Applying AMD's "Kaveri" APU for Heterogeneous Computing

DAN BOUVIER, BEN SANDER
AUGUST 2014
## OUR DESIGN CHOICES

### REDESIGNED COMPUTE CORES
- All new architecture for 45%\(^1\) more GPU performance

### HSA FEATURES TO UNLOCK GFLOPS
- Featuring shared system memory
- Heterogeneous Queuing

### ADDED THE LATEST GAMING TECHNOLOGY
- GCN Architecture
- AMD TrueAudio technology *
- Mantle
- PCI-Express® Gen 3

### ENERGY EFFICIENCY
- 95W to 15W solutions featuring configurable TDP

---

*AMD TrueAudio technology is offered by select AMD Radeon™ R9 and R7 200 Series GPUs and select AMD A-Series APUs and is designed to improve acoustic realism. Requires enabled game or application. Not all audio equipment supports all audio effects; additional audio equipment may be required for some audio effects. Not all products feature all technologies—check with your component or system manufacturer for specific capabilities.
A-SERIES REDEFINES COMPUTE

Kaveri

<table>
<thead>
<tr>
<th>4 “Steamroller” CPU Cores</th>
<th>Multimedia</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>AMD TrueAudio technology</td>
</tr>
<tr>
<td></td>
<td>UVD</td>
</tr>
<tr>
<td></td>
<td>VCE</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>8 GCN GPU Cores</th>
<th>Connectivity</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PCIe Gen 3</td>
</tr>
<tr>
<td></td>
<td>PCIe Gen 2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>hUMA Shared Memory Controller</th>
<th>Display</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>DisplayPort 1.2</td>
</tr>
<tr>
<td></td>
<td>HDMI 1.4a</td>
</tr>
<tr>
<td></td>
<td>DVI</td>
</tr>
</tbody>
</table>

MAXIMUM COMPUTE PERFORMANCE

- Up to 12 compute cores*
  - 4 "Steamroller" CPU cores
  - 8 GCN GPU cores
  - HSA enabled

ENHANCED USER EXPERIENCES

- Video acceleration
- AMD TrueAudio technology
- 4 display heads

HIGH PERFORMANCE CONNECTIVITY

- 128bits DDR3 up to 2133
- PCI-Express® Gen3 x16 for discrete graphics upgrade
- PCI-Express® for direct attach NVMe SSD

* For more information visit amd.com/computecores
Applying AMD's "Kaveri" APU for Heterogeneous Computing

**"KAVERI"**

- **Die Size:** 245mm$^2$
- **Transistor count:** 2.41 Billion
- **Process:** 28nm

![Diagram of Kaveri APU architecture](image)
IMPROVEMENTS FOR AMD’S DUAL CPU COMPUTE CORES

- Reduced I-Cache Misses
- Reduced Mispredicted Branches
- Increase in Scheduling Efficiency
- Max-width Dispatches per Thread
- Major improvements in store handling
- Increase Instruction Cache Size

“Steamroller”

- Fetch
- Decode
- Integer Scheduler
- FP Scheduler
- Integer Scheduler
- Pipeline
- Pipeline
- Pipeline
- Pipeline
- Pipeline
- Pipeline
- Pipeline
- Pipeline
- Pipeline
- L1 DCache
- 128-bit FMAC
- 128-bit FMAC
- MMX Unit
- L1 DCache
- Shared L2 Cache
"KAVERI" GPU – GRAPHICS CORE NEXT ARCHITECTURE

47% of "Kaveri" is dedicated for GPU

- 8 compute units (512 IEEE 2008-compliant shaders)
- Device flat (generic) addressing support
- Masked Quad Sum of Absolute Difference (MQSAD) with 32b accumulation and saturation
- Precision improvement for native LOG/EXP ops to 1ULP
“KAVERI” APU ENHANCEMENTS

- Coherent Hub and IOMMU
  - Dedicated coherent transaction path

- AMD Radeon™ Memory Bus
  - High bandwidth graphics data

- PCI-Express® Gen3 for External Graphics Attach

- Four Display Engines

- AMD TrueAudio Technology
INTRODUCTION OF HARDWARE COHERENCY FOR THE GPU

- **Coherent Hub (CHUB)**
  - Compute traffic steered to dedicated coherent transaction path
  - Includes IOMMU Address Translation Cache (ATC)
  - Selectively probe CPU caches based on page attribute

- **Atomics**
  - Single cycle request
  - One at a time (no gathering at the SYS level)
  - All atomics return the original data from DRAM (success of conditional)
  - TYPES Supported:
    - Test and OR, Swap, Add, Subtract, AND, OR, XOR, Signed Min, Signed Max, Unsigned Min, Unsigned Max, Clamping Inc, Clamping Dec, Compare and Swap
IOMMUv2 – ELIMINATES DOUBLE COPY

▲ With traditional memory system
- Not all GPU memory is CPU accessible (e.g. local frame buffer memory)
- Local frame buffer may not be large enough working space
- Lack of demand-paging support
- Alignment limitations
- Data copied from unpinned to pinned regions

▲ With IOMMUv2
- Eliminate CPU & DMA copy operations (in both directions!!)
- GPU operates on unpinned region directly
IOMMUv2 Peripheral Page Faults

- Page tables
- PPR queue
- CMD queue
- Event log
- System Memory
- ATS request
- ATS response
- PRI request
- PRI response
- Evaluate ATS response
- Evaluate PRI response

ATS – Address Translation Services
PRI – Page Request Interface
PPR – Peripheral Page Request
CMD – Command Queue
SW – Software (OS or Hypervisor)

Time
High overhead to pass work to/from GPU

Diagram:
- Application:
  - Transfer Buffer to GPU
  - Queue Job
  - Get Buffer
- OS:
  - Copy/Map Memory
  - Schedule Job
  - Schedule Application
- GPU:
  - Start Job
  - Finish Job
QUEUING WITH HSA ON Kaveri

Shared Virtual Memory

- Application
  - Transfer Buffer to GPU
  - Queue Job
  - Get Buffer

- OS
  - Copy/Map Memory
  - Schedule Job
  - Schedule Application

- GPU
  - Start Job
  - Finish Job
QUEUING WITH HSA ON KAVERI

- Shared Virtual Memory
- System Coherency

Diagram:
- Application
  - Queue Job
    - Get Buffer
- OS
  - Schedule Job
  - Schedule Application
  - Copy/Map Memory
- GPU
  - Start Job
  - Finish Job

Legend:
- Transfer Buffer to GPU
- Copy/Map Memory
- Get Buffer
QUEUING WITH HSA ON KAVERI

- Shared Virtual Memory
- System Coherency
- Signaling

Diagram:

- Application
  - Queue Job
  - Get Buffer
  - Transfer Buffer to GPU

- OS
  - Schedule Job
  - Schedule Application
  - Copy/Map Memory

- GPU
  - Start Job
  - Finish Job
  - Copy/Map Memory
QUEUING WITH HSA ON KAVERI

- Shared Virtual Memory
- System Coherency
- Signaling
- User Mode Queuing

Diagram:

Application
- Transfer Buffer to GPU
- Queue Job
- Get Buffer

OS
- Copy/Map Memory
- Schedule Job
- Schedule Application

GPU
- Start Job
- Finish Job

Legend:
- Application
- OS
- GPU
QUEUING WITH HSA ON KAVERI

- Shared Virtual Memory
- System Coherency
- Signaling
- User Mode Queuing
HSA IN A NUTSHELL

HSA Hardware Building Blocks

- Shared Virtual Memory
  - Single address space
  - Coherent
  - Pageable
  - Fast access from all components
  - Can share pointers

- Architected User-Level Queues

- Signals

Provide industry-standard, architected requirements for how devices share memory and communicate with each other

HSA Software Building Blocks

- HSAIL
  - Portable, parallel, compiler IR

- HSA Runtime
  - Create queues
  - Allocate memory
  - Device discovery

- Reference High-level Compiler
  - CLANG/LLVM
  - Generate HSAIL
  - OpenCL, C++AMP

Provide industry-standard compiler IR and runtime to enable existing programming languages to target the GPU
EVOLUTION OF THE SOFTWARE STACK

Driver Stack

- Domain Libraries
- OpenCL™, DX Runtimes, User Mode Drivers
- Graphics Kernel Mode Driver
- Apps

HSA Software Stack

- HSA Domain Libraries, OpenCL™ 2.x Runtime
- Task Queuing Libraries
- HSA JIT
- HSA Kernel Mode Driver
- Apps

Hardware - APUs, CPUs, GPUs

- User mode component
- Kernel mode component
- Components contributed by third parties
WHAT IS HSAIL?

- Intermediate language for parallel compute in HSA
- Generated by a “High-level Compiler” (GCC, LLVM, Java VM, etc)
- Expresses parallel regions of code
- Portable across vendors and stable across product generations
- Goal: Bring parallel acceleration to mainstream programming languages (OpenMP, C++AMP, Java)

```
main() {
...
#pragma omp parallel for
for (int i=0; i<N; i++) {
}
...
}
```
PROGRAMMING LANGUAGES PROLIFERATING ON HSA

OpenCL™ App
Java App
C++ AMP App
Python App

OpenCL Runtime
Java JVM (Sumatra)
Various Runtimes
Fabric Engine RT

HSA Helper Libraries
HSA Core Runtime
HSA Finalizer

Kernel Fusion Driver (KFD)
HSA ENABLEMENT OF JAVA

JAVA 9 – HSA-ENABLED JAVA (SUMATRA)

- Adds native APU acceleration to Java Virtual Machine (JVM)
- Developer uses Lambda, Stream API
- JVM generates HSAIL automatically
### USE CASES SHOWING HSA ADVANTAGE ON KAVERI

<table>
<thead>
<tr>
<th>Programming Technique</th>
<th>Use Case Description</th>
<th>HSA Advantage</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Data Pointers</strong></td>
<td><strong>Binary tree searches</strong>&lt;br&gt;GPU performs searches in a CPU-created binary tree</td>
<td>GPU can access existing data structures containing pointers&lt;br&gt;Higher performance through parallel operations</td>
</tr>
<tr>
<td><strong>Platform Atomics</strong></td>
<td><strong>Binary tree updates</strong>&lt;br&gt;CPU and GPU operating simultaneously on the tree, both doing modifications</td>
<td>CPU and GPU can synchronize using Platform Atomics&lt;br&gt;Higher performance through parallel operations</td>
</tr>
<tr>
<td><strong>Large Data Sets</strong></td>
<td><strong>Hierarchical data searches</strong>&lt;br&gt;Applications include object recognition, collision detection, global illumination, BVH</td>
<td>GPU can operate on huge models in place&lt;br&gt;Higher performance through parallel operations</td>
</tr>
<tr>
<td><strong>CPU Callbacks</strong></td>
<td><strong>Middleware user-callbacks</strong>&lt;br&gt;GPU processes work items, some of which require a call to a CPU function to fetch new data</td>
<td>GPU can invoke CPU functions from within a GPU kernel&lt;br&gt;Simpler programming does not require “split kernels”&lt;br&gt;Higher performance through parallel operations</td>
</tr>
</tbody>
</table>
Data Pointers
DATA POINTERS

SYSTEM MEMORY

GPU

GPU MEMORY

FLAT TREE
RESULT BUFFER

SYSTEM MEMORY

TREE

RESULT BUFFER

GPU

KERNEL

Flat Tree

Result Buffer
DATA POINTERS

Legacy

SYSTEM MEMORY

GPU

KERNEL

GPU MEMORY

FLAT TREE

RESULT BUFFER
DATA POINTERS

Legacy

SYSTEM MEMORY

GPU

KERNEL

GPU MEMORY

FLAT TREE

RESULT BUFFER

RESULT BUFFER

DATA POINTERS

Legacy

SYSTEM MEMORY

GPU

KERNEL

GPU MEMORY

FLAT TREE

RESULT BUFFER

RESULT BUFFER
DATA POINTERS

HSA

SYSTEM MEMORY

TREE

RESULT BUFFER

GPU

KERNEL

DATA POINTERS

HSA

SYSTEM MEMORY

TREE

RESULT BUFFER

GPU

KERNEL
DATA POINTERS

SYSTEM MEMORY

TREE

RESULT BUFFER

GPU

KERNEL

HSA

HOT CHIPS 26 - AUGUST 2014
static void run_hsa_path()
{
    /* Allocation and initialization */
    tree = (node *) c1SVMAlloc(context, CL_MEM_READ_ONLY, num_nodes * sizeof(node), 0);
    initialize_nodes(tree, num_nodes);
    root = construct_BST(num_nodes, tree);

    search_keys = (int *) c1SVMAlloc(context, CL_MEM_READ_ONLY, num_search_keys * sizeof(int), 0);
    initialize_search_keys(search_keys, num_search_keys, sort_input);

    found_key_nodes = (node **) c1SVMAlloc(context, CL_MEM_WRITEONLY, num_search_keys * sizeof(node), 0);
    memset(found_key_nodes, 0, num_search_keys * sizeof(node));

    /* GPU work enqueue */
    c1SetKernelArgSVMPointer(search_kernel, 0, root);
    c1SetKernelArgSVMPointer(search_kernel, 1, search_keys);
    c1SetKernelArgSVMPointer(search_kernel, 2, &num_search_keys);
    c1SetKernelArgSVMPointer(search_kernel, 3, found_key_nodes);

    c1EnqueueNDRangeKernel(queue, search_kernel, 1, NULL, &num_search_keys, &preferredLocalSize, 0, NULL, &kernel_event);

    c1Finish(queue);

    /* Cleanup */
    c1SVMFree(context, tree);
    c1SVMFree(context, search_keys);
    c1SVMFree(context, found_key_nodes);
}
DATA POINTERS - PERFORMANCE

Binary Tree Search

Search rate (nodes/ms)

Tree size (# nodes)

1M 5M 10M 25M

Search rate

CPU (1 core)
CPU (4 core)
Legacy APU
HSA APU

*Measured in AMD labs

Tree size

1M 5M 10M 25M

Search rate
HSA and full OpenCL 2.0

Both CPU+GPU operating on same data structure concurrently

INPUT BUFFER

GPU

KERNEL

CPU 0

CPU 1

TREE
Both CPU+GPU operating on same data structure concurrently
AMD’S UNIFIED SDK

- Access to AMD APU and GPU programmable components
- Component installer - choose just what you need
- Initial release includes:
  - APP SDK v2.9
  - Media SDK 1.0

APP SDK 2.9

- Web-based sample browser
- Supports programming standards: OpenCL™, C++ AMP
- Code samples for accelerated open source libraries:
  - OpenCV, OpenNI, Bolt, Aparapi
- OpenCL™ source editing plug-in for visual studio
- Now supports Cmake

MEDIA SDK 1.0

- GPU accelerated video pre/post processing library
- Leverage AMD’s media encode/decode acceleration blocks
- Library for low latency video encoding
- Supports both Windows Store and classic desktop
### ACCELERATED OPEN SOURCE LIBRARIES

<table>
<thead>
<tr>
<th>Library</th>
<th>Features</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>OpenCV</strong></td>
<td>Most popular computer vision library</td>
</tr>
<tr>
<td></td>
<td>Now with many OpenCL™ accelerated functions</td>
</tr>
<tr>
<td><strong>Bolt</strong></td>
<td>C++ template library</td>
</tr>
<tr>
<td></td>
<td>Provides GPU off-load for common data-parallel algorithms</td>
</tr>
<tr>
<td></td>
<td>Now with cross-OS support and improved performance/functionality</td>
</tr>
<tr>
<td><strong>clMath</strong></td>
<td>AMD released APPML as open source to create clMath</td>
</tr>
<tr>
<td></td>
<td>Accelerated BLAS and FFT libraries</td>
</tr>
<tr>
<td></td>
<td>Accessible from Fortran, C and C++</td>
</tr>
<tr>
<td><strong>Aparapi</strong></td>
<td>OpenCL accelerated Java 7</td>
</tr>
<tr>
<td></td>
<td>Java APIs for data parallel algorithms (no need to learn OpenCL)</td>
</tr>
</tbody>
</table>
KAVERI OPENS THE GATES TO PERFORMANCE

EQUAL ACCESS TO ENTIRE MEMORY

GPU and CPU have uniform visibility into entire memory space

UNLOCKING APU GFLOPS

Access to full potential of Kaveri APU compute power

ALL-PROCESSORS-EQUAL

GPU and CPU have equal flexibility to be used to create and dispatch work items
**END RESULT: HSA RESULTS IN MORE ENERGY EFFICIENT COMPUTATION**

What does this mean for power?

- Many important workloads execute many times more efficiently using GPU compute resources than CPU only
  - E.g. video indexing, natural human interfaces, pattern recognition

- For the same power, much better performance → finish early (computation, web page render, display update) and go to sleep

**COMPUTE CAPACITY Trend in PCs**

Source: AMD Internal data
Over the last 6 years (2008-2014), AMD achieved a 10x improvement in platform energy efficiency*

Enabled by:
- Intelligent dynamic power management
- Further integration of system components (NorthBridge, GPU, SouthBridge)
- Silicon power optimizations
- Process scaling improvements

Energy use drops ↓ While performance increases ↑ = Increased efficiency +

*Typical-use Energy Efficiency as defined by taking the ratio of compute capability as measured by common performance measures such as SpecIntRate, PassMark and PCMark, divided by typical energy use as defined by $E_{TEC}$ (Typical Energy Consumption for notebook computers) as specified in Energy Star Program Requirements Rev 6.0 10/2013
HSA A KEY ENABLER FOR AN ENERGY EFFICIENT ROADMAP

- Power efficient APUs
- Heterogeneous compute with energy efficient accelerators
- Smart power management
- Integration and miniaturization
With HSA features, Kaveri is an optimized platform for Heterogeneous Computing

- HSA features make Kaveri the FIRST full OpenCL 2.0 capable chip
  - Fine Grained SVM (hUMA)
  - C11 Atomics (Platform Atomics)
  - Dynamic Parallelism (hQ)
  - Pipes (hUMA, hQ)
ENDNOTES


2System configuration “Kaveri” A10-95W TDP, CPU Speed 3.7GHz/4.0 GHz, GPU speed 720MHz, Memory 2x4GB DDR3-1600, Disk HDD, Video Driver 13.35/HAS Beta 2.2, test dates January 1-3, 2014, Microsoft Windows 8.1 (64-bit)
DISCLAIMER & ATTRIBUTION

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

© 2014 Advanced Micro Devices, Inc. All rights reserved. AMD, the AMD Arrow logo and combinations thereof are trademarks of Advanced Micro Devices, Inc. in the United States and/or other jurisdictions. PCI Express is a registered trademark of PCI-SIG Corporation. OpenCL is a trademark of Apple Inc. used by permission by Khronos. Other names are for informational purposes only and may be trademarks of their respective owners.