Hardware Overview

Andreas Götz

Material by
Robert Sinkovits,
Mary Thomas,
Mahidhar Tatineni

SDSC Summer Institute
August 7, 2023

SDSC SAN DIEGO SUPERCOMPUTER CENTER

at the UNIVERSITY OF CALIFORNIA; SAN DIEGO

UCSD

Introduction

- Most of you are here because you are computational scientists
  - Have a specific scientific problem you are trying to solve
  - Support researchers from a variety of domains
- The actual hardware is probably of secondary interest
  - Hardware is interesting but not *that* interesting
- Nonetheless, it is still helpful to know a bit about hardware
- In this talk we will learn
  - Expanse hardware
  - Getting information about your system
  - Using some common usage monitoring tools
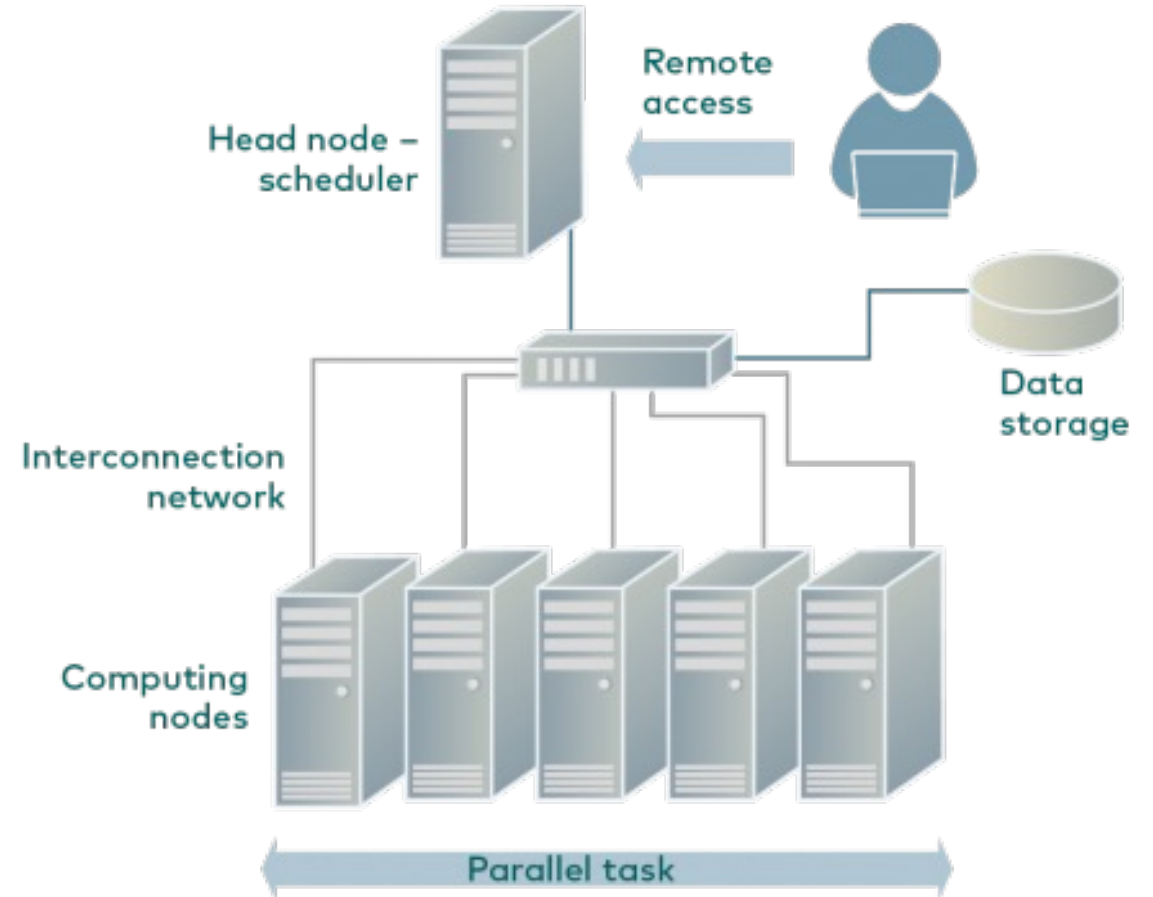
Outline

- Expanse
  - System Overview
  - AMD EPYC Processor Architecture
- Obtaining Hardware Information
  - CPU
  - GPU
  - Memory
  - Cache
  - SCSI, df, network, OS
- top and htop tools

Outline

- Expanse
  - System Overview
  - AMD EPYC Processor Architecture
- Obtaining Hardware Information
  - CPU
  - GPU
  - Memory
  - Cache
  - SCSI, df, network, OS
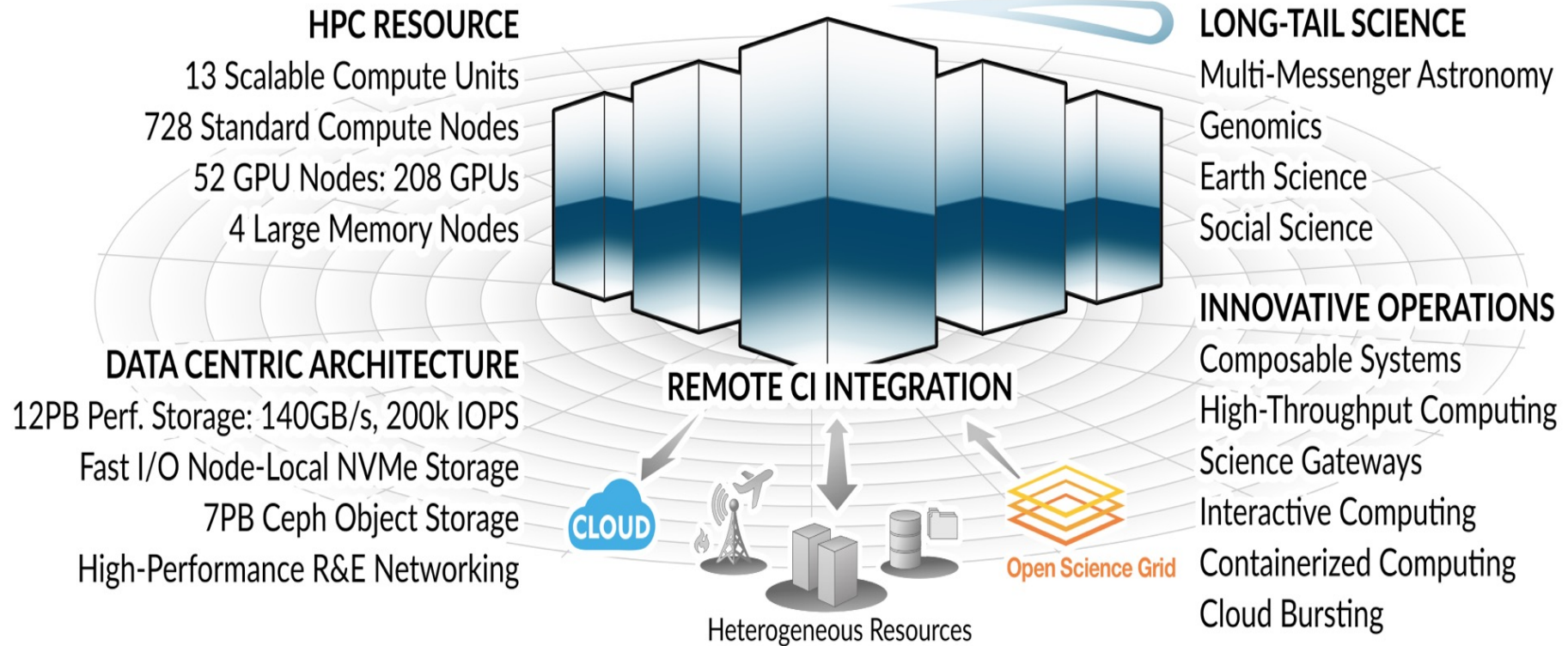- top and htop tools

System overview

- Linux/Mac –
  - use terminal + installed ssh app
- Windows:
  - Win10 terminal app + installed ssh app
  - Older Windows OS's: ssh clients apps Putty, Cygwin

- Login hostname for SDSC Expanse:
  - login.expanse.sdsc.edu
  - 198.202.100.13

Head node – scheduler

Remote access

Interconnection network

Data storage

Computing nodes

Parallel task

EXPANSE

COMPUTING WITHOUT BOUNDARIES
5 PETAFLOP/S HPC and DATA RESOURCE

HPC RESOURCE
13 Scalable Compute Units
728 Standard Compute Nodes
52 GPU Nodes: 208 GPUs
4 Large Memory Nodes

DATA CENTRIC ARCHITECTURE
12PB Perf. Storage: 140GB/s, 200k IOPS
Fast I/O Node-Local NVMe Storage
7PB Ceph Object Storage
High-Performance R&E Networking

LONG-TAIL SCIENCE
Multi-Messenger Astronomy
Genomics
Earth Science
Social Science

INNOVATIVE OPERATIONS
Composable Systems
High-Throughput Computing
Science Gateways
Interactive Computing
Containerized Computing
Cloud Bursting

REMOTE CI INTEGRATION

CLOUD

Heterogeneous Resources

Open Science Grid

For more details see the Expanse user guide @ https://www.sdsc.edu/support/user_guides/expanse.html
and the "Introduction to Expanse" webinar @ https://www.sdsc.edu/event_items/202006_Introduction_to_Expanse.html
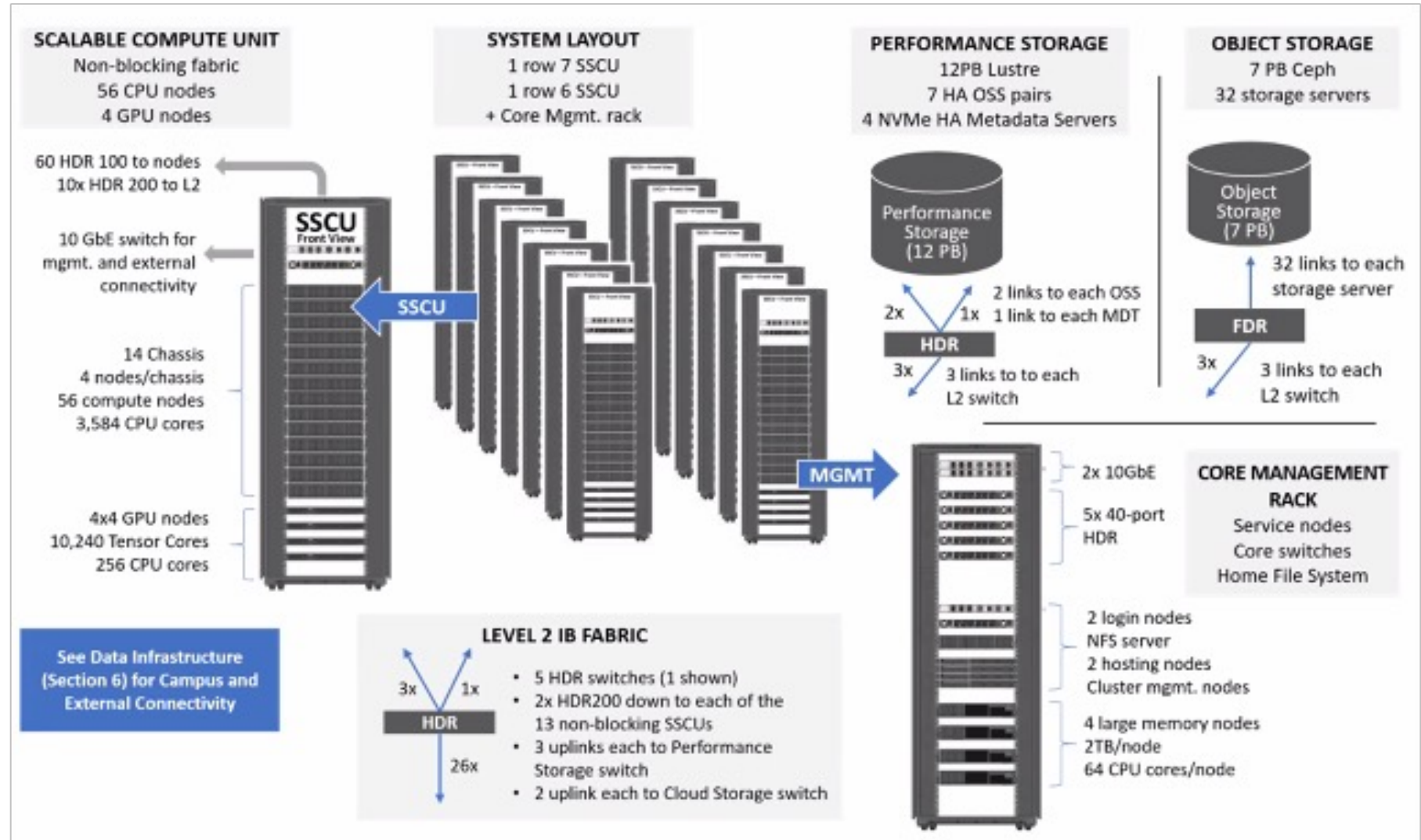
SDSC
SAN DIEGO SUPERCOMPUTER CENTER

at the UNIVERSITY OF CALIFORNIA; SAN DIEGO

UCSD

Expanse

Expanse Heterogeneous Architecture

System Summary

- 13 SDSC Scalable Compute Units (SSCU)
- 728 x 2s Standard Compute Nodes
- 93,184 Compute Cores
- 200 TB DDR4 Memory
- 52x 4-way GPU Nodes w/NVLINK
- 208 V100s
- 4x 2TB Large Memory Nodes
- HDR 100 non-blocking Fabric
- 12 PB Lustre High Performance
- Storage
- 7 PB Ceph Object Storage
- 1.2 PB on-node NVMe
- Dell EMC PowerEdge
- Direct Liquid Cooled

SCALABLE COMPUTE UNIT
Non-blocking fabric
56 CPU nodes
4 GPU nodes

SYSTEM LAYOUT
1 row 7 SSCU
1 row 6 SSCU
+ Core Mgmt. rack

PERFORMANCE STORAGE
12PB Lustre
7 HA OSS pairs
4 NVMe HA Metadata Servers

OBJECT STORAGE
7 PB Ceph
32 storage servers

60 HDR 100 to nodes
10x HDR 200 to L2

10 GbE switch for mgmt. and external connectivity

SSCU
Front View

SSCU

14 Chassis
4 nodes/chassis
56 compute nodes
3,584 CPU cores

4x4 GPU nodes
10,240 Tensor Cores
256 CPU cores

See Data Infrastructure (Section 6) for Campus and External Connectivity

Performance Storage (12 PB)

2x   1x   2 links to each OSS
1 link to each MDT
HDR
3x   3 links to to each
L2 switch

Object Storage (7 PB)

32 links to each storage server
FDR
3x   3 links to each
L2 switch

MGMT

2x 10GbE
5x 40-port HDR

CORE MANAGEMENT RACK
Service nodes
Core switches
Home File System

LEVEL 2 IB FABRIC
3x   1x
HDR
26x

- 5 HDR switches (1 shown)
- 2x HDR200 down to each of the 13 non-blocking SSCUs
- 3 uplinks each to Performance Storage switch
- 2 uplink each to Cloud Storage switch

2 login nodes
NFS server
2 hosting nodes
Cluster mgmt. nodes

4 large memory nodes
2TB/node
64 CPU cores/node

Using Login Nodes Properly

- The login nodes are meant for file editing, simple data analysis, & tasks that use minimal compute resources.

- All computationally demanding jobs should be submitted and run through the batch queuing system.

- Do not use the login nodes for:
  - computationally intensive processes,
  - as hosts for running workflow management tools
  - as primary data transfer nodes for large or numerous data transfers
  - as servers providing other services accessible to the Internet.
  - running Jupyter notebooks

- Login nodes are not the same as the batch nodes.
  - Users should request an interactive sessions to compile large programs.

Outline

- Expanse
  - System Overview
  - AMD EPYC Processor Architecture
- Obtaining Hardware Information
  - CPU
  - GPU
  - Memory
  - Cache
  - SCSI, df, network, OS
- top and htop tools

AMD EPYC 7742 Processor Architecture

- Zen 2 (Rome) microarchitecture

- 8 Core Complex Dies (CCDs).

- CCDs connect to memory, I/O, and each other through the I/O Die.

- 8 memory channels per socket.

- DDR4 memory at 3200MHz.

- PCI Gen4, up to 128 lanes of high speed I/O.

- Memory and I/O can be abstracted into separate quadrants each with 2 DIMM channels and 32 I/O lanes.

Reference: https://developer.amd.com/wp-content/resources/56827-1-0.pdf
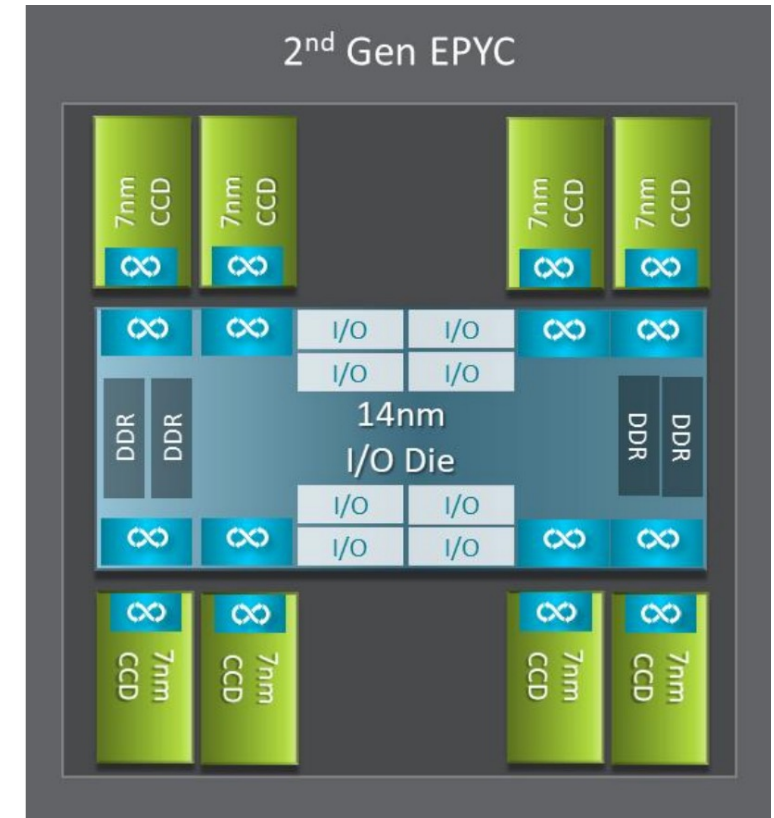
AMD EPYC 7742 Processor Architecture

- 2 Core Complexes (CCXs) per CCD

- 4 Zen2 cores in each CCX shared a 16M L3 cache. Total of 16 x 16 = 256MB L3 cache.

- Each core includes a private 512KB L2 cache.

Reference: https://developer.amd.com/wp-content/resources/56827-1-0.pdf

AMD EPYC 7742 Processor Architecture

- The four logical quadrants allow the processor to be partitioned into different NUMA domains. Options set in BIOS.

- Domains are designated as NUMA per socket (NPS).

- **NPS4:** Four NUMA domains per socket is the typical HPC configuration.
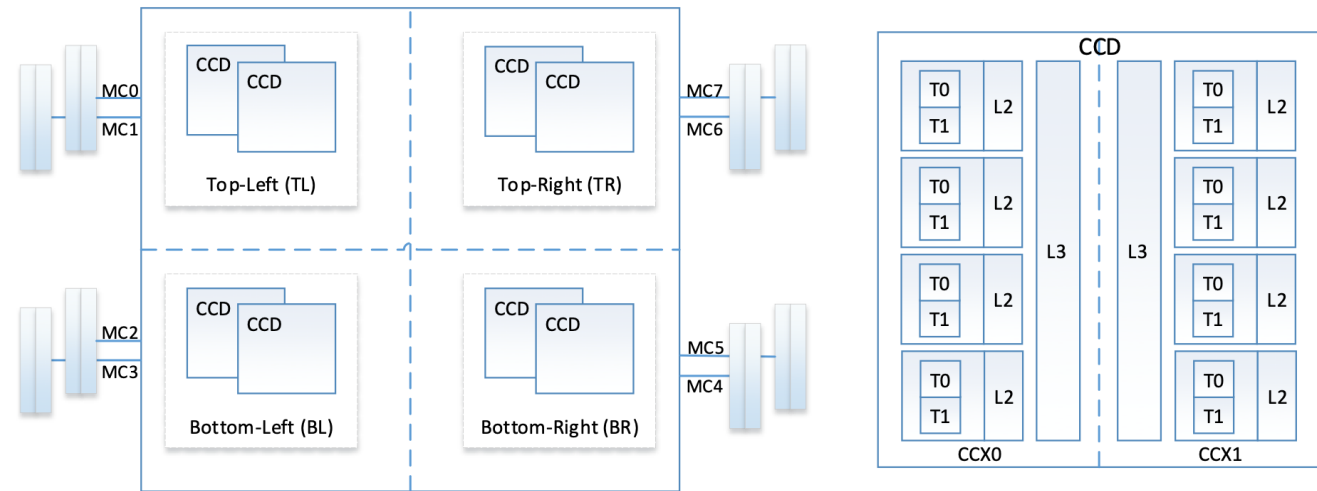
https://developer.amd.com/wp-content/resources/56338_1.00_pub.pdf

AMD EPYC 7742 Processor Architecture

- The processor is partitioned into four NUMA domains.

- Each logical quadrant is a NUMA domain.

- Memory is interleaved across the two memory channels

- PCIe devices will be local to one of four NUMA domains (the IO die that has the PCIe root for the device)

- ***This is the typical HPC configuration*** as workload is NUMA aware, ranks and memory can be pinned to cores and NUMA nodes.
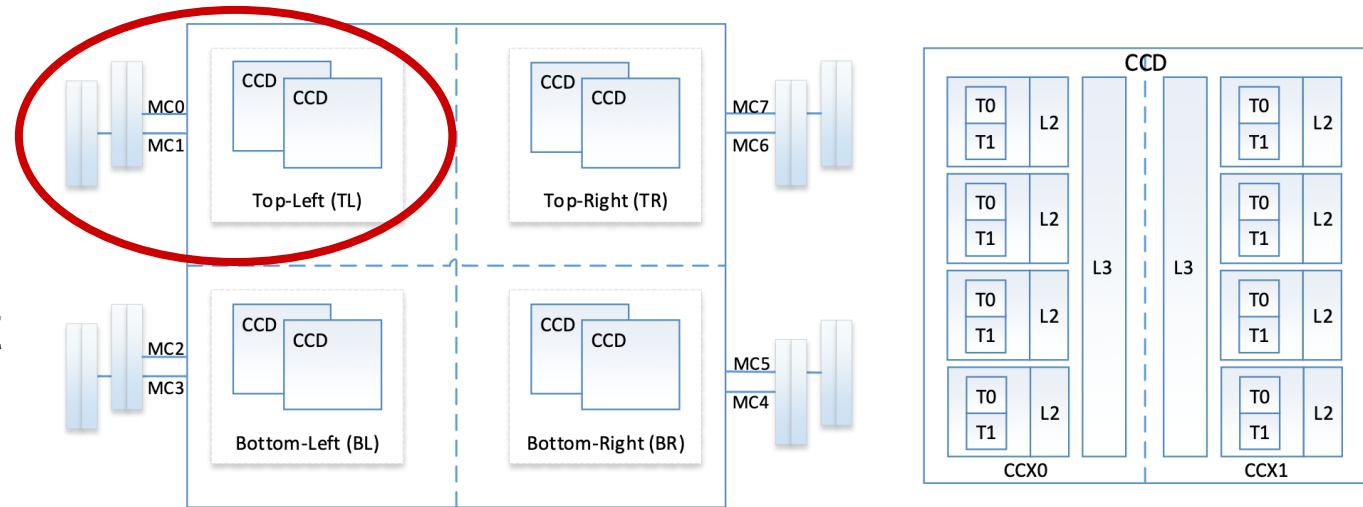
https://developer.amd.com/wp-content/resources/56338_1.00_pub.pdf

Expanse Compute Node

The AMD processors on Expanse

- Have 64 cores in total.

- Consist of 8 core complex dies (CCDs) per processor, with 2 core complexes (CCXs) per CCD. Four cores in a CCX share L3 cache.

- There are 4 NUMA domains per processor.

Each Expanse compute node

- Has 2 AMD processors, for a total of 128 cores.

- Has 2 GB RAM per core, for a total of 256 GB RAM.

- Has 1 TB NVMe storage (for fast scratch I/O)

Compute nodes are interconnected with HDR Infiniband.

Expanse storage

It is important to understand the storage hierarchy on Expanse (more later).

- The home file system is mounted via NFS and used to store smaller amounts of important data like source code files. This is slow I/O storage, not suitable for heavy I/O when running jobs.

- The Lustre file system is suitable for parallel I/O and large amount of project files like simulation output data.

- The local NVMe (SSD) storage is limited in space but very fast and should be used when doing heavy I/O during simulations.

Outline

- Expanse
  - System Overview
  - AMD EPYC Processor Architecture
- Obtaining Hardware Information
  - CPU
  - GPU
  - Memory
  - Cache
  - SCSI, df, network, OS
- top and htop tools

Getting hardware information – why do I care?

- You may be asked to report details of your hardware in a manuscript, presentation, proposal or request for computer time

- You'll know what you're running on and can answer questions like

  - Is the login node the same as the compute nodes?

  - How does one machine compare to another?

- It will give you a way of estimating performance or at least bounds on performance relative to another system. *All else being equal*, jobs will run at least as fast on hardware with

  - Faster CPU clock speeds

  - Larger caches

  - Faster local drives

Computers are like hammers!

Just like hammers, there is a wide variety of computer hardware. You can probably get away with using the wrong one, but your performance may be suboptimal and you might end up using a bigger tool than you need.
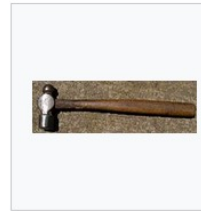
Ball-peen hammer   Ball peen hammer   Bush hammer   Claw hammer   Cross-peen hammer   Dog-head hammer (blacksmithing)
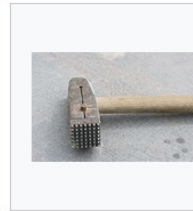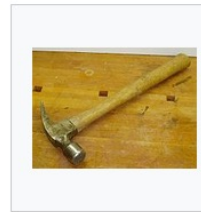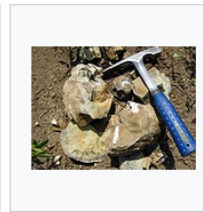
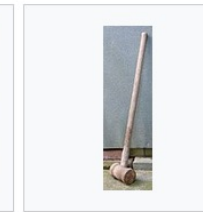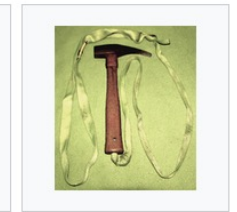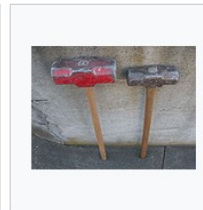Framing hammer   Geologist's hammer   HiFIT-hammer for aftertreatment of weld transitions   Long cross-face hammer (blacksmithing)   Post maul   Rock climbing hammer
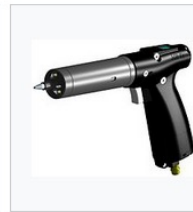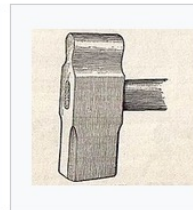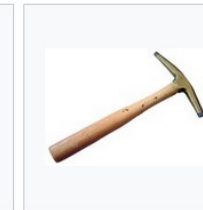
Rubber mallet   Sledgehammers   Straight pane sledgehammer   Twist hammer (blacksmithing)   Upholstery hammer   Wooden mallet

Wikipedia

Processor specifications: lscpu

On Linux systems, the **lscpu** command lists key processor information

- Number of processors (sockets)
- Processor type or model
- Nominal clock speed
- Number of cores per processor
- Cache sizes
- Instruction set architecture
- NUMA nodes

Processor specifications: Expanse compute node

```
Architecture:          x86_64
CPU op-mode(s):        32-bit, 64-bit
Byte Order:            Little Endian
CPU(s):                128
On-line CPU(s) list:   0-127
Thread(s) per core:    1
Core(s) per socket:    64
Socket(s):             2
NUMA node(s):          8
Vendor ID:             AuthenticAMD
CPU family:            23
Model:                 49
Model name:            AMD EPYC 7742 64-Core Processor
Stepping:              0
CPU MHz:               3257.493
BogoMIPS:              4491.71
Virtualization:        AMD-V
L1d cache:             32K
L1i cache:             32K
L2 cache:              512K
L3 cache:              16384K
```

```
NUMA node0 CPU(s):     0-15
NUMA node1 CPU(s):     16-31
NUMA node2 CPU(s):     32-47
NUMA node3 CPU(s):     48-63
NUMA node4 CPU(s):     64-79
NUMA node5 CPU(s):     80-95
NUMA node6 CPU(s):     96-111
NUMA node7 CPU(s):     112-127
Flags:                 fpu vme de pse tsc msr pae mce
cx8 apic sep mtrr pge mca cmov pat pse36 clflush mmx
fxsr sse sse2 ht syscall nx mmxext fxsr_opt pdpe1gb
rdtscp lm constant_tsc rep_good nopl xtopology
nonstop_tsc cpuid extd_apicid aperfmperf pni
pclmulqdq monitor ssse3 fma cx16 sse4_1 sse4_2
x2apic movbe popcnt aes xsave avx f16c rdrand
lahf_lm cmp_legacy svm extapic cr8_legacy abm sse4a
misalignsse 3dnowprefetch osvw ibs skinit wdt tce
topoext perfctr_core perfctr_nb bpext perfctr_llc
mwaitx cpb cat_l3 cdp_l3 hw_pstate sme ssbd mba sev
ibrs ibpb stibp vmmcall fsgsbase bmi1 avx2 smep bmi2
cqm rdt_a rdseed adx smap clflushopt clwb sha_ni
xsaveopt xsavec xgetbv1 xsaves cqm_llc cqm_occup_llc
cqm_mbm_total cqm_mbm_local clzero irperf xsaveerptr
wbnoinvd arat npt lbrv svm_lock nrip_save tsc_scale
vmcb_clean flushbyasid decodeas
```

Processor specifications: Expanse compute node

```
Architecture:          x86_64
CPU op-mode(s):        32-bit, 64-bit
Byte Order:            Little Endian
CPU(s):                128
On-line CPU(s) list:   0-127
Thread(s) per core:    1
Core(s) per socket:    64
Socket(s):             2
NUMA node(s):          8
Vendor ID:             AuthenticAMD
CPU family:            23
Model:                 49
Model name:            AMD EPYC 7742 64-Core Processor
Stepping:              0
CPU MHz:               3257.493
BogoMIPS:              4491.71
Virtualization:        AMD-V
L1d cache:             32K
L1i cache:             32K
L2 cache:              512K
L3 cache:              16384K
```

```
NUMA node0 CPU(s):     0-15
NUMA node1 CPU(s):     16-31
NUMA node2 CPU(s):     32-47
NUMA node3 CPU(s):     48-63
NUMA node4 CPU(s):     64-79
NUMA node5 CPU(s):     80-95
NUMA node6 CPU(s):     96-111
NUMA node7 CPU(s):     112-127
Flags:                 fpu vme de pse tsc msr pae mce
cx8 apic sep mtrr pge mca cmov pat pse36 clflush mmx
fxsr sse sse2 ht syscall nx mmxext fxsr_opt pdpe1gb
rdtscp lm constant_tsc rep_good nopl xtopology
nonstop_tsc cpuid extd_apicid aperfmperf pni
pclmulqdq monitor ssse3 fma cx16 sse4_1 sse4_2
x2apic movbe popcnt aes xsave avx f16c rdrand
lahf_lm cmp_legacy svm extapic cr8_legacy abm sse4a
misalignsse 3dnowprefetch osvw ibs skinit wdt tce
topoext perfctr_core perfctr_nb bpext perfctr_llc
mwaitx cpb cat_l3 cdp_l3 hw_pstate sme ssbd mba sev
ibrs ibpb stibp vmmcall fsgsbase bmi1 avx2 smep bmi2
cqm rdt_a rdseed adx smap clflushopt clwb sha_ni
xsaveopt xsavec xgetbv1 xsaves cqm_llc cqm_occup_llc
cqm_mbm_total cqm_mbm_local clzero irperf xsaveerptr
wbnoinvd arat npt lbrv svm_lock nrip_save tsc_scale
vmcb_clean flushbyasid decodeas
```

Processor specifications: /proc/cpuinfo

On Linux systems, the **/proc/cpuinfo** pseudo-file contains pretty much the same information that you get from lscpu, but with a few differences

- Information is listed per core

- Access to instantaneous clock speeds

- Bugs detected / addressed
  see discussion: https://unix.stackexchange.com/questions/456425/what-does-the-bugs-section-of-proc-cpuinfo-actually-show

- Microcode, TLB size , power management, cache line flush sizes and other low-level details that you probably don't need to know about

Processor specifications: /proc/cpuinfo

```
$ grep 'cpu MHz' /proc/cpuinfo  | head -5
cpu MHz         : 3325.325
cpu MHz         : 2239.263
cpu MHz         : 3374.887
cpu MHz         : 2360.551
cpu MHz         : 2393.493
```

```
--- selected output ---
microcode         : 0x8301038
bugs              : sysret_ss_attrs spectre_v1 spectre_v2 spec_store_bypass
TLB size          : 3072 4K pages
clflush size      : 64
cache_alignment   : 64
address sizes     : 43 bits physical, 48 bits virtual
power management : ts ttp tm hwpstate cpb eff_freq_ro [13] [14]
```

A brief aside on nomenclature
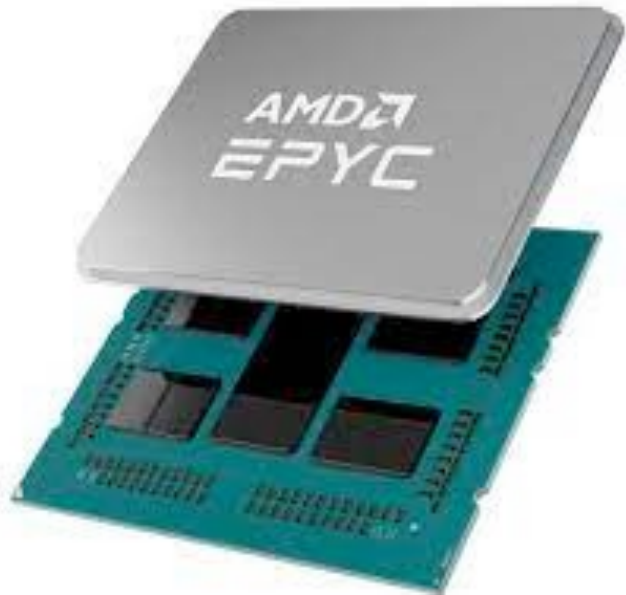
We normally think of the multicore unit that plugs into the motherboard as the "processor"

/proc/cpuinfo uses processor in a different way to mean compute core, counted across all the cores available in a node. lscpu uses more intuitive terminology.

```
processor       : 0
physical id     : 0
cpu cores       : 64
```

Quick aside on simultaneous multithreading

Simultaneous multithreading, abbreviated as SMT, is the process of a CPU splitting each of its physical cores into virtual cores, which are known as threads. This is done in order to increase performance and allow each core to run two instruction streams at once.

Intel branded this process as hyper-threading, but hyper-threading is the same thing as simultaneous multithreading. For example, AMD CPUs with four cores use simultaneous multithreading to provide eight threads, and most Intel CPUs with two cores use hyper-threading to provide four threads.

https://www.tomshardware.com/reviews/simultaneous-multithreading-definition,5762.html

SDSC does not enable hyperthreading on its systems. When hyperthreading is enabled, core count will appear to be doubled.

A brief aside on pseudo-files

Up to this point, we've been using the term pseudo-file without defining what it is. Recall that in the UNIX/Linux world, everything is treated as a file (files, directories, devices, etc.)

/proc and /sys are just interfaces to the Linux kernel data structures in a convenient and familiar file system format

```
$ ls -ld /proc
dr-xr-xr-x 2258 root root 0 Jul 28 09:27 /proc
[sinkovit@login01 ~]$ ls -ld /proc/cpuinfo
-r--r--r-- 1 root root 0 Jul 28 16:56 /proc/cpuinfo

$ head /proc/cpuinfo
processor   : 0
vendor_id   : AuthenticAMD
cpu family : 23
Model.      : 49
model name : AMD EPYC 7742 64-Core Processor
stepping    : 0
microcode  : 0x8301038
```

What's in a name?

We usually think of processors in terms of their codenames, such as Rome or Milan. Unfortunately, the /proc/cpuinfo pseudo-file returns something a little more opaque such as "AMD EPYC 7742 64-Core Processor". A quick Google search helps

In November 2018 AMD announced Epyc 2 at their Next Horizon event, the second generation of Epyc processors code-named "Rome" and based on the Zen 2 microarchitecture.[19] The processors feature up to eight 7 nm-based "chiplet" processors with a 14 nm-based IO chip providing 128 PCIe lanes in the center interconnected via Infinity Fabric. The processors support up to 8 channels of DDR4 RAM up to 4 TB, and introduce support for PCIe 4.0. These processors have up to 64 cores with 128 SMT threads per socket.[20] The 7 nm "Rome" is manufactured by TSMC.[11] It was released on August 7, 2019.[21]

https://en.wikipedia.org/wiki/Epyc

Advanced Vector Extensions (AVX, AVX2, AVX512)

The Advanced Vector Extensions (AVX) are an extension to the x86 microprocessor architecture that allows a compute core to perform up to 8 floating point operations per cycle. Previous limit was 4/core/cycle

- AVX2 improves this to 16 Flops/cycle/core (Comet, Expanse)

- AVX512 further improves to 32 Flops/cycle/core (Intel ≥ Skylake)

These were developed partially in response to challenges in increasing CPU clock speeds
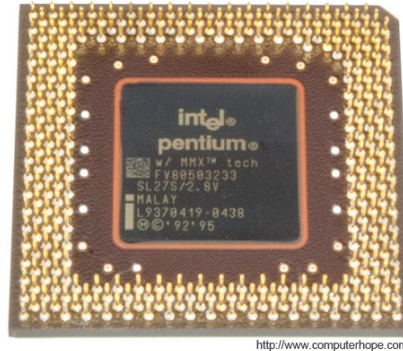
March 6, 2000 8:00 AM PST

AMD makes move to 1-GHz chip

By Joe Wilcox and Michael Kanellos
Staff Writers, CNET News

Advanced Vector Extensions (AVX, AVX2, AVX512)

- Can theoretically obtain a 2x speedup when going from a non-AVX processor to an AVX capable processor (all else being equal)
  - And another 2x from AVX to AVX2
  - And another 2x from AVX2 to AVX512

- But don't get too excited (or worried that Expanse doesn't have AVX512)
  - It's difficult enough to make good use of AVX and even harder to make good use of AVX2 or AVX512.
  - Need long loops with vectorizable content. Memory bandwidth not keeping up with gains in computing power.
  - On Skylake, clock speed scaled down when executing AVX512 instructions

Getting memory information: /proc/meminfo

On Linux machines, the /proc/meminfo pseudo-file lists key memory specs. More information than you probably want, but at least one bit of useful data

```
MemTotal:       263698228 kB  (total physical memory)
MemFree:        251035032 kB
MemAvailable:   250623760 kB
Buffers:            12824 kB
Cached:           3126364 kB
SwapCached:             0 kB
Active:           1301564 kB  (pretty good approximation to used memory)
Inactive:         2990668 kB
Active(anon):     1240284 kB
Inactive(anon):   2890076 kB
Active(file):       61280 kB
Inactive(file):    100592 kB
Unevictable:            0 kB
Mlocked:                0 kB
SwapTotal:              0 kB
SwapFree:               0 kB
Dirty:                 32 kB
Writeback:              0 kB
AnonPages:        1151660 kB
```

For more details, see http://www.redhat.com/advice/tips/meminfo.html

SDSC SAN DIEGO SUPERCOMPUTER CENTER

at the **UNIVERSITY OF CALIFORNIA; SAN DIEGO**

UCSD

Getting memory information (/proc/meminfo)

Using a simple script, you can monitor total memory usage for all processes as a function of time. Note that there is a lot of discussion on how to precisely measure memory (http://stackoverflow.com/search?q=measuring+memory+usage). The following should be good enough if you're on a dedicated node.

```perl
#!/usr/bin/perl
use strict;
use warnings;
my $count = 0;
print (" time(s)    Memory (GB)\n");
while(1) {
    sleep(1);
    $count++;
    open(MI, "/proc/meminfo");
    while(<MI>) {
        if (/Active:/) {
            my (undef, $active, undef) = split();
            $active = $active /        1048576.0;
            printf("%6d      %f\n", $count, $active);
        }
    }
    close(MI);
}
```

More memory information - dmidecode

If you really need to dig deeper and get more details on memory configuration, you can run the dmidecode command. You'll need root privileges to do this.

Output shows the results for one DIMM slot.

dmidecode --type memory

```
Memory Device
        Array Handle: 0x001D
        Error Information Handle: No Error
        Total Width: 72 bits
        Data Width: 64 bits
        Size: 16384 MB
        Form Factor: DIMM
        Set: None
        Locator: DIMM_A1
        Bank Locator: CPU1
        Type: DDR4
        Type Detail: Synchronous Registered (Buffered)
        Speed: 2133 MHz
        Manufacturer: 0xCE00
        Serial Number: 0x394FECDD
        Asset Tag: Unknown
        Part Number: M393A2G40DB0-CPB
        Rank: 1
        Configured Clock Speed: 2133 MHz
        Minimum Voltage:  1.2 V
        Maximum Voltage:  1.2 V
        Configured Voltage:  1.2 V
```

Getting GPU information

If you're using GPU nodes, you can use nvidia-smi (NVIDIA System Management Interface program) to get GPU information (type, count, etc.)

When running in gpu-shared partition, will only see the GPUs you had requested (typically one)

```
$ nvidia-smi
Mon Aug  2 10:16:29 2021
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 460.32.03    Driver Version: 460.32.03    CUDA Version: 11.2     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|                               |                      |               MIG M. |
|===============================+======================+======================|
|   0  Tesla V100-SXM2...  On   | 00000000:18:00.0 Off |                    0 |
| N/A   40C    P0    67W / 300W |      0MiB / 32510MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                                  |
|  GPU   GI   CI        PID   Type   Process name                  GPU Memory |
|        ID   ID                                                   Usage      |
|=============================================================================|
|  No running processes found                                                 |
+-----------------------------------------------------------------------------+
```

Finding cache information

On Linux systems, can obtain cache properties through the /sys pseudo filesystem. Details may vary slightly by O/S version and vendor, but basic information should be consistent. Goes deeper into cache properties than lscpu

```
$ pwd
/sys/devices/system/cpu

$ ls
cpu0    cpu17   cpu25   cpu33   cpu41   cpu5    cpu58   cpu9        offline
cpu1    cpu18   cpu26   cpu34   cpu42   cpu50   cpu59   cpufreq     online
cpu10   cpu19   cpu27   cpu35   cpu43   cpu51   cpu6    cpuidle     possible
cpu11   cpu2    cpu28   cpu36   cpu44   cpu52   cpu60   hotplug     power
...

$ cd cpu0/cache
$ ls
index0   index1   index2   index3   power   uevent

$ cd index0
$ ls
coherency_line_size    physical_line_partition   size
id                     power                     type
level                  shared_cpu_list           uevent
number_of_sets         shared_cpu_map            ways_of_associativity
```

Expanse Cache properties – AMD Rome (AMD EPYC 7742)

level	type	line size	sets	associativity	size (KB)
L1	data	64	64	8	32
L1	instruction	64	64	8	32
L2	unified	64	1024	8	512
L3	unified	64	16384	16	16384

L1 and L2 caches are per core

L3 cache shared between 4 cores on a core complex

line size x sets x associativity = size

L2 cache size = 64 x 1024 x 8 = 524288 = 512 K

Comet Cache properties – Intel Haswell (Intel Xeon E5-2680)

level	type	line size	sets	associativity	size (KB)
L1	data	64	64	8	32
L1	instruction	64	64	8	32
L2	unified	64	512	8	256
L3	unified	64	24576	20	30720

L1 and L2 caches are per core

L3 cache shared between all 12 cores in socket

line size x sets x associativity = size

L2 cache size = 64 x 512 x 8 = 262144 = 256 K

Gordon Cache properties – Intel Sandy Bridge (Intel Xeon E5-2670)

level	type	line size	sets	associativity	size (KB)
L1	data	64	64	8	32
L1	instruction	64	64	8	32
L2	unified	64	512	8	256
L3	unified	64	16384	20	20480

L1 and L2 caches are per core

L3 cache shared between all 8 cores in socket

line size x sets x associativity = size

L2 cache size = 64 x 512 x 8 = 262144 = 256 K

Trestles Cache properties – AMD Magny-Cours (AMD Opteron Processor 6136)

level	type	line size	sets	associativity	size (KB)
L1	data	64	512	2	64
L1	instruction	64	512	2	64
L2	unified	64	512	16	512
L3	unified	64	1706	48	5118

L1 and L2 caches are per core

L3 cache shared between all 8 cores in socket

line size x sets x associativity = size

L2 cache size = 64 x 512 x 16 = 524288= 512K

Impact of cache size on performance

Note – example based on old systems, but still very illustrative

Based on the clock speed and instruction set, program run on single core of Gordon should be 2.26x faster than on Trestles. The larger L1 and L2 cache sizes on Trestles mitigate performance impact for very small problems.

DGSEV (Ax=b) wall times as function of problem size

N	t (Trestles)	t (Gordon)	ratio	KB
62	0.000117	0.000086	1.36	30
125	0.000531	0.000384	1.38	122
250	0.002781	0.001542	1.80	488
500	0.016313	0.007258	2.24	1953
1000	0.107222	0.046252	2.31	7812
2000	0.744837	0.331818	2.24	31250
4000	5.489990	2.464218	2.23	125000

Finding SCSI device information

SCSI (Small Computer System Interface) is a common interface for mounting peripheral, such as hard drives and SSDs. The lsscsi or /proc/scsi/scsi file command will provide info on SCSI devices.

```
$ lsscsi
[3:0:0:0]     disk     ATA          SSDSC2KB480G8R    DL67   /dev/sda
[N:0:0:1]     disk     Dell Express Flash NVMe P4510 1TB SFF__1    /dev/nvme0n1
```


Dell 1TB PCIe
NVMe Read

df provides information on filesystem usage

Local scratch (SSDs)

```
$ df -h
Filesystem                                                Size   Used  Avail Use% Mounted on
/dev/nvme0n1p1                                            916G    77M   870G    1% /scratch
ps-071.sdsc.edu:/ps-data/community-sw                    1.0T   102G   923G   10% /expanse/community
10.21.0.21:6789,10.21.11.7:6789,10.21.11.8:6789:/        1.7T   553G   1.2T   33% /cm/shared
10.22.101.123@o2ib:10.22.101.124@o2ib:/expanse/projects   11P   1.4P   9.3P   13%
/expanse/lustre/projects
10.22.101.123@o2ib:10.22.101.124@o2ib:/expanse/scratch    11P   1.4P   9.3P   13%
/expanse/lustre/scratch
10.22.100.113:/pool3/home/sinkovit                       209T   4.1T   205T    2% /home/sinkovit


--- only selected filesystems shown ---
```

df provides information on filesystem usage

Community and SDSC maintained software stacks

```
$ df -h
Filesystem                                              Size  Used  Avail  Use%  Mounted on
/dev/nvme0n1p1                                          916G   77M   870G    1%  /scratch
ps-071.sdsc.edu:/ps-data/community-sw                  1.0T  102G   923G   10%  /expanse/community
10.21.0.21:6789,10.21.11.7:6789,10.21.11.8:6789:/      1.7T  553G   1.2T   33%  /cm/shared
10.22.101.123@o2ib:10.22.101.124@o2ib:/expanse/projects 11P  1.4P   9.3P   13%
/expanse/lustre/projects
10.22.101.123@o2ib:10.22.101.124@o2ib:/expanse/scratch  11P  1.4P   9.3P   13%
/expanse/lustre/scratch
10.22.100.113:/pool3/home/sinkovit                     209T  4.1T   205T    2%  /home/sinkovit

--- only selected filesystems shown ---
```

df provides information on filesystem usage

```
$ df -h
Filesystem                                              Size  Used  Avail Use% Mounted on
/dev/nvme0n1p1                                           916G   77M   870G   1% /scratch
ps-071.sdsc.edu:/ps-data/community-sw                   1.0T  102G   923G  10% /expanse/community
10.21.0.21:6789,10.21.11.7:6789,10.21.11.8:6789:/       1.7T  553G   1.2T  33% /cm/shared
10.22.101.123@o2ib:10.22.101.124@o2ib:/expanse/projects  11P  1.4P   9.3P  13%
/expanse/lustre/projects
10.22.101.123@o2ib:10.22.101.124@o2ib:/expanse/scratch   11P  1.4P   9.3P  13%
/expanse/lustre/scratch
10.22.100.113:/pool3/home/sinkovit                      209T  4.1T   205T   2% /home/sinkovit


--- only selected filesystems shown ---
```

Lustre scratch and project filesystems

df provides information on filesystem usage

```
$ df -h
Filesystem                                               Size  Used  Avail Use% Mounted on
/dev/nvme0n1p1                                            916G   77M   870G   1% /scratch
ps-071.sdsc.edu:/ps-data/community-sw                    1.0T  102G   923G  10% /expanse/community
10.21.0.21:6789,10.21.11.7:6789,10.21.11.8:6789:/        1.7T  553G   1.2T  33% /cm/shared
10.22.101.123@o2ib:10.22.101.124@o2ib:/expanse/projects   11P  1.4P   9.3P  13%
/expanse/lustre/projects
10.22.101.123@o2ib:10.22.101.124@o2ib:/expanse/scratch    11P  1.4P   9.3P  13%
/expanse/lustre/scratch
10.22.100.113:/pool3/home/sinkovit                       209T  4.1T   205T   2% /home/sinkovit

--- only selected filesystems shown ---
```

Home filesystem

SDSC **SAN DIEGO SUPERCOMPUTER CENTER**

at the UNIVERSITY OF CALIFORNIA; SAN DIEGO

UCSD

Finding network information

The ip command (/sbin/ip) is normally used by sys admins, but regular users can use it to learn about networking information

```
$ /sbin/ip link
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN mode DEFAULT
group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: eno1: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN mode DEFAULT group
default qlen 1000
    link/ether 6c:2b:59:bb:61:24 brd ff:ff:ff:ff:ff:ff
3: eno33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 9000 qdisc mq state UP mode
DEFAULT group default qlen 1000
    link/ether 1c:34:da:62:a8:50 brd ff:ff:ff:ff:ff:ff
4: eno34: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN mode DEFAULT group
default qlen 1000
    link/ether 1c:34:da:62:a8:51 brd ff:ff:ff:ff:ff:ff
5: ib0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 4092 qdisc mq state UP mode DEFAULT
group default qlen 256
    link/infiniband 20:00:11:07:fe:80:00:00:00:00:00:00:1c:34:da:03:00:5d:53:90
brd 00:ff:ff:ff:ff:12:40:1b:ff:ff:00:00:00:00:00:00:ff:ff:ff:ff
6: eno33.450@eno33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 9000 qdisc noqueue
state UP mode DEFAULT group default qlen 1000
    link/ether 1c:34:da:62:a8:50 brd ff:ff:ff:ff:ff:ff
```

Finding OS and kernel information

Use uname to get information on the Linux kernel

```
$ uname -r
4.18.0-147.el8.x86_64
$ uname -o
GNU/Linux
$ uname -a
Linux login01 4.18.0-147.el8.x86_64 #1 SMP Wed Dec 4 21:51:45 UTC 2019 x86_64
x86_64 x86_64 GNU/Linux
```

Look in /etc/centos-release to get the Linux distribution (will vary by Linux distro)

```
$ cat /etc/centos-release
CentOS Linux release 8.1.1911 (Core)
```

Machine info - overkill?

- We've probably gone a little deeper than is necessary for you to be an effective supercomputer user.

- Think of this as a way to round out your HPC knowledge. You're learning a little bit about the tools of the trade, getting comfortable poking around on a system, acquiring the knowledge that will make it easier to work with your sys admin and picking up the background that will help you to make intelligent decisions in the future.

- Exercise: grab an interactive node (or just the login node) on Expanse and experiment with what we've covered. Cheat sheet on the next slide.

- Note that login and compute nodes are somewhat different

Machine info – cheat sheet

File or command	Information provided
less /proc/cpuinfo or lscpu	CPU specs
less /proc/meminfo	Memory specs and usage
nvidia-smi	GPU specs and usage
cd /sys/devices/system/cpu/cpu0/cache … then look at directory contents	Cache configuration
less /proc/scsi/scsi or lsscsi	Peripherals (e.g. SSDs)
less /etc/mtab	Mounted file systems
df -h	File system usage (readable format)
/sbin/ip link	Networking information
uname -a	OS information
less /etc/centos-release	Centos version

Outline

- Expanse
  - System Overview
  - AMD EPYC Processor Architecture
- Obtaining Hardware Information
  - CPU
  - GPU
  - Memory
  - Cache
  - SCSI, df, network, OS
- top and htop tools

Using the Linux top/htop utility

The top utility is found on all Linux systems and provides a high level view of running processes. Does not give any information at the source code level (profiling), but can still be very useful for answering questions such as

- How many of my processes are running?
- What are the states of the processes (running, sleeping, etc.)?
- Which cores are being utilized?
- Are there any competing processes that may be affecting my performance?
- What fraction of the CPU is each process using?
- How much memory does each process use?
- Is the memory usage growing over time? (Useful for identifying memory leaks)
- How many threads are my processes using?

Customizing top

Top has the following defaults, but is easily customizable

- Processes only (no threads)
- To toggle threads display, type "H" while top is running

- Information for all users
- Can restrict to a single user by launching with "top -u username"

- Process ID, priority, 'nice' level, virtual memory, physical memory, shared memory, state, %CPU, %memory, CPU time, command
- To modify, type "f" while top is running and toggle fields using letters

- Update information every 3 seconds
- Change refresh rate by launching with "top -d *n*"

- Ordered by CPU usage
- Type "M" to order by memory usage

Non-threaded code


```
●●●                    🏠 stivoknis — sinkovit@gcn-17-57:~ — ssh — 94×33
top - 08:37:00 up 60 days, 14:23,  1 user,  load average: 15.32, 10.36, 6.12
Tasks: 624 total,  17 running, 607 sleeping,   0 stopped,   0 zombie
Cpu(s): 68.7%us,  1.3%sy,  0.0%ni, 29.9%id,  0.1%wa,  0.0%hi,  0.0%si,  0.0%st
Mem:  66054160k total, 37885796k used, 28168364k free,     8808k buffers
Swap:  2097144k total,    13400k used,  2083744k free, 32927192k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
70388 sinkovit  20   0  194m  76m 1612 R 100.0  0.1  1:31.06 lobfaster.pl
72547 sinkovit  20   0  120m 2976 1612 R 100.0  0.0  0:01.49 lobfaster.pl
72516 sinkovit  20   0  127m 9.9m 1608 R 100.0  0.0  0:02.09 lobfaster.pl
72526 sinkovit  20   0  121m 3388 1612 R 100.0  0.0  0:01.84 lobfaster.pl
72535 sinkovit  20   0  121m 4208 1612 R 100.0  0.0  0:01.73 lobfaster.pl
72565 sinkovit  20   0  120m 3212 1612 R 100.0  0.0  0:01.01 lobfaster.pl
72268 sinkovit  20   0  130m  12m 1612 R 98.9   0.0  0:11.96 lobfaster.pl
72359 sinkovit  20   0  123m 5976 1612 R 98.9   0.0  0:09.77 lobfaster.pl
72460 sinkovit  20   0  127m  10m 1612 R 98.9   0.0  0:08.38 lobfaster.pl
72481 sinkovit  20   0  131m  13m 1612 R 98.9   0.0  0:07.44 lobfaster.pl
72529 sinkovit  20   0  122m 4576 1612 R 98.9   0.0  0:01.82 lobfaster.pl
72439 sinkovit  20   0  130m  12m 1612 R 97.0   0.0  0:08.64 lobfaster.pl
72590 sinkovit  20   0  120m 3140 1612 R 71.7   0.0  0:00.37 lobfaster.pl
72602 sinkovit  20   0  120m 2576 1612 R 38.8   0.0  0:00.20 lobfaster.pl
72605 sinkovit  20   0  120m 2528 1600 R 34.9   0.0  0:00.18 lobfaster.pl
72608 sinkovit  20   0  119m 2340 1600 R 21.3   0.0  0:00.11 lobfaster.pl
```

16 processes, each using anywhere from 21.3% to 100% of a compute core.

Memory footprint (RES) is minimal, with each process only using up to 76 MB.

CPU times ranging from 0.11s (just started) to 1:31

SDSC

SAN DIEGO SUPERCOMPUTER CENTER

at the **UNIVERSITY OF CALIFORNIA; SAN DIEGO**

UCSD

Threaded code (thread display off)

```
● ● ●        🏠 stivoknis — sinkovit@gcn-17-57:~ — ssh — 87×33
Tasks: 592 total,   2 running, 590 sleeping,   0 stopped,   0 zombie
Cpu(s): 99.8%us,  0.2%sy,  0.0%ni,  0.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Mem:  66054160k total, 16519596k used, 49534564k free,    11248k buffers
Swap: 2097144k total,     13400k used,  2083744k free,  7563960k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
81007 sinkovit  20   0 6872m 5.8g 1412 R 1595.9  9.1   5:56.48 lob_constructio
```

Threaded code with thread display toggled to the "off" position. Note the heavy CPU usage, very close to 1600%

Threaded code (thread display on)

```
● ● ●            🏠 stivoknis — sinkovit@gcn-17-57:~ — ssh — 87×33
Tasks: 626 total,  17 running, 609 sleeping,   0 stopped,   0 zombie
Cpu(s): 15.8%us,  0.2%sy,  0.0%ni, 84.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Mem:  66054160k total, 17495556k used, 48558604k free,    11552k buffers
Swap:  2097144k total,    13400k used,  2083744k free,  8478752k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
81007 sinkovit  20   0 6927m 5.8g 1412 R 99.5  9.2  8:37.91 lob_constructio
81096 sinkovit  20   0 6927m 5.8g 1412 R 10.5  9.2  1:13.43 lob_constructio
81105 sinkovit  20   0 6927m 5.8g 1412 R 10.5  9.2  1:13.43 lob_constructio
81107 sinkovit  20   0 6927m 5.8g 1412 R 10.5  9.2  1:13.43 lob_constructio
81097 sinkovit  20   0 6927m 5.8g 1412 R 10.2  9.2  1:13.40 lob_constructio
81099 sinkovit  20   0 6927m 5.8g 1412 R 10.2  9.2  1:13.39 lob_constructio
81100 sinkovit  20   0 6927m 5.8g 1412 R 10.2  9.2  1:13.44 lob_constructio
81101 sinkovit  20   0 6927m 5.8g 1412 R 10.2  9.2  1:13.44 lob_constructio
81102 sinkovit  20   0 6927m 5.8g 1412 R 10.2  9.2  1:13.43 lob_constructio
81103 sinkovit  20   0 6927m 5.8g 1412 R 10.2  9.2  1:13.45 lob_constructio
81106 sinkovit  20   0 6927m 5.8g 1412 R 10.2  9.2  1:13.44 lob_constructio
81108 sinkovit  20   0 6927m 5.8g 1412 R 10.2  9.2  1:13.44 lob_constructio
81109 sinkovit  20   0 6927m 5.8g 1412 R 10.2  9.2  1:13.29 lob_constructio
81110 sinkovit  20   0 6927m 5.8g 1412 R 10.2  9.2  1:13.39 lob_constructio
81098 sinkovit  20   0 6927m 5.8g 1412 R  9.9  9.2  1:13.44 lob_constructio
81104 sinkovit  20   0 6927m 5.8g 1412 R  9.9  9.2  1:13.38 lob_constructio
```

16 threads, with only one thread making good use of CPU

Total memory usage 5.8 GB (9.2% of available)

Threaded code (thread display on)

```
● ● ●                🏠 stivoknis — sinkovit@gcn-17-57:~ — ssh — 87×33
Tasks: 626 total,  17 running, 609 sleeping,   0 stopped,   0 zombie
Cpu(s): 90.9%us,  0.1%sy,  0.0%ni,  9.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Mem:  66054160k total, 17628152k used, 48426008k free,   11496k buffers
Swap:  2097144k total,   13400k used,  2083744k free,  8396488k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM   TIME+  COMMAND
81007 sinkovit  20   0 7132m 6.0g 1412 R 100.0  9.5   7:54.98 lob_constructio
81110 sinkovit  20   0 7132m 6.0g 1412 R 90.4   9.5   0:51.15 lob_constructio
81096 sinkovit  20   0 7132m 6.0g 1412 R 90.1   9.5   0:51.17 lob_constructio
81098 sinkovit  20   0 7132m 6.0g 1412 R 90.1   9.5   0:51.19 lob_constructio
81099 sinkovit  20   0 7132m 6.0g 1412 R 90.1   9.5   0:51.14 lob_constructio
81100 sinkovit  20   0 7132m 6.0g 1412 R 90.1   9.5   0:51.18 lob_constructio
81101 sinkovit  20   0 7132m 6.0g 1412 R 90.1   9.5   0:51.18 lob_constructio
81102 sinkovit  20   0 7132m 6.0g 1412 R 90.1   9.5   0:51.18 lob_constructio
81103 sinkovit  20   0 7132m 6.0g 1412 R 90.1   9.5   0:51.19 lob_constructio
81104 sinkovit  20   0 7132m 6.0g 1412 R 90.1   9.5   0:51.14 lob_constructio
81105 sinkovit  20   0 7132m 6.0g 1412 R 90.1   9.5   0:51.19 lob_constructio
81106 sinkovit  20   0 7132m 6.0g 1412 R 90.1   9.5   0:51.18 lob_constructio
81107 sinkovit  20   0 7132m 6.0g 1412 R 90.1   9.5   0:51.18 lob_constructio
81108 sinkovit  20   0 7132m 6.0g 1412 R 90.1   9.5   0:51.18 lob_constructio
81097 sinkovit  20   0 7132m 6.0g 1412 R 89.8   9.5   0:51.15 lob_constructio
81109 sinkovit  20   0 7132m 6.0g 1412 R 89.8   9.5   0:51.08 lob_constructio
```

16 threads, all making good (but not ideal) use of the compute cores

htop – like top, but with better interface, vertical and horizontal scrolling, process tree view, etc.

htop – like top, but with better interface, vertical and horizontal scrolling, process tree view, etc.

top/htop exercise

- Once we do the hands on exercises for the other sessions, open another terminal and run top or htop

- In the meantime, try running on the login node or an interactive node and play around with the options

- Useful tip – on Expanse, once Slurm has allocated a node to your account, you can then directly login to the node