



Introduction to OmniTrace

Gina Sitaraman, Suyash Tandon, George Markomanolis,
Jonathan Madsen, Austin Ellis, Bob Robey

AMD Profiling Training, Pawsey
April 26, 2023

AMD 
together we advance_

[Public]

Background – AMD Profilers

ROC-profiler (rocprof)

Omnitrace

Omniperf

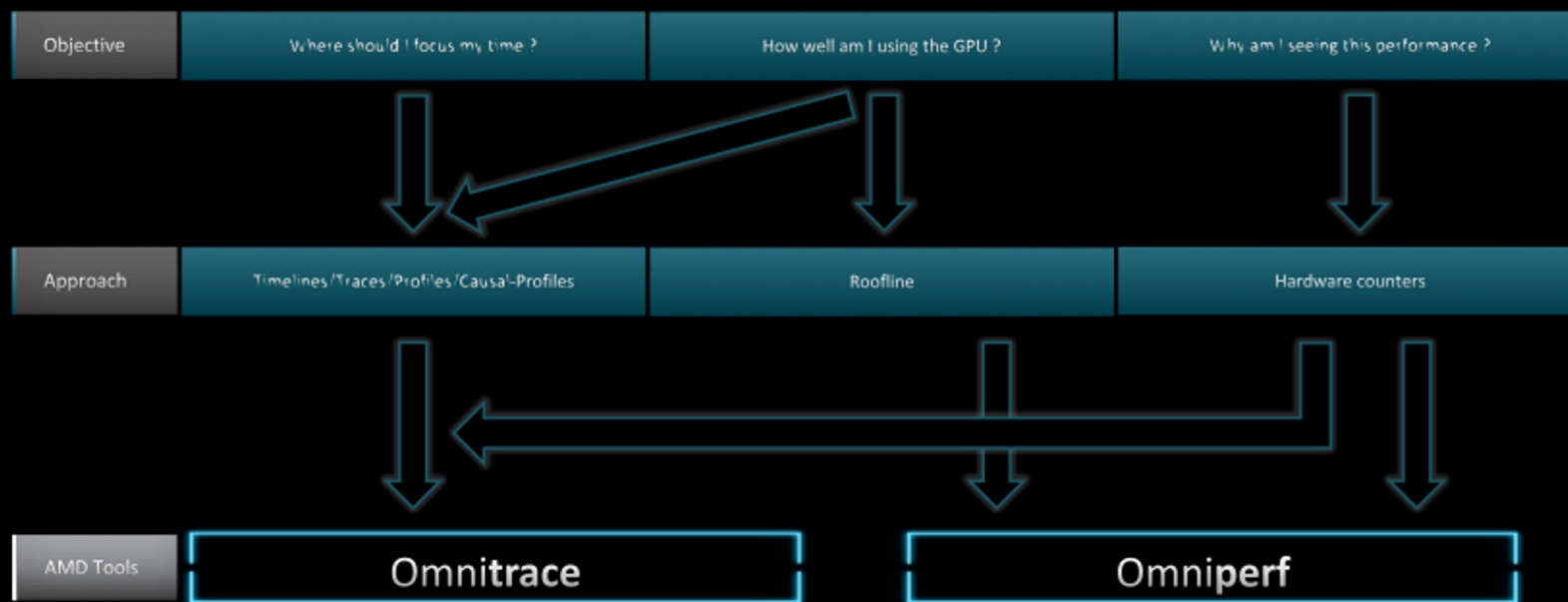
Hardware Counters	Raw collection of GPU counters and traces Counter collection with user input files Counter results printed to a CSV	Trace collection	Comprehensive trace collection CPU GPU		Performance Analysis	Automated collection of hardware counters Analysis Visualisation	
Traces and timelines	Trace collection support for CPU copy HIP API HSA API GPU Kernels	Supports	CPU copy OpenMP*	HIP API MPI Kokkos	HSA API p-threads	GPU Kernels multi-GPU	Supports Speed of Light Memory chart Rooflines Kernel comparison
Visualisation	Traces visualized with Perfetto	Visualisation	Traces visualized with Perfetto		Visualisation	With Grafana or standalone GUI	

Name	Calls	TotalDura	Average%	Percentage
10 hipMemcpyHtoH	99	3.22E+00	3.25E+08	64.148172
11 hipMemcpyHtoD	205	2.42E+00	7.0390517	13.225
12 hipMemcpyHtoA	87	7.79E+09	80232086	15.849513
13 hipMemcpyAtoH	9	5.42E+09	6.01E+08	7.415194
14 hipDeviceSynchronize	28	5.32E+09	47066288	9.801515
15 hipMemset	17	1.0E+09	61534688	4.40004
16 hipMemcpy	40	6.10E+08	18791876	1.113383
17 hipLaunchKernel	1856	58082863	31284	0.679676
18 hipStreamCreate	2	46980834	21290417	0.063625
19 hipEvent	2	38847286	74218123	0.024858
20 hipStreamDestroy	2	11581838	7918689	0.020828
21 hipFree	38	8269713	217626	0.013344
22 hipStreamRelease	330	2520095	7636	0.003437
23 hipEventWait	30	1484884	49493	0.002021
24 hipPushCallConfig	1856	229159	123	0.000114
25 hipStreamWaitEvent	1856	224177	120	0.000098
26 hipStreamWaitEvent	1856	150458	67	0.000138
27 hipStreamCreate	100	76676	232	0.000125
28 hipStreamDestroy	100	64671	195	8.87E-05
29 hipGetDeviceProperties	67	12808	1002	7.11E-05
30 hipGetDevice	64	11611	281	1.59E-05
31 hipGetDevice	5	401	401	5.50E-07
32 hipGetDeviceCount	1	230	230	3.02E-07



[Public]

Background – AMD Profilers



[Public]

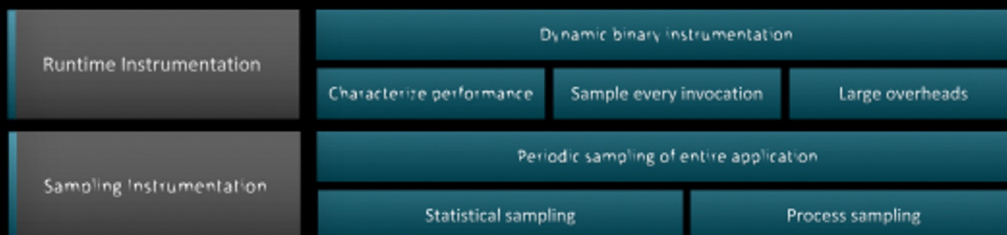
Omnitrace: Application Profiling, Tracing, and Analysis

AMD Research Tool	Repository: https://github.com/AMDRResearch/omnitrace					
	✘ Not part of ROCm stack					
Language Support	C/C++	Fortran	Python	OpenCL™		
Data Collection Modes	Dynamic instrumentation	Statistical/process sampling		Causal Profiling		
Data Analysis	High-level summary	Comprehensive trace		Critical trace analysis		
Parallelism Support	MPI	OpenMP®	Pthreads	HIP	HSA	Kokkos
GPU Metrics	HW counters	HSA API	HIP API	HIP trace	HSA trace	Memory & thermal
CPU Metrics	HW counters	Timing metrics	Memory access	Network	I/O	more...

Refer to [current documentation](#) for recent updates

[Public]

Omnitrace instrumentation Modes



Basic command-line syntax:

```
$ omnitrace [omnitrace-options] -- <CMD> <ARGS>
```

For more information or help use -h/--help/? flags:

```
$ omnitrace -h
```

Can also execute on systems using a job scheduler. For example, with SLURM, an interactive session can be used as:

```
$ srun [options] omnitrace [omnitrace-options] -- <CMD> <ARGS>
```

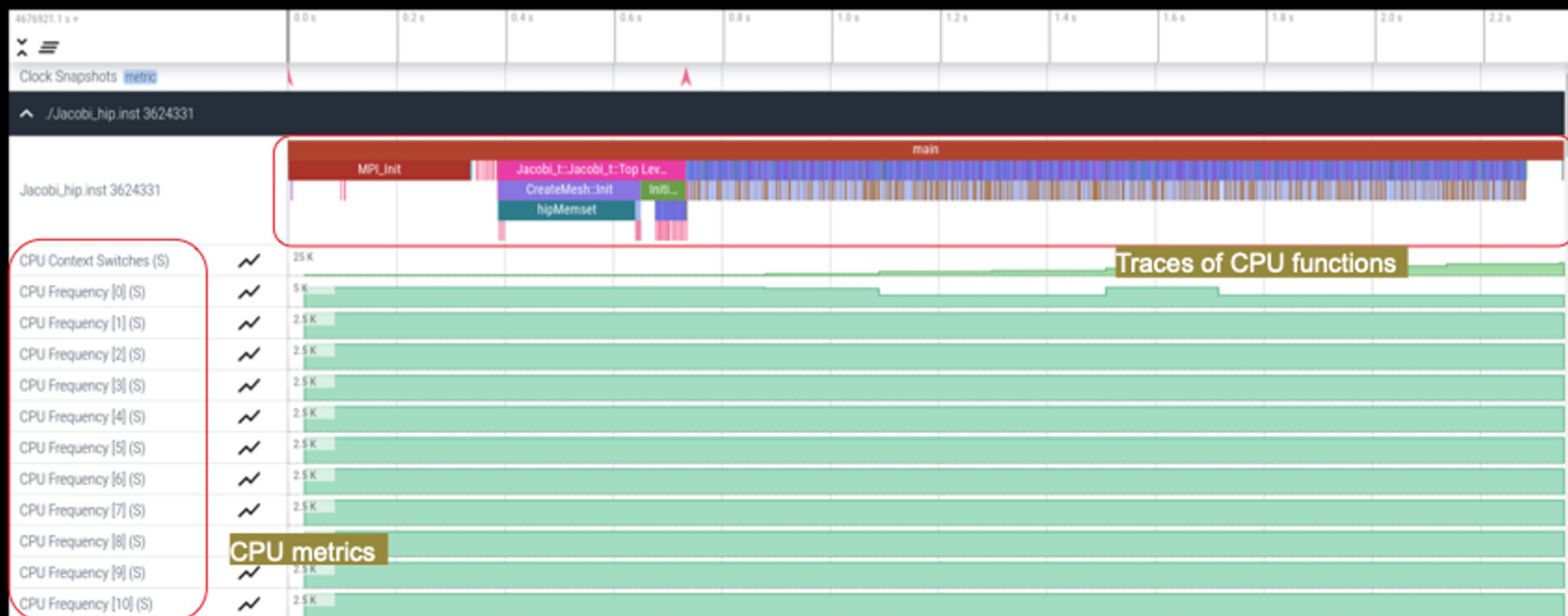
For problems, create an issue here: <https://github.com/AMDResearch/omnitrace/issues>
Documentation: <https://amdresearch.github.io/omnitrace/>

[Public]

Visualizing Trace

Use Perfetto

Copy perfetto-trace-0.proto to your laptop, go to <https://ui.perfetto.dev/>, Click "Open trace file", select perfetto-trace-0.proto



[Public]

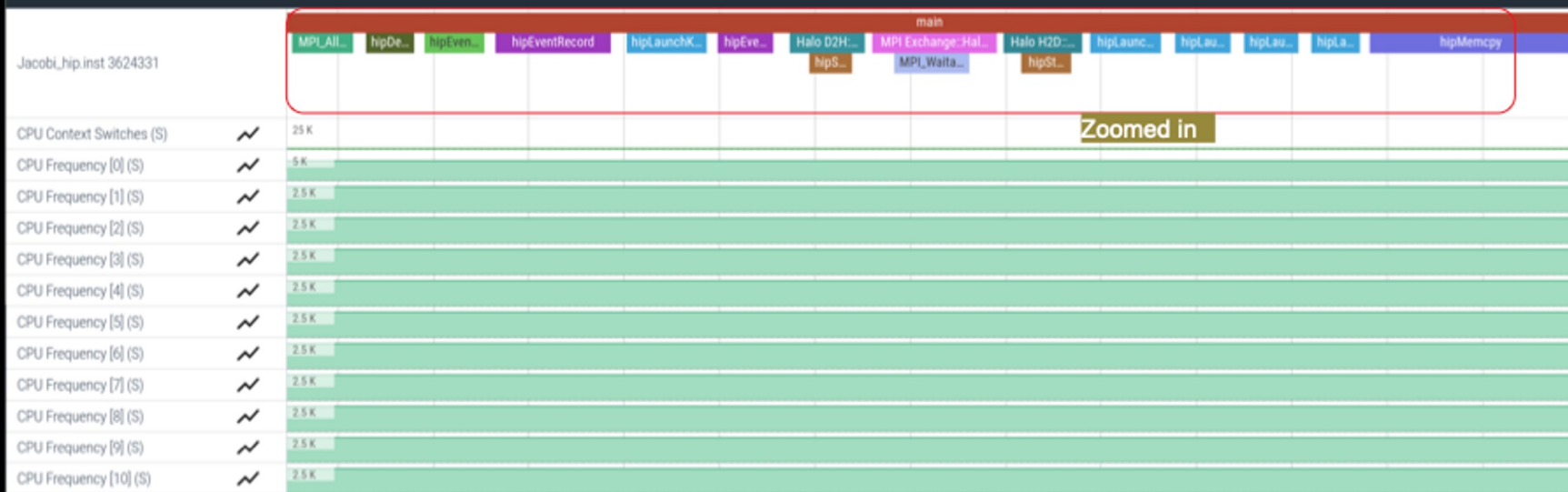
Visualizing Trace

Use Perfetto

Zoom in to investigate regions of interest



^ ./Jacobi_hip.inst 3624331

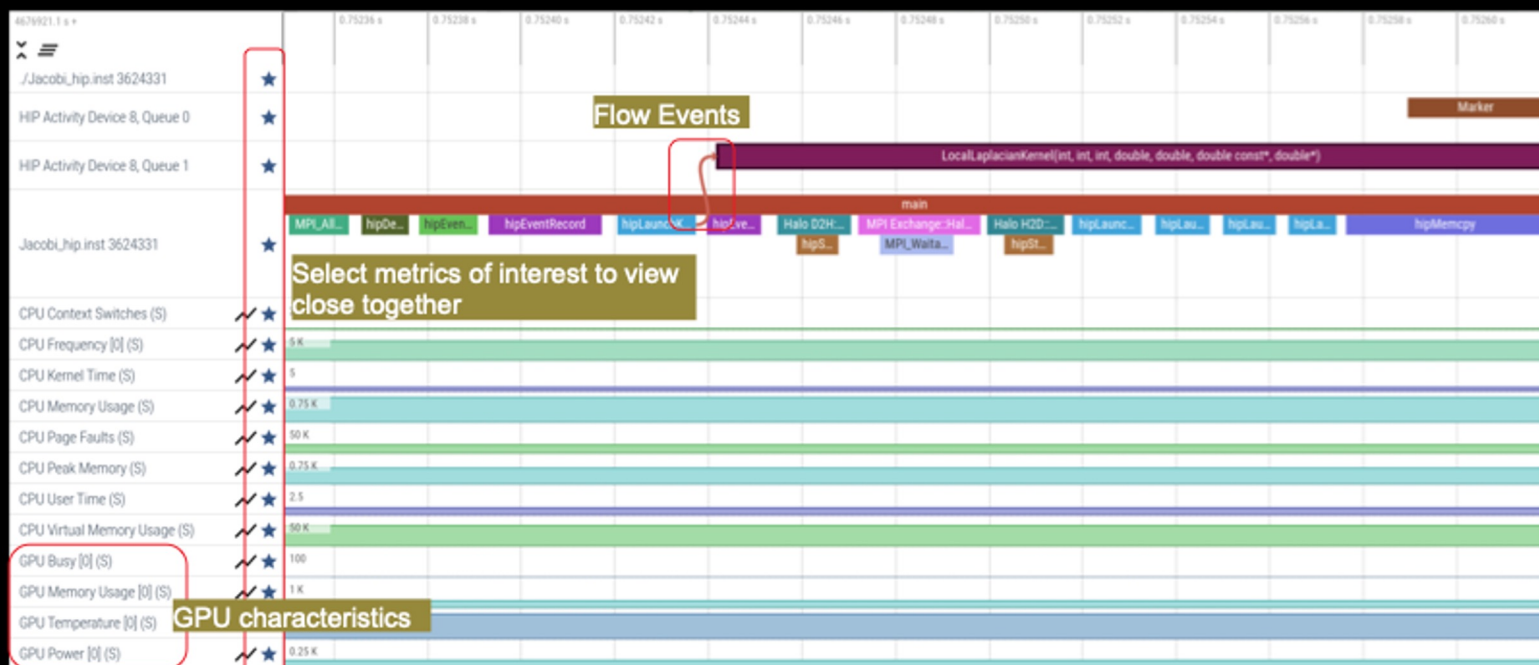


[Public]

Visualizing Trace

Use Perfetto

Zoom in to investigate regions of interest



[Public]

Commonly Used GPU Counters

VALUUtilization	The percentage of ALUs active in a wave. Low VALUUtilization is likely due to high divergence or a poorly sized grid
VALUBusy	The percentage of GPUTime vector ALU instructions are processed. Can be thought of as something like compute utilization
FetchSize	The total kilobytes fetched from global memory
WriteSize	The total kilobytes written to global memory
L2CacheHit	The percentage of fetch, write, atomic, and other instructions that hit the data in L2 cache
MemUnitBusy	The percentage of GPUTime the memory unit is active. The result includes the stall time
MemUnitStalled	The percentage of GPUTime the memory unit is stalled
WriteUnitStalled	The percentage of GPUTime the write unit is stalled

Modify config file

Create a config file in \$HOME:

```
$ omnitrace-avail -G $HOME/.omnitrace.cfg
```

Modify the config file \$HOME/.omnitrace.cfg to add desired metrics and for concerned GPU#ID:

```
...  
OMNITRACE_ROCM_EVENTS = GPUBusy:device=0,  
Wavefronts:device=0, MemUnitBusy:device=0  
...
```

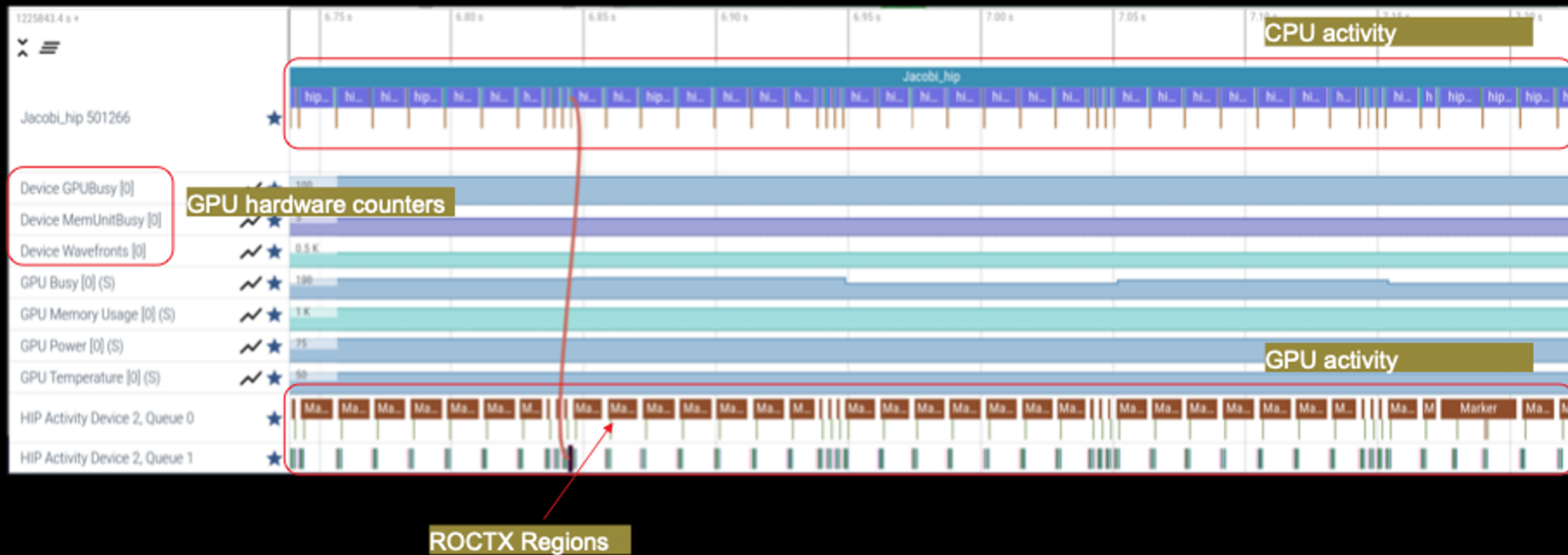
To profile desired metrics for all participating GPUs:

```
...  
OMNITRACE_ROCM_EVENTS = GPUBusy, Wavefronts,  
MemUnitBusy  
...
```

Full list at: <https://github.com/ROCm-Developer-Tools/rocprofiler/blob/amd-master/test/tool/metrics.xml>

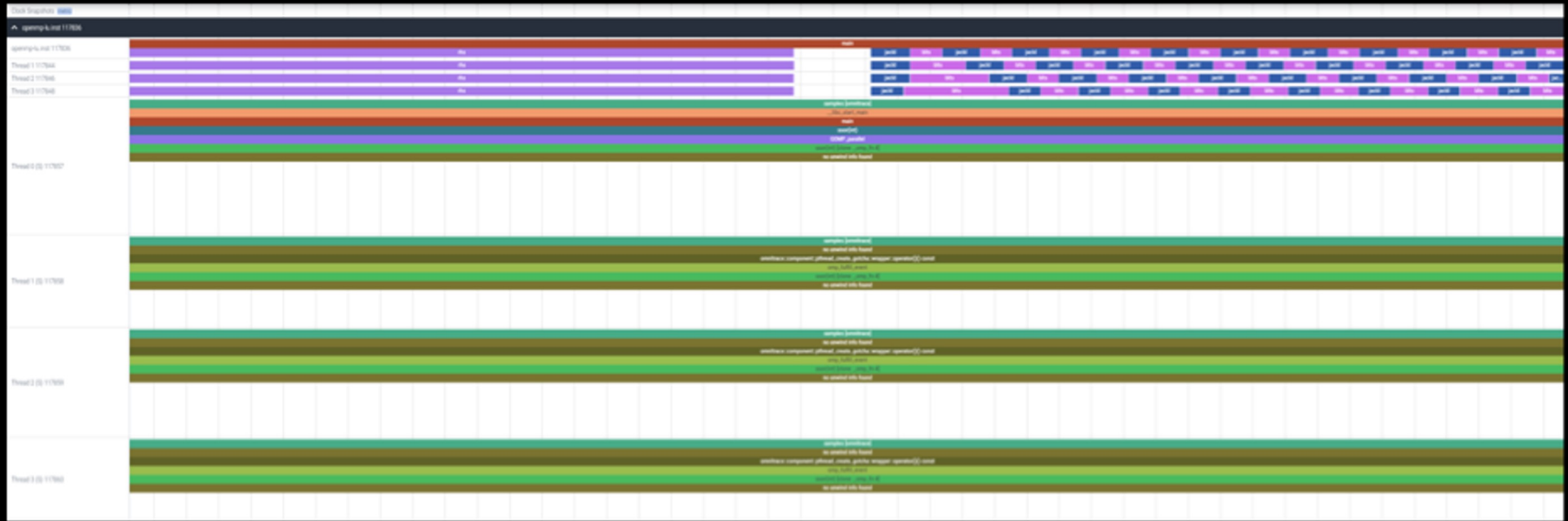
[Public]

Visualization with Hardware Counters



[Public]

OpenMP® Visualization



[Public]

Summary

- **OmniTrace is a powerful tool to understand CPU + GPU activity**
 - Ideal for an initial look at how an application runs
- **Leverages several other tools and combines their data into a comprehensive output file**
 - Some tools used are AMD uProf, rocprof, rocm-smi, roctracer, perf, etc.
- **Easy to visualize traces in Perfetto**
- **Includes several features:**
 - Dynamic Instrumentation either at Runtime or using Binary Rewrite
 - Statistical Sampling for call-stack info
 - Process sampling, monitoring of system metrics during application run
 - Causal Profiling
 - Critical Path Tracing

[Public]

DISCLAIMERS 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's products are as set forth in a signed agreement between the parties or in AMD's Standard Terms and Conditions of Sale. GD-18

THIS INFORMATION IS PROVIDED 'AS IS.' AMD MAKES NO REPRESENTATIONS OR WARRANTIES WITH RESPECT TO THE CONTENTS HEREOF AND ASSUMES NO RESPONSIBILITY FOR ANY INACCURACIES, ERRORS, OR OMISSIONS THAT MAY APPEAR IN THIS INFORMATION. AMD SPECIFICALLY DISCLAIMS ANY IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY, OR FITNESS FOR ANY PARTICULAR PURPOSE. IN NO EVENT WILL AMD BE LIABLE TO ANY PERSON FOR ANY RELIANCE, DIRECT, INDIRECT, SPECIAL, OR OTHER CONSEQUENTIAL DAMAGES ARISING FROM THE USE OF ANY INFORMATION CONTAINED HEREIN, EVEN IF AMD IS EXPRESSLY ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

© 2023 Advanced Micro Devices, Inc. All rights reserved.

AMD, the AMD Arrow logo, Radeon™, Instinct™, EPYC, Infinity Fabric, ROCm™, and combinations thereof are trademarks of Advanced Micro Devices, Inc. Other product names used in this publication are for identification purposes only and may be trademarks of their respective companies.