

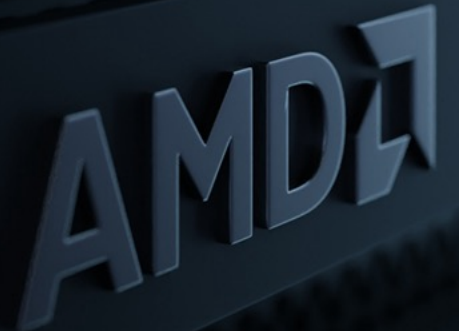


Understanding GPU activity & checking jobs

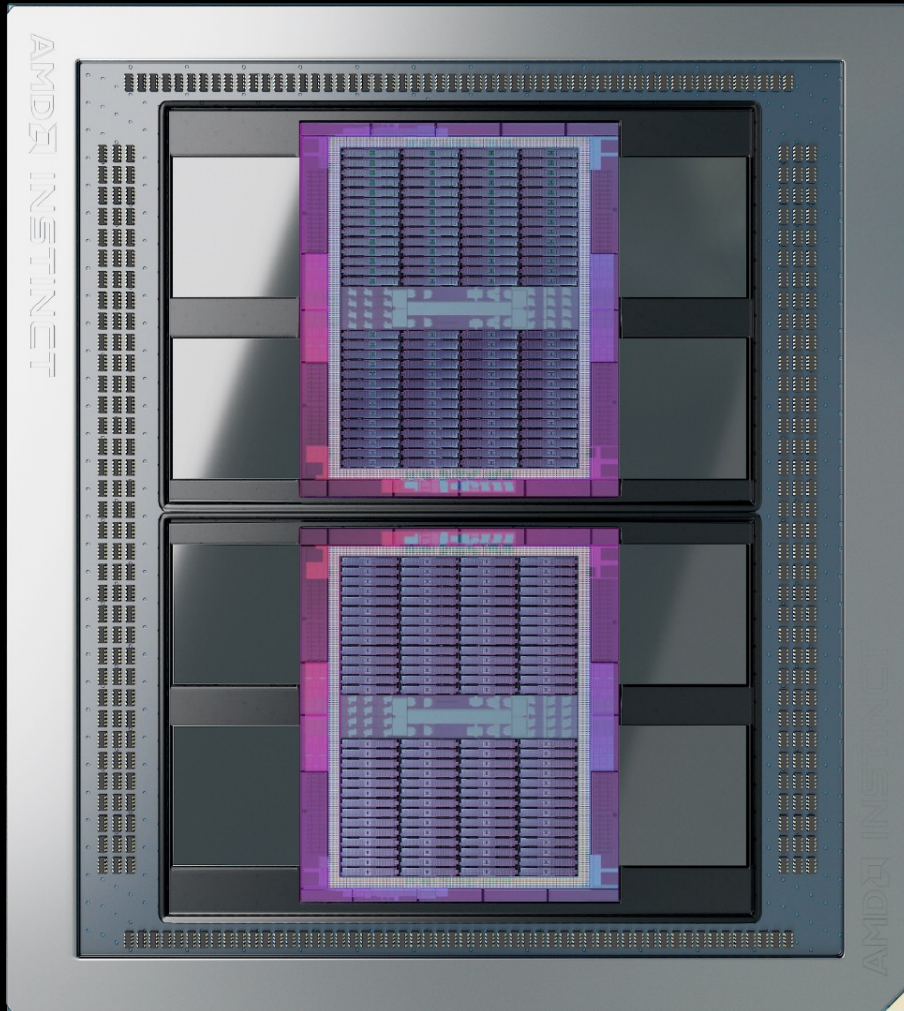
Samuel Antao

LUMI AI Workshop

Copenhagen, Denmark, May. 29-30th, 2024



AMD Instinct™ GPUs



AMD INSTINCT™ MI250X

TWO COMPUTE CHIPLETS – 2 GCDs

58B

Transistors in 6nm

220

Compute Units

880

2nd Gen Matrix Cores

128

GB HBM2E @ 3.2 TB/s

<https://www.amd.com/system/files/documents/amd-cdna2-white-paper.pdf>

AMD Instinct™ GPUs

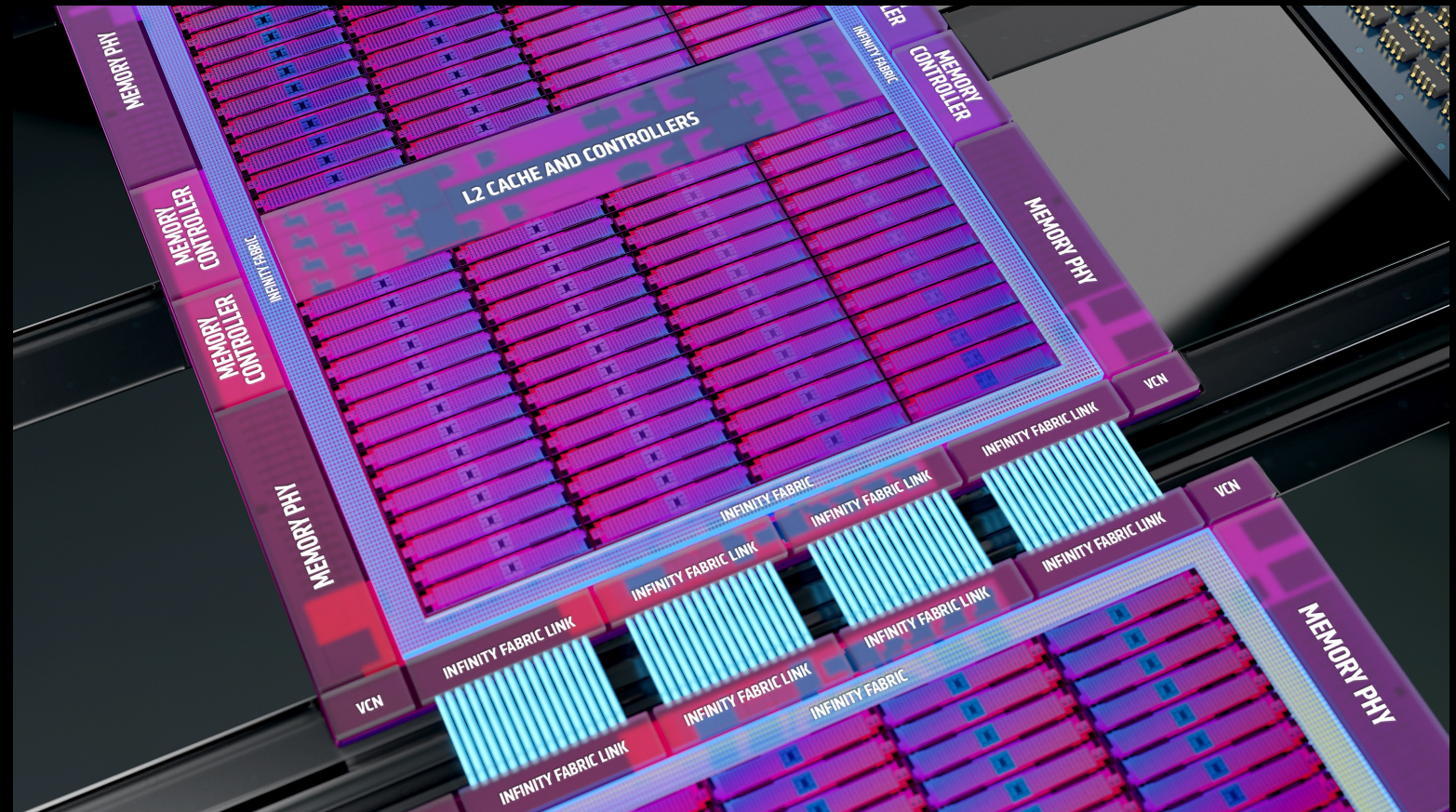
MULTI-CHIP DESIGN

TWO GPU DIES IN PACKAGE TO MAXIMIZE COMPUTE & DATA THROUGHPUT

INFINITY FABRIC FOR
CROSS-DIE
CONNECTIVITY

4 LINKS RUNNING
AT 25GBPS

400GB/S OF BI-
DIRECTIONAL BANDWIDTH



Multiple GCD design has implications on monitoring strategy!

- GPUs have a a given power budget for the two GCDs.
- What is happening in one GCD will limit power in the other.
- Drawn power is the best indicator of GPU activity:
 - A kernel waiting idle for data shows in the driver as 100% GPU utilization
 - Drawn power oscillating around 500W is good indication that compute capabilities in the full GPU are being leveraged
 - For single GCD, 300W should be a good indication.
- rocm-smi is que easiest way to peek at GPU utilization – but not the most accurate!

As reported by the driver – doesn't indicate how well the resource is used.

Average power consumption

```

===== ROCm System Management Interface =====
===== Concise Info =====
GPU   Temp   AvgPwr  SCLK      MCLK      Fan   Perf   PwrCap  VRAM%  GPU%
0     58.0c  324.0W  1650Mhz   1600Mhz   0%   manual 500.0W  98%    100%
1     49.0c  N/A     800Mhz    1600Mhz   0%   manual 0.0W   0%     0%
=====
===== End of ROCm SMI Log =====

```

Frequency will shift to observe GPU power/thermal budget.

Starting a SLURM parallel session

- Starting session in specific nodes to monitor

- For first node of allocation:

```
srun --interactive \  
    --pty \  
    /bin/bash
```

- For other nodes (GPU's won't be visible):

```
srun --pty \  
    --jobid <jobid> \  
    -w <target_node> \  
    --mem=0 \  
    --oversubscribe \  
    --interactive \  
    -n 1 -c 56 \  
    --gpus-per-task=0 \  
    /usr/bin/bash
```

This doesn't seem
to be supported
anymore...

Get your job ID and
allocated nodes
(`squeue -me`)

Start parallel session
(`srun -interactive...`)

Monitor node activity:
`rocm-smi` for GPU
`top` or similar for CPU

Logging from the environment

- HIP runtime and GPU dispatch information can be logged with AMD_LOG_LEVEL=4

```
:3:hip_module.cpp      :662 : 117659918626 us: 8088 : [tid:0x14b2015e9700]
  hipLaunchKernel ( 0x14b5ec183ed0, {32768,1,1}, {512,1,1}, 0x14b2015e71b0, 0, stream:<null> )
...
:3:rocvirtual.cpp      :786 : 117659918634 us: 8088 : [tid:0x14b2015e9700] Arg0: = val:16777216
:3:rocvirtual.cpp      :786 : 117659918636 us: 8088 : [tid:0x14b2015e9700] Arg1: = val:22689590804480
... ShaderName : _ZN2at6native6legacy18elementwise_kernelIli512ELi1EZNS0_15gpu_kernel_implIIZZNS0_23direct_copy_kernel

:3:hip_module.cpp      :663 : 117659918649 us: 8088 : [tid:0x14b2015e9700] hipLaunchKernel: Returned hipSuccess :
```

Number of blocks and threads of the dispatch

Arguments

Kernel mangled name

Return error.

Background – AMD Profilers

ROC-profiler (rocprof)

Hardware Counters

Raw collection of GPU counters and traces

Counter collection with user input files

Counter results printed to a CSV

Traces and timelines

Trace collection support for

CPU copy

HIP API

HSA API

GPU Kernels

Visualisation

Traces visualized with Perfetto

	A	B	C	D	E
1	Name	Calls	TotalDura	AverageN	Percentage
2	hipMemcpyAsync	99	3.22E+10	3.25E+08	44.14872
3	hipEventSynchronize	330	2.42E+10	73394557	33.225
4	hipMemsetAsync	87	7.76E+09	89232696	10.64953
5	hipHostMalloc	9	5.41E+09	6.01E+08	7.415198
6	hipDeviceSynchronize	28	1.32E+09	47006288	1.805515
7	hipHostFree	17	1.05E+09	61534688	1.435014
8	hipMemcpy	41	8.11E+08	19791876	1.112161
9	hipLaunchKernel	1856	58082083	31294	0.079676
10	hipStreamCreate	2	46380834	23190417	0.063625
11	hipMemset	2	18847246	9423623	0.025854
12	hipStreamDestroy	2	15183338	7591669	0.020828
13	hipFree	38	8269713	217624	0.011344
14	hipEventRecord	330	2520035	7636	0.003457
15	hipMalloc	30	1484804	49493	0.002037
16	__hipPopCallConfigura	1856	229159	123	0.000314
17	__hipPushCallConfigur	1856	224177	120	0.000308
18	hipGetLastError	1494	100458	67	0.000138
19	hipEventCreate	330	76675	232	0.000105
20	hipEventDestroy	330	64671	195	8.87E-05
21	hipGetDevicePropertie	47	51808	1102	7.11E-05
22	hipGetDevice	64	11611	181	1.59E-05
23	hipSetDevice	1	401	401	5.50E-07
24	hipGetDeviceCount	1	220	220	3.02E-07

Omnitrace

Trace collection

Comprehensive trace collection

CPU

GPU

Supports

CPU copy

HIP API

HSA API

GPU Kernels

OpenMP®

MPI

Kokkos

p-threads

multi-GPU

Visualisation

Traces visualized with Perfetto



Omniperf

Performance Analysis

Automated collection of hardware counters

Analysis

Visualisation

Supports

Speed of Light

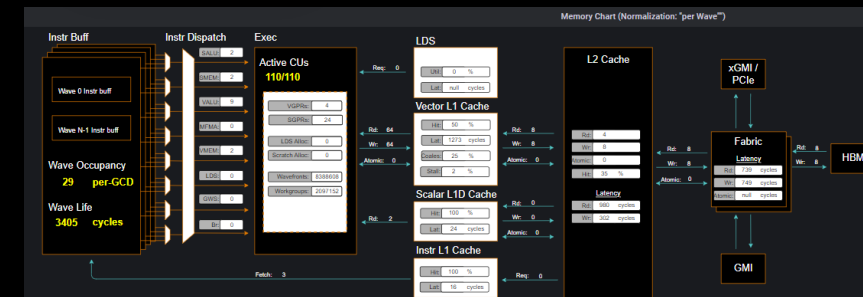
Memory chart

Rooflines

Kernel comparison

Visualisation

With Grafana or standalone GUI



Profiling with Rocprof

- Rocprof profiler client is the easiest way to get started with GPU profiling.
- It is available as part of the ROCm stack and, therefore, available in the containers
- It is seldomly useful to profile every single process/rank of your app:
 - Profiling more than needed = more potential profiling overhead
 - Misleading conclusions



```
pcmd=' '  
if [ $RANK -eq 2 ] ; then  
    pcmd='rocprof --hip-trace'  
fi
```

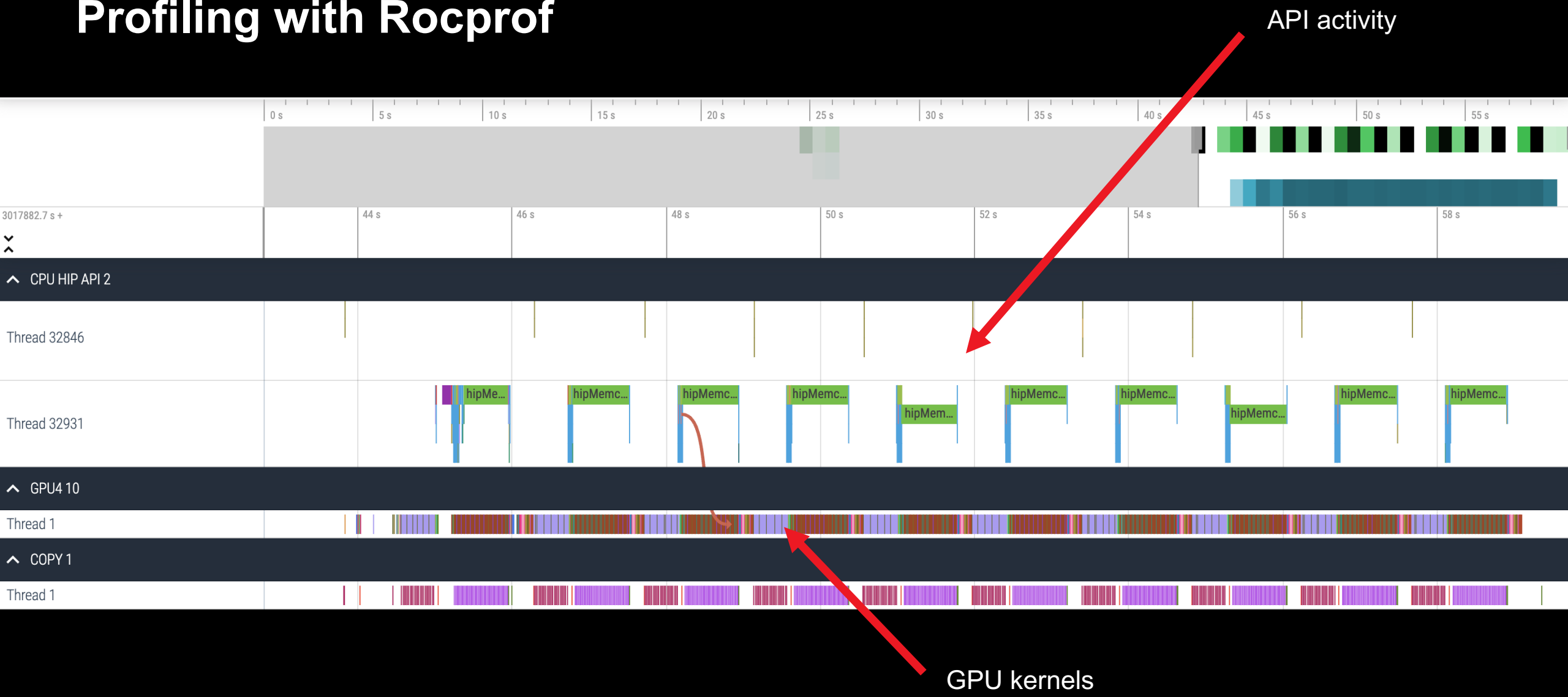
Command to prepend to my application instantiation

We want to profile only for one rank – in this case rank #2

Run command as before except to the prepended profiling command

```
$pcmd python -u myapp.py
```


Profiling with Rocprof



API activity

GPU kernels

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

©2024 Advanced Micro Devices, Inc. All rights reserved. AMD, the AMD Arrow logo, AMD Instinct, EPYC, Infinity Fabric, ROCm, and combinations thereof are trademarks of Advanced Micro Devices, Inc. PCIe is a registered trademark of PCI-SIG Corporation. OpenCL™ is a registered trademark used under license by Khronos. The OpenMP name and the OpenMP logo are registered trademarks of the OpenMP Architecture Review Board. TensorFlow, the TensorFlow logo and any related marks are trademarks of Google Inc. PyTorch, the PyTorch logo and any related marks are trademarks of Facebook, Inc.

Other product names used in this publication are for identification purposes only and may be trademarks of their respective companies.

