White Paper | COMPUTE CORES

TABLE OF CONTENTS

A NEW ERA OF COMPUTING

Over the past 40 years the mechanisms used to advance microprocessor performance and capabilities have evolved greatly, from executing a single instruction at a time on a single processor to executing many parallel instructions on many symmetric cores. Most recently a microprocessor capability might have been determined by its operating frequency and/or the number of CPU cores. With the continued integration of different processor cores, accelerators, and other processing elements, the traditional labeling has become less representative of a processor's capability. Now with the introduction of single chip heterogeneous computing, a new measure of identification and comparison is required. To this end, the best general measure for the next generation of processors is to count all execution units that are capable of performing in the same compute capacity as the traditional central processing unit (CPU). In this paper, these will be referred to as "Compute Cores".

HISTORY OF PROCESSORS

The CPU Era
The first microprocessors began as stand-alone/ discrete CPUs, often executing less than a single instruction per clock cycle. While CPUs might be thought of in relation to PCs, the early versions were used in less sophisticated applications like calculators and other control applications. Advancements in manufacturing technology quickly led to more general purpose and programmable microprocessors. While it was natural to add memory to the CPU, most other functions, even other mathematical compute functions like floating-point processing, remained as separate devices to the CPU. Despite the limited functionality of these first microprocessors, they represented the birth of computing as we know it today.

The rapid progression of semiconductor photolithography and manufacturing technology has enabled a reduction in die size effectively doubling the number of transistors in a device area every 18 to 24 months. This trend, referred to as "Moore's Law", has been a driving factor in the evolution of the microprocessor. The challenge presented to microprocessor architects and software developers has been how to harness the logarithmic increase in transistors to the benefit of consumers.

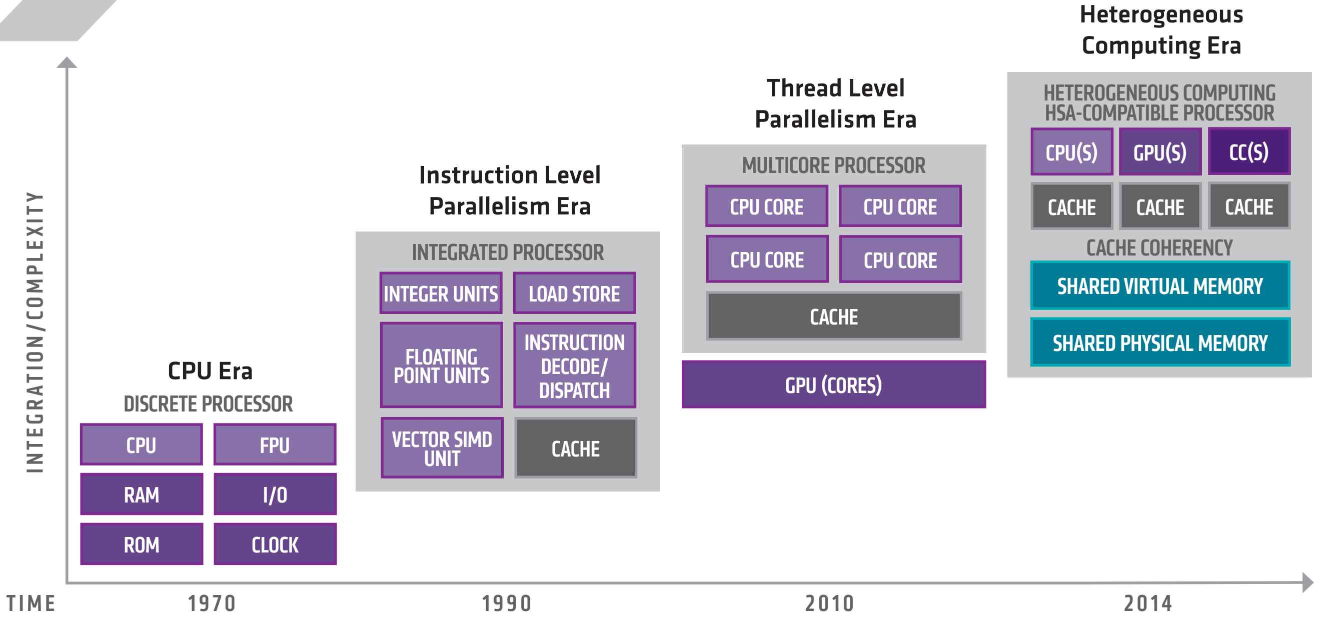
The early years of processor development were directed toward adding more sophisticated instructions, increasing the execution register width, and finally integration of a floating point math unit.

The Instruction-Level-Parallelism Era
The early processors executed instructions in the order placed by their programmers.It was noted in most programs, instruction sequences existed with no direct data dependencies to each other. In such cases, performance could be increased by executing these sequences of instructions in parallel. This led to more sophisticated microarchitectures with parallel execution units exploiting this instruction-level-parallelism (ILP). Combined with out-of-order execution and super-pipelining a rapid increase occurred in not only the number of instructions per cycle (IPC) executed but also the clock frequency of the microprocessor. The microprocessors capability and performance became labeled by its operating frequency.

The next functional unit added to microprocessors was the single-instruction-multiple-data (SIMD) vector unit. In this unit a single instruction causes the same operation to multiple data elements in parallel. This significantly increased the throughput of the microprocessor core.

Thread Level Parallelism Era
In the mid 2000's the complexity and speeds of a single processor core microarchitecture reached an apex where it became impractical to gain performance at the same historical rate through use of the same class of mechanisms. A new scheme would be necessary to continue the performance trend.

In the 1990's operating systems became more sophisticated enabling more than one application to be active simultaneously. This new generation of operating system further allowed applications to be broken into independent threads that could work together in parallel. Threads could be distributed across multiple discrete microprocessors in a system thus exploiting thread-level-parallelism (TLP) to gain performance. In order to maintain consistent operating results the processors deployed were expected to be identical to each other forming a symmetric-multi-processor (SMP) system. With the continued growth in available transistors it was a natural next step to combine multiple cores into a



single multicore device. The generational frequency uplift trend had slowed driving need for a different labeling scheme. This new generation of processors was to be classified by CPU core count.

Besides including general purpose integer and floating point math execution units, each core in these multicore processors contained parallel vector SIMD execution units such as the x86 Advanced Vector Extensions (AVX.) Including these SIMD units was thought to be the answer to driving continued performance efficiency. However, unleashing this performance by the programmer was a tedious task due to limited working register resources and working in the confines of a load/store programming model of a general purpose CPU.

The GPU for general purpose processing

The increase in graphic-intensive content in the mid 1990's led to the addition of a specialized graphics processing unit to the system. The GPU was tailored to manipulate memory to accelerate image creation in a frame buffer for display. Graphics processing evolved to include 3D, rich texture mapping, and visualization effect generation. The GPU architecture evolved in flexibility to handle many different data types and processing scenarios with high throughput efficiency. The capability of the GPU was soon recognized and adopted into other non-graphics applications that exhibited similar attributes. Combining such discrete GPUs into systems with CPUs opened up new options for heterogeneous processing. In this way application code could be partitioned into those elements best executed on a traditional microprocessor and those more amenable to a GPU.

Deployment of the GPU in general computing applications presented a challenge to the programmer. Programming languages and a software interface to host code running on the main microprocessor, was needed to exploit this new level of parallelism. The resulting approaches born out were the CUDA and OpenCL programming models.

The Heterogeneous Computing Era

Once again the increase in available transistors provided the opportunity to integrate CPU with GPU into a single device. The close proximity of these two processor types offered the ability to more tightly couple the memory system to enable direct sharing of data structures.

Faced with limitations in silicon scaling (maintaining Moore's Law) and the driving need to improve performance and efficiency, the electronics industry is banding together to change the very nature of computing. This is the transition to "heterogeneous computing" where the various execution units are more tightly integrated and share system responsibility and resources. The critical part of this is elevating the other programmable execution units like the GPU to the same level of the CPU for memory access, queuing, and execution. In other words, rather than having a CPU and various co-processors, these various processor elements can be referred to in combination as "Compute Cores."

> Compute Core: *Any core capable of running at least one process in its own context and virtual memory space, independently from other cores.*

The Compute Core represents a change in the hardware and software architecture. In a heterogeneous processor, merely representing the number of CPU cores would misrepresent the potential compute capability of the complete device. As a result, AMD is changing its nomenclature to meet the requirements of a heterogeneous computing platform.

THE COMPUTE CORE NOMENCLATURE

Beginning in 2014, products from various semiconductor companies, particularly those that are members of the HSA Foundation, will begin introducing heterogeneous devices. To assist in evaluating these new processor solutions, AMD is adopting the Compute Core nomenclature as a designation for processors that meet the HSA specifications and as a general method of comparing these solutions. With the first generation of heterogeneous processors based on the architecture of the APU codenamed "Kaveri", AMD will begin designating the number of compute cores in the following manner.

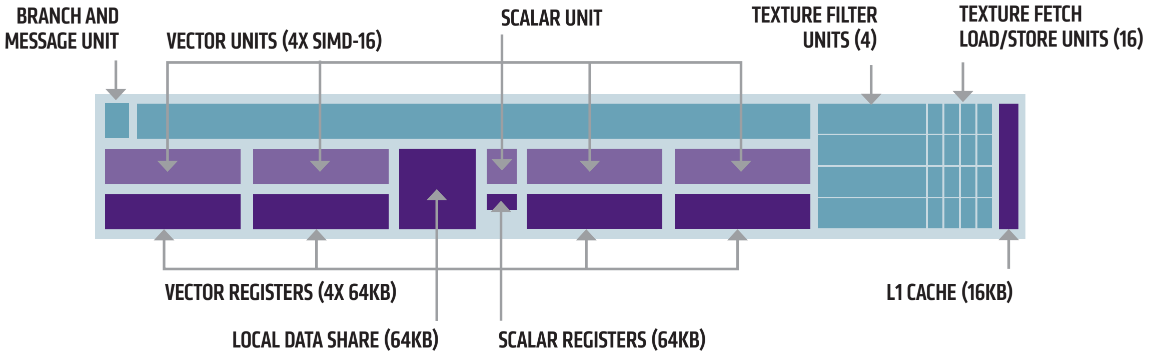
AMD A10-7850K APU with Radeon™ R7 graphics
12 Compute Cores (4 CPU + 8 GPU)

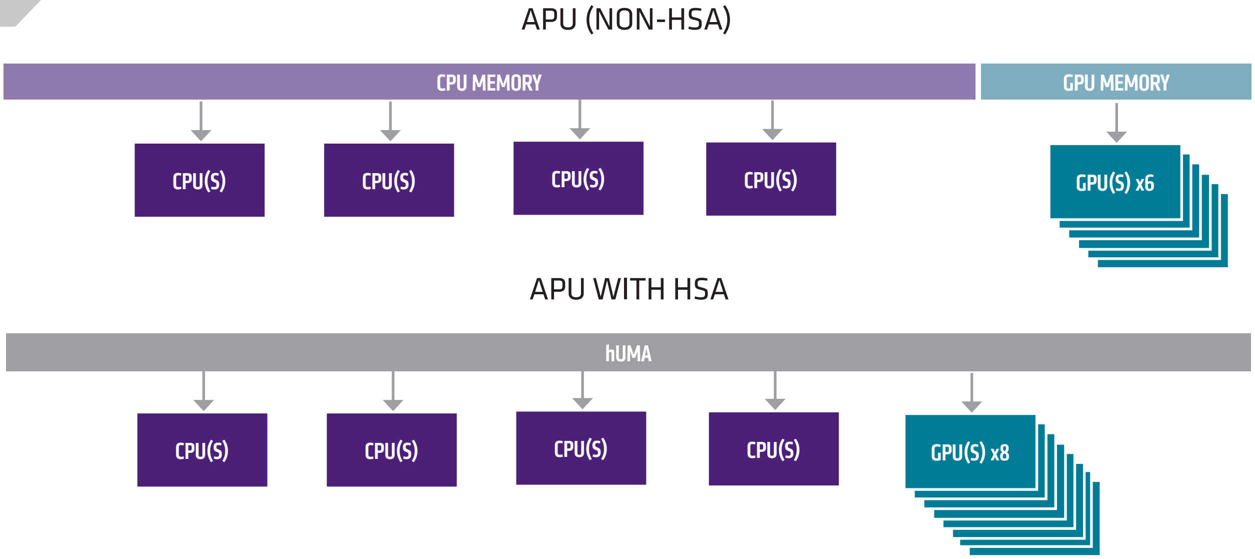
AMD'S HETEROGENEOUS PLATFORM

"Kaveri" represents the latest in AMD's processor technology and first heterogeneous computing platform by combining up to four high-performance Steamroller CPU cores with eight Graphics Core Next (GCN) GPU cores. AMD's Steamroller compute cores are highly optimized x86 cores for parallel instruction execution. The cores are paired together in modules and feature independent instruction decoders, enhanced branch prediction units, redesigned caches, and ALUs.

AMD's GCN compute units are grouped together by a dedicated scheduler that feeds four 16-wide SIMD vector processors, a scalar processor, local data registers and data share memory, a branch & message processor, 16 texture fetch or load/store units, four texture filter units, and a texture cache. Some of these characteristics are very similar to how CPU cores are grouped together, and offer similarities of being able to independently execute work-groups in parallel. As a result, each one of these GPU compute units operates in a manner comparable to a CPU core. However, unlike a CPU core which is designed and optimized to handle serial tasks, the "GPU core" is designed and optimized to handle parallel tasks.

"Kaveri" combines these CPU and GPU technologies with an integrated hardware scheduler and the supporting libraries and tools for the industry's first heterogeneous computing platform as defined by the upcoming HSA Foundation specification.





SUMMARY

Thus far, the processor has evolved through increases in performance and integration, but the operation and execution has remained relatively constant. Heterogeneous computing represents a major change in both the hardware architecture of processors as well as the structure of software operating on them. Processors such as AMD's new "Kaveri" will usher in a new era of processor performance and efficiency.

To coincide with this change, the use of the compute cores nomenclature will provide a more accurate representation of a heterogeneous processor. AMD recognizes that not all computing cores are the same and the nomenclatures should not be considered the measurement as the overall performance of a processor.

The HSA Foundation seeks to create applications that seamlessly blend scalar processing on the CPU, parallel processing on the GPU, and optimized processing on the DSP via high bandwidth shared memory access enabling greater application performance at low power consumption. The Foundation is defining key interfaces for parallel computation utilizing CPUs, GPUs, DSPs, and other programmable and fixed-function devices, thus supporting a diverse set of high-level programming languages and creating the next generation in general-purpose computing.

Most importantly, the HSA Foundation is driving greater developer productivity in heterogeneous computing by removing the barriers of traditional heterogeneous programing and allowing developers to do what they do best, focus on algorithms instead of managing system resources.

THE HSA FOUNDATION

The HSA (Heterogeneous System Architecture) Foundation is a not-for-profit consortium of SoC IP vendors, OEMs, academia, SoC vendors, OSVs and ISVs that is redefining how the system architecture can integrate CPUs, GPUs, DSPs, and accelerators to dramatically ease programming on heterogeneous parallel devices.

The HSA Foundation members are building a heterogeneous compute software ecosystem built on open, royalty-free industry standards and open-source software: the HSA runtimes and compilation tools are based on open-source technologies such as LLVM and GCC.

www.amd.com/computecores