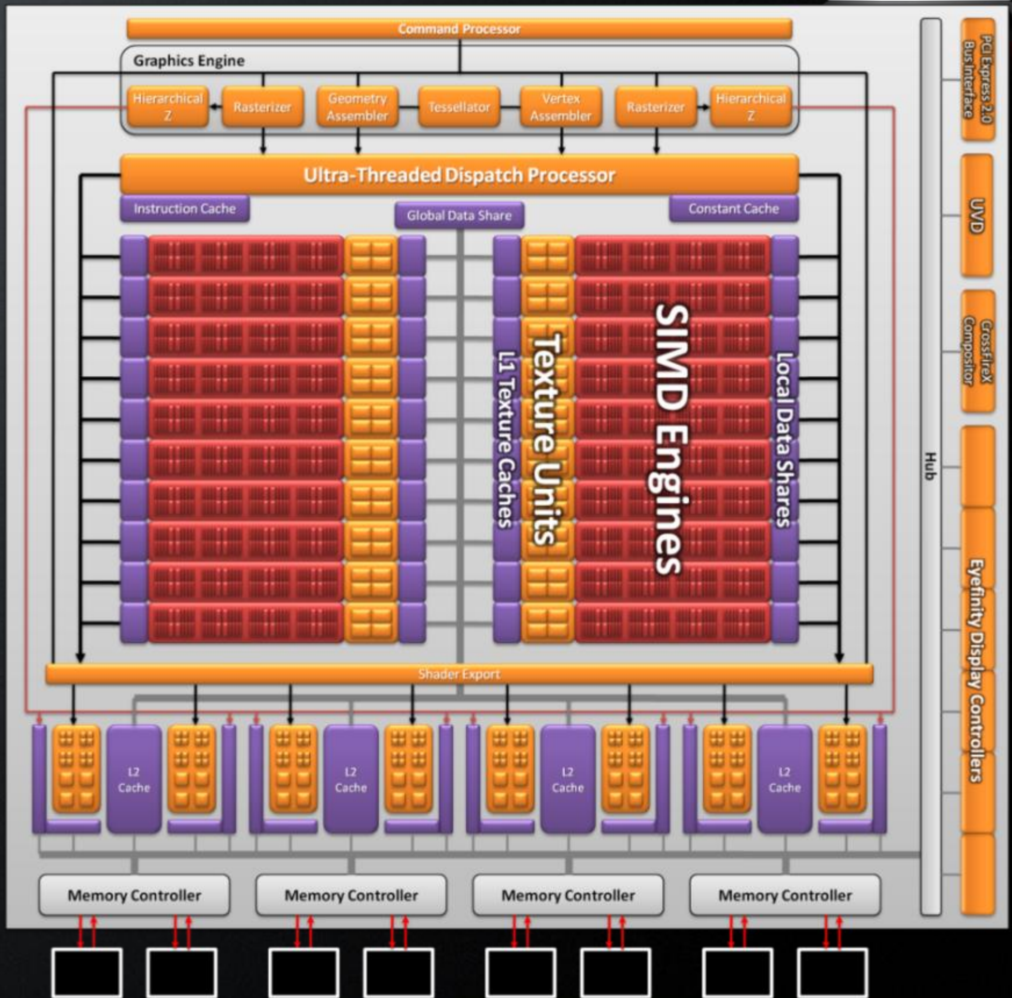
*Based on comparison of consumer client single-GPU configurations as of 12/08/09. ATI Radeon™ HD 5870 provides 14.47 GFLOPS/W and 7.90 GFLOPS/mm² vs. NVIDIA GTX 285 at 5.21 GFLOPS/W and 2.26 GFLOPS/mm².

AMD The future is fusion

ATI Radeon™ HD 5870 (“Cypress”) Architecture (2009)

2.72 Teraflops Single Precision, 544 Gigaflops Double Precision

- Full Hardware Implementation of DirectCompute 11 and OpenCL™ 1.0
- IEEE754-2008 Compliance Enhancements
- Additional Compute Features:
  - 32-bit Atomic Operations
  - Flexible 32kB Local Data Shares
  - 64kB Global Data Share
  - Global synchronization
  - Append/consume buffers



SIMD Cores

Each core:

- Includes 80 scalar stream processing units in total + 32KB Local Data Share
- Has its own control logic and runs from a shared set of threads
- Has dedicated fetch unit w/ 8KB L1 cache
- Communicates with other SIMD cores via 64KB global data share

Thread Processors

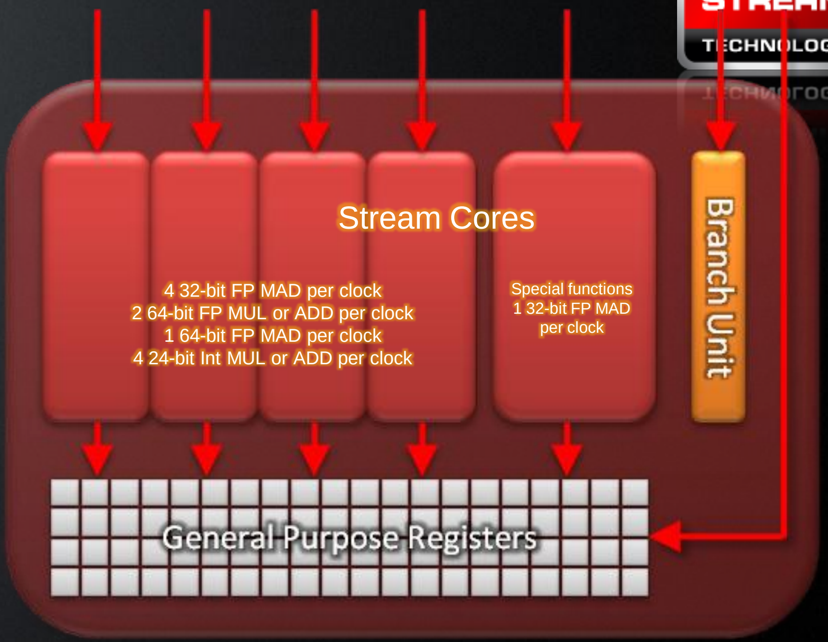
2.7 TeraFLOPS Single Precision

544 GigaFLOPS Double Precision

- 7x more than Nvidia Tesla C1060*

Increased IPC

- More flexible dot products
- Co-issue MUL & dependent ADD in a single clock
- Sum of Absolute Differences (SAD)
  - 12x speed-up with native instruction
  - Used for video encoding, computer vision
  - Exposed via OpenCL extension
- DirectX 11 bit-level ops
  - Bit count, insert, extract, etc.
- Fused Multiply-Add
- Improved IEEE-754 FP compliance
  - All rounding modes
  - FMA (Cypress only)
  - Denorms (Cypress only)
  - Flags

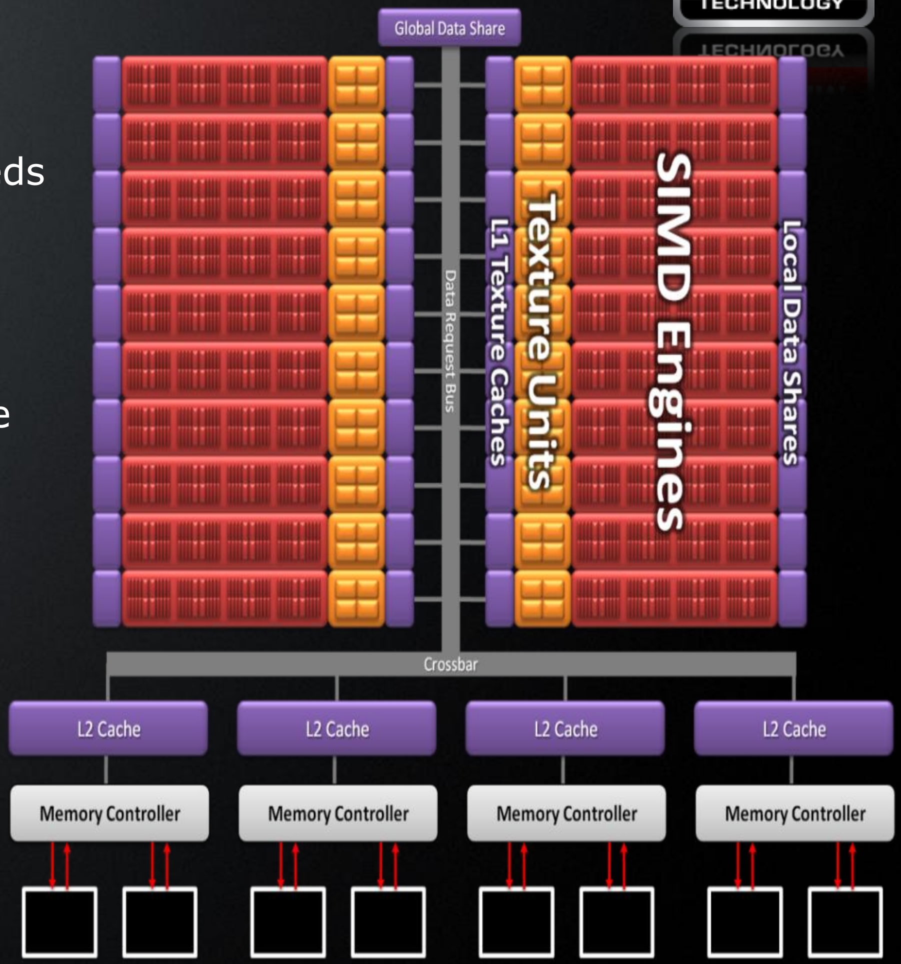


- Each Thread Processor includes:
  - 4 Stream Cores + 1 Special Function Stream Core
  - Branch Unit
  - General Purpose Registers

* Based on published figure of 78 GigaFLOPS

Memory Hierarchy

- Optimized memory controller area
- EDC (Error Detection Code)
  - CRC Checks on Data Transfers for Improved Reliability at High Clock Speeds
- GDDR5 Memory Clock temperature compensation
  - Enables Speeds Approaching 5 Gbps
- Fast GDDR5 Link Retraining
  - Allows Voltage & Clock Switching on the Fly without Glitches
- Increased texture bandwidth
  - Up to 68 billion bilinear filtered texels/sec
  - Up to 272 billion 32-bit fetches/sec
- Increased cache bandwidth
  - Up to 1 TB/sec L1 texture fetch bandwidth
  - Up to 435 GB/sec between L1 & L2
- Doubled L2 cache
  - 128kB per memory controller
- New DirectX 11 texture features
  - 16k x 16k max resolution
  - New 32-bit and 64-bit HDR block compression modes (BC6/7)



OpenCL™: Game-Changing Development

Enabling Broad Adoption of GP-GPU Capabilities

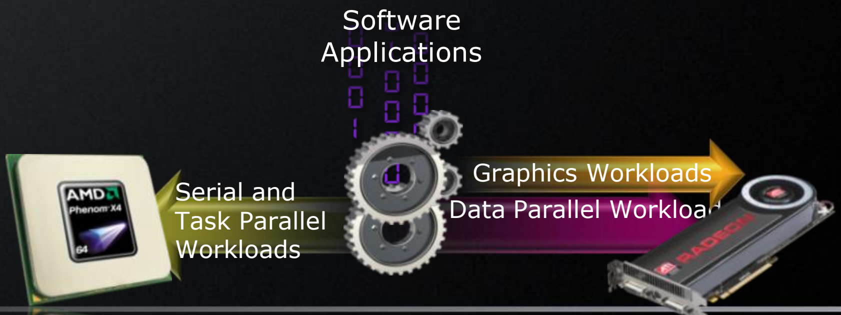
- Industry standard API: Open, multiplatform development platform for heterogeneous architectures
- The power of Fusion: Leverages CPUs and GPUs for balanced system approach
- Broad industry support: Created by architects from AMD, Apple, IBM, Intel, Nvidia, Sony, etc.
- Fast track development: Ratified in December 2008; AMD is the first company to provide a complete OpenCL solution
- Momentum: Enormous interest from mainstream developers and application ISVs

→ More stream-enabled applications across all markets

ATI Stream SDK v2.01:
OpenCL™ For Multicore x86 CPUs and GPUs

The Power of Fusion: Developers leverage heterogeneous architecture to enable superior user experience

- **First complete OpenCL™ development platform**
- **Certified OpenCL™ 1.0 compliant by the Khronos Group**[1]
- Write code that can scale well on multi-core CPUs and GPUs
- AMD delivers on the promise of support for OpenCL™, with both high-performance CPU and GPU technologies
- Available for download now – includes documentation, samples, and developer support



Product Page: http://developer.amd.com/stream

[1] Conformance logs submitted for the ATI Radeon™ HD 5800 series GPUs, ATI Radeon™ HD 5700 series GPUs, ATI Radeon™ HD 4800 series GPUs, ATI FirePro™ V8700 series GPUs, AMD FireStream™ 9200 series GPUs, ATI Mobility Radeon™ HD 4800 series GPUs and x86 CPUs with SSE3.

Anatomy of OpenCL™

Language Specification

- C-based cross-platform programming interface
- Subset of ISO C99 with language extensions - familiar to developers
- Well-defined numerical accuracy - IEEE 754 rounding behavior with defined maximum error
- Online or offline compilation and build of compute kernel executables
- Includes a rich set of built-in functions

Platform Layer API

- A hardware abstraction layer over diverse computational resources
- Query, select and initialize compute devices
- Create compute contexts and work-queues

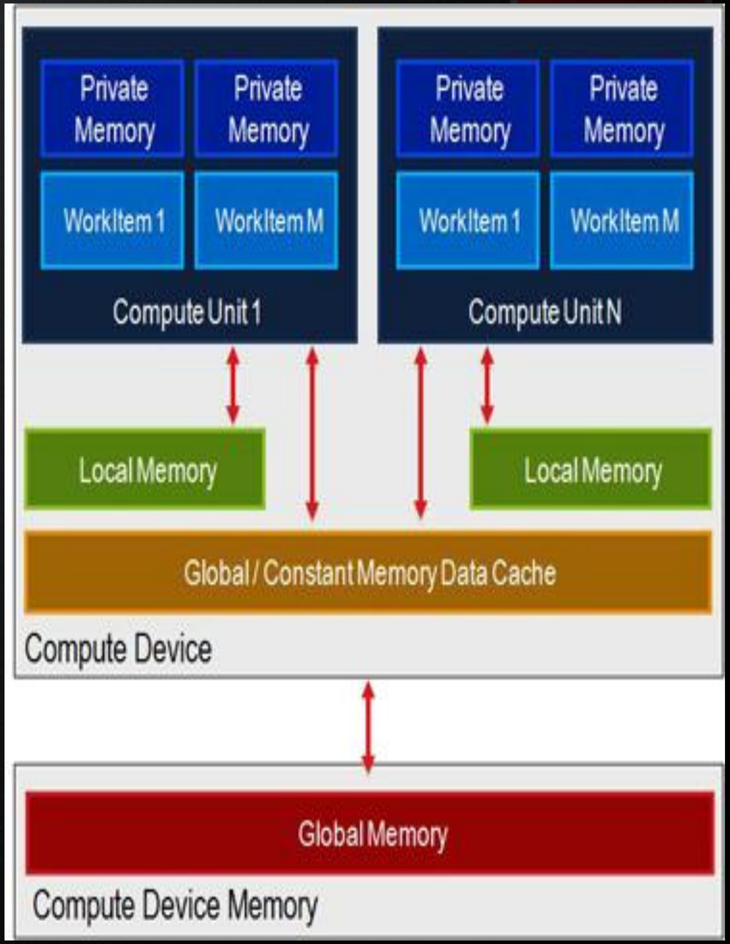
Runtime API

- Execute compute kernels
- Manage scheduling, compute, and memory resources

OpenCL™ Programming Model

Execution Model

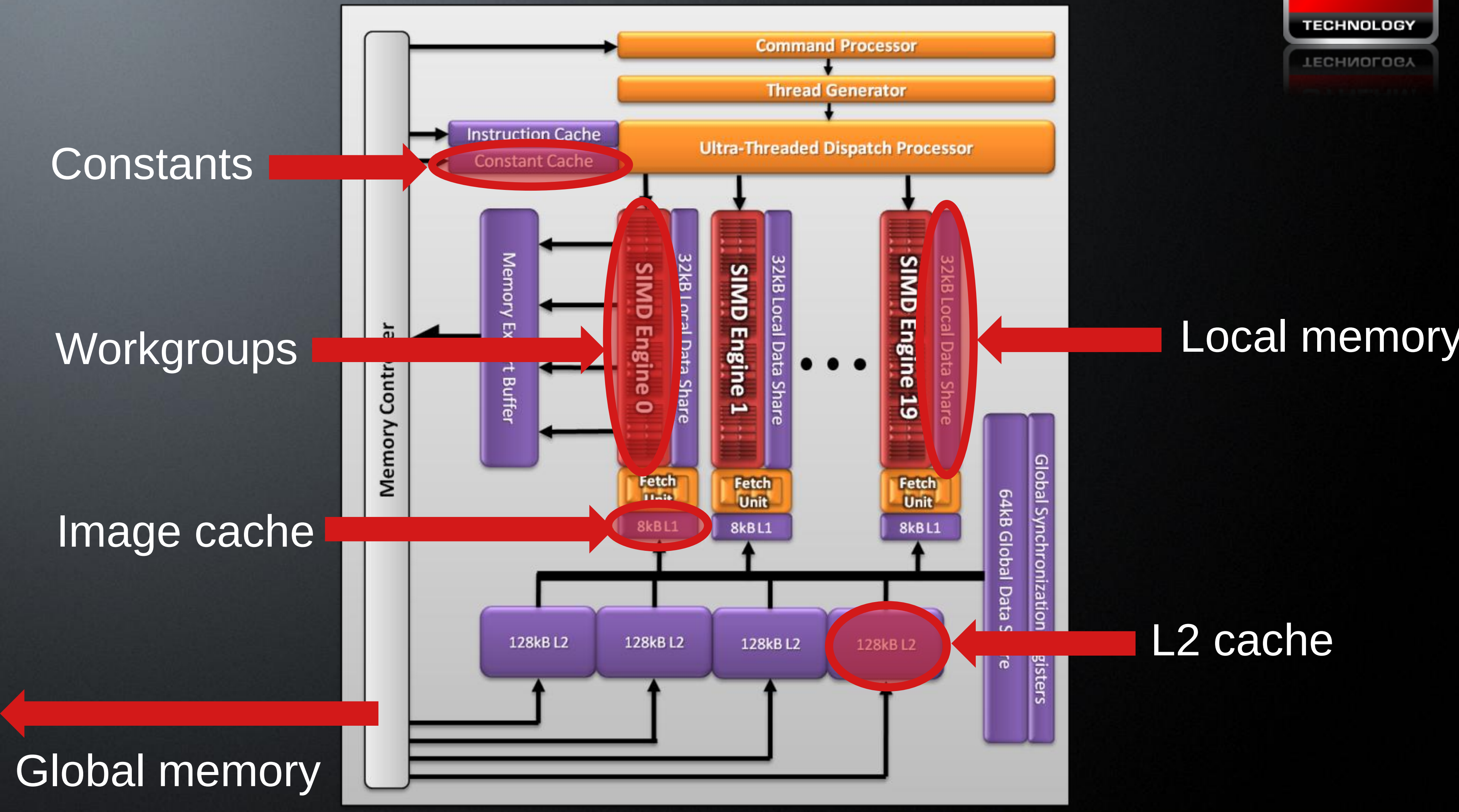
- *Compute kernel* is basic unit of execution
- Execution can occur in-order or out-of-order
- Kernel can be *data-parallel* (GPU) or *task-parallel* (CPU)
- N-dimensional *execution domain* for kernels
- Ability to group *work-items* into *work-groups* for sync/comm



Memory Model

- Multi-level memory model: *private*, *local*, *constant* and *global*

OpenCL™ View



Constants
Workgroups
Image cache
Global memory
Local memory
L2 cache
Command Processor
Thread Generator
Instruction Cache
Constant Cache
Ultra-Threaded Dispatch Processor
Memory Export Buffer
SIMD Engine 0
32kB Local Data Share
SIMD Engine 1
32kB Local Data Share
SIMD Engine 19
32kB Local Data Share
Fetch Unit
8kB L1
128kB L2
128kB L2
128kB L2
128kB L2
64kB Global Data Share
Global Synchronization Registers

OpenCL™ Memory space on AMD GPU

Registers/LDS

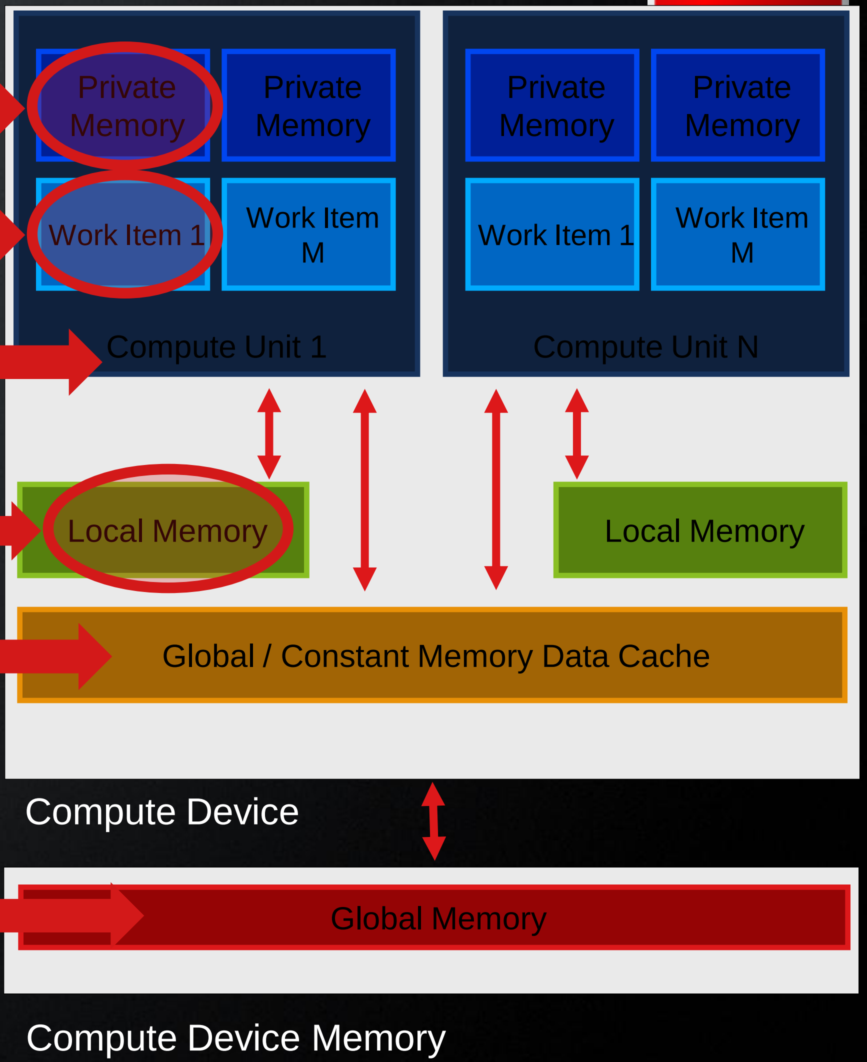
Thread Processor Unit

SIMD

Local Data Share

Board Mem/Constant Cache

Board Memory



Example Walk Through – Kernel

```
__kernel void vec_add (__global const float *a,
                       __global const float *b,
                       __global float *c)
{
    int gid = get_global_id(0);
    c[gid] = a[gid] + b[gid];
}
```

Example Walk Through – Host Code (Init)

```
// create the OpenCL context on a GPU device
cl_context = clCreateContextFromType(0, CL_DEVICE_TYPE_GPU,
                                     NULL, NULL, NULL);

// get the list of GPU devices associated with context
clGetContextInfo(context, CL_CONTEXT_DEVICES, 0, NULL, &cb);
devices = malloc(cb);
clGetContextInfo(context, CL_CONTEXT_DEVICES, cb, devices, NULL);

// create a command-queue
cmd_queue = clCreateCommandQueue(context, devices[0], 0, NULL);

// allocate the buffer memory objects
memobjs[0] = clCreateBuffer(context, CL_MEM_READ_ONLY |
                                     CL_MEM_COPY_HOST_PTR,
                            sizeof(cl_float)*n, srcA, NULL);}
memobjs[1] = clCreateBuffer(context, CL_MEM_READ_ONLY |
                                     CL_MEM_COPY_HOST_PTR,
                            sizeof(cl_float)*n, srcB, NULL);
memobjs[2] = clCreateBuffer(context, CL_MEM_WRITE_ONLY,
                            sizeof(cl_float)*n, NULL, NULL);
```

Example Walk Through – Host Code (Compile)

```
// create the program
program = clCreateProgramWithSource(context, 1, &program_source,
                                    NULL, NULL);

// build the program
err = clBuildProgram(program, 0, NULL, NULL, NULL, NULL);

// create the kernel
kernel = clCreateKernel(program, "vec_add", NULL);
```

Example Walk Through – Host Code (Run)

```
// set the args values
err  = clSetKernelArg(kernel, 0, (void *)&memobjs[0],
sizeof(cl_mem));
err |= clSetKernelArg(kernel, 1, (void *)&memobjs[1],
sizeof(cl_mem));
err |= clSetKernelArg(kernel, 2, (void *)&memobjs[2],
sizeof(cl_mem));

// set work-item dimensions
global_work_size[0] = n;

// execute kernel
err = clEnqueueNDRangeKernel(cmd_queue, kernel, 1, NULL,
                             global_work_size,
                             NULL, 0, NULL, NULL);

// read output array
err = clEnqueueReadBuffer(context, memobjs[2], CL_TRUE, 0,
                          n*sizeof(cl_float),
                          dst, 0, NULL, NULL);
```

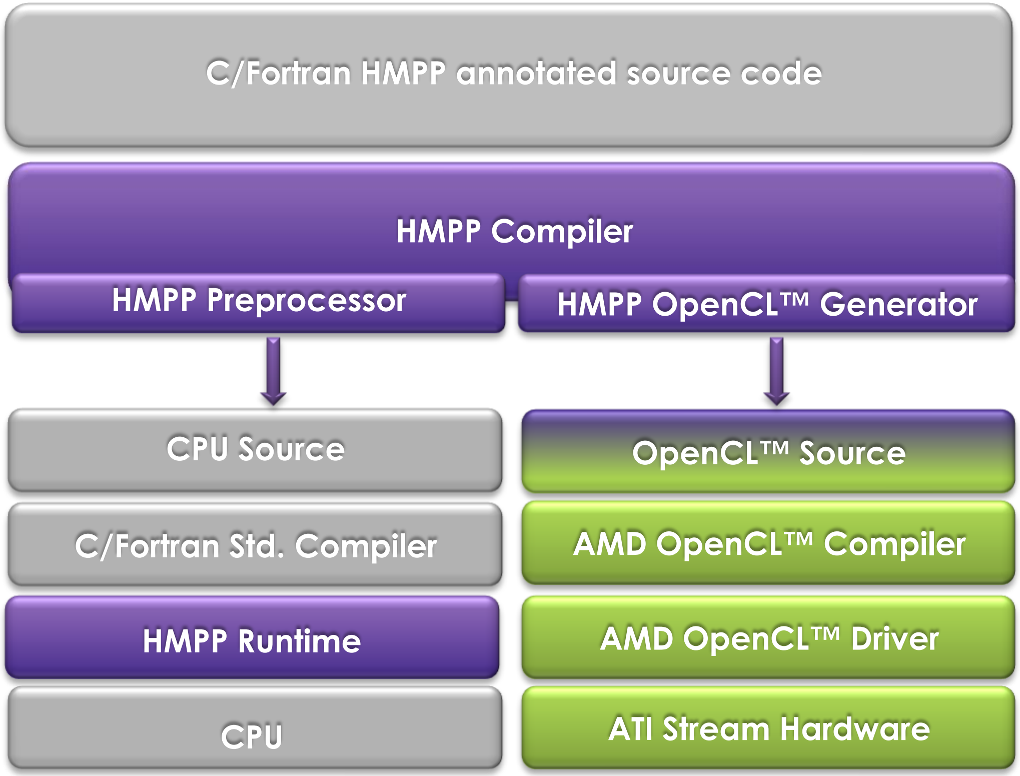
OpenCL™ Development Directions

Khronos Group is working to evolve specification to support future architectural models and features

- Moving compute-oriented optional features into the core specification
  - Double Precision, Atomics
- Developing extensions to support specific applications
  - Video, Physics, etc.
- Improving cross-platform interoperability
- Tightening mathematical precisions
- Developing more advanced scheduling models

OpenCL™ Backend for HMPP (Scheduled to be released)

- A compiler integrating OpenCL™ stream generator
  - Build portable CPU and GPU hardware specific computations
- C & Fortran programming directives
  - High level programming interface for scientific applications
- Runtime library
  - Ease application deployment on multi-GPUs systems



Comparing OpenCL™ and DirectX® 11 DirectCompute

How will developers choose between OpenCL™ and DirectX® 11 DirectCompute?

- Feature set is similar in both APIs

DirectX® 11 DirectCompute

- Easiest path to add compute capabilities to existing DirectX® applications
- Windows Vista® and Windows® 7 only

OpenCL™

- Ideal path for new applications porting to the GPU for the first time
- True multiplatform: Windows®, Linux®, MacOS
- Natural programming without dealing with a graphics API

ATI Stream Technology
Enabled Multimedia Applications

CyberLink

MediaShow 5
MediaShow Espresso
PowerDirector 8
PowerDirector 7

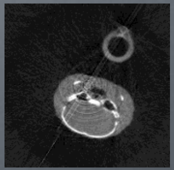
ArcSoft

SimHD™ Plug-in for TotalMedia Theatre

ROXIO

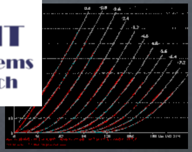
Roxio Creator™ 2010
Roxio Creator™ 2010 Pro

GPU Acceleration in Technical Applications



Tomographic Reconstruction: Alain Bonissent, Centre de Physique des Particules de Marseille

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



EDA Simulation: ACCIT

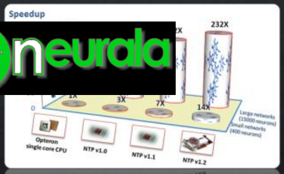
- Currently beta testing applications and reporting >10x speedup**

Seismic Processing: Brown Deer

- Achieving 120x speedup vs CPU on 3D 2nd order finite-difference time-domain (FDTD) seismic processing algorithm

Options Trading: Scotia Capital:

- Reported a 28x speedup over a quad-core CPU.



Neural Networks: Neurala

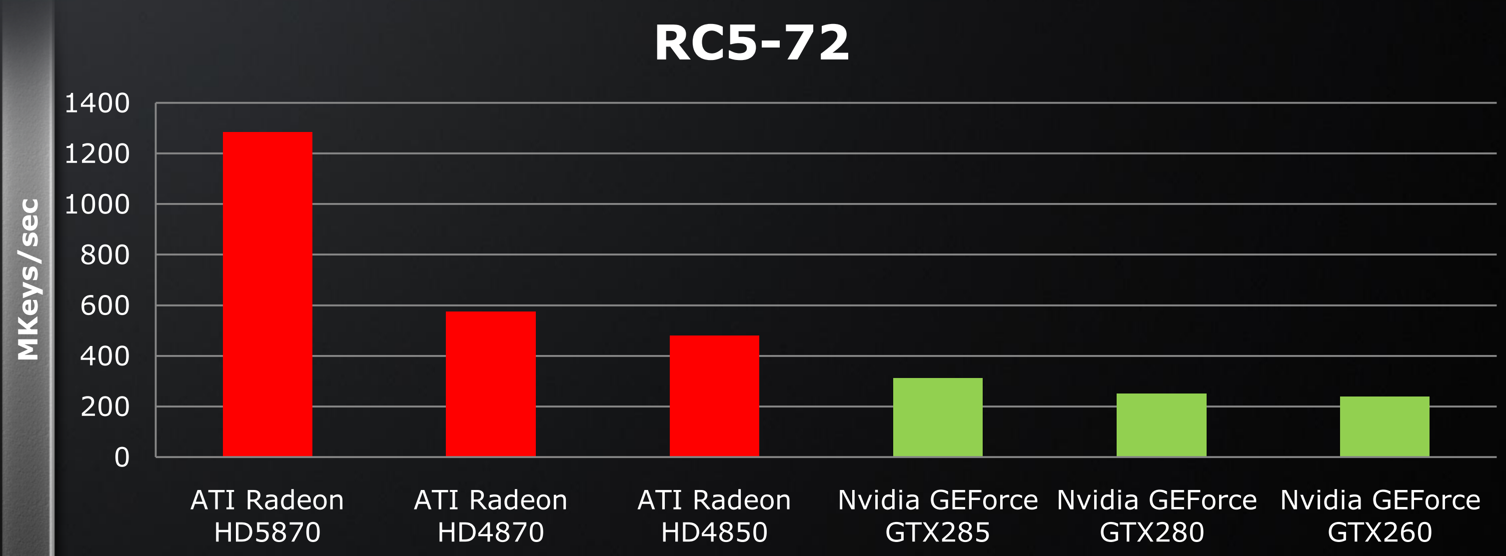
Developing Neurala Technology Platform for advanced brain-based machine learning applications

- Report achieving 10-200x speedups on biologically inspired neural models

High Performance With ATI Stream Technology: Cryptography through Distributed.net

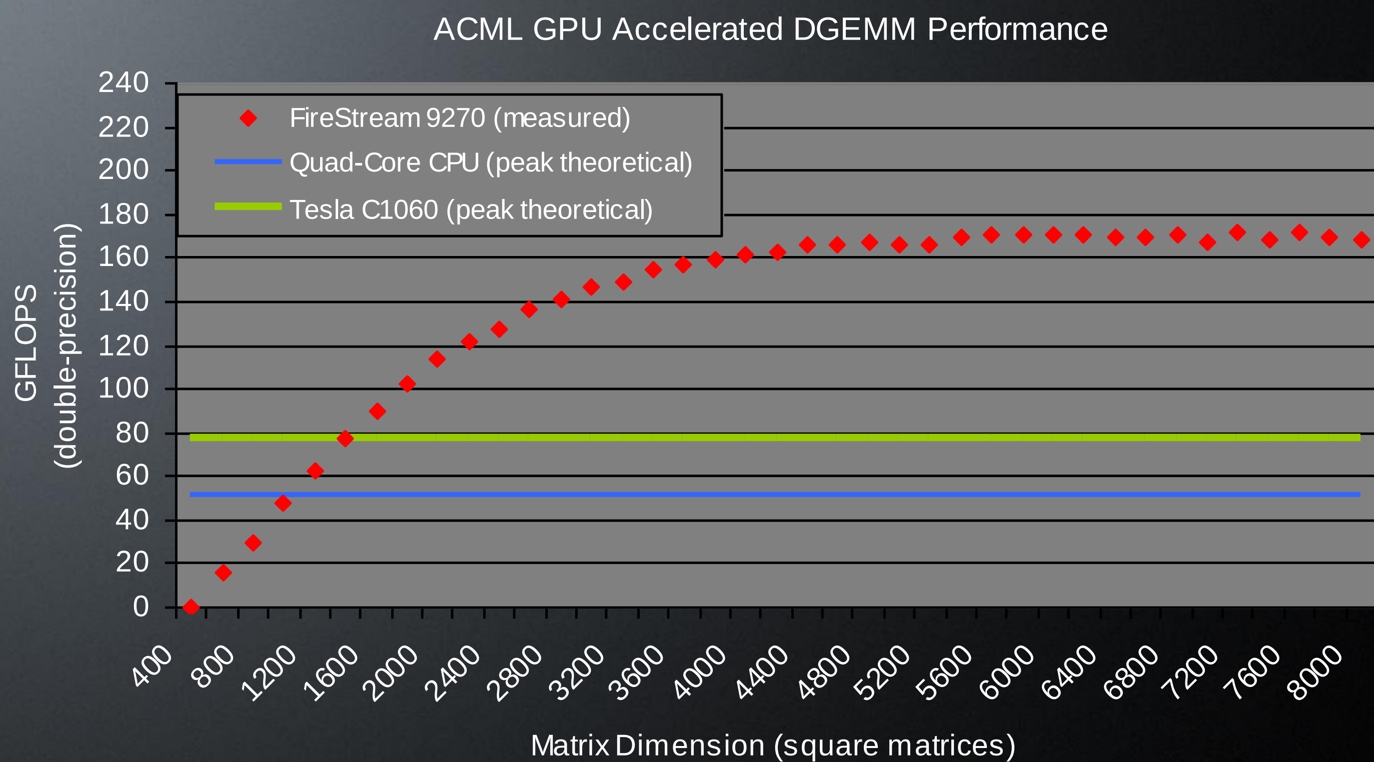
Distributed.net provides a distributed model allowing users to donate compute cycles to large compute-intensive projects

RC5-72 – Cryptography algorithm that searches for encryption keys



*Based on AMD internal testing using RC5-72 clients as of 9/04/09. Results shown in MKeys evaluated per second. Configuration: AMD Phenom™ X4 9950 Black Edition processor, 8GB DDR2 RAM, Windows Vista® 32-bit. AMD drivers: ATI Catalyst™ 9.8 (ATI Radeon™ HD 48xx), prerelease driver (ATI Radeon HD 5870). Nvidia driver: GeForce 190.62. AMD client: [x86/Stream], v2.9106.513 (beta8). Nvidia client: [x86/CUDA-2.2], v2.9105.512 (beta8).

Application Acceleration



- •AMD FireStream™ 9270 on AMD Phenom™ X4 9950/790FX/4GB DDR2 running RHEL 5.1 x86_64
- •AMD FireStream measured performance includes transfer of operand and result matrices
- •Quad-Core peak theoretical performance quoted for 3.2GHz Nehalem processor
- •C1060 peak theoretical performance derived from published specifications
- •ACML-GPU library freely available from: http://developer.amd.com/gpu/acmlgpu

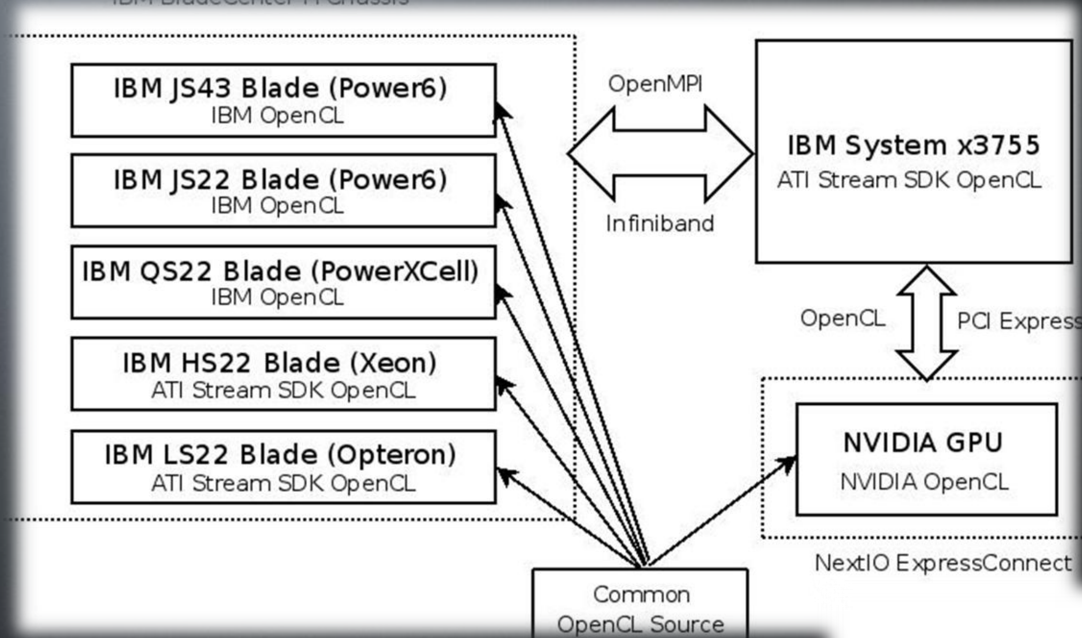
NUDT's Tianhe-1

1.206 Pflops peak - 563.1 Tflops LINPACK

6,144 Intel CPUs - ***5,120 ATI RV770 GPUs***

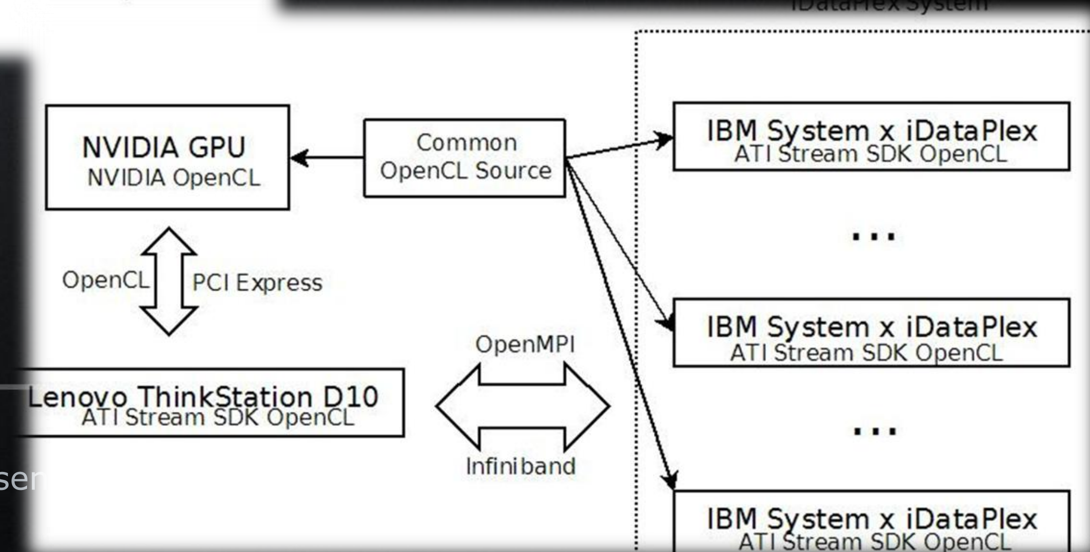
Hybrid Parallel Gas Dynamics in OpenCL™

LANL, IBM, AMD, NVIDIA Booths at SC09



- SC09 demos on
  - •x86 CPU – Opteron
  - •x86 CPU – Xeon
  - •GPU – NVIDIA
  - •GPU – AMD
  - •Power6
  - •PowerXCell

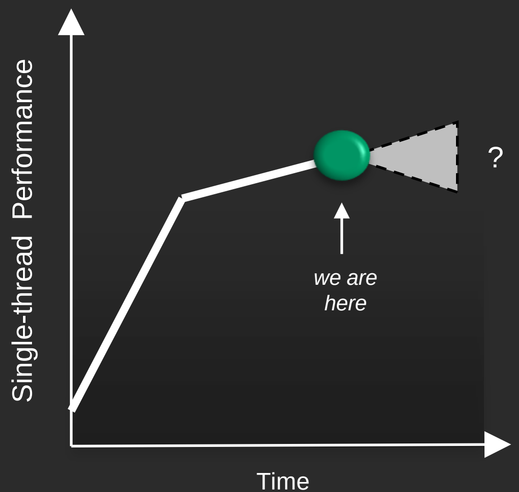
Common Source!!



A New Era of Processor Performance

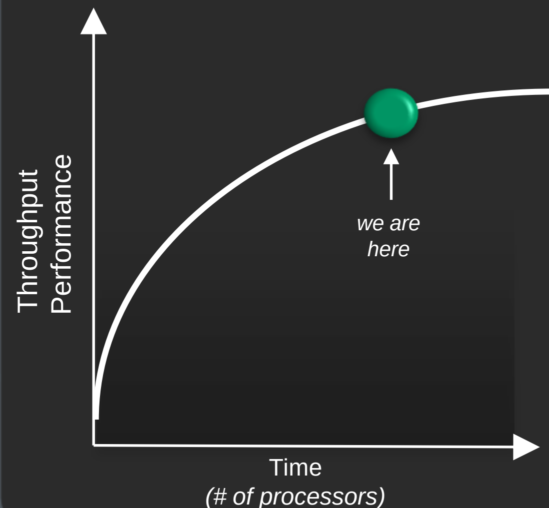
Single-Core Era

Constrained by:
- ✗ Power
- ✗ Complexity



Multi-Core Era

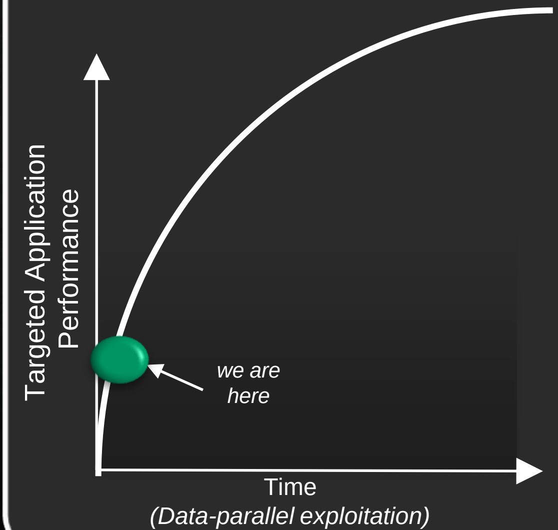
Constrained by:
- ✗ Power
- ✗ Parallel SW availability
- ✗ Scalability



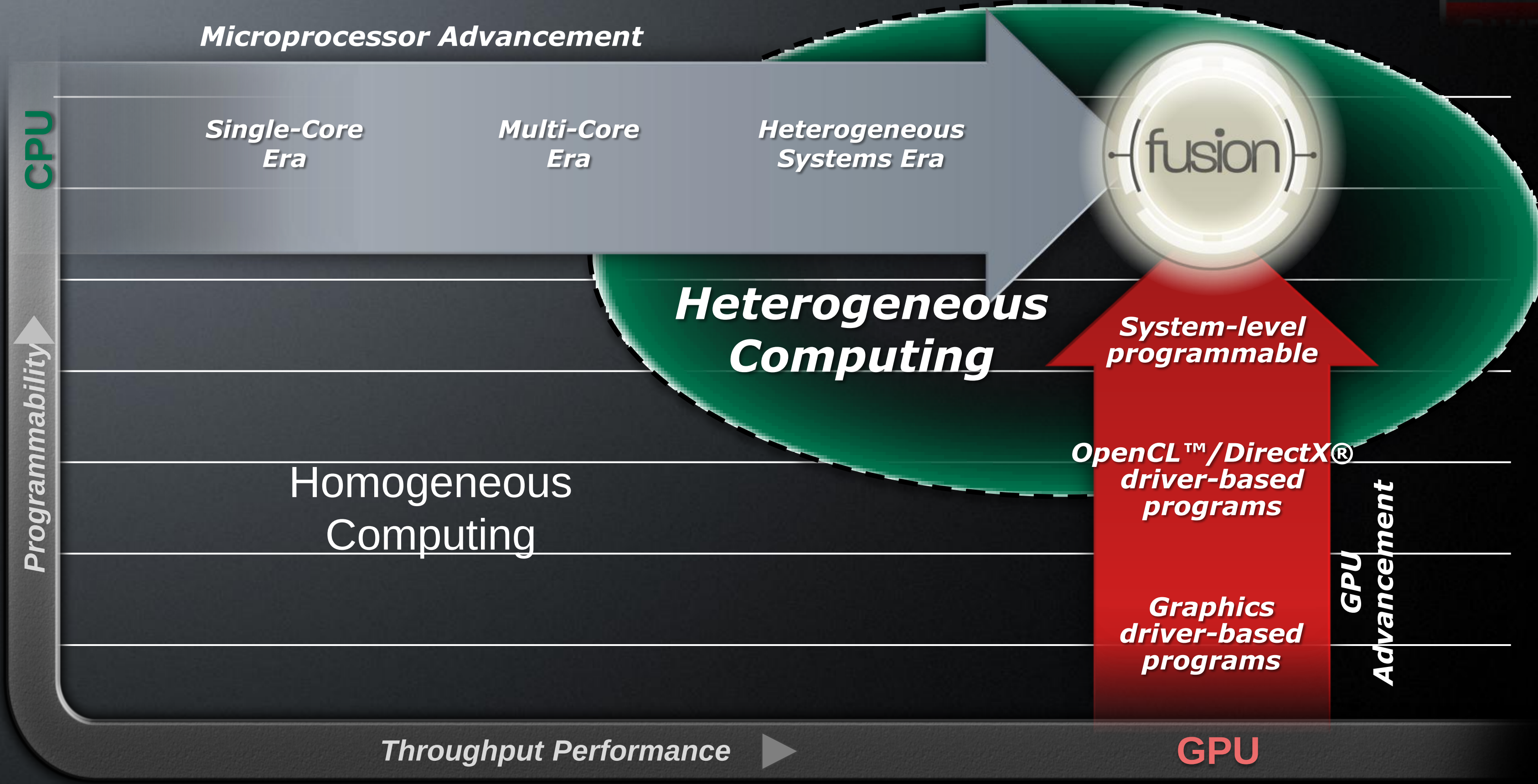
Heterogeneous Systems Era

Enabled by:
- ✓ Abundant data parallelism
- ✓ Power efficient GPUs

Constrained by:
- ✗ Programming models



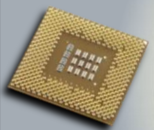
A New Era of Processor Performance

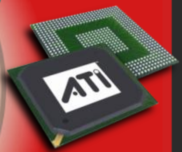


AMD Fusion™ APUs Fill the Need

x86 CPU owns the Software World

- Windows®, MacOS and Linux® franchises
- Thousands of apps
- Established programming and memory model
- Mature tool chain
- Extensive backward compatibility for applications and OSs
- High barrier to entry

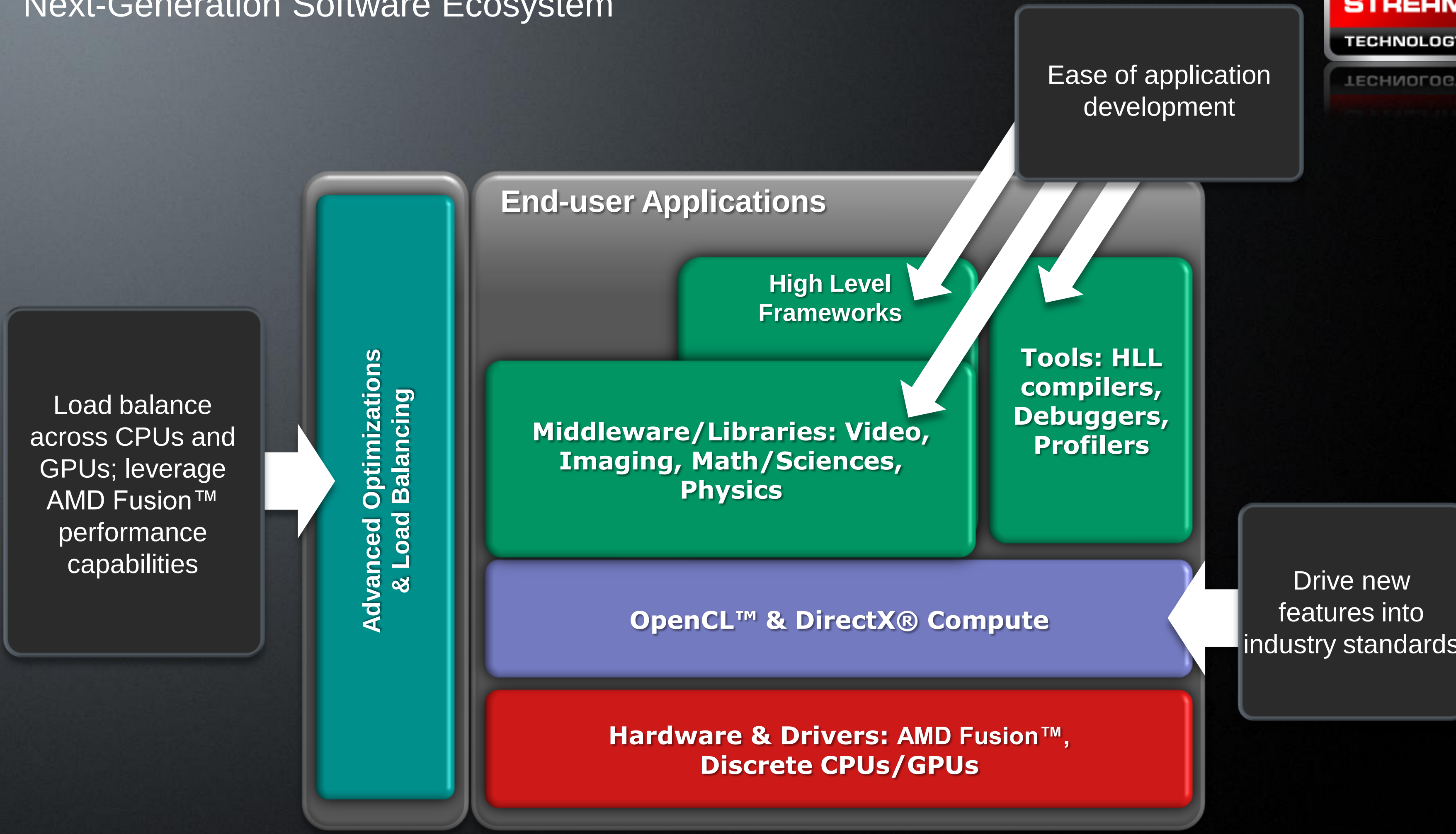




GPU Optimized for Modern Workloads

- Enormous parallel computing capacity
- Outstanding performance-per - watt-per-dollar
- Very efficient hardware threading
- SIMD architecture well matched to modern workloads: video, audio, graphics

Heterogeneous Computing:
Next-Generation Software Ecosystem



ONLY AMD!

AMD
The future is fusion

Backup Slides

Training and Related Resources

- Training Resources
  - Introductory Tutorial to OpenCL™
  - AMD Developer Inside Track: Introduction to OpenCL™
  - ATI Stream OpenCL™ Technical Overview Video Series
  - Porting CUDA to OpenCL™
  - Image Convolution Using OpenCL™ - A Step-by-Step Tutorial
  - OpenCL™ Tutorial: N-Body Simulation
- Related Resources
  - OpenCL™: The Open Standard for Parallel Programming of GPUs and Multi-core CPUs
  - The Khronos™ Group – OpenCL™ Overview Page
  - ATI Stream Profiler Product Page
  - ACML-GPU Product Page
  - ATI Stream Power Toys Product Page
  - ATI Stream Developer Articles & Publications
  - ATI Stream Developer Showcase
  - ATI Stream Developer Training Resources
  - KB75 - Tips and suggestions for running SiSoftware Sandra 2010 OpenCL™ GPGPU benchmarks
- ATI Stream SDK v2.01 Documentation

OpenCL™ vs. CUDA

Feature	OpenCL™	CUDA
Compilation Methods	Online + Offline	Offline Only
Mathematical Precision	Well Defined	*Undefined*
Math Libraries	Defined Standard	Proprietary
CPU Support	OpenCL™ CPU Device	No CPU Support
Native Host Task Support	Task Parallel Compute Model w/ Ability To Enqueue Native Threads	No Native Thread Support
Extension Mechanism	Defined Mechanism	Proprietary
Vendor Support	Industry-Wide Support AMD, Apple, etc.	NVIDIA Only
C Language Support	Yes	Yes

Disclaimer & Attribution

DISCLAIMER

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

The information contained herein is subject to change and may be rendered inaccurate for many reasons, including but not limited to product and roadmap changes, component and motherboard version changes, new model and/or product releases, product differences between differing manufacturers, software changes, BIOS flashes, firmware upgrades, or the like. AMD assumes no obligation to update or otherwise correct or revise this information. However, AMD reserves the right to revise this information and to make changes from time to time to the content hereof without obligation of AMD to notify any person of such revisions or changes.

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

The future is fusion