# Basics of Computing

Peter Ruprecht

peter.ruprecht@colorado.edu

www.rc.colorado.edu

Slides:
https://github.com/ResearchComputing/Basics_Supercomputing

# Outline

- Hardware hierarchy: supercomputer to core
- Components of a compute node
- Interconnect
- Storage and filesystems
- Coprocessors
- Operating systems

# Compute Hardware Architecture

Processing: Supercomputer – Node – Socket/CPU – Core

Memory: Distributed – RAM – L3 – L2 – L1

Interconnect network

Storage (disk)

Coprocessors

Non-cluster supercomputers (IBM Blue Gene, SGI UV)

Cloud?

*Parallelism at all levels of the computing system is the name of the game today … SIMD, OpenMP, MPI*

# Clusters

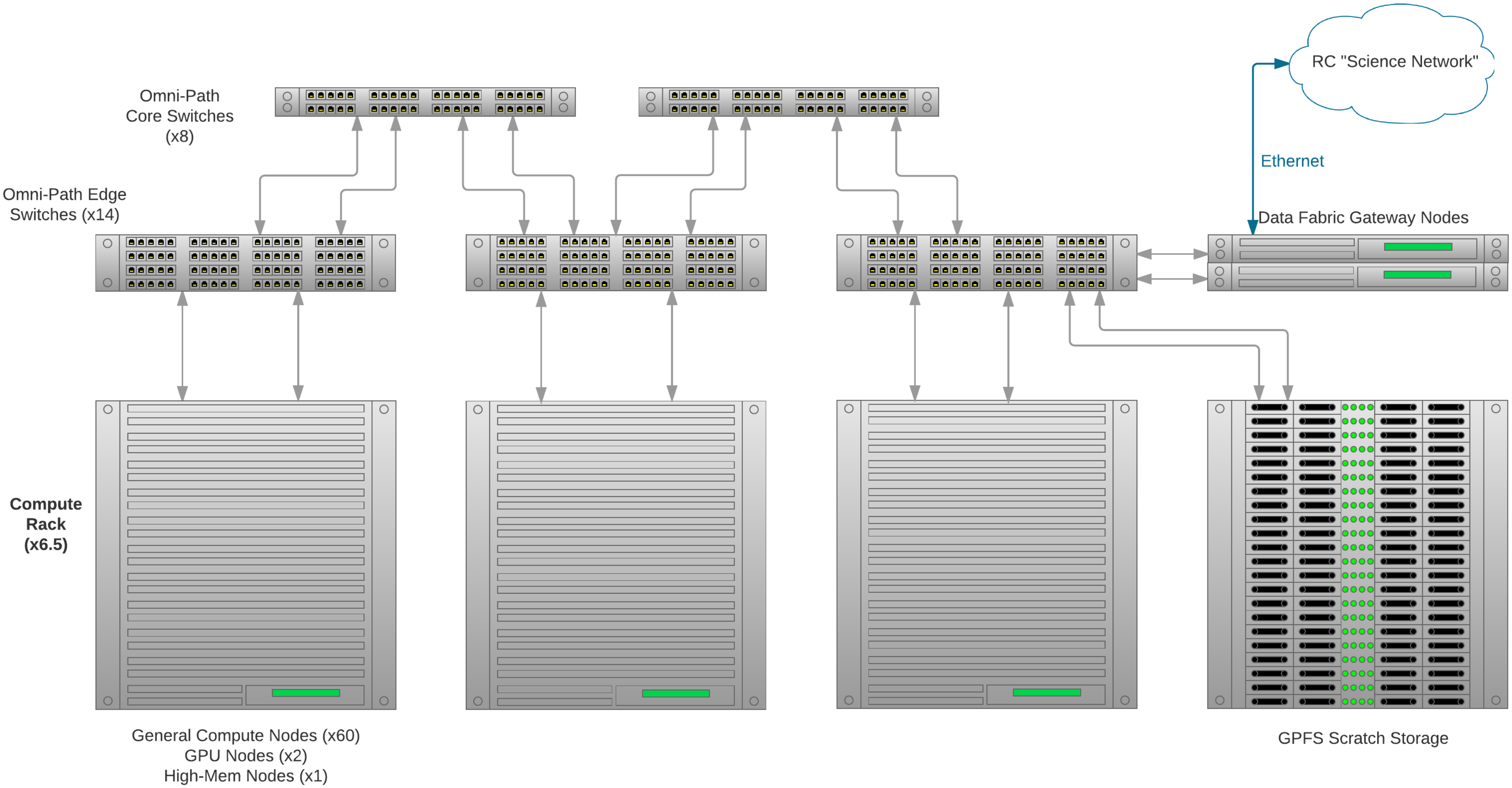
Stampede (TACC)

6400 nodes
- 2 sockets per node, 8 cores per socket

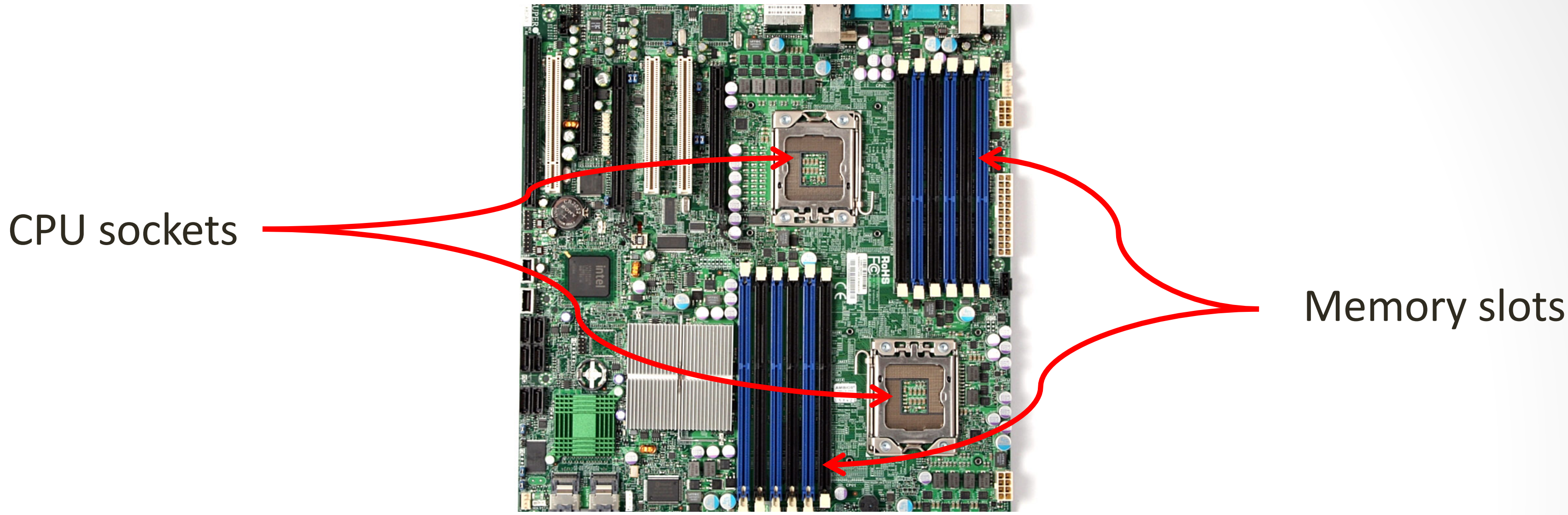
56 Gbps Infiniband interconnect
14 PB parallel storage system

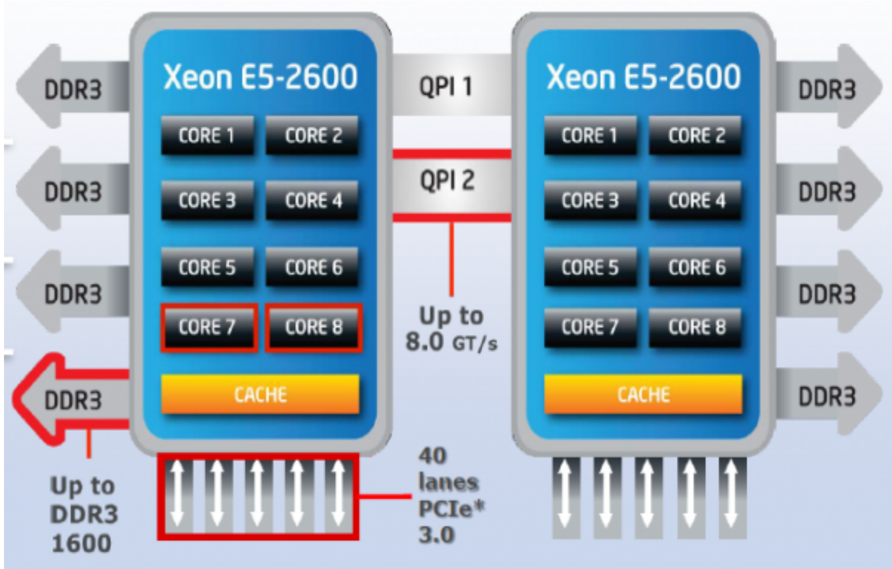
# SUMMIT SCHEMATIC

Peter Ruprecht | July 15, 2016



# Nodes and Processors




Images: Supermicro, Intel

# Host Processor Types

- x86_64 (Intel, AMD)
  - Most likely to encounter
  - General purpose
- Xeon Phi (Intel)
- POWER (IBM)
  - May have tight integration with coprocessor
- ARM
  - RISC, means fewer transistors
  - Low power

All processors operate at a certain frequency (several GHz) and can perform several instructions per cycle.

# Computer Bus Layout

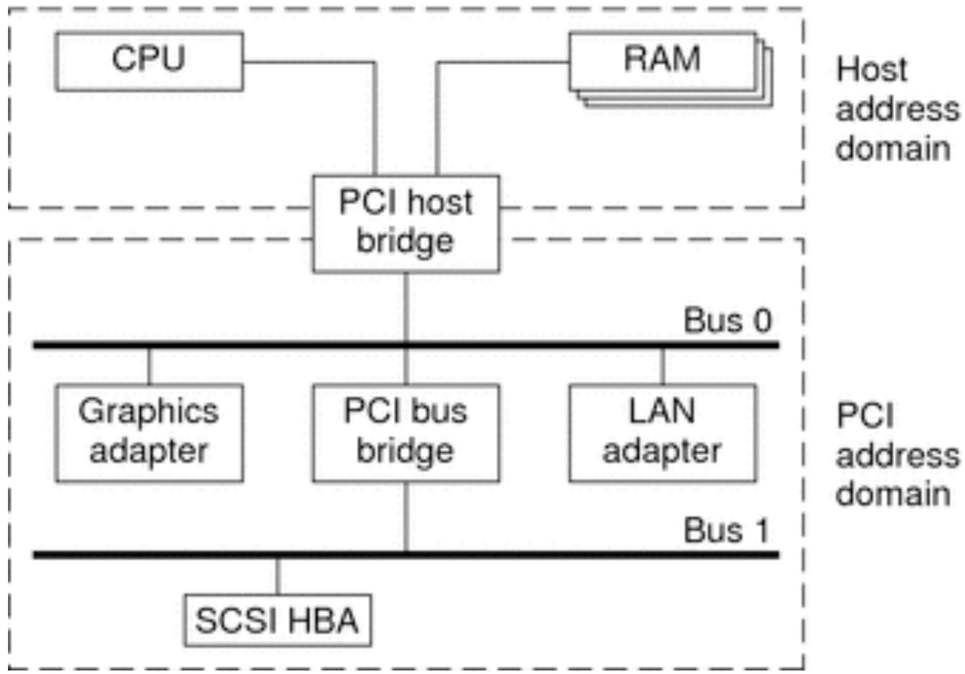



Image: Oracle

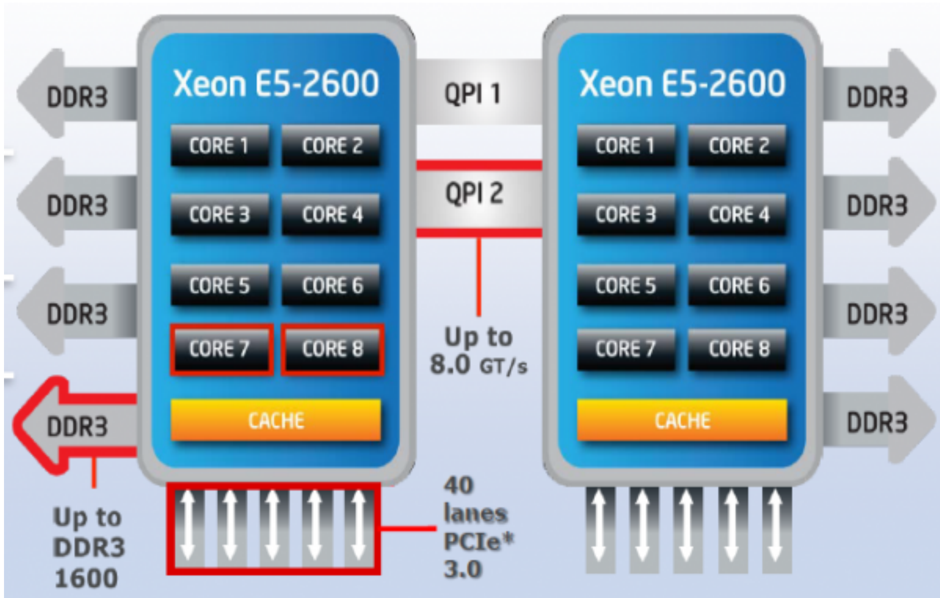
# Memory

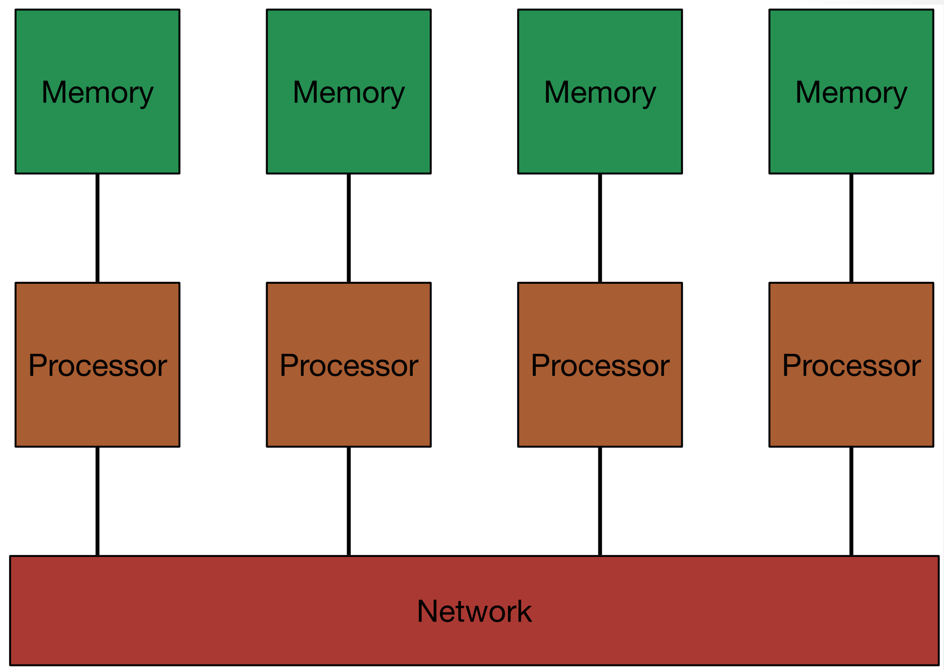
- Holds data that is being calculated on, as well as computational instructions
- Closest memory to CPU is actually on the processor die
  - Cache: Level 1 and 2 are dedicated to a single core
  - Level 3 is larger and shared between all cores in a socket
- Next closest is RAM
- Can also access memory on other nodes, via RDMA (remote direct memory access)

*Shared memory* is local to one node and several process threads can share the same data addresses.

*Distributed memory* is on multiple nodes and each process normally has its own copy or part of the data.

# Shared and Distributed Memory



# Interconnect

- How to access distributed memory across nodes?
- Need a fast, *low-latency* network.
- Current interconnect technologies
  - InfiniBand (40, 56, 100 Gbps)
  - Ethernet (10, 40, 100 Gbps)
  - Aries/Gemini (Cray)
  - OmniPath (100 Gbps)

- RDMA normally requires special interconnect-aware libraries; frequently these are included with MPI

Interconnect is what makes a bunch of nodes into a supercomputer!

# Storage

Different types of “disk” for different needs

- Local disk in the node, often SSD
- Shared scratch
  - Often accessed over cluster interconnect
  - Parallel filesystems, eg Lustre or GPFS
  - Traditionally tuned for high bandwidth, not high IOPS
  - May have a “burst buffer” layer in front of it
  - Short-term storage only!!
- Longer-term or archive
  - Often uses Hierarchical Storage Management

# CPU pipeline

# Optimizing for Data Access

- Page Fault, file on IDE disk: 1,000,000,000 cycles
- Page Fault, file in buffer cache: 10,000 cycles
- Page Fault, file on ram disk: 5,000 cycles
- Page Fault, zero page: 3,000 cycles
- Main memory access: about 200 cycles
- L3 cache hit: about 52 cycles
- L1 cache hit: 2 cycles

**The Core i7 can issue 4 instructions per cycle. So a penalty of 2 cycles for L1 memory access means a missed opportunity for 7 instructions.**

# Co-Processors

Specialized, usually massively multi-core, separate from host processor (CPU).

- GPU (Graphics Processing Unit, “video card”)
  - Thousands of cores
  - Great for vectorizable or embarrassingly-parallel apps
  - Programming frameworks include CUDA, OpenACC, OpenCL
  - Many applications (eg, Matlab) support CUDA directly
- Xeon Phi (being phased out)
  - Dozens of cores
  - Existing x86 code can be directly recompiled to run on Phi
  - Function offload

# Operating Systems in HPC

Operating system: software that manages computer hardware and the processes running on it.

- MS Windows – hasn't really gotten a foothold
- Mac OSX – even less of a factor
- Unix / Linux – some variation will be on virtually any HPC system you use
  - May be a stripped-down version optimized for the particular hardware, or
  - May be a full distribution

# What is Linux?

- Part of the Unix family of operating systems.
- Started in early '90s by Linus Torvalds.
- Technically refers only to the kernel; software from the GNU project and elsewhere is layered on top to form a complete OS. Most is open source.
- Several full distributions are available – from enterprise-grade, like RHEL or SUSE, to more consumer-focused, like Ubuntu.
- Runs on everything from embedded systems to supercomputers.

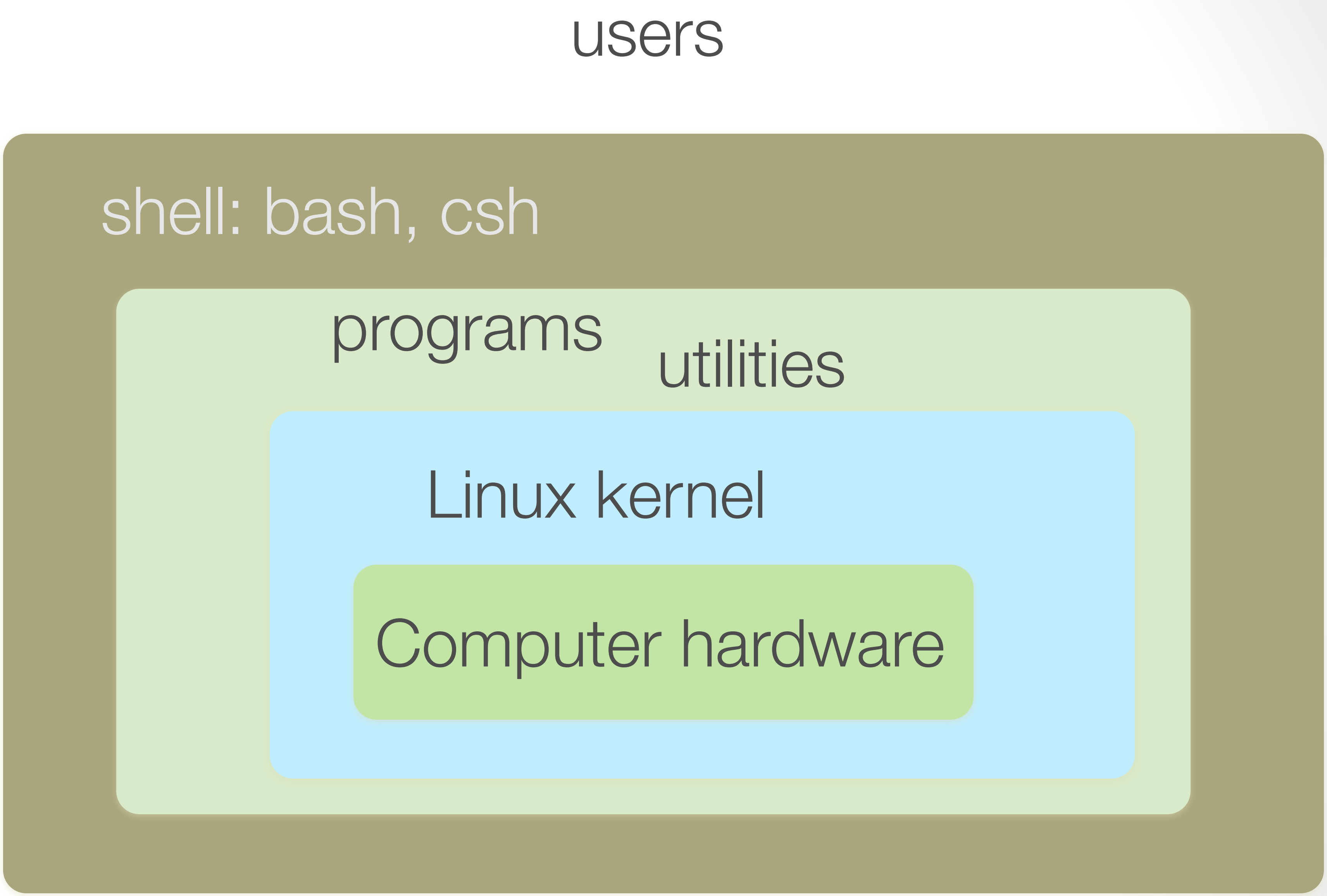
# Why use Linux?

- Linux command-line syntax may seem overwhelming to the new user, but:
- It's the default operating system on virtually all HPC systems
- It's extremely flexible
- It tries not to get in your way
- It's fast and powerful
- Open-source scientific applications are developed predominantly for Linux

- You can get started with a few basic commands and build from there

# Alternatives to the Command Line

- Domain-specific portals
  - CIPRES
  - SEAgrid
  - Many, many others
- Notebooks
  - Jupyter (“Julia, Python, R” , but many others)
- Domain-independent portal environments
  - Sandstone HPC
  - Viewpoint


users
shell: bash, csh
programs
utilities
Linux kernel
Computer hardware

# References


1. Eijkhout, Victor. *Introduction to High-Performance Scientific Computing*, 2015. https://bitbucket.org/VictorEijkhout/hpc-book-and-course/src
2. Hager, G, and G Wellein. *Introduction to High Performance Computing for Scientists and Engineers*, CRC Press, 2010.
3. Levesque, John, and Gene Wagenbreth. *High Performance Computing*. CRC Press, 2010.
4. Neeman, Henry. *Supercomputing in Plain English*, High Performance Computing Workshop Series, http://www.oscer.ou.edu/education.php

# Questions?

- Email rc-help@colorado.edu

- Link to survey on this topic:
  http://tinyurl.com/curc-survey16
  Speaker: Pete Ruprecht
  Title: Computing Basics (July 2017 BSW)

- Slides:
  https://github.com/ResearchComputing/Basics_Supercomputing