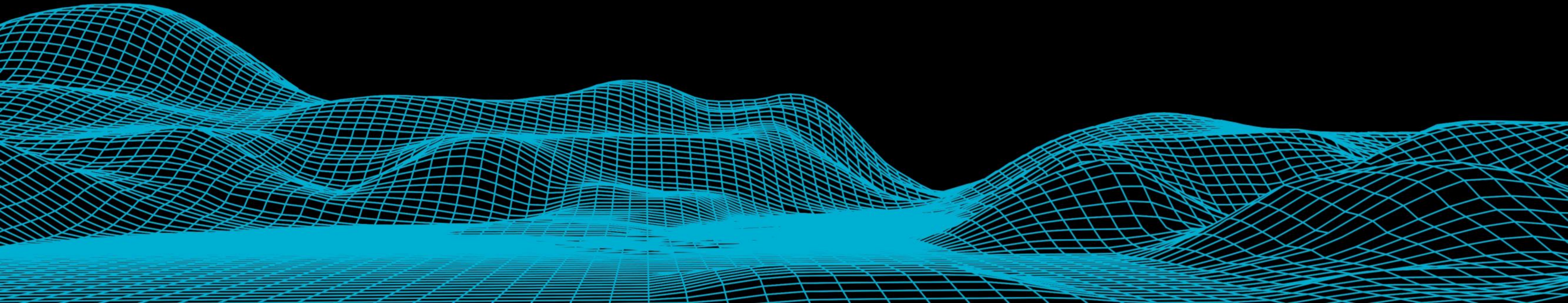
Microsoft Azure

AMD EPYC for HPC

Overview, Strategies, and Best Practices for the HPC Community
Part 1

Summer 2021



Agenda

Overview of AMD EPYC in HPC Space

What's all the fuss?
History of EPYC
HPC community adoption

Buying/Using EPYC for HPC

What problems are you trying to solve?
What does someone new to EPYC need to know?
What to buy and why?
What traps to avoid?

Best Practices

BIOS, Networking, OS
Applications

Evan Burness

Humble employee of ….

Lifelong Blue Devil ….

Grew up in Durham, North Carolina

Duke '05 Grad

5 National Titles, a million amazing memories

Still can't believe we lost to Uconn in '99 and '04 Final Fours

Evan Burness

Principal Program Manager, Azure HPC (2017 – present)

- Lead for Azure H-series (CPUs + RDMA networking)

Director, HPC Solutions, Cycle Computing (2016-2017)

National Center for Supercomputing Applications, Univ Illinois (2009-2016)

What this Talk Is and Is Not

Is Not

Clever Azure marketing ploy
Advertisement for AMD
Anti-Intel rant
PhD-level thesis

Is

Contribution to broader HPC Community from a group that’s deployed a lot of AMD EPYC for HPC/AI
Digestible, pragmatic guidance for those thinking of buying or have bought AMD EPYC
Recommendations/data to help answer common Q’s, save you time, and support HPC workloads
Open invite to ask questions and get my best, most data-driven answers

What’s all the fuss about AMD?

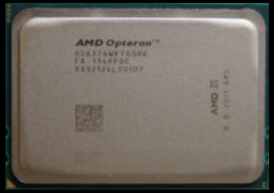
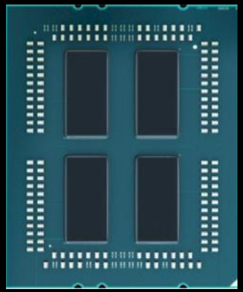
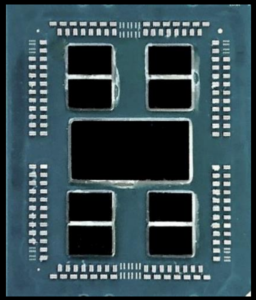
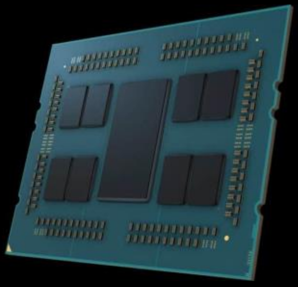
TL;DR - “EPYC” CPU is a credible alternative to Intel in the datacenter for buyers and users of HPC

- Leadership memory bandwidth & IO
- Competitive FLOPS
- x86 compatibility
- Very good power efficiency
- Highly competitive economics

All things we in the HPC world really like!

Quick History of AMD EPYC

50,000 foot view

2012	2017	2019	2021
			
"Abu Dhabi" core uArch	"Zen1" core uArch	"Zen2" core uArch	"Zen3" core uArch
"Piledriver" SoC	"Naples" SoC	"Rome" SoC	"Milan" SoC
Up to 16 cores	Up to 32 cores	Up to 64 cores	Up to 64 cores
~64 GB DRAM B/w	~260 GB DRAM B/W	~340 GB DRAM B/W	~340 GB DRAM B/W
PCIe 2.0	Up to 4MB L3/core	Up to 16MB L3/core	Up to 32 MB L3/core
	PCIe 3.0	PCIe 4.0	PCIe 4.0

“Chiplets” 101

TL;DR

Chiplets help to increase fab yields and schedule, lower cost, improve socket level performance and power efficiency

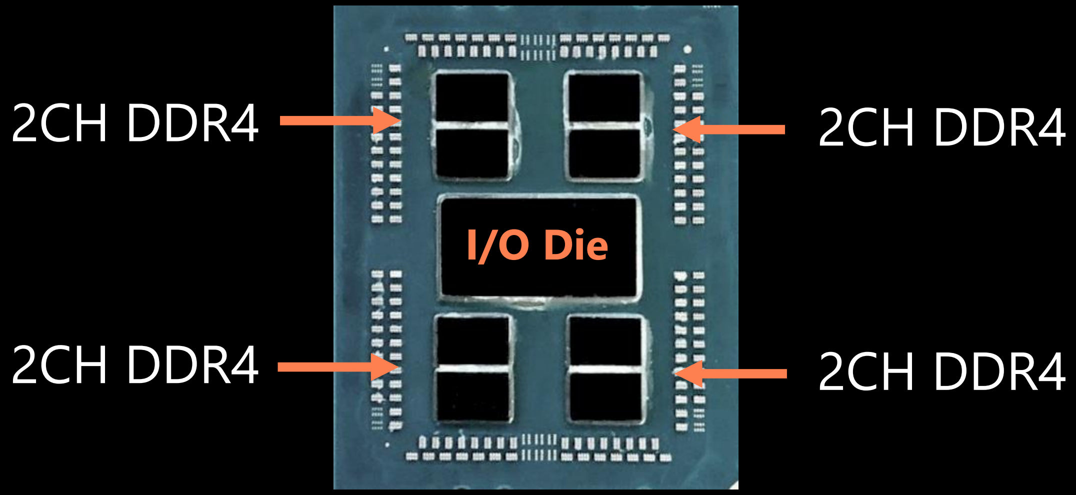
Pros

All of the above are good!

Tradeoffs

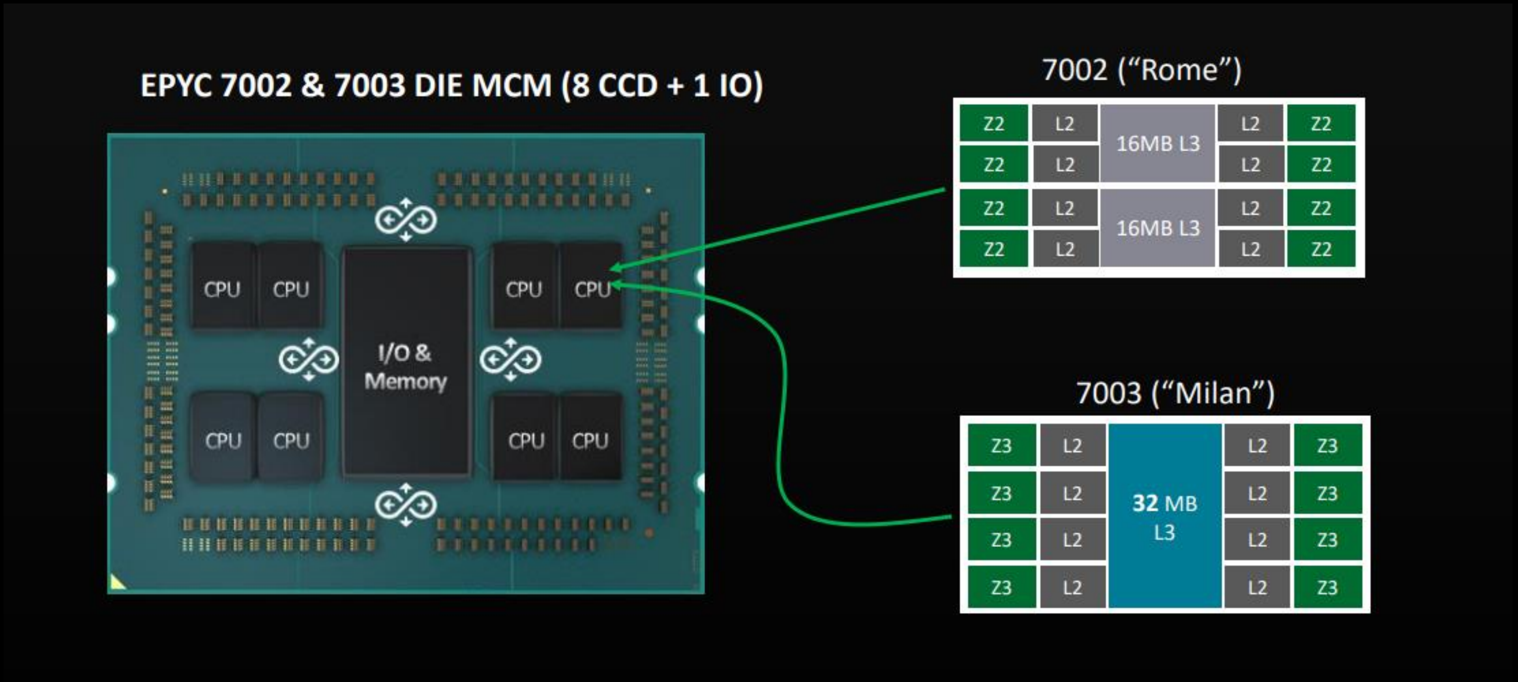
Users and developers need to think of EPYC CPUs as almost “clusters on a chip” and have awareness as to how best to overlay software on top of this kind of hardware

EPYC “Rome” Die Shot



E.g. Above is more “4 * 2 CH memory” rather than “8 channel”

Milan v. Rome



7002 ("Rome")

Z2	L2	16MB L3	L2	Z2
Z2	L2		L2	Z2
Z2	L2	16MB L3	L2	Z2
Z2	L2		L2	Z2

7003 ("Milan")

Z3	L2	32 MB L3	L2	Z3
Z3	L2		L2	Z3
Z3	L2		L2	Z3
Z3	L2		L2	Z3

Similarities

Same core counts

Same 280w TDP max

Same PCIe 4.0 support

Same 8ch DDR4 3200 (2/quadrant)

Same 16 GT/s xGMI

Differences

2x addressable cache/core

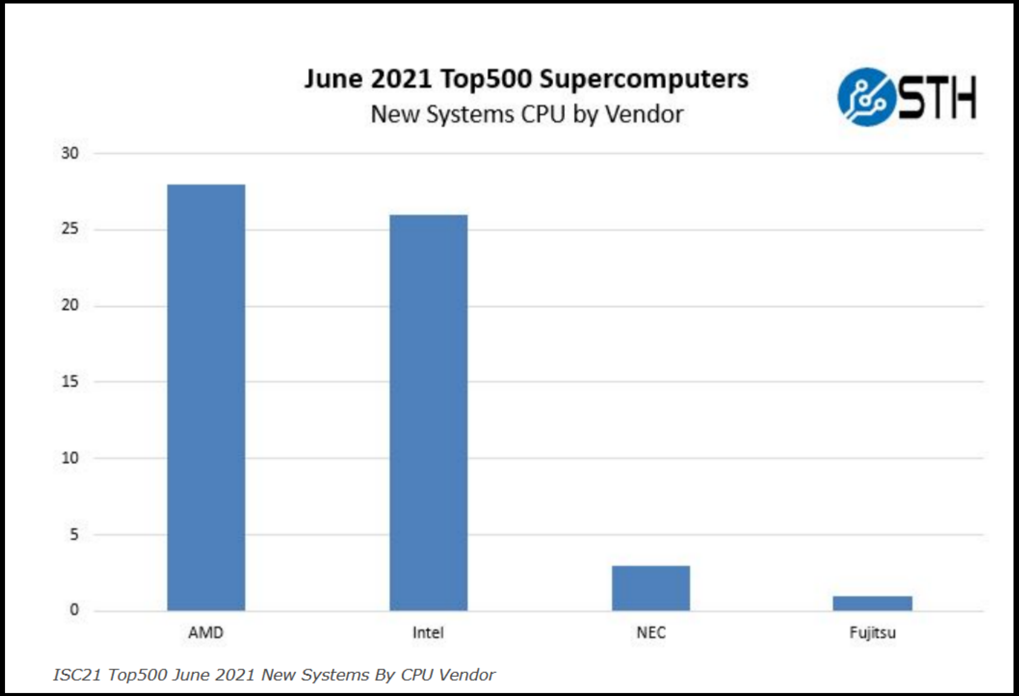
19% higher IPC from Zen3 v. Zen2

Higher frequencies

Better memory latencies

Trends in HPC Usage of AMD

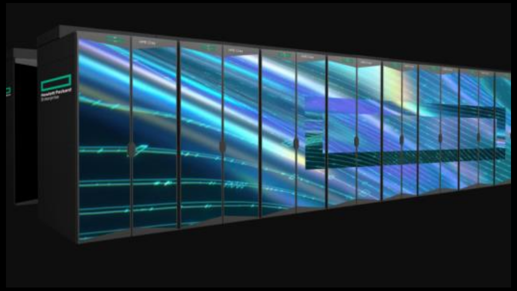
Top500 – New Systems



https://www.servethehome.com/top500-june-2021-our-new-systems-analysis-amd-wins-big/

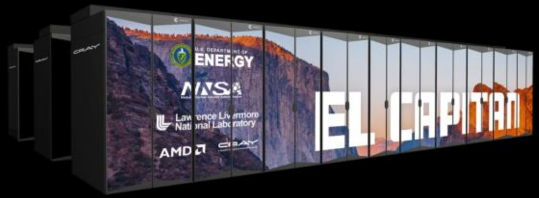
Pre-Exascale and Exascale

~89PF Peak



~552PF Peak

>1.5 EF Peak



> 2 EF Peak

Azure Using AMD for HPC & AI

HB-series VMs

CPU-based HPC

HBv1

EPYC Gen1 “Naples”
100 Gb EDR
Q2 2019

HBv2

EPYC Gen2 “Rome”
200 Gb HDR
Q1 2020

HBv3

EPYC Gen3 “Milan”
200 Gb HDR
Q1 2021

ND A100 v4-series VMs

GPU-based HPC/AI

NDv4

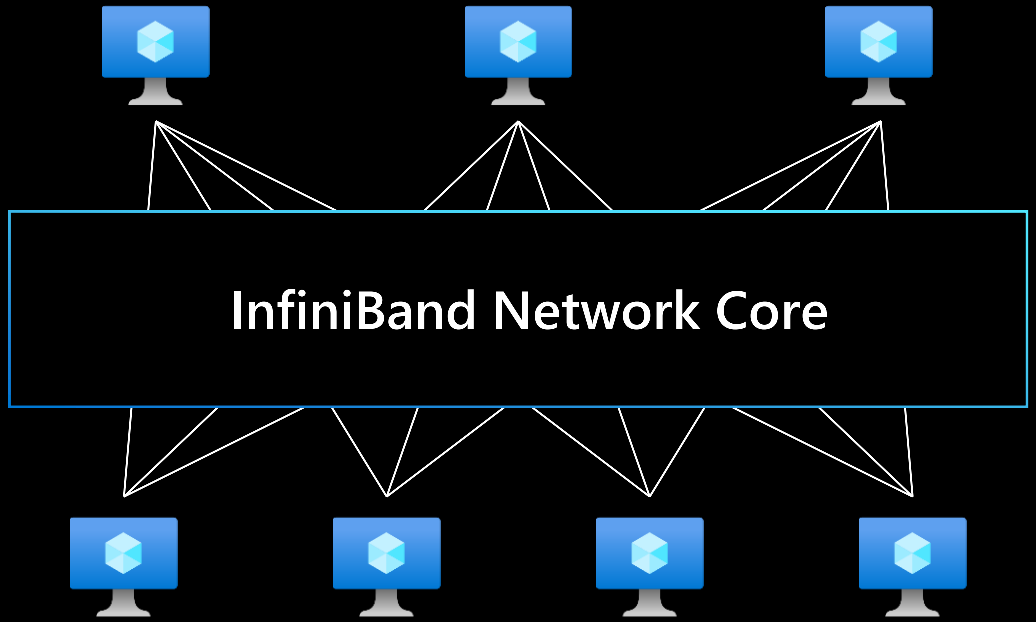
EPYC Gen2 “Rome”
8 * Nvidia A100 NVLINK 40 GB
8 * 200 Gb GDR
Q3 2021

HBv1 docs: ***https://bit.ly/3CQblox***
HBv2 docs: ***https://bit.ly/3iT23Wi***
HBv3 docs: ***https://bit.ly/37REkQ5***
NDv4 docs: ***https://bit.ly/3xSd1zE***

Non-blocking Fat Tree topology

Hardware offload of MPI collectives

Supports all MPI



InfiniBand Network Core

1.3 microsecond latencies

Dynamic Connected Transport

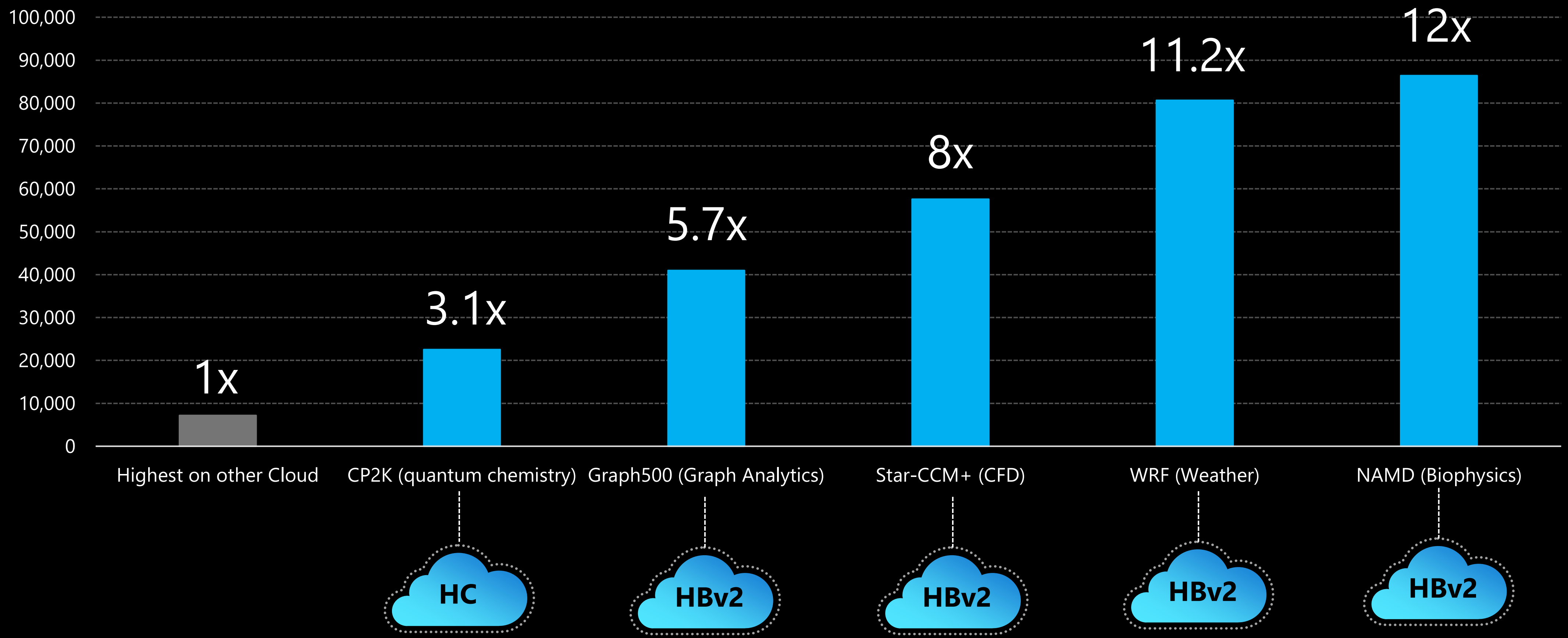
Up to 200 Gb HDR

Bare-metal passthrough

Intelligent Adaptive Routing

HPC Application Scaling on Azure

Max Demonstrated # parallel processes for MPI job



EPYC’s Relevance to You?

“It Depends” (of course)

First Step – What are the most important problems you are trying to solve for? How do you stack rank?

- What is the relevant level(s) of scale?
- Pure performance ? Performance/€ ? Cost/Performance?
- Simplest possible HPC evolution for my users?
- Supported platform by ISVs and/or required SW toolchains?
- Platform for accelerators?
- Lowest possible cost?
- Something else?

Frequent answer from Azure HPC customers: “best performance and cost/performance for my main workloads, with as minimal user education as possible”

Maybe Buying or Bought EPYC ?

What You Should Know

EPYC performance can be extremely good for a CPU

- Typical Haswell/Broadwell → Rome/Milan will seem like enormous leap for most workloads
- How good depends on what your workload scales with (memory bandwidth? L3? Compute? Frequency?)

Realize/explain performance or cost per job is what matters

- Infrastructure doesn't scale by "cores", you buy/rent servers (nodes)
- Clock frequency ≠ performance (don't just "MOAR GIGAHURTZ!!")
- Perf scales per server (or VM), or by N* scalable network endpoints (MPI)
- Doesn't matter if you used all the cores (do you do this for RAM? Cache? CUDA cores in a GPU? RDMA B/W?)
- Exception scenario: using expensive SW licensed per core

Affinitize processes explicitly and with understanding of hardware topology

- Generally advisable to evenly distributed processes by physical L3 boundaries (4 cores/Rome, 8 cores/Milan)
- Don't just throw N processes at the server and assume app/OS will automagically figure out placement for you

"Per Core" Performance Is Never Fixed

Per core performance depends heavily on how X number of cores in a node sub-divide global shared assets that have significant impact on performance

- DRAM bandwidth
- L3 cache capacity and bandwidth
- On-die and inter-socket bandwidth ("GMI" and "xGMI")
- Power and thermal headroom to increase clock frequencies
- For MPI workloads, network bandwidth/latency

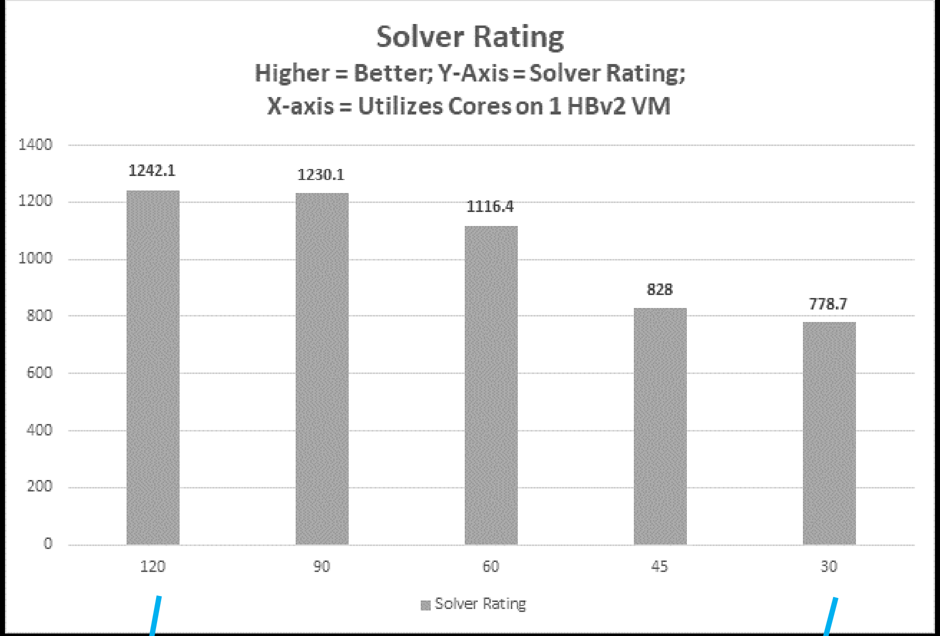
In right circle, the cores appear to be ~2.5x faster than the cores on the right. Are they?

No, they're exact same cores in exact same server, just getting different allocations of global shared assets

ANSYS Fluent 2019.5

Aircraft 14m cell case, 1* Azure HBv2 VM

Scaling from 1-4 processes per NUMA



100% of best performance possible, but 4x the cores and licenses used. Still just 1 node of infrastructure.

63% of best performance possible, but ¼ of the cores per node and licenses used

“Per Core” Performance Is Never Fixed

Even for compute bound apps, per core performance depends on whether and to what degree global shared assets are being exhausted

Same phenomenon will generally occur on other CPUs, too (e.g. Intel Xeon)

Lesson – Target a EPYC CPU model with a core count that returns commensurate value for increase in cost

Linpack – Compute (TFLOPS)

Cores per Milan CCD	1 per CCD	2 per CCD	4 per CCD	6 per CCD
Benchmark	**16C**	**32C**	**64C**	**96C**
HBv3 HPL	0.76	1.40	2.23	2.86
Bare metal HPL	0.76032	1.4257152	2.25344	2.87232
Expected HPL Efficiency	90%	90%	87%	85%
VM as a % of Metal	1x	0.98x	0.99x	1x

Note decline in expected and delivered HPL efficiency; this is due to gradually running out of data fabric (GMI) bandwidth

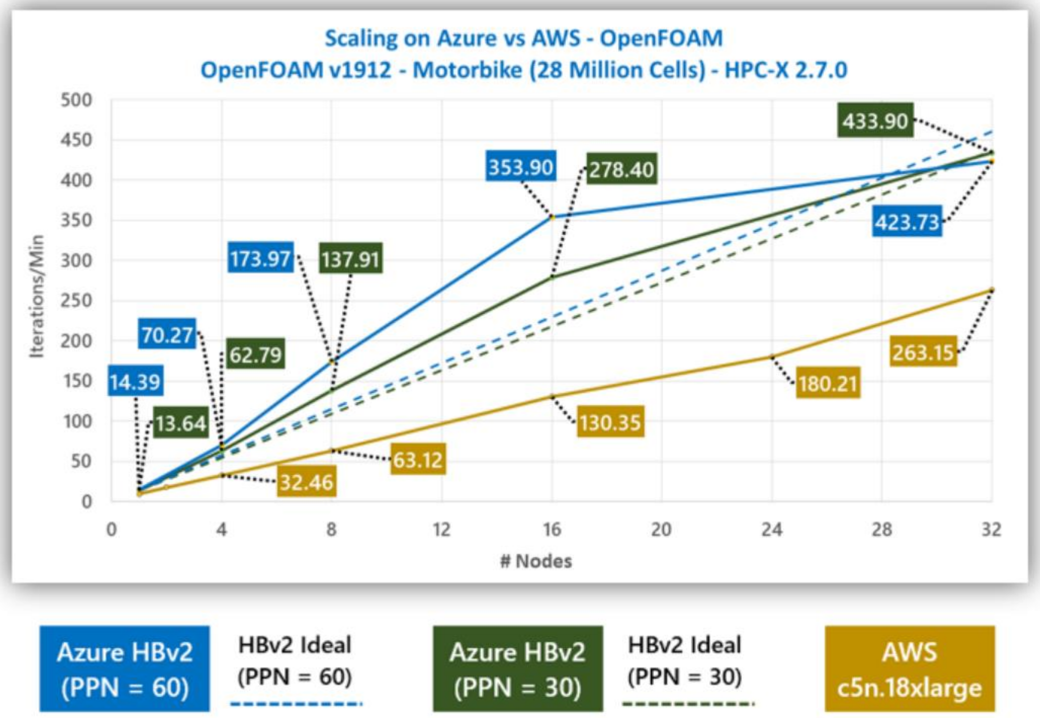
EPYC v. Xeon – Per Core Perf

TL;DR – EPYC packs in so much memory bandwidth , L3 cache, data fabric perf, etc. that for many HPC apps, even at ISO core counts, it will often outperform Xeon

Disclaimer - not showing this for “Azure v. AWS” purposes” (Azure Skylake in HC-series would look similar to AWS’ Skylake in this case)

Nor using Skylake as representative of all Intel Xeon (e.g. IceLake would do better than Skylake here)

Nor saying OpenFOAM is indicative of every HPC workload



(AWS Source: AWS-Auto-May2020-CFDMotorsport.pdf (awscloud.com), p. 21)

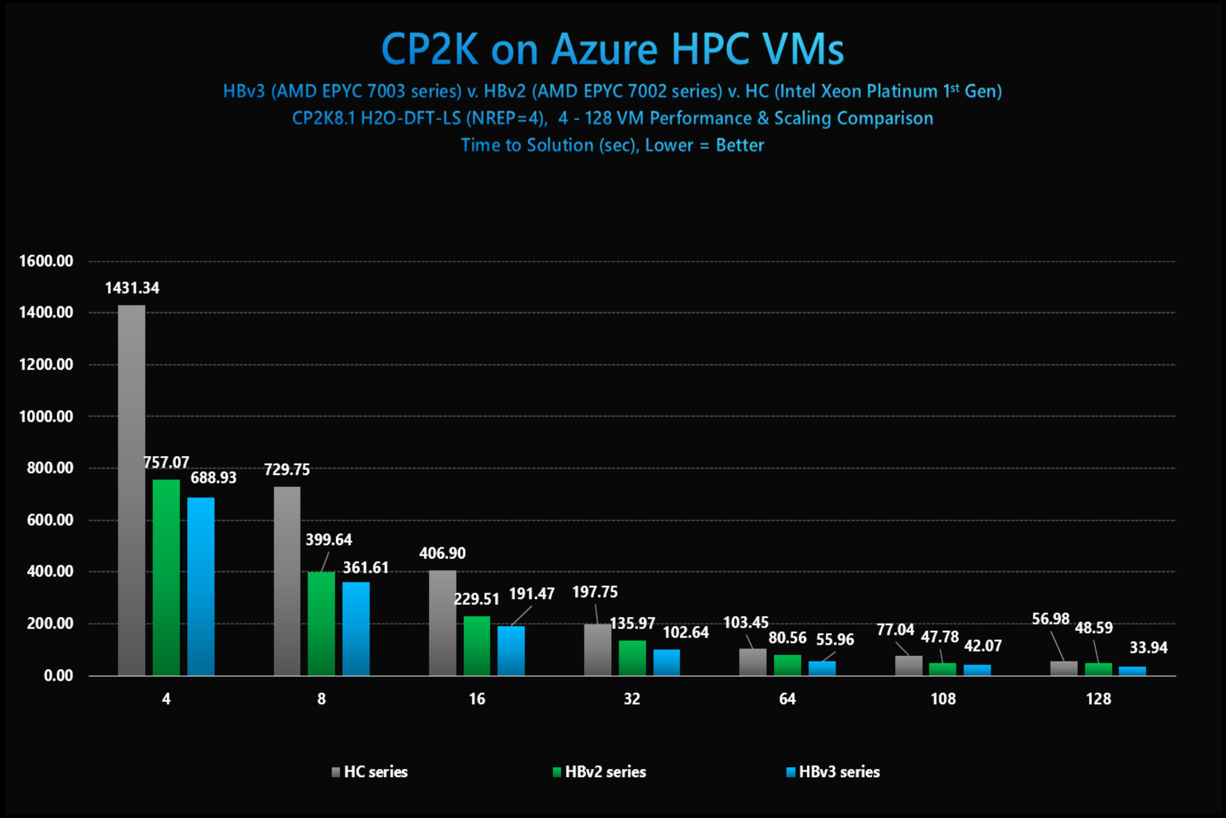
Figure 3: Azure HBv2 v. AWS C5n.18xlarge

Optimizing OpenFOAM Performance and Cost on Azure HBv2 VMs - https://bit.ly/3xUbOYo

Why Scale Really Matters

Know Your Target(s)

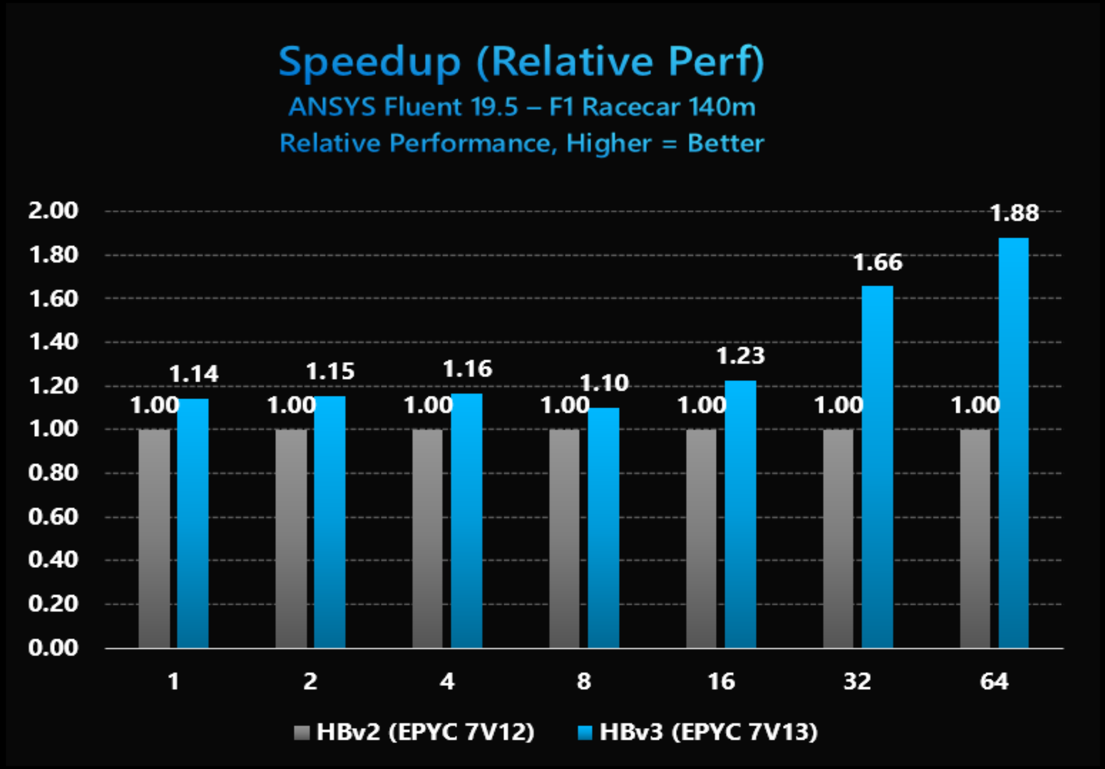
Big differences in 1 node perf can reduce at scale….



CP2K on Azure HPC VMs

HBv3 (AMD EPYC 7003 series) v. HBv2 (AMD EPYC 7002 series) v. HC (Intel Xeon Platinum 1st Gen)
CP2K8.1 H2O-DFT-LS (NREP=4), 4 - 128 VM Performance & Scaling Comparison
Time to Solution (sec), Lower = Better

…and small differences in 1 node per can increase at scale



Speedup (Relative Perf)

ANSYS Fluent 19.5 – F1 Racecar 140m
Relative Performance, Higher = Better

Both scenarios can change calculus of which CPU platform to invest in, and how to configure those platforms

What About AVX512 Support?

TL;DR - Not *likely* a big deal

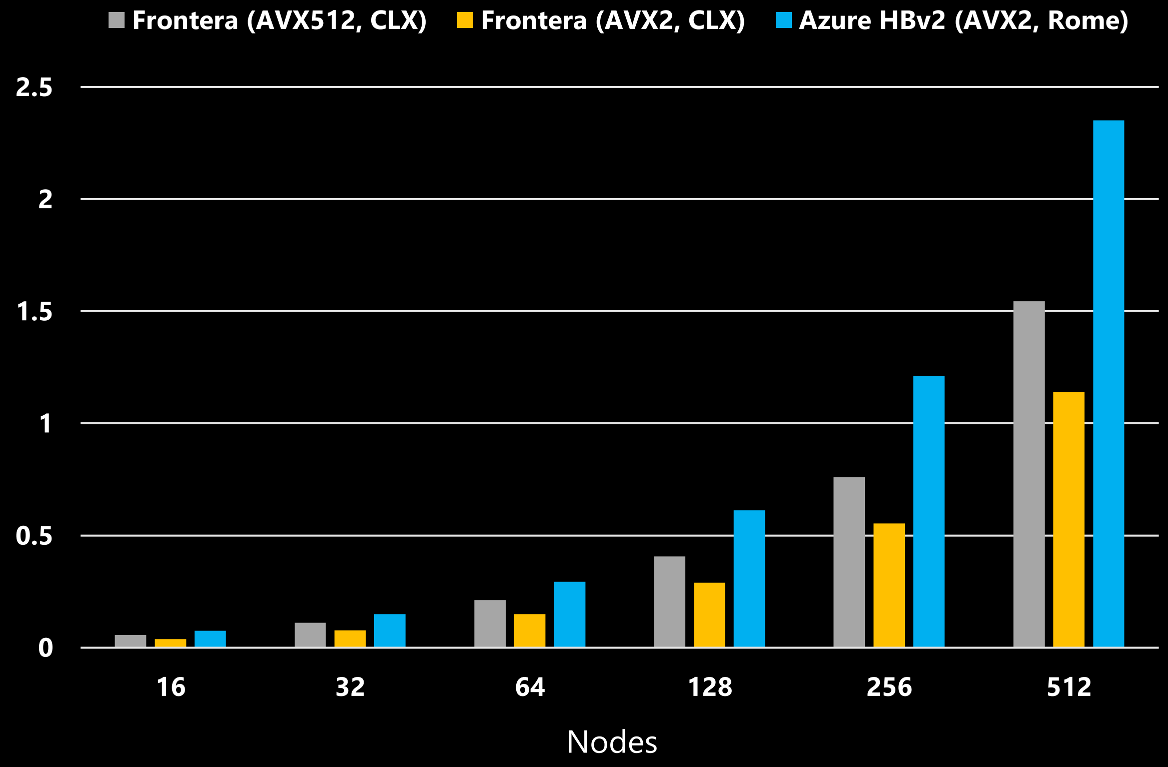
- Few HPC apps support AVX512 as is, even fewer are heavily optimized for it
- Anything that supports AVX512 also likely has AVX2 binary (e.g. GROMACS)
- For those that are, EPYC core count advantage makes up the difference (and with no need for AVX512 support)
  - Scenario 1: (2 CPUs/server) * (28 cores/Cascade Lake 8280) * (32 ops/cycle) * (~1.9 GHz SIMD bound freq) = ~3.4 teraFLOPS FP64 (peak)
  - Scenario 2: (2 CPUs/server) * (64 cores/Rome 7742) * (16 ops/cycle) * (~2.2 GHz SIMD bound freq) = ~4.5 teraFLOPS FP64 (peak)

Exception: You have AVX512 app *AND* it's licensed per core *AND* SW costs dominate TCO *AND* problem is not communications bound

Big picture - If your app is so purely compute bound, you probably want a GPU anyway

Azure v. a TOP10 Supercomputer

NAMD, nanoseconds/day, higher = better



What About MKL Support?

TL;DR - Very much a “it depends”

In general, MKL will run just fine on EPYC

If access to source, AMD libraries are optimized and well supported for EPYC

https://developer.amd.com/amd-aocl/

Backup option for MKL (prior to 2020) is to use Debug Mode Type=5 (not necessarily recommended, though)

But some apps will take hard dependency on MKL and as a result deliver better perf, perf/unit of cost, and cost/performance on Intel Xeon

Intel MKL and AMD BLIS on EPYC 7742 (Rome)

Library	**MKL (Debug Mode enabled)**	**MKL (Debug Mode disabled)**	**BLIS**
Single-core DGEMM	51.36 GigaFLOPS	47.684 GigaFLOPS	50.65 GigaFLOPS
Multi-core DGEMM	3239 GigaFLOPS	1778 GigaFLOPS	4020 GigaFLOPS

BIOS Settings to Know

L3 as NUMA → Defines NUMA boundary

- Enabled = 1 NUMA for every L3 slice
- Disabled = # of NUMA will be how you define NPS (**recommended**)

Nodes per Socket – Determines how Interleaving is done

- NPS1 = simplest presentation
- NPS2 = 2-way interleaving/socket (**recommended**)
- NPS4 = 4-way interleaving/socket
  - NPS4 not an option on 6 CCD EPYC

Determinism Mode

- Performance = bring every CPU in cluster to lowest common denominator of silicon yield
- Power = let motherboard drive CPU to best frequencies based on frequency/power curve of given CPU (**recommended**)

C-States

- Enabled – best "Fmax" (**recommended**)
- Disabled – limited "Fmax"

BIOS Settings to Know

Preferred IO

- key PCIe device (e.g. InfiniBand NIC (**recommended**)

LCLK for Key PCIe device

- Set to 593 (improves NIC latency)

Simultaneous Multi-threading (SMT)

- Enabled = 2 threads/core
- Disabled = 1 thread/core

cTDP

- Configurable power range

Package Power Limit (PPL)

- Hard governor of socket power limit
- Depends on your DC power limit and how you are assessed OPEX costs

Additional Resources

High Performance Computing (HPC) Tuning Guide for AMD EPYC™ 7003 Series Processors - https://bit.ly/3k0oiZL

High Performance Computing (HPC) Tuning Guide for AMD EPYC™ 7002 Series Processors - https://bit.ly/3xRJzd1

HPC Performance and Scalability Results with Azure HBv2 VMs - https://bit.ly/2XD7Ebj

HPC Performance and Scalability Results with Azure HBv3 VMs - https://bit.ly/37PyCOM

AMD Presentation to NASA – “Why AMD for HPC” - https://go.nasa.gov/3CVOMEz

AMD Optimizing CPU Libraries (AOCL) - https://bit.ly/3m9afnf

Optimizing OpenFOAM Performance and Cost on Azure HBv2 VMs - https://bit.ly/3xUbOYo

Microsoft Azure

Thank you!

Feedback

Q & A