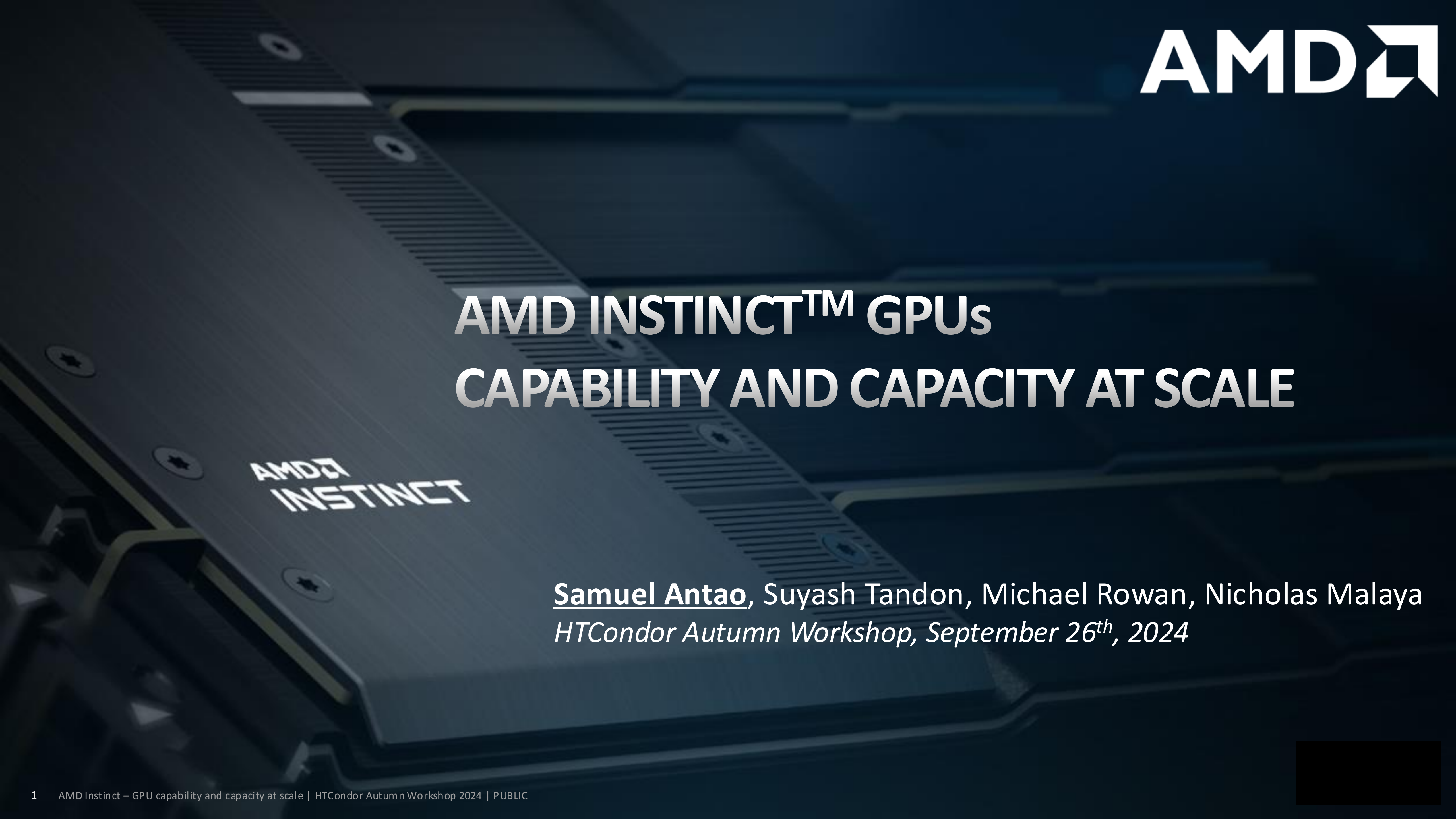


AMD INSTINCT™ GPUs CAPABILITY AND CAPACITY AT SCALE

Samuel Antao, Suyash Tandon, Michael Rowan, Nicholas Malaya

HTCondor Autumn Workshop, September 26th, 2024

[redacted]

AMD INSTINCT™ GPU FOOTPRINT IN HPC & AI

PROVEN AT-SCALE WINS

INSTINCT POWERS SEVERAL OF TOP SUPERCOMPUTERS INCLUDING #1 FRONTIER

TOP500 The List.

FRONTIER

LUMI

GENCI

CINES

Microsoft Azure

KUNGL TEKNISKA HÖGSKOLAN

PAWSEY supercomputing centre

PARTNERED WITH AI THOUGHT-LEADERS & CLOUD PROVIDERS

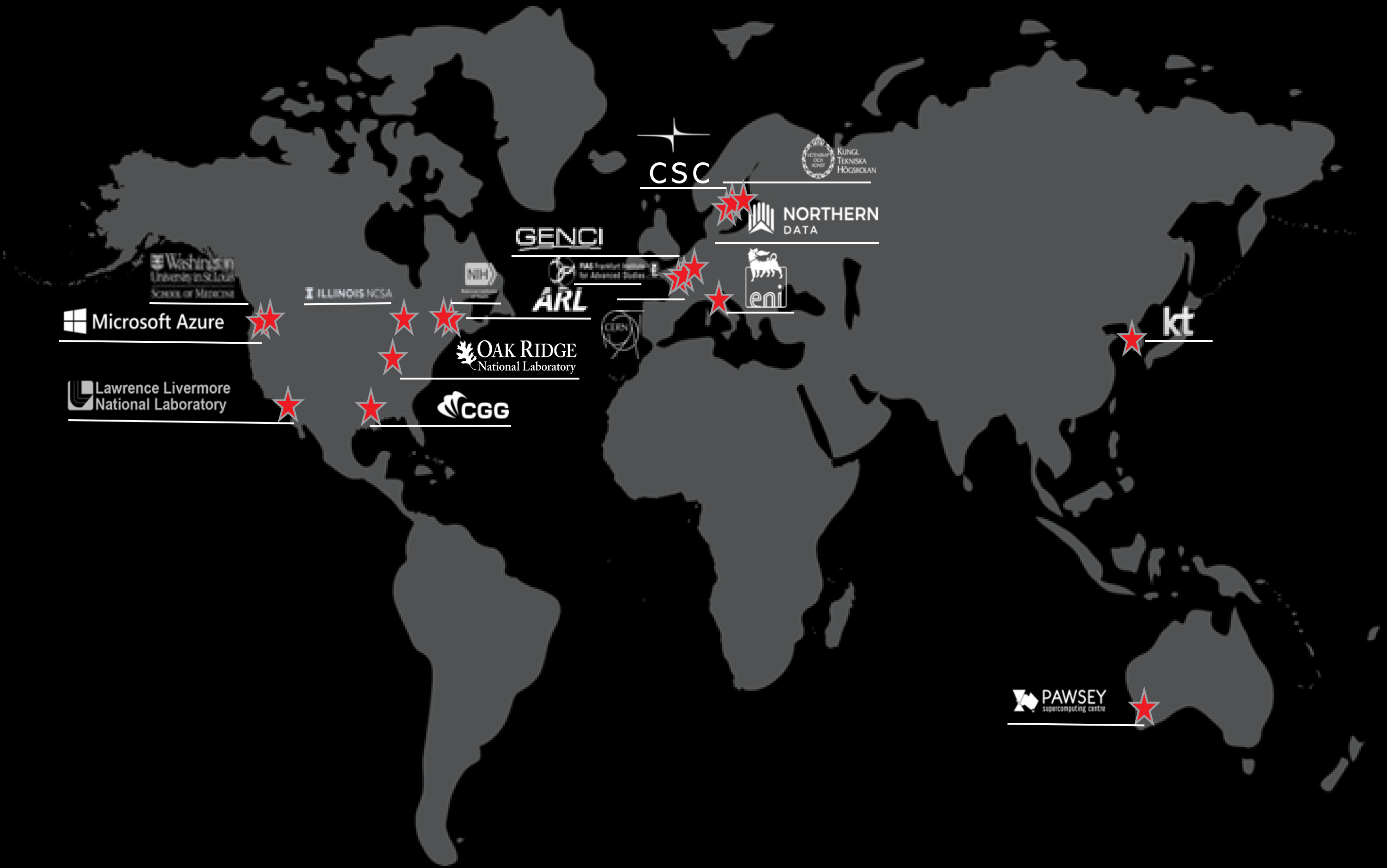
Microsoft Azure

kt

AI2

LANDING AI

+ More to be Announced


CSC
KUNGL TEKNISKA HÖGSKOLAN
NORTHERN DATA
GENCI
Washington University in St. Louis School of Medicine
ILLINOIS NCSA
NIH
FIAS Frankfurt Institute for Advanced Studies
ARL
eni
Microsoft Azure
CERN
kt
OAK RIDGE National Laboratory
Lawrence Livermore National Laboratory
CGG
PAWSEY supercomputing centre

Source: TOP500, Green500, and HPL-MxP lists, as of June, 2024

AMD PLATFORM FOR ACCELERATED COMPUTING

LEADERSHIP IN HPC & AI FOR EXASCALE-CLASS COMPUTING

AMD CDNA

WORKLOAD-OPTIMIZED COMPUTE ARCHITECTURE

PURPOSE-BUILT, OPTIMIZED ARCHITECTURE
DESIGNED TO DO ONE THING EXTREMELY WELL: COMPUTE INTENSIVE HPC AND AI WORKLOADs

AMD ROCm

OPEN & PORTABLE SOFTWARE

SIMPLIFIED PROGRAMMABILITY AND INCREASED USABILITY
THROUGH A GROWING SOFTWARE ECOSYSTEM AND A RICH SUITE OF OPTIMIZED LIBRARIES, FRAMEWORKS AND TOOLS

DEEP PARTNERSHIP WITH LEADING HPC CENTERS & AI THOUGHT- LEADERS

OAK RIDGE National Laboratory

CSC

PAWSEY supercomputing centre

GENCI

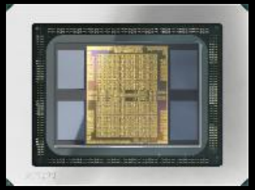
CINES

Lawrence Livermore National Laboratory

Microsoft Azure

AMD
together we advance_

OUR JOURNEY IN GPU ACCELERATION

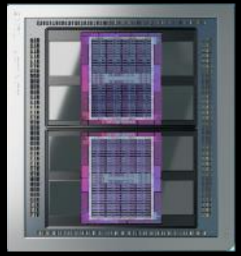


AMD Instinct™ MI100

AMD CDNA™

Ecosystem Growth

First purpose-built GPU architecture for the data center

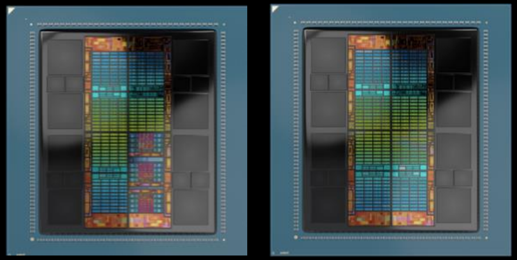


AMD Instinct™ MI200

AMD CDNA™ 2

Driving HPC and AI to a New Frontier

First purpose-built GPU powering discovery at Exascale



AMD Instinct™ MI300

AMD CDNA™ 3

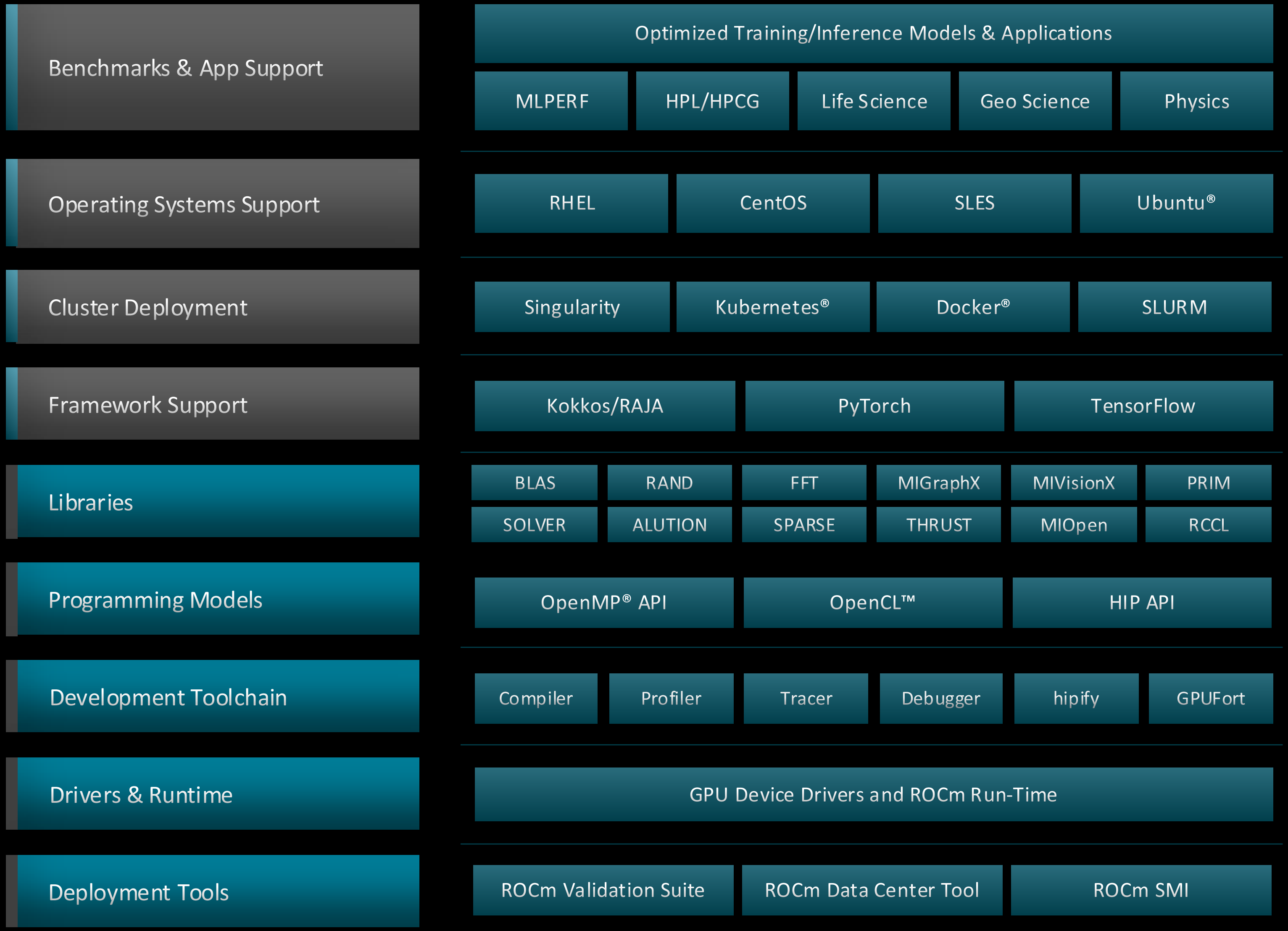
Data Center APU & Discrete GPU

Breakthrough architecture designed for leadership efficiency and performance for AI and HPC

OPEN SOFTWARE PLATFORM FOR GPU COMPUTE

AMD ROCm

- Unlocked GPU Power To Accelerate Computational Tasks
- Optimized for HPC and Deep Learning Workloads at Scale
- Open Source Enabling Innovation, Differentiation, and Collaboration



Benchmarks & App Support	Optimized Training/Inference Models & Applications MLPERF HPL/HPCG Life Science Geo Science Physics
Operating Systems Support	RHEL CentOS SLES Ubuntu®
Cluster Deployment	Singularity Kubernetes® Docker® SLURM
Framework Support	Kokkos/RAJA PyTorch TensorFlow
Libraries	BLAS RAND FFT MIGraphX MIVisionX PRIM SOLVER ALUTION SPARSE THRUST MIOpen RCCL
Programming Models	OpenMP® API OpenCL™ HIP API
Development Toolchain	Compiler Profiler Tracer Debugger hipify GPUFort
Drivers & Runtime	GPU Device Drivers and ROCm Run-Time
Deployment Tools	ROCm Validation Suite ROCm Data Center Tool ROCm SMI

TRANSITIONING WORKLOADS TO INSTINCT GPUS

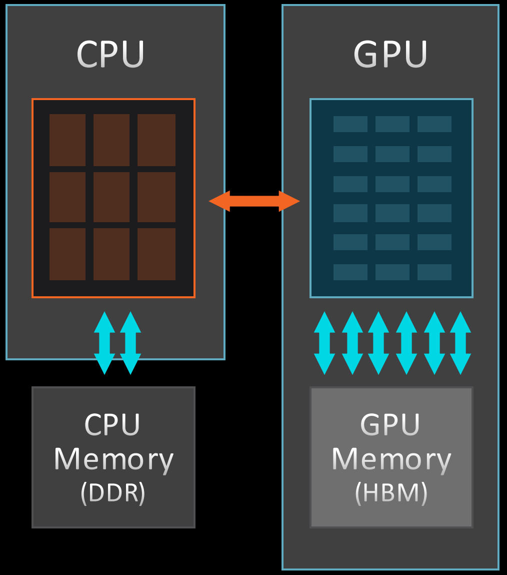
LOW FRICTION SOFTWARE PORTING FOR EXISTING NVIDIA USERS TO AMD

	NVIDIA		AMD
ML Frameworks Python™	PyTorch TensorFlow ONNX JAX	→ DROP IN Out-of-the-box Support	PyTorch TensorFlow ONNX JAX
HPC/ML Kernel Development C++, Triton IR	TRITON KERNELS → CUDA Triton Backend → GPU	DROP-IN →	TRITON KERNELS → ROCM Triton Backend → GPU
	CUDA CUSTOM KERNELS → NVCC → GPU	→ PORT/OPTIMIZE	CUDA CUSTOM KERNELS → *HIPIFY* → HIPCC → GPU
HPC/ML Libraries C++	cuBLAS, cuSparse, cuFFT, NCCL, cuDNN… GPU	→ MIRROR Equivalent Libraries	rocBLAS, rocSparse, rocFFT, RCCL, MIOpen… GPU

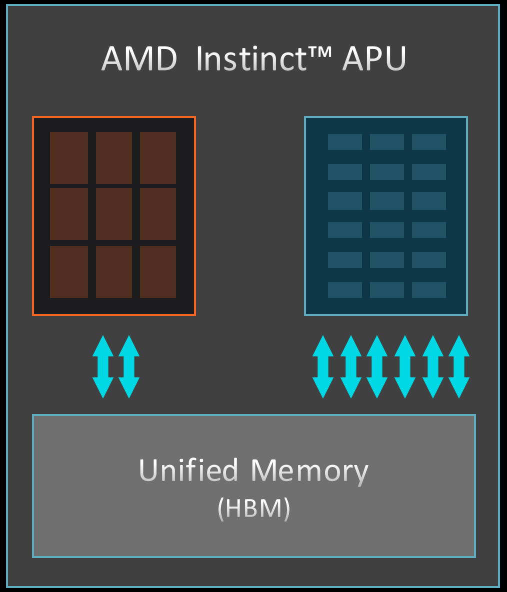
AMD

MI300A UNIFIED MEMORY APU ARCHITECTURE BENEFITS

AMD CDNA™ 2 Coherent Memory Architecture → AMD CDNA™ 3 Unified Memory APU Architecture



- **Eliminate Redundant Memory Copies**
- **No programming distinction between host and device memory spaces**
- **High performance, fine-grained sharing between CPU and GPU processing elements**
- **Single process can address all memory, compute elements on a socket**



AMD together we advance_

APU PROGRAMMING MODEL

CPU CODE

```
double* in_h = (double*)malloc(Msize);
double* out_h = (double*)malloc(Msize);


for (int i=0; i<M; i++) // initialize
   in_h[i] = …;

cpu_func(in_h, out_h, M);


for (int i=0; i<M; i++) // CPU-process
  … = out_h[i];
```

GPU CODE

```
double* in_h = (double*)malloc(Msize);
double* out_h = (double*)malloc(Msize);
hipMalloc(&in_d, Msize);
hipMalloc(&out_d, Msize);

for (int i=0; i<M; i++) // initialize
   in_h[i] = …;
hipMemcpy(in_d,in_h,Msize);
gpu_func<< >>(in_d, out_d, M);
hipDeviceSynchronize();
hipMemcpy(out_h,out_d,Msize);

for (int i=0; i<M; i++) // CPU-process
  … = out_h[i];
```

APU CODE

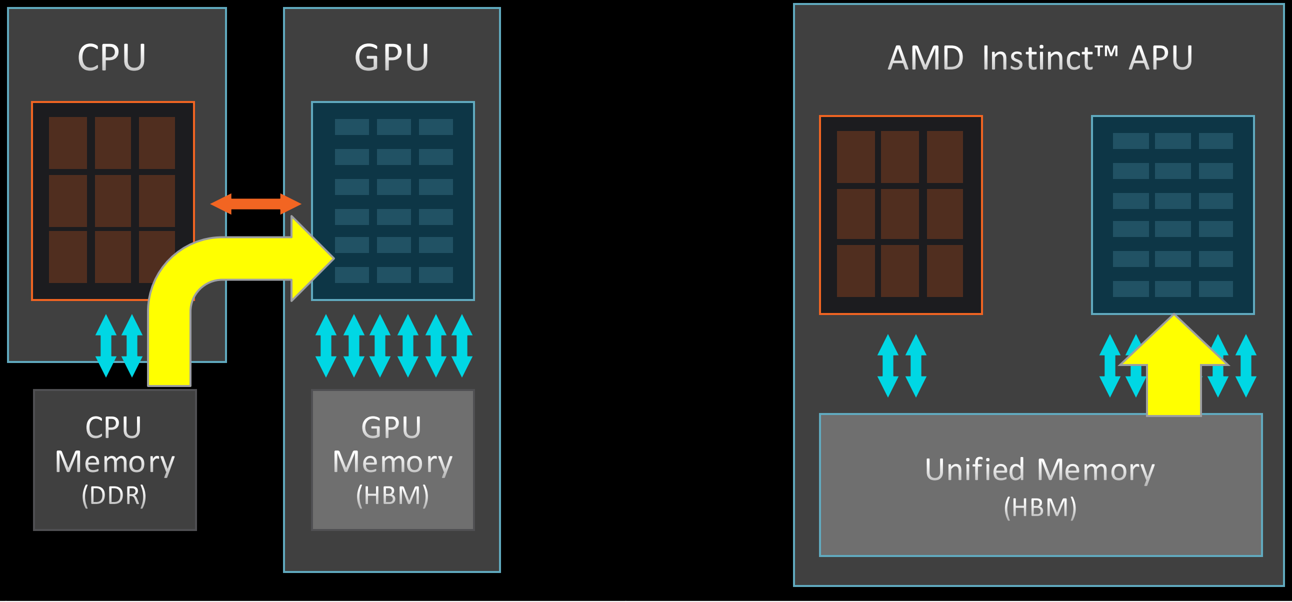
```
double* in_h = (double*)malloc(Msize);
double* out_h = (double*)malloc(Msize);


for (int i=0; i<M; i++) // initialize
   in_h[i] = …;

gpu_func<< >>(in_h, out_h, M);
hipDeviceSynchronize();


for (int i=0; i<M; i++) // CPU-process
  … = out_h[i];
```

- GPU memory allocation on Device
- Explicit memory management between CPU & GPU
- Synchronization Barrier

APU PROGRAMMING: PERFORMANCE IMPLICATIONS

GPU CODE

```
double* in_h = (double*)malloc(Msize);
double* out_h = (double*)malloc(Msize);
hipMalloc(&in_d, Msize);
hipMalloc(&out_d, Msize);


for (int i=0; i<M; i++) //initialize
   in_h[i] = …;
hipMemcpy(in_d,in_h,Msize);
gpu_func<< >>(in_d, out_d, M);
hipDeviceSynchronize();
hipMemcpy(out_h,out_d,Msize);

for (int i=0; i<M; i++) // CPU-process
  … = out_h[i];
```



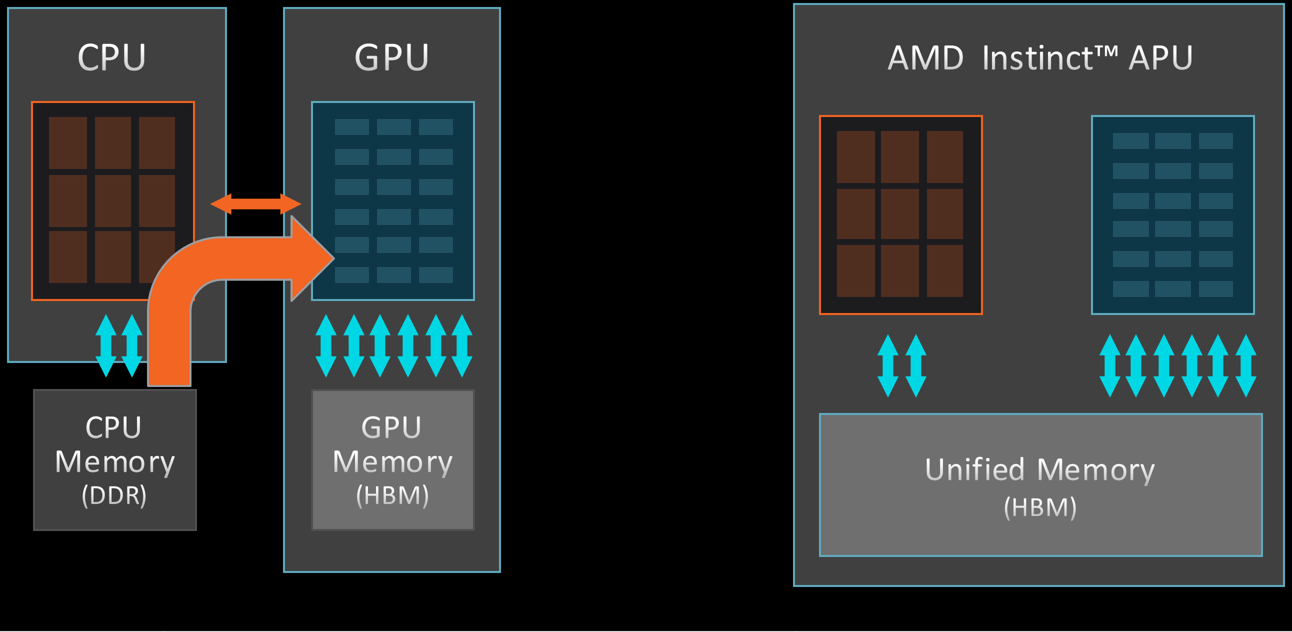
Operation	MI250X (MCM)	MI300A
H2D Copy	O(10) GB/s	O(TB/s)

- GPU memory allocation on Device
- Explicit memory management between CPU & GPU
- Synchronization Barrier

APU PROGRAMMING: PERFORMANCE IMPLICATIONS

APU CODE

```
double* in_h = (double*)malloc(Msize);
double* out_h = (double*)malloc(Msize);




for (int i=0; i<M; i++) //initialize
   in_h[i] = …;

gpu_func<< >>(in_h, out_h, M);
hipDeviceSynchronize();


for (int i=0; i<M; i++) // CPU-process
  … = out_h[i];
```



Operation	MI250X (MCM)	MI300A
Coherent Access	O(10) GB/s	N/A

- ~~GPU memory allocation on Device~~
- ~~Explicit memory management between CPU & GPU~~
- Synchronization Barrier

PROGRAMMING ACROSS FRAMEWORKS/COMPILERS

OpenMP® CODE

```
#pragma omp requires unified_shared_memory
 double* in_h = (double*)malloc(Msize);
double* out_h = (double*)malloc(Msize);

for (int i=0; i<M; i++) // initialize
   in_h[i] = …;

#pragma omp target
{ … }

for (int i=0; i<M; i++) // CPU-process
  … = out_h[i];
```

RAJA CODE

```
double* in_h = (double*)malloc(Msize);
double* out_h = (double*)malloc(Msize);

for (int i=0; i<M; i++) // initialize
   in_h[i] = …;

RAJA::forall< exec_policy >(arange, [=]
(int i) { … } );

for (int i=0; i<M; i++) // CPU-process
  … = out_h[i];
```

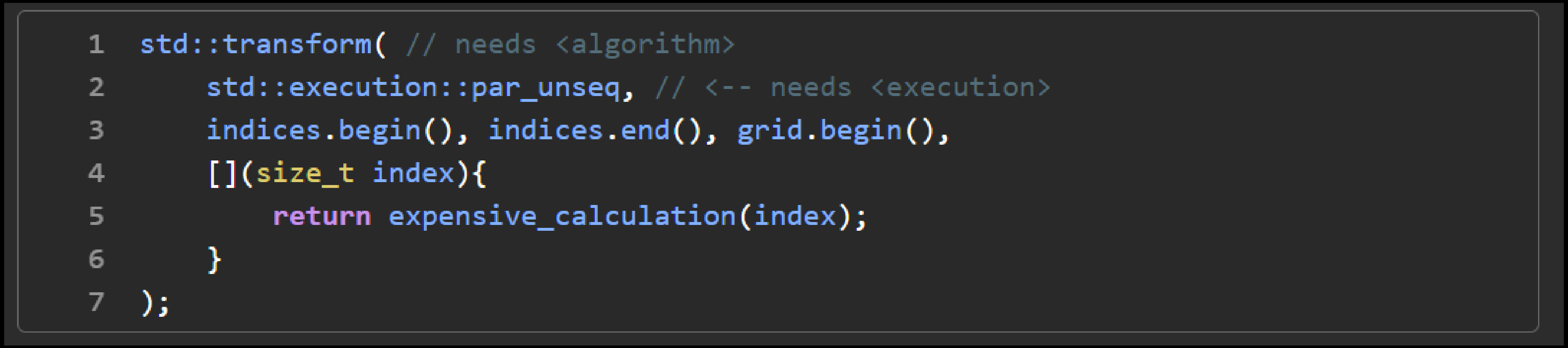
KOKKOS CODE

```
double* in_h = (double*)malloc(Msize);
double* out_h = (double*)malloc(Msize);

for (int i=0; i<M; i++) // initialize
   in_h[i] = …;

Kokkos::parallel_for(M, [=] (const  int
i){ … };
Kokkos::fence();

for (int i=0; i<M; i++) // CPU-process
  … = out_h[i];
```

- ~~GPU memory allocation on Device~~
- ~~Explicit memory management between CPU & GPU~~
- Synchronization Barrier

HIP STANDARD PARALLELISM

- AMD providing support for advanced C++ in LLVM: today, entirely Open Source Software (OSS)
  - Only supports par_unseq acceleration currently
  - Compiler support implementation upstreamed
  - Available in recent ROCm releases
  - Re-uses HIP support in CLANG/LLVM and algorithms from libraries (rocThrust)
  - Available today: https://github.com/ROCmSoftwarePlatform/roc-stdpar
    - Several applications tested and running (LULESH, etc.)



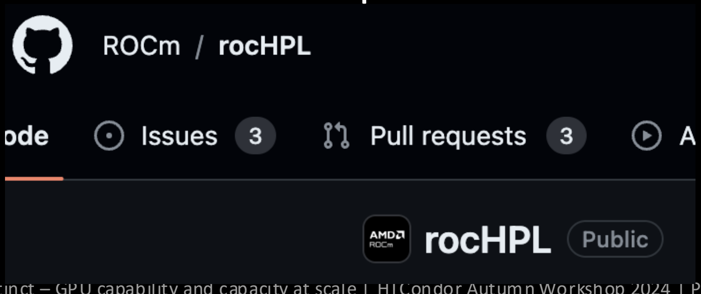
```
std::transform( // needs <algorithm>
    std::execution::par_unseq, // <-- needs <execution>
    indices.begin(), indices.end(), grid.begin(),
    [](size_t index){
        return expensive_calculation(index);
    }
);
```

AMD PARTNERSHIPS IN HPC AND AI

- Centers of Excellence
- In 2023, AMD HPC teams supported many focused training activities
  - >100 days of training, >3000 participants
  - Trainings, hackathons, private hackathons, virtual hackathons
- Focused on application porting, tuning, and analysis
  - Learnings are used to influence our hardware and software roadmap (co-design)
- Papers and dissemination of best practices

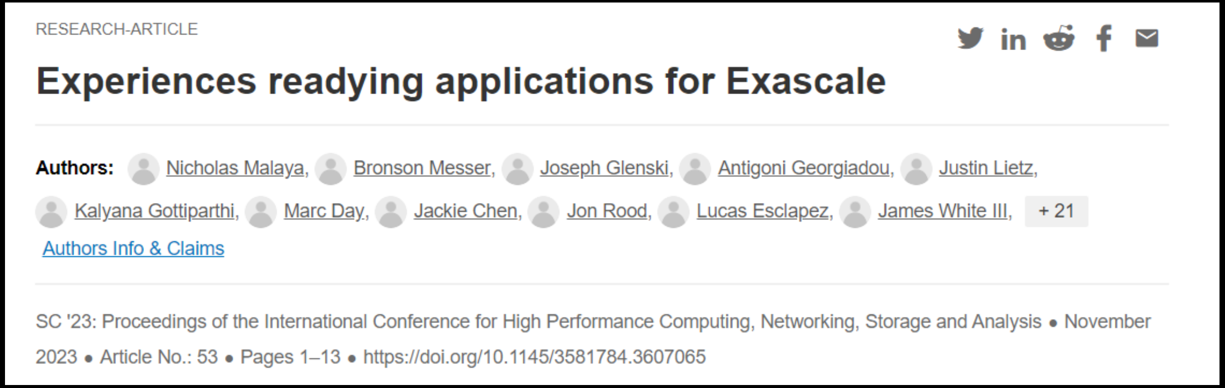
LLVM HIP OpenMP

- Commitment to Open Source Software


ROCm / rocHPL

ode · Issues 3 · Pull requests 3 · A

rocHPL Public

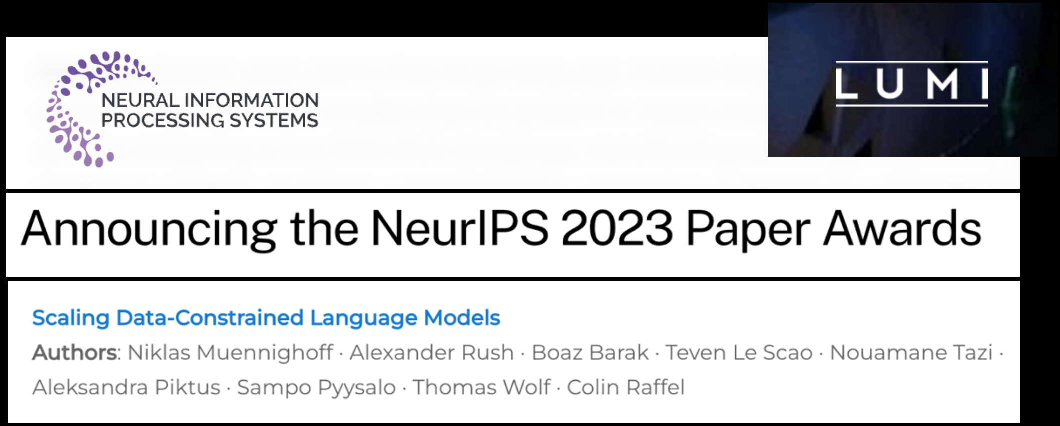

RESEARCH-ARTICLE

Experiences readying applications for Exascale

Authors: Nicholas Malaya, Bronson Messer, Joseph Glenski, Antigoni Georgiadou, Justin Lietz, Kalyana Gottiparthi, Marc Day, Jackie Chen, Jon Rood, Lucas Esclapez, James White III, + 21

Authors Info & Claims

SC '23: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis • November 2023 • Article No.: 53 • Pages 1–13 • https://doi.org/10.1145/3581784.3607065


NEURAL INFORMATION PROCESSING SYSTEMS

LUMI

Announcing the NeurIPS 2023 Paper Awards

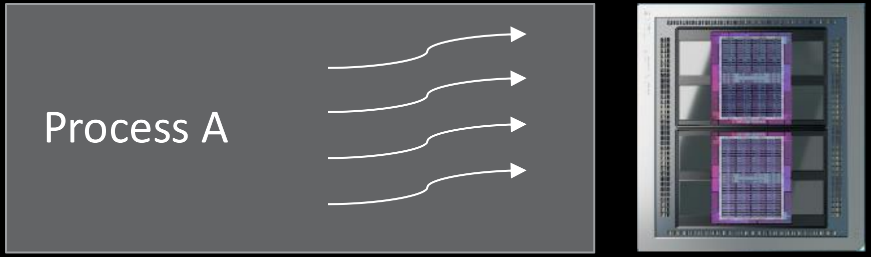
Scaling Data-Constrained Language Models

Authors: Niklas Muennighoff · Alexander Rush · Boaz Barak · Teven Le Scao · Nouamane Tazi · Aleksandra Piktus · Sampo Pyysalo · Thomas Wolf · Colin Raffel

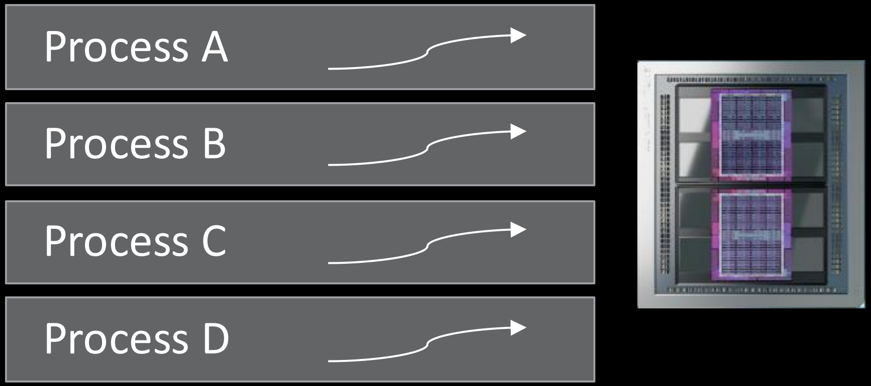
OVERSUBSCRIBING THE GPU

IMPROVING KERNEL THROUGHPUT

- Instinct GPUs provide HW support to run concurrently multiple contexts
- Process isolation – management of virtual memory per process
- Number of queues can be controlled at runtime with the environment variable GPU_MAX_HW_QUEUES



- Throughput within a process
  - Multiple host threads concurrently use the same HIP context
  - Use case:
    - Several tiny kernels and data transfers running concurrently
    - Typical when CPU applications are being ported to GPUs

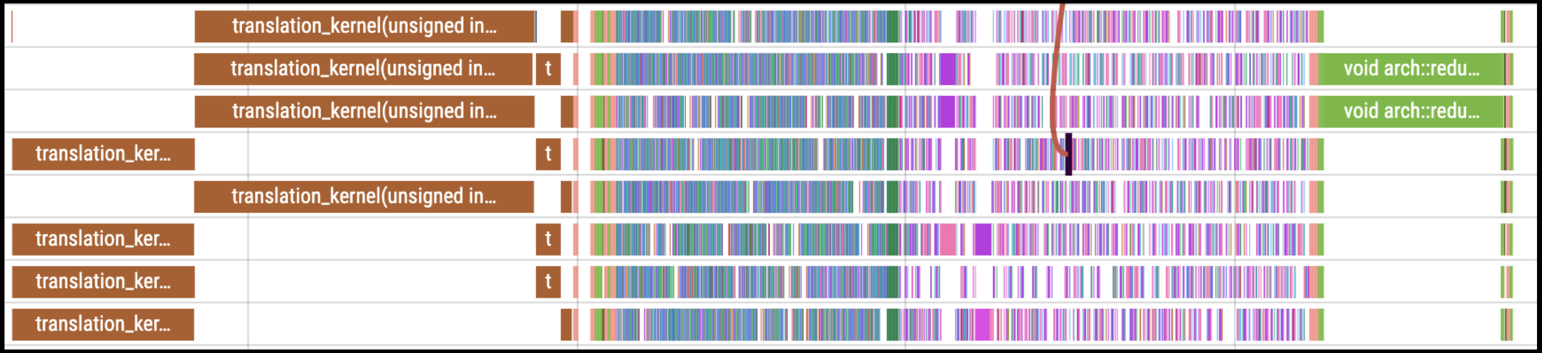


- Throughput across processes
  - Same GPU supporting multiple HIP contexts from different processes
  - Reliance on GPU driver to multiplex contexts
  - No need for software solutions to multiplex contexts in time

OVERSUBSCRIBING THE GPU

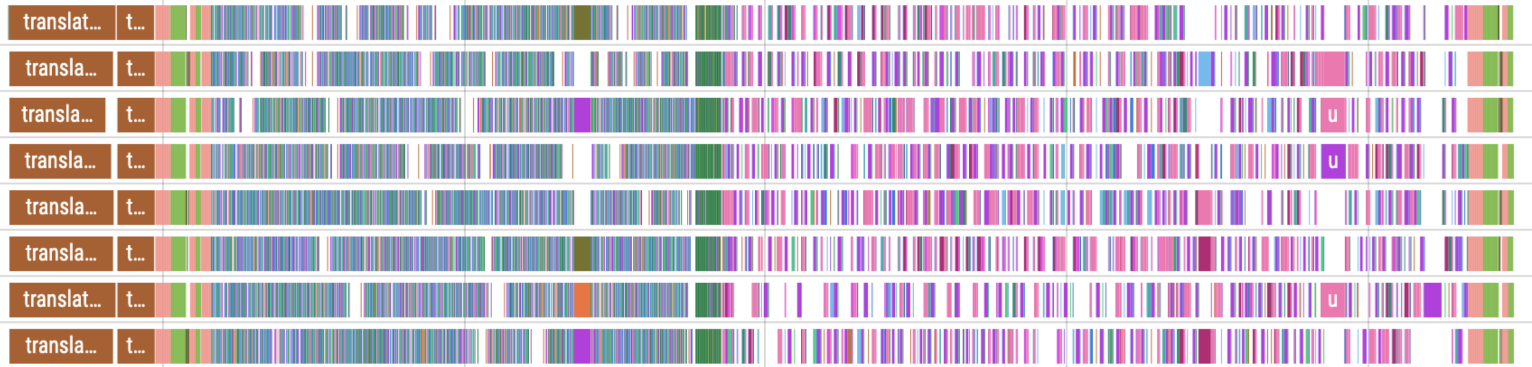
IMPROVING KERNEL THROUGHPUT – MULTIPLE THREADS

- Allowing a process to use more GPU HW queues (MI250X Instinct GPU)
  - export GPU_MAX_HW_QUEUES=4 (default)


translation_kernel(unsigned in...
translation_kernel(unsigned in...
t
void arch::redu...
translation_kernel(unsigned in...
void arch::redu...
translation_ker...
t
translation_kernel(unsigned in...
translation_ker...
t
translation_ker...
t
translation_ker...

Before: 4.578s

  - export GPU_MAX_HW_QUEUES=22



After: 2.497s

Same workload

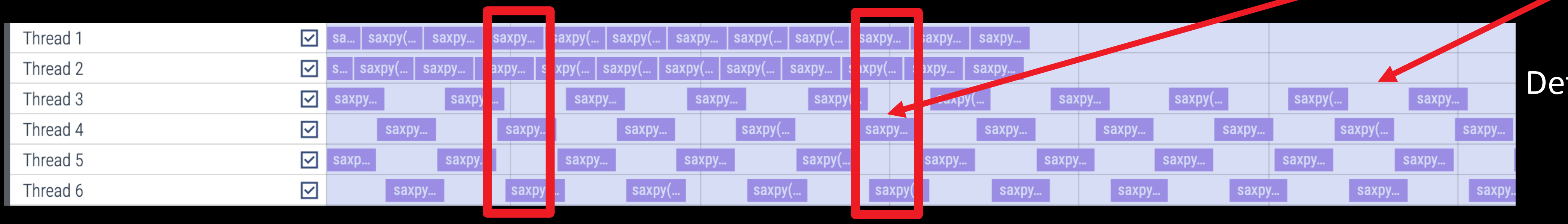
Speedup is 1.9x

More overlap of the activity from different streams

Better GPU occupancy

OVERSUBSCRIBING THE GPU

IMPROVING KERNEL THROUGHPUT – MULTIPLE THREADS VS PROCESS

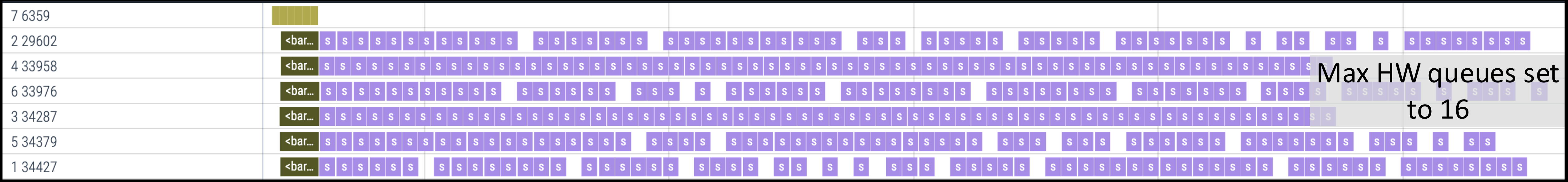


Maximum overlap of 4 kernels

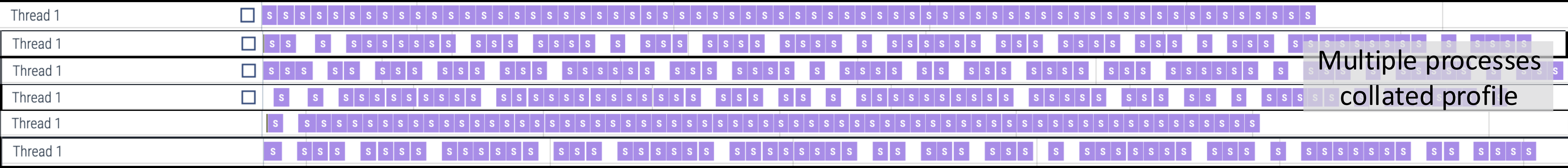
Tail effect

Default max (4) HW queues

- We can optimize throughput and be less prone to tail effects by increasing number of queues properly
- Throughput observed with threading similar to throughput over multiple processes:



Max HW queues set to 16



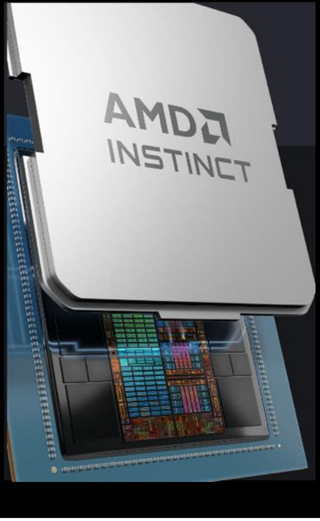
Multiple processes collated profile

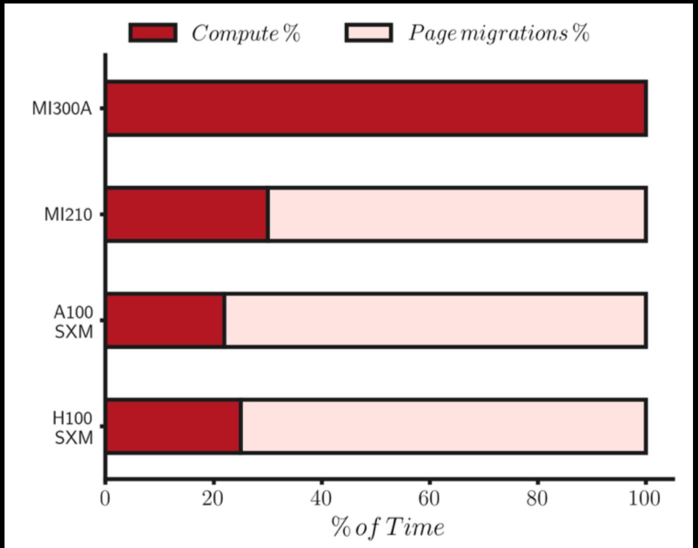
AMD
together we advance_

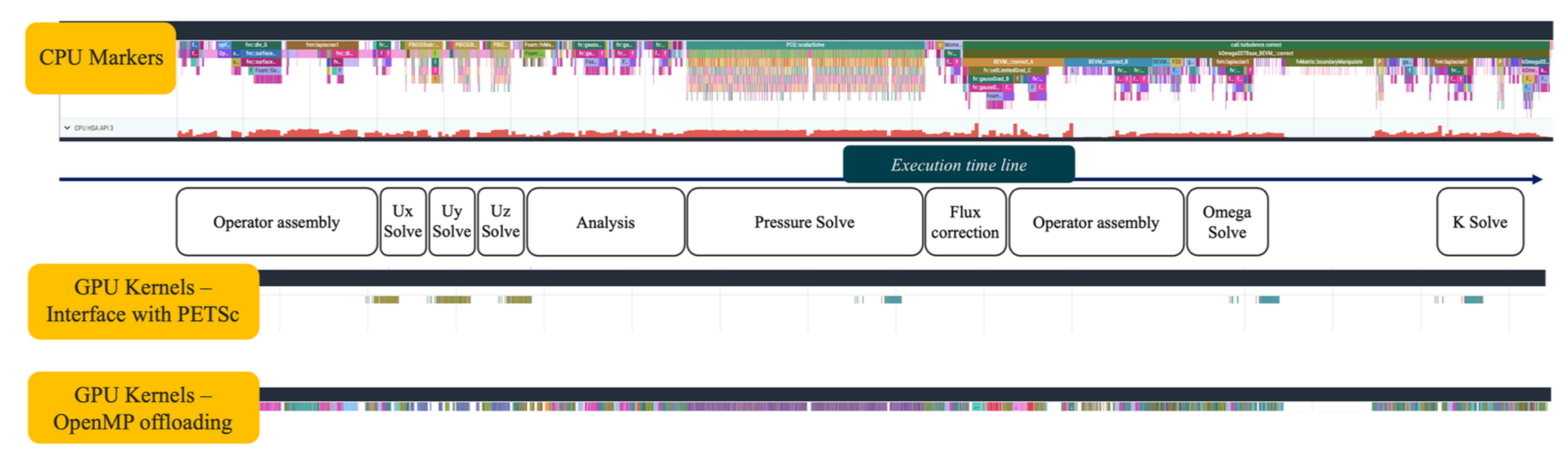
LEVERAGING THE APU PROGRAMMING MODEL

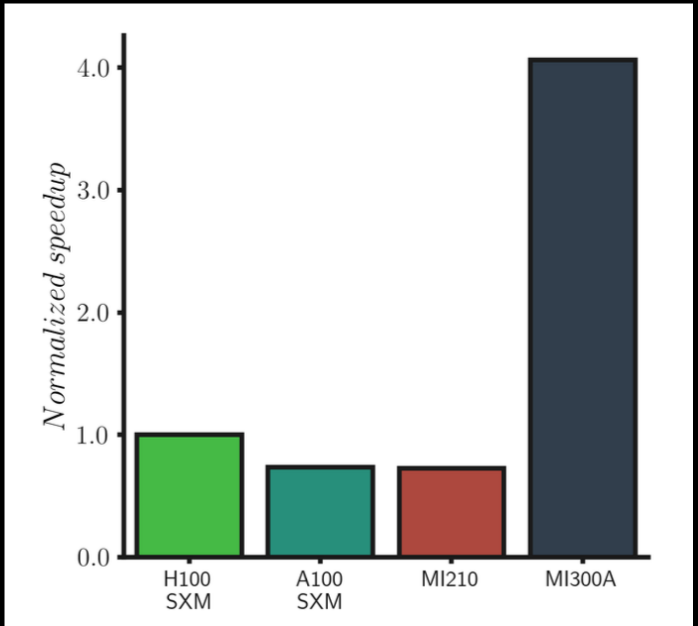
- APU removing bottle necks in OpenFOAM simulations
  - OpenMP to enable offloading larger portions of code
  - APU avoids time spend in data migration

- Tandon et al. ***Porting HPC Applications to AMD Instinct MI300A Using Unified Memory and OpenMP***
  - https://arxiv.org/abs/2405.00436









AMD ROCM DEVELOPER HUB

Engage with AMD ROCm™ Platform Experts

Participate in AMD ROCm Webinar Series

Post questions, view FAQs in Community Forum

Increase Understanding

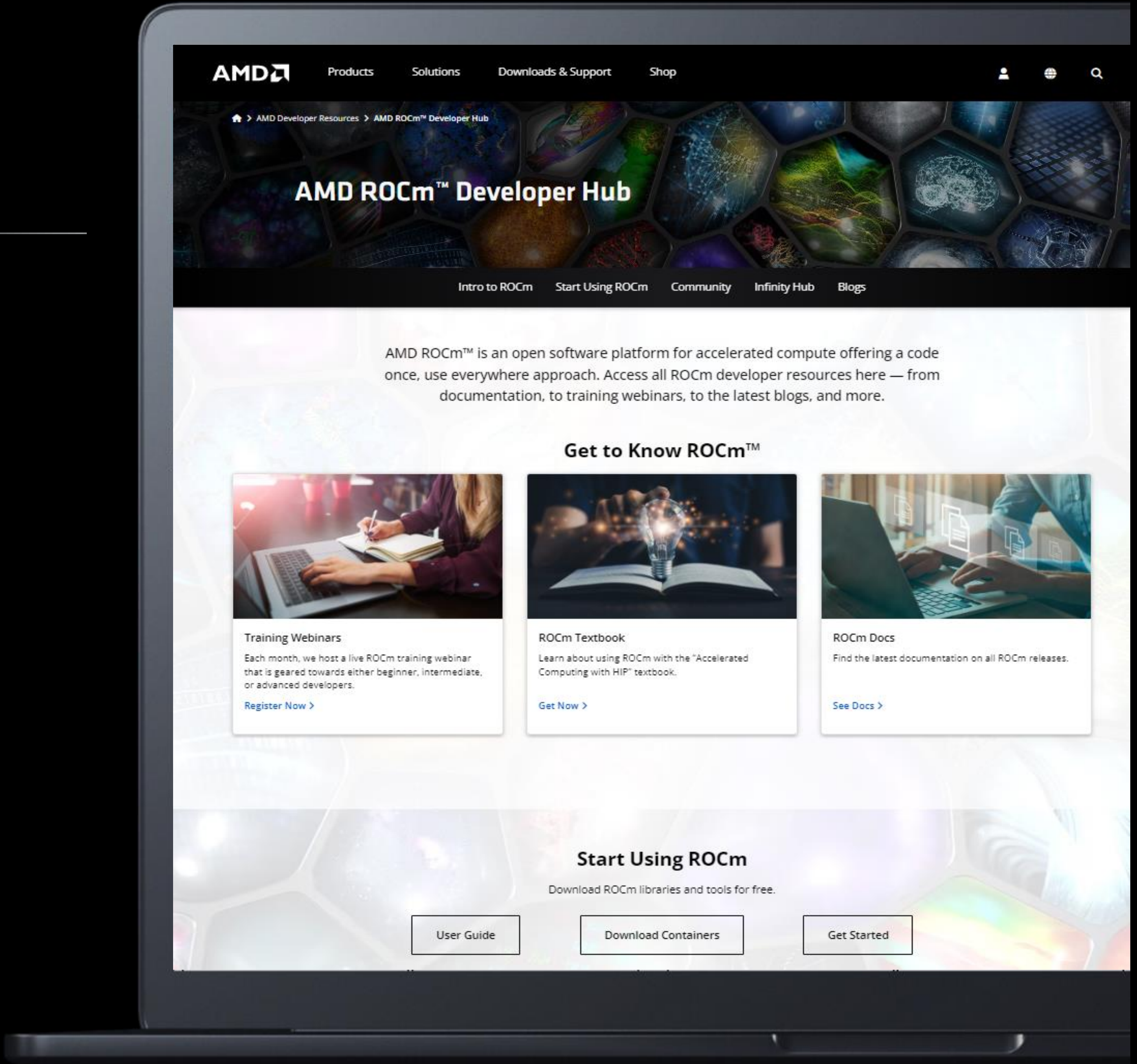
Purchase AMD ROCm Text Book

View the latest news and good practices in the Blogs

Get Started Using AMD ROCm™

AMD ROCm Documentation on GitHub

Download the Latest Version of AMD ROCm



AMD ROCm™ Developer Hub

AMD ROCm™ is an open software platform for accelerated compute offering a code once, use everywhere approach. Access all ROCm developer resources here — from documentation, to training webinars, to the latest blogs, and more.

Get to Know ROCm™

Training Webinars

Each month, we host a live ROCm training webinar that is geared towards either beginner, intermediate, or advanced developers.

Register Now >

ROCm Textbook

Learn about using ROCm with the "Accelerated Computing with HIP" textbook.

Get Now >

ROCm Docs

Find the latest documentation on all ROCm releases.

See Docs >

Start Using ROCm

Download ROCm libraries and tools for free.

AMD
together we advance_

DISCLAIMER AND ATTRIBUTIONS

The information contained herein is for informational purposes only and is subject to change without notice. While every precaution has been taken in the preparation of this document, it may contain technical inaccuracies, omissions and typographical errors, and AMD is under no obligation to update or otherwise correct this information. Advanced Micro Devices, Inc. makes no representations or warranties with respect to the accuracy or completeness of the contents of this document, and assumes no liability of any kind, including the implied warranties of noninfringement, merchantability or fitness for particular purposes, with respect to the operation or use of AMD hardware, software or other products described herein. No license, including implied or arising by estoppel, to any intellectual property rights is granted by this document. Terms and limitations applicable to the purchase or use of AMD products are as set forth in a signed agreement between the parties or in AMD's Standard Terms and Conditions of Sale. GD-18

AMD