Realizing the AMD Exascale Heterogeneous Processor Vision

Industry Product

Alan Smith, Gabriel H. Loh, Michael J. Schulte, Mike Ignatowski, Samuel Naffziger, Mike Mantor, Mark Fowler
Nathan Kalyanasundharam, Vamsi Alla, Nicholas Malaya, Joseph L. Greathouse, Eric Chapman, Raja Swaminathan
Advanced Micro Devices, Inc.

***Abstract* – AMD had previously detailed its exascale research journey from initial targets and requirements to the development and evolution of its vision of a high-performance computing (HPC) accelerated processing unit (APU), dubbed the Exascale Heterogeneous Processor or EHP. At the conclusion of that work, the learnings were integrated into the design of the node architecture that went into the Frontier supercomputer, the world's first exascale machine. However, while the Frontier node architecture embodied many of the attributes of the EHP concept, advanced heterogeneous integration capabilities at the time were not yet sufficiently mature to realize our vision of a fully-integrated APU for HPC and AI. In this paper, we finish the EHP's story by digging deeper into why an APU was not the right solution at the time of our first exascale architecture, what the shortcomings were of previous EHP concepts, and how AMD further evolved the concept into the AMD Instinct™ MI300A APU. MI300A is the culmination of years of AMD developments in advanced packaging technologies, its APU hardware and software, and the next step in our highly effective chiplet strategy to not only deliver a groundbreaking design for exascale computing, but to also meet the demands of new large-language model and generative AI applications.**

I. Introduction

Over a decade ago, the U.S. Department of Energy (DOE) created an ambitious set of goals to guide their exascale program [7]. In particular, the DOE was concerned that technical challenges like the end of Moore's Law [23] and Dennard scaling [9], power consumption, the memory wall [43], and reliability at scale could significantly delay the introduction of exascale-class supercomputers. The DOE administered a series of advanced research public-private partnerships to enable and accelerate the pre-exascale research needed to achieve exascale compute capabilities.

AMD was selected to participate in these research programs. Key aspects of our exascale research vision and how they evolved and ultimately led to the world's first exascale supercomputer, Frontier at Oak Ridge National Laboratory [28][29][30], were detailed in our ISCA 2023 Industry Track paper [22]. In particular, our research vision proposed the "Exascale Heterogeneous Processor" (EHP), a high-performance accelerated processing unit (APU) consisting of CPU and GPU compute along with multiple stacks of high-bandwidth memory (HBM) and other system-on-a-chip (SoC) components all integrated into a single physical processor package. For a variety of factors to be discussed in this paper, what was ultimately deployed in the Frontier supercomputer's node architecture had a logical organization that reflected many of the key architectural attributes of the EHP, but was constructed out of separate, discrete packages.

*This paper is part of the Industry Track of ISCA 2024's program

Lawrence Livermore National Laboratory announced that their upcoming El Capitan supercomputer will utilize AMD Instinct™ MI300A APUs [37]. The MI300A APU represents the realization and embodiment of the original EHP concept, combining compute and memory components in a single advanced package. This paper provides an inside view into: why AMD did not initially pursue the EHP in the first generation exascale machine; remaining challenges with the original EHP concept; technical details of the AMD MI300A APU and how it ultimately surpasses the EHP; and the evolution of the AMD chiplet strategy into a modular 3D chiplet platform that also enables the AMD Instinct™ MI300X accelerator for targeting the rapid explosion in computational demands for machine learning (ML) large-language models (LLM) and generative AI workloads.

II. Background and Context

This section provides a brief review of the AMD EHP concept to provide the necessary context for Section III, where we discuss why the EHP was not utilized in the first generation AMD exascale approach. For the full story of the development of the original EHP and the Frontier supercomputer, we refer the reader to our previous retrospective paper [22].

A. EHP Overview

Our original EHP research concepts changed through several different incarnations based on evolving information over time about technology capabilities and limitations, alignment with product roadmaps, and overall project performance and power targets. An APU is attractive for HPC supercomputer scenarios because the GPU components provide very high computational throughput per Watt and per unit volume (m^3) to maximize performance within the constraints of datacenter compute density and power consumption limits. The CPU components are still critical for code sections that are irregular or otherwise difficult to parallelize on the GPU (Amdahl's Law must still be reckoned with), as well as in the dispatch and orchestration of GPU compute tasks. Co-packaging the CPU and GPU can also reduce data movement latencies, improve bandwidth between components, and improve synchronization and coordination overheads. The EHP also proposed using in-package HBM, which enables "zero copy" memory allocation and management between the CPU and GPU. The CPU can initialize data directly in the HBM, and then kernels dispatched to the GPU can likewise directly access the same physical memory without explicitly copying the data from host to device memory as is typically needed when the CPU and GPU are implemented in discrete packages. Beyond the performance and power benefits of avoiding the data copies, this unified shared memory architecture also provides programmability benefits, as users are now

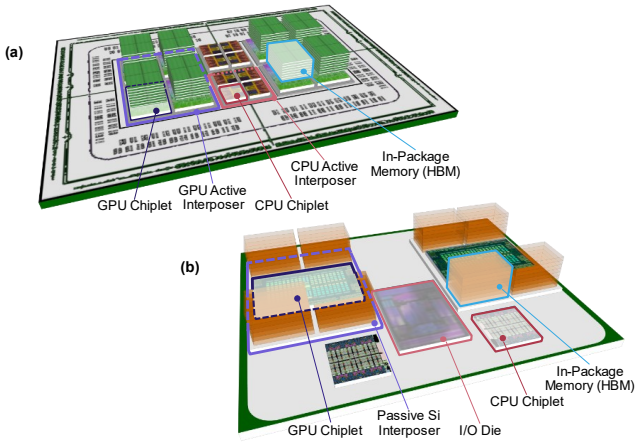

Figure 1. (a) "Version 3" of the Exascale Heterogeneous Processor (EHP) concept from the AMD pre-exascale research explorations [42], and (b) "Version 4" of the EHP [22].

freed from orchestrating the data copies from the CPU to the GPU and back. Placing everything in the same package also introduces new opportunities to dynamically shift power between the components based on workload characteristics.

Figure 1(a) shows "version 3" of the EHP (EHPv3), which included the use of active silicon interposers 3D-stacked with compute chiplets and high-bandwidth memory (HBM) [20][31][42]. The main reason for the aggressive use of stacking in this concept was to maximize the total compute and memory within the processor package. Due to the projected technologies that would be sufficiently mature for high-volume manufacturing in the timeframe of our first exascale architecture, our final "version 4" of the EHP (EHPv4), shown in Figure 1(b), backed away from a 3D-stacked approach and instead relied on 2D chiplets on an organic substrate [27] combined with 2.5D passive silicon interposer integration [12] of the GPU and HBM. EHPv4 still maintained the key attributes of combining CPU, GPU, and HBM in a single package, but it was less aggressive in terms of its use of advanced integration and packaging technologies compared to EHPv3.

B. Frontier Node Architecture Overview

For a variety of factors to be discussed in Section III, the node architecture used in the Frontier supercomputer did not utilize an APU. Figure 2 shows the actual node architecture consisting of a CPU and four discrete GPU accelerators connected with AMD Infinity Fabric™ coherent interconnect [30][32]. As discussed in our prior retrospective paper [22], the underlying components of the Frontier node architecture, which has been widely deployed in several other supercomputers and datacenters [23], effectively consist of four instances of the EHP conjoined by a common I/O die (IOD). For example, the components within each of the four different-colored boxes in Figure 2 consist of two CPU chiplets, two large GPU die, and eight stacks of HBM, which match the compute and memory components of one EHPv4. A flat memory address space with cache coherence between the CPU and GPU portions enabled an APU-like view of the different components: architecturally unified although implemented in physically distinct packages.

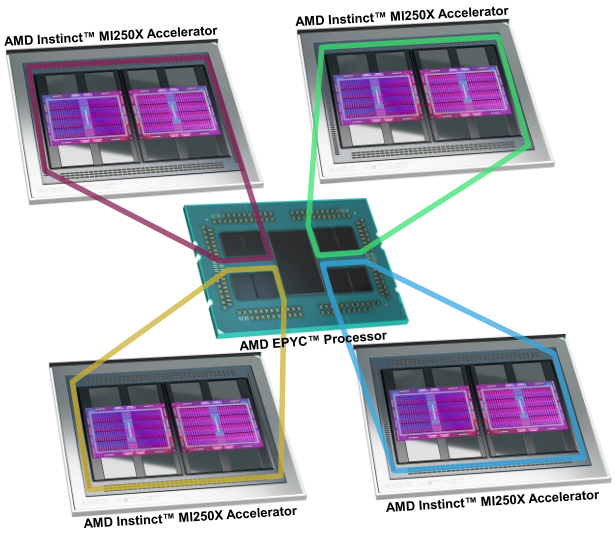

Figure 2. The Frontier Node Architecture's compute components.

C. Dawn of the Modern ML Era

At the start of the exascale program, the focus was on HPC and scientific computing. During this decade of research leading up to our first exascale architecture, the industry witnessed the explosion of the modern ML revolution. This created new requirements for our products, and AMD evolved our vision to devise a solution to provide world-class performance and power efficiency for *both* HPC and ML. There are many common requirements between HPC and ML, in particular very high memory bandwidth and compute throughput. However, ML also brings new requirements like lower-precision arithmetic not traditionally emphasized in HPC and greater memory capacity to hold growing model sizes, although there is now also significant activity toward adopting ML techniques for HPC use cases [34]. The market demand for world-class ML products steered the AMD strategy for its second generation exascale plans to support both sets of market needs. This necessitated taking the AMD chiplet strategy to new levels of 3D modularity.

III. EHP: Almost There, But Not Quite

We first provide some retrospective insights into why AMD did not pursue building either EHPv3 or EHPv4 for the Frontier supercomputer. The overall vision for an HPC APU was and remains technically sound, but the industry's heterogeneous integration capabilities at the time were not yet sufficiently mature to support high-volume productization of our exascale APU concepts. This section also sets the stage for understanding our approach for the MI300A processor.

A. Aggressive Technology Needs for EHPv3

In some ways, EHPv3 was more desirable compared to EHPv4 due to its significantly higher levels of integration density. While AMD has been successfully utilizing hybrid-bonding 3D stacking [14] in its CPU products enhanced with V-Cache™ technology, EHPv3's stacking is significantly more aggressive than our V-Cache products.

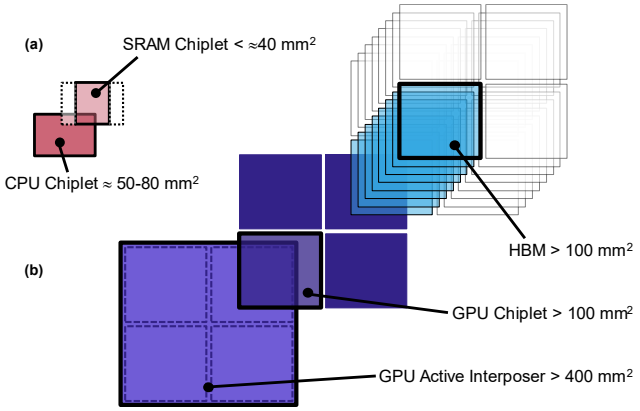

Figure 3. (a) AMD V-Cache technology SRAM chiplet stacked on a CPU chiplet, and (b) the stacked components of the GPU complex in EHPv3. (Illustration is approximately to scale.)

For context, the V-cache technology used in AMD EPYC™ and Ryzen™ processors stacks an SRAM chiplet on top of a CPU chiplet that is only tens of mm^2 in size [26][44], as shown in Figure 3(a). The 3D stacking assembly process for EHPv3 would be significantly more complex. First each GPU chiplet in Figure 3(b) would be equal to or larger than the footprint of an HBM stack (on the order of 100 mm^2 per stack) [40]. Four of these GPU chiplets would then need to be stacked on top of an active interposer die that would have to be over 400 mm^2 to have enough room. Furthermore, on top of each GPU chiplet, we would also need to stack the HBM.

The amount of time needed to mature the overall assembly flow for EHPv3 would not have aligned with Frontier's schedule. The reasons for this are primarily due to the number of additional processing steps required, the number of separate dies/stacks that need to be individually handled and tested, additional die-thinning and TSV construction for going beyond a two-high stack, and the larger overall size of the structure. The heat dissipation through this 3D structure would have also exceeded contemporary cooling capabilities. Since the original Frontier plans were set, AMD continued to evolve and mature its 3D stacking technology with multiple 3D-stacked Ryzen and EPYC products in collaboration with its foundry partners, and these improved capabilities were leveraged in our approach beyond Frontier.

Beyond the 3D-stacking of the components of the EHPv3, the multiple complexes, each comprised of an active interposer, chiplets, and HBM, also need to be co-packaged together. Heterogenous integration (HI) combining multiple different integration and packaging technologies has recently been gaining much attention [19], and AMD has already been mixing multiple forms of die-stacking and advanced packaging for several years. The EPYC and Ryzen V-Cache processors combine 3D-hybrid bonded components with conventional organic substrate-based 2D chiplet packaging. The AMD Instinct™ MI250X accelerator mixes two large GPU chiplets with 2.5D elevated fan-out bridge silicon die and 3D microbump-based HBM stacks [1][2][36]. However, none of these compare to the scale and size of heterogeneous integration required by EHPv3; it would have been too much to practically handle in the Frontier timeframe.

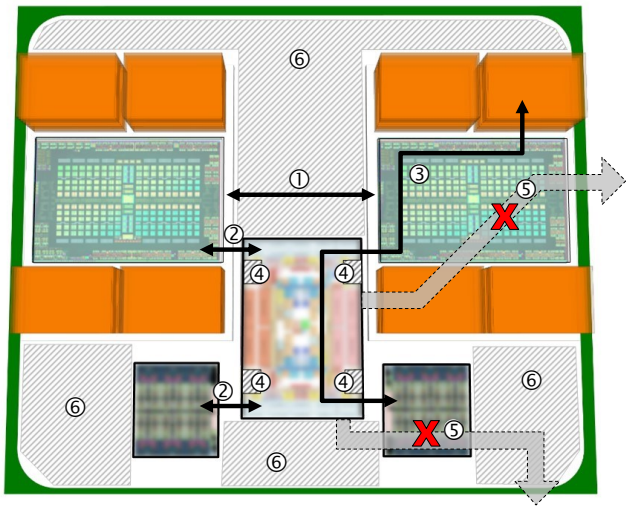

Figure 4. Various remaining challenges in EHPv4.

B. Shortcomings of the EHPv4

The EHPv4 had backed off from many of the more aggressive packaging concepts presented by EHPv3 in an attempt to better intercept the Frontier timeline, but it still retained many compelling attributes to create a powerful and efficient HPC APU. However, the final EHPv4 still had several design aspects and some remaining technology challenges that prevented it from being exactly what was needed.

The EHPv4 attempted to reuse the IOD from our mainstream EPYC server processors. While silicon reuse is generally desirable [27], the use cases must line up. In particular for EHPv4, the server-derived IOD introduced some challenges. First, the size and placement of die-to-die interfaces in the server IOD forced a non-optimal overall chiplet topology for EHPv4. The two GPU portions ended up being separated by a substantial distance that limits bandwidth and increases power for memory accesses from a GPU die to HBM stacks on the other side, labeled as ① in Figure 4. This long distance prevents the effective usage of higher-bandwidth packaging solutions such as passive silicon interposers or Elevated Fanout Bridge (EFB) interconnects [36] (which are typically only utilized for directly adjacent/abutting die such as the GPU and HBM), and instead the design would be limited to interconnect options such as the 2D AMD Infinity Fabric™ (IF) links used between the GCDs in MI250X.

The server IOD's IF links between chiplets was also not ideal for our HPC APU. The server IOD's IF links ② were originally provisioned to handle DDR levels of memory bandwidth, and as such they can become bottlenecks for a system that is focused on in-package HBM.

The overall chiplet topology creates a relatively long path from the CPU chiplets to the HBM ③, requiring two die-to-die IF hops (plus the data fabric/network-on-chip traversals within the IOD and GPU) to get to the memory, and then pay that price again for the data response. The IOD would also

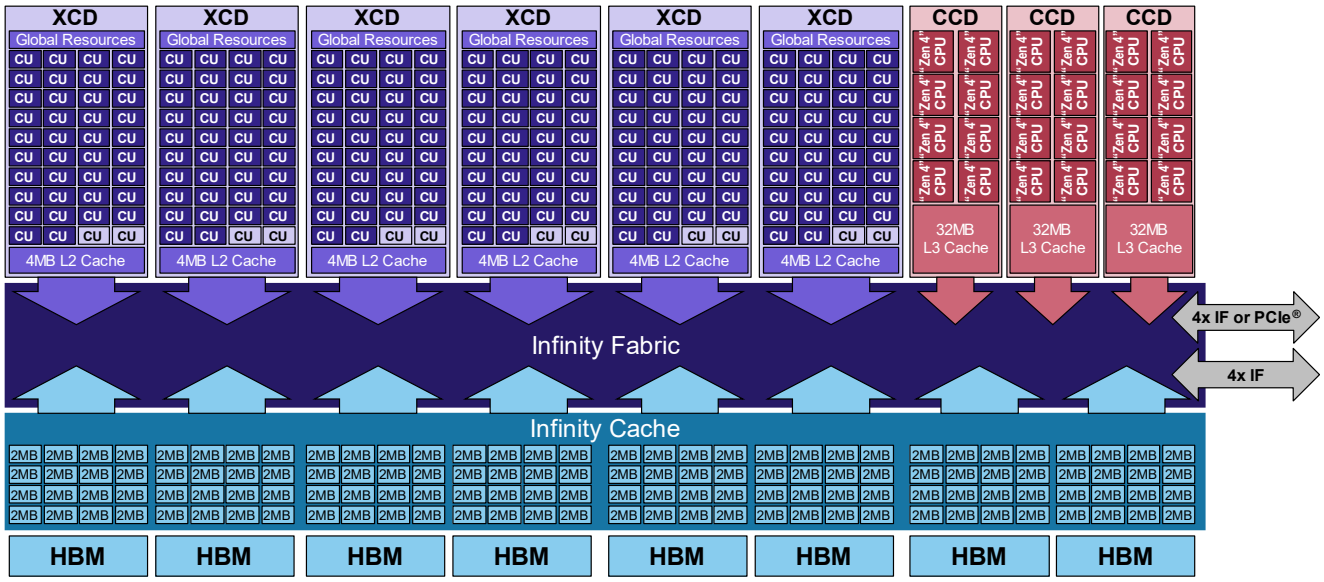

Figure 5. Logical/architectural block diagram of the MI300A accelerated processing unit.

have been inefficiently utilized as several of the 4th generation EPYC IOD's twelve IF links [41] ④ would not have been connected to any chiplets and therefore represent wasted die area. In a similar vein, the placement of the GPU and HBM complexes would have blocked the signal escape routes for some of the IOD's DDR memory channels and I/O lanes ⑤, and so those interfaces would also become unutilized die area. When building a solution to achieve the highest possible performance, one should aim to maximize the utilization of the precious package area, but EHPv4 leaves several regions of the package empty ⑥. Reusing the server IOD also limits its connections to the other chiplets to 2D organic substrate-based packaging (because an unmodified server IOD would not have the interfaces for, e.g., EFB-based links), resulting in sub-optimal bandwidth and energy costs for the EHP's data movement.

As a result, the combination of EPYC CPUs and Instinct MI250X GPUs in a coherent, heterogeneous node architecture was the right approach for our first-generation exascale architecture [22][32]. However, the success of the Frontier approach utilizing different combinations of advanced packaging technologies combined with the learnings from these products increased confidence and reduced risk to pave the path to an APU for HPC and ML [23][24].

IV. MI300A Architectural Organization

The AMD Instinct™ MI300A APU builds on the EHP concept. This section describes the architectural organization of the MI300A, and Section V covers its physical construction.

A. Overview

The MI300A APU consists of a mix of CPU and GPU chiplets along with a shared AMD Infinity Cache memory-side cache, eight stacks of HBM, I/O interfaces, and Infinity Fabric (IF) interconnect to provide the data movement among all of these. Figure 5 shows a block diagram view of the MI300A APU. MI300A has a total of six accelerator complex dies (XCDs) that provide the APU's highly-parallel and energy-efficient GPU compute engines (the term "GPU" is somewhat of a misnomer as the CDNA™ 3 architecture in the XCDs optimized out the traditional graphics-specific hardware for pixel processing and ray tracing). The MI300A APU also utilizes three CPU complex dies (CCDs) that provide the CPU cores. The XCDs and CCDs all connect to the IF interconnect that provides the SoC's network-on-chip (NoC) functionality. Due to the physical construction of MI300A (Section V) the "NoC" spans multiple chips. The IF routes memory requests to and from the system's 128 HBM memory channels as well as the I/O interfaces. Each memory channel is also paired with a 2MB slice of the Infinity Cache (256MB total capacity). The following subsections provide more details on each of these components.

	Vector		Matrix						
	FP64	FP32	FP64	FP32	TF32	FP16	BF16	FP8	INT8
CDNA 2	128	128	256	256	n/a	1024	1024	n/a	1024
CDNA 3	**128**	**256**	**256**	**256**	**1024**	**2048**	**2048**	**4096**	**4096**

Table 1. Peak operations-per-clock-per-CU rates for the CDNA 2 CUs in MI250X versus the CDNA 3 CUs in MI300A.

B. XCD

The MI300A's XCDs are implemented in a 5 nm technology process, and each XCD provides 38 compute units (CUs) for a total of 228 CUs across the entire APU. Note that each XCD physically implements 40 CUs, but only 38 are utilized to improve yield (i.e., up to two CUs can be defective, as indicated by the lightly-shaded CUs in Figure 5). Each XCD contains shared global resources, including the scheduler, hardware queues, and four Asynchronous Compute Engines (ACE) that send compute shader workgroups to the CUs. The CUs within an XCD share a 4MB L2 cache that serves to coalesce all of the memory traffic for the die.

Each CU is a highly-threaded processor including an instruction fetch unit, scheduler, scalar/vector/matrix execution units, and load/store pipelines with a 32KB L1 data cache and

a 64KB Local Data Share (LDS) that form the start of the memory hierarchy. The L1 data cache line size has been increased to 128B, and the bus widths to/from the data cache have correspondingly increased, effectively doubling the cache bandwidth compared to the CDNA 2 architecture [2]. Each pair of CUs shares a 64KB, 8-way set associative instruction cache. For GPU workloads, the overwhelmingly common case is that the stream gets executed by groups of CUs, so sharing the instruction cache increases the cache hit rate with minimal impact on die area.

Compared to the CDNA 2 architecture used in the MI250X accelerator, the MI300A CDNA 3 architecture significantly increases the performance of the Matrix Cores' support for machine learning data types for training and inference. Table 1 shows the operations-per-clock-per-CU rates for CDNA 2 and CDNA 3 architectures. The table also highlights the additional support for FP8 data types in the MI300A XCDs. Not reflected in the table is that the CDNA 3 Matrix Cores also support 4:2 sparsity; under such conditions, the peak throughput can reach as high as 8192 ops/cycle/CU (for FP8 and INT8). Additional details on CDNA 3 are available [5].

C. CCD

The MI300A APU leverages the "Zen 4" CCD [26] from the AMD 4th generation EPYC CPU [4] with modifications for hybrid bonding. Like the XCDs, the CCDs are implemented in a 5 nm technology. Each CCD provides eight "Zen 4" cores that share a 32 MB L3 cache. The MI300A APU provides three CCDs for a total of 24 "Zen 4" cores. Full details of the "Zen 4" microarchitecture can be found in prior work [14][41]. A few key highlights of "Zen 4" over the prior generation "Zen 3" design include doubling the per-core L2 cache size to 1MB, improvements in branch prediction accuracy, increases in the sizes of many of the microarchitectural structures (op cache, retire queue, integer and floating point register files), clock frequency improvements, performance-per-Watt improvements, and the addition of ISA support for AVX 512 instructions. The CCDs execute all of the traditional x86-based code, including everything necessary for the operating system as well as all portions of user codes that have not been offloaded to the XCDs.

D. In-package Infinity Fabric and Memory Organization

The CCDs and XCDs share a unified HBM-based memory system. This enables direct load-store accesses to the HBM by either type of processor without any data copying as is typically required in systems that use discrete GPUs with their own video memory distinct from the host CPU's memory. The CPU can initialize memory and then directly launch a kernel on the GPU components. This provides both performance and programmability benefits, which will be discussed further in Section VI. The CPUs are hardware coherent with all CPUs and GPUs using the same type of probe filter-based coherence protocol as in EPYC CPUs. The GPUs are software-coherent to GPUs in other sockets (to reduce hardware coherence bandwidth needs) and directory-based hardware coherent within a socket using a slightly simpler protocol than the CPUs use.

A common Infinity Fabric interconnect provides data transport between the CCDs and XCDs to and from the eight stacks of HBM, as previously shown in Figure 5. In aggregate, the MI300A APU provides 128 GB of memory capacity with a total of 128 memory channels and a peak theoretical bandwidth of about 5.3 TB/s. The IF and HBM channels are relatively finely interleaved. Every 4KB of sequential physical addresses map to the same HBM stack before moving on to another HBM stack chosen based on a physical address hashing scheme. The IF also provides connectivity to the I/O interfaces (discussed in more detail in Section VIII).

Each of the 128 memory channels is paired with a 2 MB slice of the Infinity Cache (256 MB total). The Infinity Cache acts as a memory-side cache, and therefore it does not need to participate in cache coherence transactions. The primary function of the Infinity Cache is to provide bandwidth amplification for the HBM, delivering up to 17 TB/s of bandwidth. The Infinity Cache also features a hardware prefetcher to help reduce memory access latencies.

V. MI300A Physical Construction

The MI300A APU consists of about 146 billion transistors (not counting the HBM), which pushes the boundaries of manufacturing in multiple dimensions. This section discusses the AMD advancements in chiplets, 3D stacking, packaging, IP reuse, and modular architecture methodology that went into the MI300A APU.

A. MI300A Heterogeneous Integration Technologies

The sheer amount of logic represented by MI300A's architecture requires multiple reticles worth of transistors to implement. Our past work described how chiplet-based designs enable the decomposition of logically large architectures into multiple physically distinct silicon chiplets [27]. However, chiplets alone in a 2D/2.5D arrangement would still not have provided enough integration density to pack MI300A's components into a reasonably-sized package, as discussed earlier in the context of EHPv4.

To make this all work, MI300A utilizes an aggressive combination of multiple technologies, as illustrated in Figure 6. Individual XCD and CCD chiplets utilize 3D hybrid bonding technology to vertically interface to multiple active interposer IOD. Hybrid bonding uses direct-contact bonding (as opposed to placing individual microbumps) where metal bonding pads on either die are atomically fused together. This results in dense vertical interconnects (9 μm pitch for both AMD V-Cache products and MI300A) and superior thermal conduction properties compared to microbump-based 3D stacking [44]. The cross-sectional view illustrates the entire stack of passive and active components in MI300A, including structural and carrier silicon components required for the mechanical stability of the entire complex. (Note that the placement of the HBM in the cross-sectional view has been pushed to the "sides" for illustrative purposes.)

The amount of "beachfront" perimeter required to interface with eight stacks of HBM as well as to provide all of the I/O

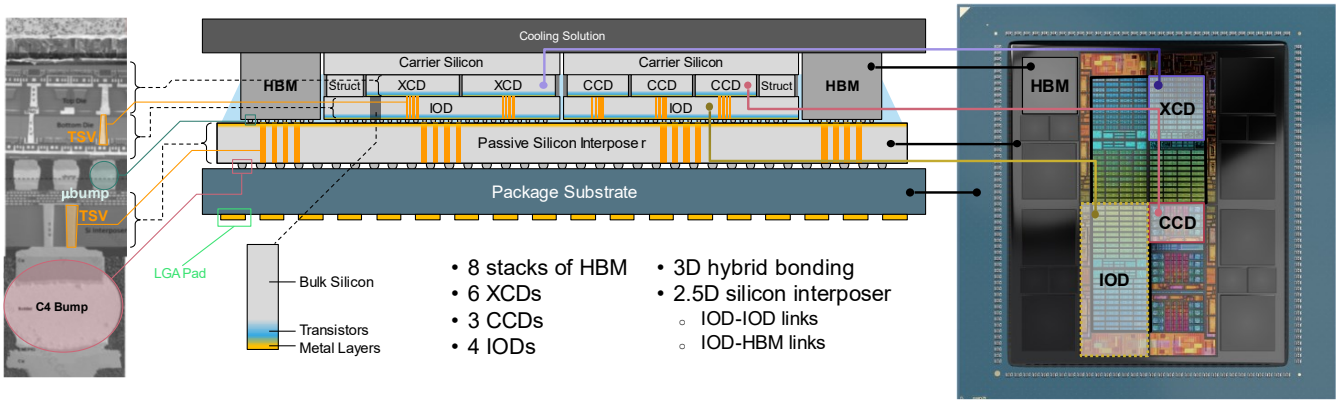

Figure 6. Details of the heterogeneous integration construction of the MI300A APU utilizing multiple advanced integration and packaging technologies, chiplets, and 3D memory stacks. (Cross-sectional view is not drawn to scale.)

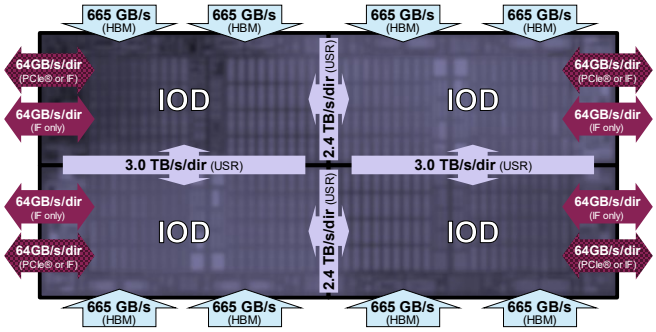

Figure 7. MI300A IOD bandwidths across various interfaces.

interfaces would have required a massive IOD well exceeding a standard lithographic reticle's size. In the past, AMD utilized chiplets for compute elements (e.g., CCDs in our EPYC and Ryzen CPUs), and MI300A exploits a similar partitioning approach and employs four separate IODs (visible in the areal views on the right of Figure 6, with the bottom-left IOD highlighted, and also shown in Figure 7). Each IOD (with multiple CCDs or XCDs hybrid bonded on top) is then 2.5D-integrated on a silicon interposer with the eight HBM stacks. Enabled by the tight spacing between adjacent IODs, we implement very low power and low latency ultra-short reach (USR) PHYs for data movement between IODs. The USR PHYs deliver more than a 10× improvement in area bandwidth density (Tbps/mm^2) versus conventional SerDes [27] while reducing power consumption (0.4 mW/Gbps). The USR interface has a minimum microbump pitch of 35 µm. In total, the USR interfaces deliver multiple TB/s of bandwidth (Figure 7) so that the HBM can be accessed as if the Infinity Fabric were implemented on a single monolithic IOD.

B. 3D Interfaces

The XCD was specifically designed for MI300A, and therefore we could directly place its 3D interfaces to align with the corresponding connection points on the IODs below.

For silicon reuse, MI300A adapts the same CCDs that can be found in AMD EPYC™ CPUs. In the EPYC processor configurations, the CCD uses an IF SerDes interface designed for 2D organic substrate packaging, as highlighted in Figure 8(a). However, for MI300A's 3D-stacked organization, we

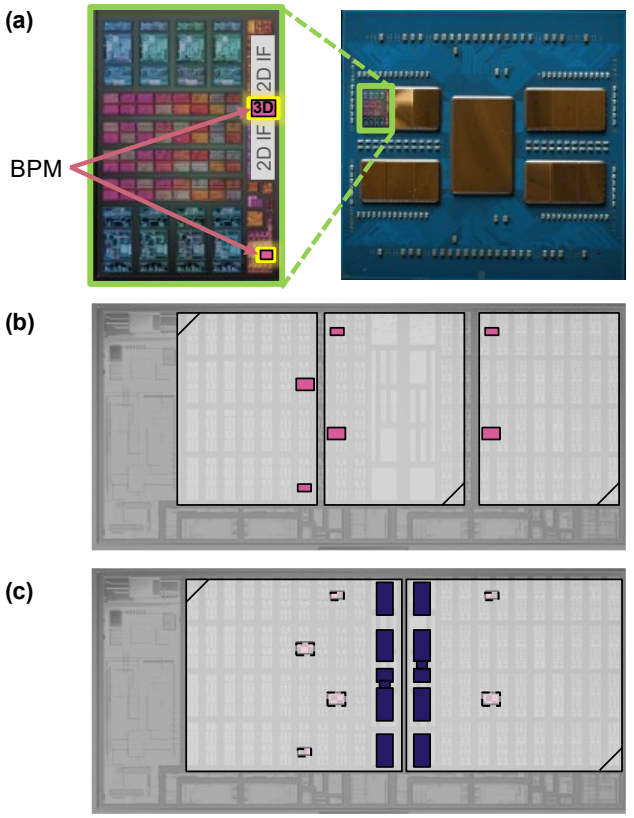

Figure 8. (a) Illustration of 2D and 3D interfaces on the CCD, (b) IOD's CCD 3D interfaces, and (c) IOD's XCD 3D interfaces.

needed to add bond pad metal (BPM) landing sites for the hybrid bonding stacking process. Figure 8(a) shows how we squeezed in the 3D interfaces within whitespace in the CCD floorplan. This also illustrates just how much denser the TSV-based 3D interfaces are compared to organic substrate-based 2D interconnect solutions. Within the CCD, we utilize simple interface multiplexing between the 2D and 3D interfaces depending on which product the CCD is deployed in.

On the IOD side, the corresponding 3D interfaces are implemented to line up with the three CCDs, as shown in Figure 8(b). Note that two of the CCDs are rotated 180° in relation

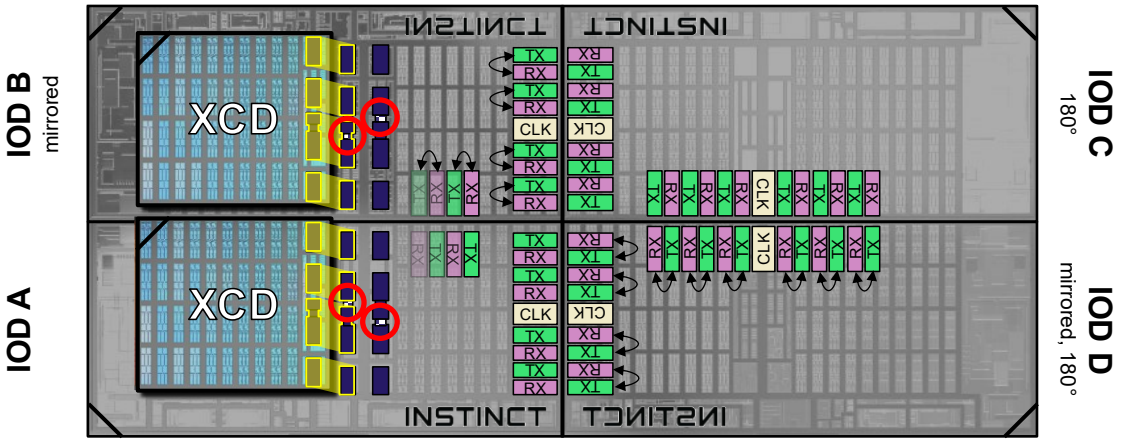

Figure 9. TSV replication (red circles) to align with non-mirrored chiplets, and IOD mirroring with USR PHY alignment.

to the other CCD, and the IOD's interfaces are similarly rotated to maintain correct alignment.

MI300A utilizes the same IOD physical design whether a given IOD has CCDs or XCDs stacked on top. The CCDs and XCDs have different sizes, different bandwidths/interface widths, and different total chiplet counts (i.e., three CCDs on an IOD versus two XCDs). To support this, the IOD implements the superset of interfaces as shown in Figure 8(c). When the two XCDs are stacked on the IOD, the middle set (dark blue) of TSV interfaces are used, and the three sets of CCD interfaces are disabled (dashed/hashed boxes). Similar to the CCDs, one of the XCDs in the figure is rotated 180°, and therefore the corresponding TSV interface is likewise rotated to match. The careful co-planning of the XCD, CCD, and IOD interfaces enables this seamless modular swapping of compute chiplets on top of the IOD.

C. IOD Mirroring

There are four total instances of the IOD silicon in MI300A. Two of the IOD are "mirrored" copies of the same physical design with minimal modifications. This enabled us to reuse almost all of the floorplanning, physical design, and engineering effort across the two very similar tapeouts. Figure 9 shows the four IODs: two normal versions and two mirrored versions, with one of each IOD type rotated 180° ("Instinct" logos and alignment markers in the corners of the IODs are provided to discern orientation and mirroring).

To enable the IODs to seamlessly interface with each other, the USR transmit (TX) and receive (RX) modules needed to be swapped on the mirrored IOD as indicated by the bidirectional arrows on the mirrored IOD B and IOD D. In this fashion, each TX module on an IOD is directly paired with its corresponding RX module in the adjacent IODs. This is an example of how the IODs are not exact geometric mirrors of each other, but such modifications can be supported with relatively simple algorithmic changes to the design.

The mirrored IOD also created an interesting interface challenge for the 3D-stacked chiplets. While the floorplan for the IOD is mirrored in a symmetric fashion, we do not implement mirrored versions of the CCDs and XCDs. Therefore, the interfaces needed to be designed so that unmirrored CCDs and XCDs could properly land on and remain aligned with

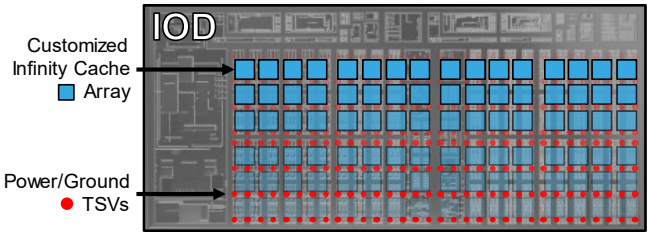

Figure 10. Coordinated floorplanning of chiplet power-delivery TSVs and Infinity Cache SRAM array macros.

the IODs underneath, regardless of whether the IOD below is an original or mirrored instance. The left side of Figure 9 shows two XCDs (note the same, non-mirrored orientation for both) stacked on top of IOD A and IOD B. Red circles on IOD A indicate redundant TSV interfaces that are not used on the non-mirrored IOD instances. However, when the IOD is mirrored (e.g., IOD B), these previously redundant TSVs are now aligned with the non-mirrored XCD, thereby enabling the XCD to interface to mirrored and non-mirrored IOD. Note that this type of TSV redundancy is limited to the 3D signal interfaces (power/ground TSVs are discussed in the next section). The carefully choreographed alignment of IOD, XCD, and CCD interfaces enables MI300A to deal with all of the required combinations of rotated and/or mirrored IODs and rotated CCDs and XCDs.

D. Power Delivery

The previous discussion focused on adapting the 3D interfaces for data and control under various permutations of mirroring and rotation. However, a similar challenge exists for TSV planning in the IOD to deliver power to the chiplets. Our solution uses a uniform grid of power/ground (P/G) TSVs that was consistent for both CCDs and XCDs. This is conceptually straightforward, but the implementation requires significant advanced planning to make sure that every single P/G TSV lines up for every permutation of mirrored/rotated IOD, CCD, and XCD. The resulting IOD TSV grid can deliver more than 1.5 A/mm^2 of current with minimal I^2R loss.

The microbump interface at the bottom side of the IOD (to the passive silicon interposer) can deliver an additional 0.5 A/mm^2 (beyond the 1.5 A/mm^2 for the stacked chiplets) to power the IOD. As workloads transition between compute-

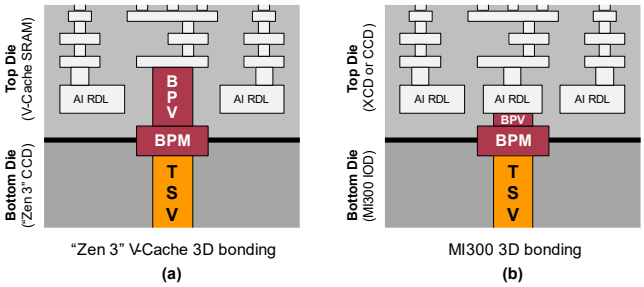

Figure 11. 3D bonding interfaces in (a) earlier V-Cache implementations and (b) the MI300A APU.

dominated and memory-intensive phases, power can be vertically shifted/reallocated between the IOD and the compute chiplets (discussed further in the next section).

Beyond the power/ground TSV planning, the Infinity Cache array macros also needed to be co-optimized with the overall power delivery solution. The arrays were customized so that they were pitch-matched to fit within the channels between the P/G TSV stripes, as shown in Figure 10.

While MI300A uses the same fundamental hybrid bonding process as our V-Cache technology, further optimization to the stacking process was required for MI300A. Figure 11(a) shows a cross-sectional view of the 3D hybrid bonding interface used in the "Zen 3" generation of CCDs with V-Cache technology [44]. The BPM at the hybrid-bonded interface is connected to the SRAM die's top-level metal by a bond-pad via (BPV). For MI300A, however, the CCDs and XCDs are stacked on top, as shown in Figure 11(b), and these have higher power requirements compared to a V-Cache SRAM die. In MI300A, the BPV lands directly on the aluminum redistribution layer (RDL), which has lower resistance and is more effective for delivering power to the compute chiplets.

E. Power Management and Thermals

With 3D stacking, power and thermal management are very important factors in the overall physical design of the system. As discussed in the previous section, power can be dynamically reallocated among the different physical components of the MI300A accelerator. Figure 12(a) shows two representative power distributions (normalized) corresponding to compute-intensive (GPU) and memory-intensive workload scenarios. In the compute-intensive case, the majority of the power can be directed to the compute chiplets. In a memory-intensive scenario, more of the power can be shifted to the memory system, data fabric, and USR links. This creates two extreme operating conditions that the power delivery system must handle.

In addition to the power delivery, the thermal solution must likewise be able to accommodate these very different operating conditions. Figure 12(b) shows thermal simulation results for the GPU-intensive scenario, where the thermal hotspots are concentrated on the XCDs. Figure 12(c) shows the case for a memory-intensive workload. The thermal distribution is somewhat more mixed here as the XCD activity is still discernable in the heat map, but the power-related impacts of data movement are more noticeable. The eight HBM PHYs

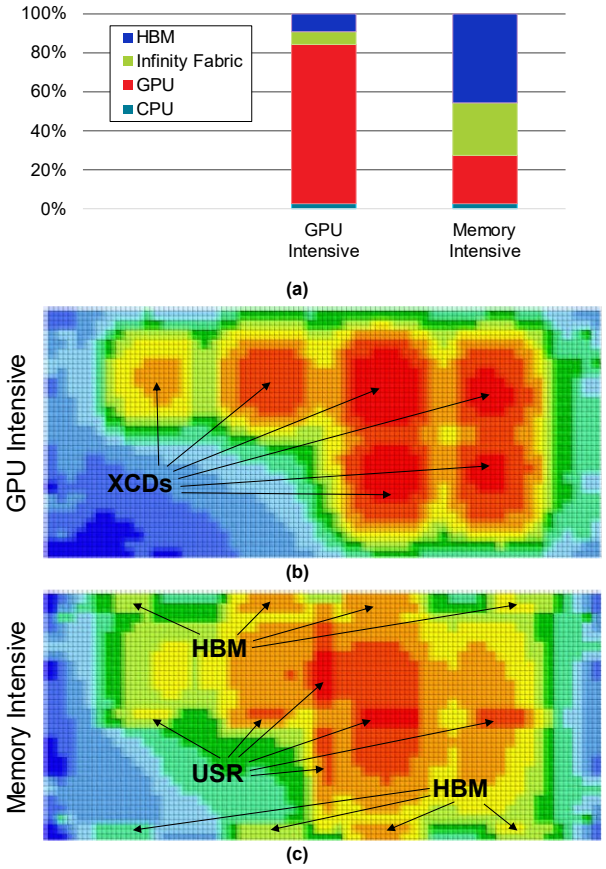

Figure 12. (a) Representative power distributions for different workload scenarios and thermal simulation projections for (b) GPU-intensive and (c) memory-intensive workload scenarios.

are visible along the periphery of the module, and the USR PHYs clearly stand out as they deliver multiple TB/s of data between the four IODs. The effective power and thermal management of MI300A was accomplished through careful engineering and co-design of both TSV placement and power density/power map planning.

F. Comparison Versus the EHP

The final design of the MI300A APU is physically quite distinct from the prior EHP concepts, yet MI300A also embodies many of the attributes originally envisioned to support exascale computing. Like the EHP, the MI300A is a single-package APU solution combining heterogeneous CPU and GPU computing components with a unified, in-package memory system. The exact chiplet count is slightly different, but both ended up with the same ratio of two GPU compute chiplets for every CCD (i.e., 4:2 in EHPv4, and 6:3 in MI300A). Both the EHP and MI300A also used the same number of HBM stacks. EHPv3 also foresaw the eventual adoption of a 3D-stacked active interposer organization that separated compute chiplets from the cache and interconnect.

The MI300A APU also features some key differences that provide significant advantages and improvements over the prior EHP concepts. Perhaps first and foremost, the availability of the combination of both hybrid-bonded 3D stacking and microbump-based 2.5D interposers enabled the

implementation of a much higher bandwidth and energy efficient data movement substrate between all of the compute chiplets, HBM, and I/O. EHPv4 had inefficiencies due to the large distance between the GPU dies, and even EHPv3's organic substrate-based links between the active interposers would have posed bandwidth and power challenges. EHPv3 required two different active interposer types, which would have increased the design cost. MI300A is able to utilize what is effectively a single IOD design with which we can mix and match different compute chiplets. Designing a new IOD for MI300A was the better choice compared to EHPv4, which attempted to reuse the IOD from the EPYC processors. While silicon reuse is generally desirable, it is only effective when there exists a good fit with the target use cases; in the case of EHPv4, the EHP's needs were too distant from the design requirements of the EPYC server products.

VI. Programming Model and Software

Software programmability was a major aspect of the EHP, and it was at the forefront of the MI300A design. To simplify software development for MI300A, we created mechanisms to present its multiple XCDs as a single, unified GPU and to share its unified HBM memory between CPU and GPU.

A. Unified Multi-chiplet Accelerator

EHP aimed to fulfill the expectation that kernels achieve greater performance in each generation of GPU. However, we found that EHPv4 was not amenable to single kernels executing on both GPU chiplets due to the lower bandwidth connections (compared to HBM) between the chiplets. For similar reasons, the AMD Instinct™ MI250X presented each GCD as a standalone accelerator. However, as the number of chiplets in a design increases, presenting each chiplet as its own accelerator is a less enticing option for developers.

The kernel launch interface between user-mode software and MI300A is a queue in user-mode visible memory that can be filled with packets that describe the kernel. The queues are defined by the Heterogeneous System Architecture (HSA) standard [18]; the packets submitted to these queues follow HSA's Architected Queueing Language (AQL) format. In contrast to lower-level packet formats that describe what values to put into which hardware registers in order to cause a particular GPU to launch a kernel, AQL packets describe a higher-level goal such as "launch kernel X with Y workgroups, each with Z threads."

As described in Section IV.B, each MI300A XCD contains the necessary hardware to handle dispatching kernels to that XCD. These Asynchronous Compute Engines (ACE) are responsible for interfacing with the user-mode queues, reading AQL packets, decoding these packets, and appropriately programming microarchitectural resources to initiate the dispatch of the kernel. Further, ACE hardware then finds space within the XCD's compute units for the workgroups, initializes wavefront register state, and gives it a program counter to begin execution. Finally, ACE hardware is responsible for detecting when all workgroups in a kernel have completed and signaling completion indicators to software. While ACEs in AMD GPUs such as the MI250X work similarly, MI300A also allows the kernel scheduling hardware on each XCD to work cooperatively with other XCDs to present to user-mode software a multi-XCD *partition*. Because the AQL packets describe a high-level request such as "launch a kernel," the ACE on each XCD can work together to process the request.

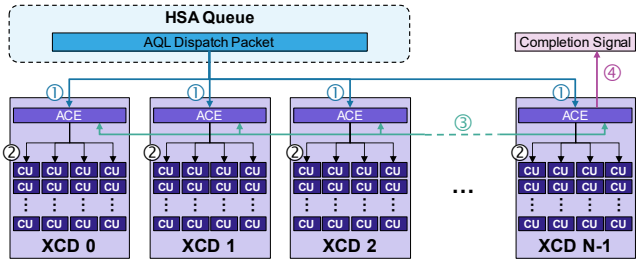

Figure 13. Multi-XCD kernel dispatch and completion flow.

For example, when a dispatch packet is submitted into the queue, an ACE in each XCD of a partition will read the AQL packet ①. All of these processors decode the packet and set up their local microarchitecture to launch a subset of the requested workgroups ②; the ACE knows how many XCDs are in the partition, so it knows that its XCD is only responsible for executing a subset of the kernel's total workgroups. The decision of which workgroups are scheduled into which XCD is configurable to allow tradeoffs between factors like inter-workgroup data reuse in the XCD's L2 cache versus initiating work on as many XCDs as possible to maximize memory bandwidth; after deciding which workgroups to launch, workgroup creation then proceeds as normal within the XCD.

At various points in the processing of a packet, the XCDs' ACEs may need to synchronize with each other ③. For example, all XCDs must indicate that their subset of a dispatch's waves have completed, and their writes are visible to the appropriate coherence scope before a nominated XCD can send a signal that indicates the kernel has completed ④. The Infinity Fabric on MI300A includes a high-priority communication channel to facilitate performant communication and synchronization between the ACEs on each XCD of a partition.

This cooperative protocol allows the partition to take a single kernel dispatch from a single software-visible queue and spread the workgroups across the XCDs, making the partition appear as a single logical GPU despite its multi-chiplet nature. This is a natural pairing to MI300A's unified memory, where the entire HBM memory space is visible to a single process.

Instead of requiring a separate scheduling chiplet, we used per-chiplet schedulers to reduce inter-chiplet wiring requirements and increase workgroup scheduling throughput as more chiplets are added. This also enables MI300A to present partitions with different numbers of chiplets (Section VIII).

Presenting the MI300A device as a single partition helps existing software easily use the entire device. Several software frameworks commonly used in HPC are supported, including OpenMP®, RAJA [6], and Kokkos [10]. The most popular AI/ML frameworks, such as PyTorch, TensorFlow, JAX [10], and Triton [38], are likewise supported. The AMD ROCm™ open compute platform [3] provides familiar

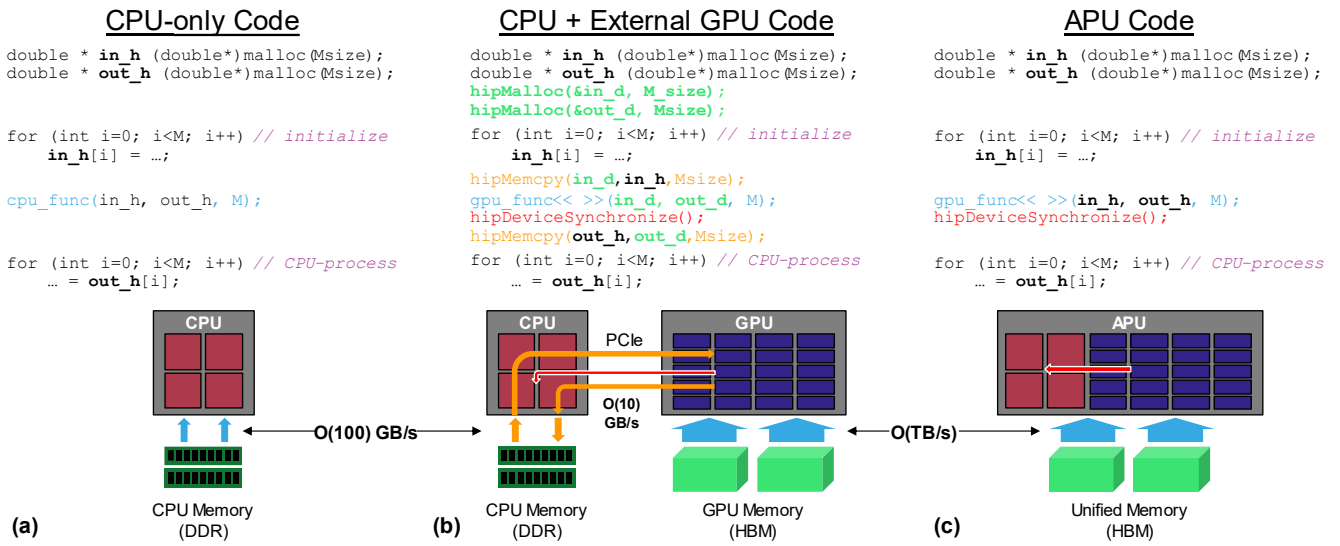

Figure 14. Example code and data movement/synchronization for (a) CPU-only, (b) CPU and a discrete/external GPU with separate memory spaces, and (c) APU with a unified memory.

optimized libraries (e.g., rocBLAS, rocDNN) to aid programmers in accessing optimized implementations of common compute tasks.

B. Unified Memory

MI300A presents a single unified memory to the CPU cores and GPU CUs, which simplifies the programming model. A unified memory enables eliminating redundant data copies, removing the distinction between host and device memory spaces, and enables fine-grained sharing between processing elements. While some platforms provide the appearance of unified memory to the software (e.g., via page migration to transparently copy data between the CPU's DDR and the GPU's HBM), MI300A avoids such data movement overheads by matching the actual physical memory organization with the programmer's view. Figure 14 illustrates a code simplification example. In the CPU-only code shown in Figure 14(a), one allocates memory (e.g., `malloc`), initializes it, and then computes on the data. With separate CPU and GPU memories, the code is more involved, shown in Figure 14(b). Memory space must first be allocated on both the CPU and GPU (`malloc` and `hipMalloc`). The CPU then initializes the memory, but then must copy the data from the CPU's memory space to the GPU memory buffer (`hipMemcpy`). The CPU can then launch a kernel on the GPU and then wait for the GPU to finish (e.g., `hipDeviceSynchronize`). Rather than immediately continuing its own post-kernel computations, the CPU first copies the results from the GPU memory back to the CPU side (another `hipMemcpy`), at which point the CPU can complete any remaining operations. Finally for an APU programming model (e.g., in MI300A) shown in Figure 14(c), the programmer can avoid the memory management steps needed for separate CPU and GPU memories. The APU code omits GPU memory allocations (no `hipMalloc`), and because there is only a single copy of the data like in the CPU-only case, there is also no

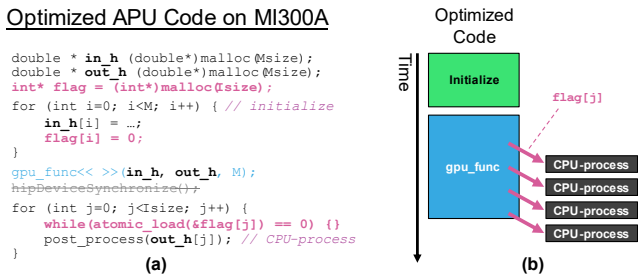

Figure 15. (a) Example code for fine-grained decoupling of GPU and CPU execution, (b) timeline showing overlapping of GPU and CPU execution, and (c) timeline of original code.

need to copy the data around (no `hipMemcpy`). Once the CPU has finished initializing the data, it can immediately launch the GPU kernel due to the APU's unified memory (including cache coherency between CPU and GPU). The CPU must still properly synchronize with the GPU, after which it can complete its post-kernel computations, again without any additional data copy operations.

Beyond programmability benefits, APUs like MI300A can also enhance performance from optimized data movement. For discrete GPUs, the bandwidth between the host CPU and an external GPU is typically limited by the PCIe® interface, typically tens of GB/s. The bottom portion of Figure 14(b) illustrates the discrete GPU scenario where the CPU can read data from DDR memory at a few hundreds of GB/s but is then bottlenecked by PCIe. In the APU situation in Figure 14(c), the CPU writes to HBM as fast as it can, and then the GPU can immediately operate on the data at HBM speeds.

The APU programming model can also open up new opportunities to further optimize how the CPU and GPU components work together. Figure 15(a) shows pseudocode where the unified memory enables overlap of GPU and CPU compute and removes coarse-grained kernel-level synchronization. This example adds an array of completion flags (one per output element), which enables the GPU to set these flags

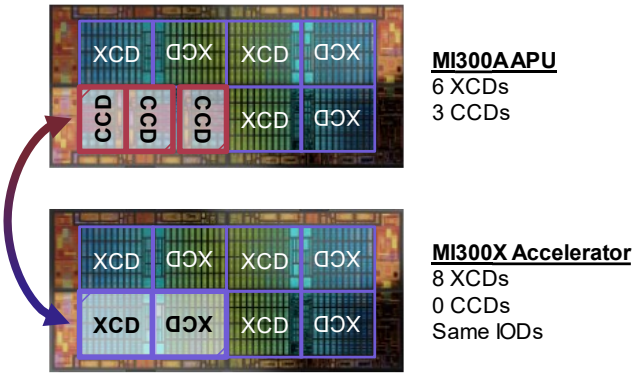

Figure 16. Modular replacement of MI300A's CCDs with XCDs to create the MI300X accelerator.

as it produces its data output, shown in Figure 15(b). On the CPU side, the CPU waits in a spin-loop thanks to the APU's cache coherent memory system. As the GPU produces individual data elements, the CPU can immediately proceed with its post-processing, thereby increasing compute concurrency.

Presenting a single accelerator with a unified memory space opens opportunities to automatically accelerate applications on an APU because data are always accessible to CPU cores or GPU CUs via the in-package HBM. This means that software largely does not need to consider non-local overhead of access across a link. This permits standard library APIs, such as BLAS or LAPACK, to be linked to both CPU and GPU libraries. The generic library calls invoke a thin shim library that dispatches the work to either the CPU or GPU processing elements depending on simple heuristics such as problem size, etc. This enables code that might be CPU-only, with frequent calls to standard libraries, to be offloaded to an APU without explicit code refactoring.

VII. MI300X

As discussed in the introduction, the industry is witnessing an explosion in compute demand for ML and generative AI workloads. Many of these customers desire solutions with both massive computational throughput and memory systems that deliver high bandwidth and large capacities. Large language models (LLMs) are a key example of this type of very demanding modern workload. Some market segments make use of AI-specific accelerators, but many users desire the easier programmability associated with more general-purpose GPU architectures such as that supported by our CDNA XCDs.

The silicon building blocks of MI300A provide a modular chiplet platform that enables stacking different compute chiplets on the IODs. By reusing the silicon components of MI300A, AMD has also designed the AMD Instinct MI300X accelerator [33]. MI300X is structurally similar to MI300A, but the three CCDs are swapped with a pair of XCDs to create an accelerator-only module, shown in Figure 16. The eight XCDs provide a total of 304 CUs, delivering more FLOPS/mm^3 than MI300A. MI300X also utilizes 12-high HBM stacks for a total of 192 GB of memory capacity (24 GB per HBM stack). MI300X is an especially strong match

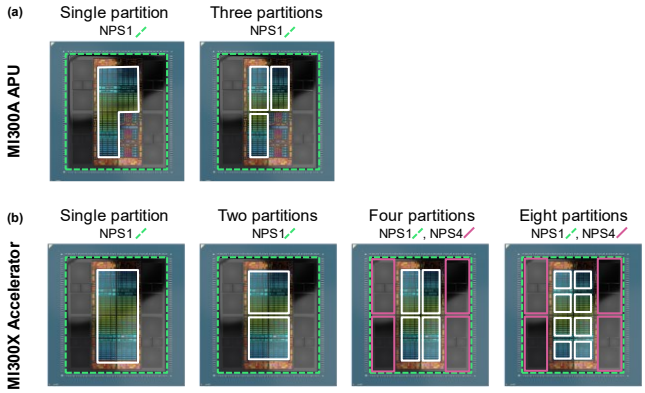

Figure 17. Compute and memory partitioning modes for (a) the MI300A APU and (b) the MI300X accelerator.

for LLM use cases where the prompt phase demands high compute throughput, the token generation phase is typically constrained by memory bandwidth, and modern LLMs demand more total capacity to handle these larger models.

Our modular design approach represents a new level in our highly successful chiplet strategy [27]. Our previous chiplet approaches reused the same chiplet types for different product lines (e.g., EPYC and Ryzen processors), but this represents a new approach to chiplet modularity wherein we can optionally swap different *types* of chiplets. AMD only needed to design the XCD and IOD, and then by leveraging mirroring of the IOD and adapting an existing CCD, we are able to utilize different combinations of these components to create both the MI300A and MI300X products without compromising either. MI300 presents a new type of modular architecture and a true embodiment of the heterogeneous integration approach combining chiplets, different technology nodes, mix-and-match chiplet interfaces, 2.5D silicon interposers, 3D HBM, and 3D hybrid bonding all in one package.

VIII. Platform Architecture and Scalability

While a chiplet-based approach provides design flexibility, MI300 also features additional capabilities to offer flexibility to users in how they deploy MI300. The first set of options allow MI300A and MI300X to support different partitioning configurations for multi-user deployments. For MI300A, the six XCDs can be used as a single compute device or as three separate partitions, shown in Figure 17(a). In both partitioning modes, the entire HBM address space is uniformly interleaved, implementing a single non-uniform memory access (NUMA) domain per socket (NPS1). The XCD-only MI300X lends itself to additional partitioning options, as illustrated in Figure 17(b). The XCDs can be partitioned in powers of two from a single unified partition down to eight separate partitions (one XCD per partition). The memory system also supports a single unified NUMA domain (NPS1) as well as sub-dividing the memory space into four NUMA domains per socket (NPS4). This scheme lends itself to PCIe® single root I/O virtualization (SR-IOV) where each PCIe virtual function (VF) can be mapped to a separate partition.

The MI300 architecture also provides I/O configurability to enable flexibility in how multiple MI300 modules can be

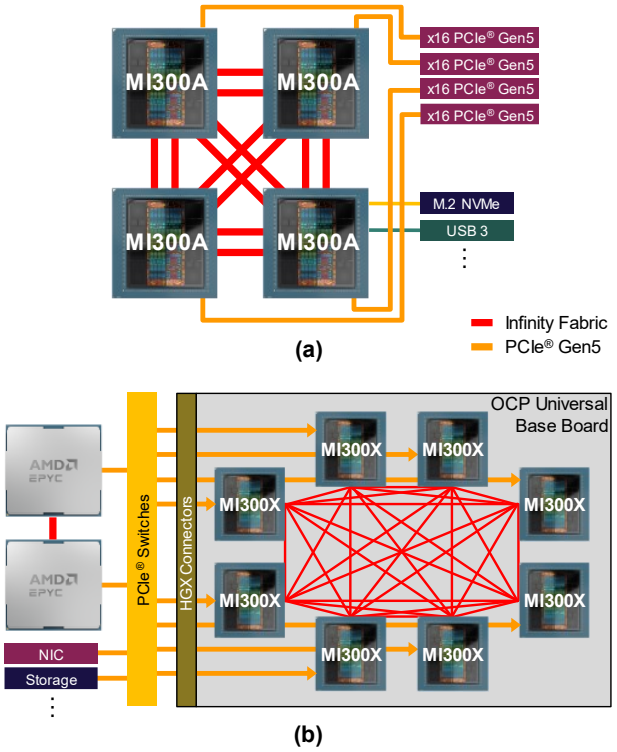

Figure 18. Exemplary node architecture configurations for (a) the MI300A APU and (b) the MI300X accelerator.

organized in a variety of scalable node topologies. Each IOD provides two x16 I/O interfaces. One of the x16 links is dedicated for Infinity Fabric connections, and the other can act as either an Infinity Fabric link or a PCIe gen5 interface. In total, each MI300 socket has eight x16 links (four of which can serve as either Infinity Fabric or PCIe). Each x16 link has a peak theoretical bidirectional bandwidth of 128 GB/s (64 GB/s per direction) for a total of 1,024 GB/s per socket.

Figure 18(a) shows an example MI300A node architecture featuring four MI300A APUs. Each MI300A has direct load-store access to all HBM across all four modules (i.e., flat physical address space). In this example, each MI300A socket uses six of its eight x16 I/O links to implement cache-coherent Infinity Fabric links in a fully-connected topology with the other MI300A modules (i.e., two x16 links between every pair of MI300A APUs). The remaining I/O interfaces can be made available for network interfaces, storage, etc.

Figure 18(b) shows a different system topology utilizing eight MI300X accelerator modules that act as PCIe devices connected to EPYC CPU hosts. In this case, seven of the eight x16 I/O links per MI300X module are utilized to create a fully-connected IF topology among the eight MI300X accelerators. The last PCIe link provides connectivity back to the CPU host, possibly through PCIe switches depending on the exact node architecture. Note that in these two scalable system topologies, both MI300A and MI300X utilize the exact same IOD designs, but we leverage the configurability of the interfaces to enable very different overall systems.

		AMD Instinct MI250X (Up to)	AMD Instinct MI300A (Up to)	AMD Instinct MI300X (Up to)
Hardware Specs	Memory Capacity	128 GB HBM2e	128 GB HBM3	192 GB HBM3
	Memory Bandwidth (Peak Theoretical)	~3.2 TB/s	~5.3 TB/s	~5.3 TB/s
	Scale-Out (Back-end) Network Bandwidth	200 Gb/s Ethernet/IB	400 Gb/s Ethernet/IB	400 Gb/s Ethernet/IB
	Max TDP/TBP	560 W	760 W	760 W
	Heterogeneous Integration	2D IF, 2.5D EFB, 3D µbump HBM	2.5D passive silicon interposer, 3D HB active interposer and multiple chiplets, 3D µbump HBM	
HPC Peak Perf. (Peak)	FP64 Vector (TFLOPS)	47.9	61.3	81.7
	FP32 Vector (TFLOPS)	47.9	122.6	163.4
	FP64 Matrix (TFLOPS)	95.7	122.6	163.4
	FP32 Matrix (TFLOPS)	95.7	122.6	163.4
AI Peak Perf. (Peak)	TF32* [TF32 Sparsity] (Matrix)	Not Supported	490.3 [980.6]	653.7 [1307.4]
	FP16 [FP16 Sparsity] (TFLOPS)	383.0 [Not Supported]	980.6 [1961.2]	1307.4 [2614.9]
	BFLOAT16 [BF16 Sparsity] (TFLOPS)	383.0 [Not Supported]	980.6 [1961.2]	1307.4 [2614.9]
	FP8* [FP8 Sparsity] (TFLOPS)	Not Supported	1961.2 [3922.3]	2614.9 [5229.8]
	INT8 [INT8 Sparsity] (TOPS)	383.0 [Not Supported]	1961.2 [3922.3]	2614.9 [5229.8]

Figure 19. Generational uplift of MI300A and MI300X over MI250X.

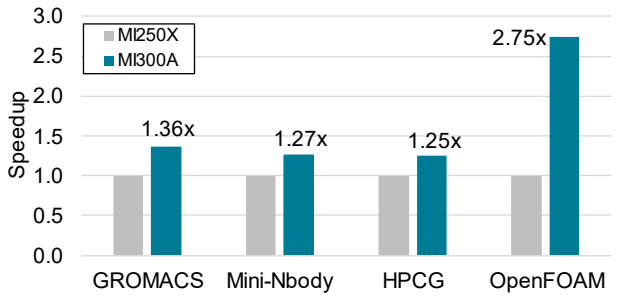

Figure 20. Measured speedups on HPC workloads of the AMD MI300A APU over an AMD MI250X accelerator.

IX. Results

The focus of this industry track paper is not on bottom-line performance, but rather on sharing with the computer architecture community our insights, learnings, and the evolution from the prior EHP research concepts to a high-volume, high-performance, highly aggressive yet still practically manufacturable series of APU and accelerator products to address HPC and AI markets. This section provides only a few results to give a flavor of the benefits of MI300A and MI300X.

The MI300A APU (and MI300X accelerator) provides generational performance improvements over the AMD Instinct MI250X accelerators. Figure 19 shows the side-by-side comparison of MI250X, MI300A, and MI300X across a variety of metrics of interest. Peak computational throughput rates are increased across the board, the peak memory bandwidth has also improved by 70%, and I/O (network) bandwidth has also doubled to support HPC and AI/ML scale out. For the MI300X accelerator, the total memory capacity is also 50% greater to better service LLMs.

Figure 20 shows performance speedups of a few HPC workloads of MI300A compared to the prior generation MI250X accelerator. The MI300A results were measured on an MI300A (128GB HBM3, 550W TDP*) bring-up reference platform running Ubuntu® Linux and ROCm 6.0, and the MI250X (128GB HBM2e, 560W TBP*) results used Ubuntu Linux with ROCm 5.4.3. All runs utilized a single APU or a single GPU. For GROMACS, the N-body kernel [16], and the HPCG [17] benchmark, MI300A delivers higher performance due to its higher compute throughput (GROMACS, N-body) and HBM3's higher memory bandwidth vs. the HBM2e memory in MI250X (HPCG). OpenFOAM® is a computational fluid dynamics workload [15] that sees a 2.75×

* TDP = Thermal Design Power, TBP = Total Board Power. Accounting for voltage regulation losses, MI300A TBP is approximately 670W.

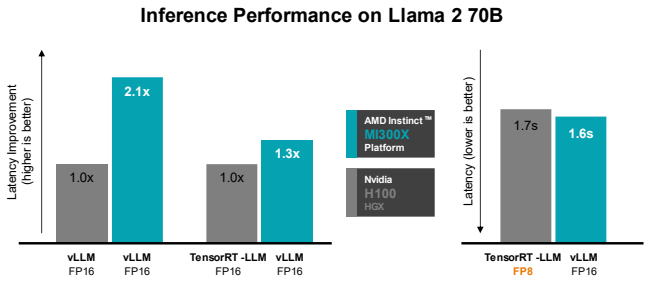

Figure 21. Llama-2 70B inference latency performance (median) using batch size 1, 2048 input tokens, and 128 output tokens.

performance improvement. This workload (on the HPC Motorbike test/input) really matches well with the APU computing paradigm as it (1) is computationally intense, (2) requires high memory bandwidth, and (3) also tends to exhibit a lot of CPU-GPU data movement in discrete-GPU implementations. MI300A's massive compute resources and eight stacks of HBM deliver on the first two needs of this workload. The APU's unified memory architecture addresses the data movement aspect, as the fastest way to move data is to not move it all if possible.

While the focus of this paper is on the MI300A APU for exascale computing, we also highlight a few results for the MI300X accelerator. Figure 21 shows performance results for inference on the Llama-2 model with 70 billion parameters [39]. For these experiments, we used a batch size of one, 2048 input tokens, and 128 output tokens. Full platform configuration details can be found online [6]. The first set of results on the left shows the improvement in median inference latency using the vLLM inference and serving library [21] for both the baseline GPU and MI300X to directly compare hardware capabilities. In this experiment, MI300X was measured to provide more than 2× improvement in inference latency. The second set of results uses the TensorRT-LLM library that is optimized specifically for the baseline GPU. Even in this scenario, MI300X still delivers a ≈30% improvement. The last set of results considers a baseline where the FP8 number format is used, which effectively allows the baseline GPU to double its peak compute and memory throughputs. The vLLM library currently does not support FP8, and so the MI300X results continue to use the FP16 format instead. Despite the difference in numerical formats, MI300X continues to demonstrate a performance advantage when measuring absolute latency.

X. Conclusions

While the seeds for our HPC APU vision were planted over a decade ago, some ideas, no matter how technically compelling, cannot be rushed to production before technology, business, and other conditions are ready. The contrast between the MI250X-based Frontier node architecture and the APU in the MI300A design just a couple years apart illustrates how roadmaps and product plans must be carefully adapted based on the specific circumstances. The decision to not implement an APU in the first-generation exascale supercomputer was not a technical indictment of the idea of an HPC APU, but it was simply not yet the right time. Technology matures, the market changes, and customer demands evolve, and now the time has finally arrived for AMD to bring forth our high-performance APU.

References

[1] Advanced Micro Devices, Inc., "AMD Instinct™ MI200 Series Accelerator," 2022, https://www.amd.com/system/files/documents/amd-instinct-mi200-datasheet.pdf.

[2] Advanced Micro Devices, Inc., "Introducing AMD CDNA™ 2 Architecture," 2021, https://www.amd.com/system/files/documents/amd-cdna-whitepaper.pdf

[3] Advanced Micro Devices, Inc., "AMD ROCm Open Software Platform for GPU Compute," http://www.amd.com/ROCm, 2022.

[4] Advanced Micro Devices, Inc., "4th-gen AMD EPYC™ Processor Architecture," https://www.amd.com/system/files/documents/4th-gen-epyc-processor-architecture-white-paper.pdf

[5] Advanced Micro Devices, Inc., "AMD CDNA™ 3 Architecture," https://www.amd.com/content/dam/amd/en/documents/instinct-tech-docs/white-papers/amd-cdna-3-white-paper.pdf

[6] Advanced Micro Devices, Inc., "Competitive performance claims and industry leading Inference performance on AMD Instinct MI300X," https://community.amd.com/t5/instinct-accelerators/competitive-performance-claims-and-industry-leading-inference/ba-p/652304, December 2023

[7] Argonne National Laboratory, Request for Information (RFI) No. 1-KD73-I-31583-00, "Project: Exascale Research and Development," July 2011.

[8] D. Beckingsale, J. Burmark, R. Hornung, H. Jones, W. Killian, A. Kunen, O. Pearce, P. Robinson, B. Ryujin, T. Scogland, "RAJA: Portable Performance for Large-Scale Scientific Applications," Int'l Workshop on Performance, Portability and Productivity in HPC, 2019.

[9] M. Bohr, "A 30 Year Retrospective on Dennard's MOSFET Scaling Paper," IEEE Solid-State Circuits Society Newsletter, vol. 12(1), 2007.

[10] J. Bradbury, R. Frostig, P. Hawkins, M. J. Johnson, C. Leary, D. Maclaurin, G. Necula, A. Paszke, J. VanderPlas, S. Wanderman-Milne, Q. Zhang, "JAX: Composable Transformations of Python+NumPy programs," http://github.com/google/jax, 2018

[11] K.-W. Chung, S. Shih, S.-T. Lu, T.-H. Chen, C.-T. Chen, J. Ho, J.-J. Chen, J.- P. Lin, "3D Stacking DRAM using TSV Technology and Microbump Interconnect," in the International Microsystems Packaging Assembly and Circuits Technology Conference, October 2010.

[12] Y. Deng and W. Maly. "Interconnect Characteristics of 2.5-D System Integration Scheme," Proceedings of the International Symposium on Physical Design, pages 171–175, April 2001.

[13] H. Carter Edwards, Christian R. Trott, and Daniel Sunderland. "Kokkos: Enabling manycore performance portability through polymorphic memory access patterns." Journal of parallel and distributed computing 74.12 (2014): 3202-3216.

[14] M. Evers, L. Barnes, M. Clark, "The AMD Next-Generation 'Zen 3' Core," in IEEE Micro, Vol. 42(3), May-June 2022.

[15] C. Greenshields, "OpenFOAM v11 User Guide," The OpenFOAM Foundation, 2023, https://doc.cfd.direct/openfoam/user-guide-v11

[16] M. Harris, "mini-nbody: A simple N-body Code," https://github.com/harrism/mini-nbody

[17] M. A. Heroux, J. Dongarra, P. Luszczek, "HPCG Technical Specification," Sandia National Laboratories Technical Report, SAND2013-8752, October 2013.

[18] HSA Foundation, "HSA Platform System Architecture Specification Version 1.2," May 2018.

[19] IEEE Electronics Packaging Society, "Heterogeneous Integration Roadmap (2021 Edition), http://eps.ieee.org/hir

[20] A. Kannan, N. Enright Jerger, G. H. Loh, "Enabling Interposer-based Disintegration of Multi-core Processors," International Symposium on Microarchitecture (MICRO), December 2015.

[21] Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Yu, Joey Gonzalez, Hao Zhang, Ion Stoica, "vLLM: Easy, Fast, and Cheap LLM Serving with PagedAttention," https://blog.vllm.ai/2023/06/20/vllm.html

[22] G. Loh, M. Schulte, M. Ignatowski, V. Adhinarayanan, S. Aga, D. Aguren, V. Agrawal, A. M. Aji, J. Alsop, P. Bauman, B. M. Beckmann, M. V. Beigi, S. Blagodurov, T. Boraten, M. Boyer, W. Brantley, N. Chalmers, S. Chen, K. Cheng, M. Chu, D. Cownie, N. Curtis, J. del Pino, N. Duong, A. Dutu, Y. Eckert, C. Erb, C. Freitag, J. Greathouse, S. Gurumurthi, A. Gutierrez, K. Hamidouche, S. Hossamani, W. Huang, M. Islam, N. Jayasena, J. Kalamatianos, O. Kayiran, J. Kotra, A. Lee, D. Lowell, N. Madan, A. Majumdar, N. Malaya, S. Manne, S. Mashimo, D. McDougall, E. Mednick, M. Mishkin, M. Nutter, I. Paul, M. Poremba, B. Potter, K. Punniyamurthy, S. Puthoor, S. Raasch, K. Rao, G. Rodgers, M. Scrbak, M. Seyedzadeh, J. Slice, V. Sridharan, R. van Oostrum, E. van Tassell, A. Vishnu, S. Wasmundt, M. Wilkening, N. Wolfe, M. Wyse, A. Yalavarti, D. Yudanov, "A Research Retrospective on AMD's Exascale Computing Journey," in the International Symposium on Computer Architecture, June 2023.

[23] N. Malaya, B. Messer, J. Glenski, A. Georgiadou, J. Lietz, K. Gottiparthi, M. Day, J. Chen, J. Rood, L. Esclapez, J. White III, G. R. Jansen, N. Curtis, S. Nichols, J. Kurzak, N. Chalmers, C. Freitag, P. Bauman, A. Fanfarillo, R. D. Budiardja, T. Papatheodore, N. Frontiere, D. Mcdougall, M. Norman, S. Sreepathi, P. Roth, D. Bykov, N. Wolfe, P. Mullowney, M. Eisenbach, M. T. Henry De Frahan, W. Joubert, "Experiences readying applications for Exascale," in the International Conference for High Performance Computing, Networking, Storage and Analysis (SC '23), November 2023.

[24] G. S. Markomanolis, A. Alpay, J. Young, M. Klemm, N. Malaya, A. Esposito, J. Heikonen, S. Bastrakov, A. Debus, T. Kluge, K. Steiniger, J. Stephan, R. Widera, M. Bussmann, "Evaluating GPU Programming Models for the LUMI Supercomputer," in Supercomputing Frontiers: 7th Asian Conference, March 2022.

[25] G. E. Moore, "Cramming More Components onto Integrated Circuits," in Electronics, Vol. 38, No. 8, April 1965.

[26] B. Munger, K. Wilcox, J. Sniderman, C. Tung, B. Johnson, R. Schreiber, C. Henrion, K. Gillespie, T. Burd, H. Fair, D. Johnson, J. White, S. McLelland, S. Bakke, J. Olson, R. McCracken, M. Pickett, A. Horiuchi, H. Nguyen, T. H. Jackson, "'Zen 4': The AMD 5nm 5.7GHz x86-64 Microprocessor Core," in the International Solid-State Circuits Conference, February 2023.

[27] S. Naffziger, N. Beck, T. Burd, K. Lepak, G. H. Loh, M. Subramony, S. White, "Pioneering Chiplet Technology and Design for the AMD EPYC™ and Ryzen™ Processor Families," in the International Symposium on Computer Architecture (ISCA), June 2021

[28] Oak Ridge National Laboratory, "No Scaling Back: DOE, Cray, AMD to Bring Exascale to ORNL," May 2019, https://www.olcf.ornl.gov/2019/05/07/no-scaling-back-doe-cray-amd-to-bring-exascale-to-ornl/

[29] Oak Ridge National Laboratory, "Frontier Supercomputer Debuts as World's Fastest, Breaking Exascale Barrier," https://www.ornl.gov/news/frontier-supercomputer-debuts-worlds-fastest-breaking-exascale-barrier

[30] Oak Ridge National Laboratory, "Frontier User Guide," 2022, https://docs.olcf.ornl.gov/systems/frontier_user_guide.html

[31] M. J. Schulte, M. Ignatowski, G. H. Loh, B. M. Beckmann, W. C. Brantley, S. Gurumurthi, N. Jayasena, I. Paul, S. K. Reinhardt, G. Rodgers, "Achieving Exascale Capabilities through Heterogeneous Computing," IEEE Micro, July-August 2015.

[32] A. Smith, N. James, "AMD Instinct™ MI200 Series Accelerator and Node Architectures," in Hot Chips, August 2022.

[33] A. Smith, E. Chapman, C. Patel, R. Swaminathan, J. J. Wuu, T. Huang, W. Jung, A. Kaganov, H. McIntyre, R. Mangaser, "AMD InstinctTM MI300 Series Modular Chiplet Package - HPC and AI Accelerator for Exa-Class Systems," in the International Solid-State Circuits Conference, February 2024.

[34] R. Stevens, V. Taylor, J. Nichols, A. B. Maccabe, K. Yelick, D. Brown, "AI for Science: Report on the Department of Energy (DOE) Town Halls on Artificial Intelligence (AI) for Science," ANL-20/17 158802. February 2020.

[35] R. Swaminathan, "Case Study: AMD Products Built with 3D Packaging (Tutorial)," in Hot Chips, August 2021.

[36] R. Swaminathan, M. J. Schulte, B. Wilkerson, G. H. Loh, A. Smith, N. James, "AMD Instinct™ MI250X Accelerator enabled by Elevated Fanout Bridge Advanced Packaging Architecture," in the International Symposium on VLSI Technology and Circuits, June 2023.

[37] J. Thomas, "LLNL and HPE to partner with AMD on El Capitan, projected as world's fastest supercomputer," Lawrence Livermore National Laboratory, March 2020, https://www.llnl.gov/archive/news/llnl-hpe-partner-amd-el-capitan-projected-worlds-fastest-supercomputer

[38] P. Tillet, H. T. Kung, D. Cox, "Triton: an intermediate language and compiler for tiled neural network computations," In Proceedings of the 3rd ACM SIGPLAN International Workshop on Machine Learning and Programming Languages, 2019

[39] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, A. Rodriguez, A. Joulin, E. Grave, G. Lample, "LLaMA: Open and Efficient Foundation Language Models", *arXiv*:2302.13971 [cs.CL], https://doi.org/10.48550/arXiv.2302.13971

[40] K. Tran, "The Era of High Bandwidth Memory," Tutorial in HotChips 28, August 2016.

[41] K. Troester, R. Bhargava, "AMD Next Generation 'Zen 4' Core and 4th Gen AMD EPYC™ 9004 Server CPU," in HotChips 2023.

[42] T. Vijayaraghavan, Y. Eckert, G. Loh, M. Schulte, M. Ignatowski, I. Paul, B. Beckmann, S. Reinhardt, W. Brantley, J. Greathouse, O. Kayiran, M. Poremba, W. Huang, A. Karunanithi, G. Sadowski, V. Sridharan, S. Raasch, M. Meswani, "Design and Analysis of an APU for Exascale Computing," in the International Symposium on High-Performance Computer Architecture (HPCA), February 2017.

[43] W. A. Wulf, S. A. McKee, "Hitting the Memory Wall: Implications of the Obvious," ACM SIGARCH Computer Architecture News, Volume 23, Issue 1, pp. 20–24, March 1995.

[44] J. J. Wuu, R. Agarwal, M. Ciraula, C. Dietz, B. Johnson, D. Johnson, R. Schreiber, R. Swaminathan, W. Walker, S. Naffziger, "3D V-Cache: the Implementation of a Hybrid-Bonded 64MB Stacked Cache for a 7nm x86-64 CPU," in the International Solid-State Circuits Conference, February 2022.