SCHOOL OF
HUMAN SCIENCES
& TECHNOLOGY

OPERATING SYSTEMS & PARALLEL COMPUTING

Final Exam Preparation

ie SCHOOL OF HUMAN SCIENCES & TECHNOLOGY

Final Exam

• Parallel Computing	20 Questions
• MPI / OpenMP / PThreads	
• Shared / Distributed / Hybrid Memory	
• Big Data	10 Questions
• GPU / NVIDIA	10 Questions
• Quantum	10 Questions
• HPC	10 Questions
• Cloud Solutions	10 Questions
• AWS, Azure, Google, Oracle, IBM	

SCHOOL OF
HUMAN SCIENCES
& TECHNOLOGY

Questions Examples

What is Parallel Computing?

Answer: Parallel Computing resembles the study of designing algorithms such that the time complexity is minimum. Thus the speed up factor is taken into consideration.

SCHOOL OF HUMAN SCIENCES & TECHNOLOGY

Differentiate between SIMD and MIMD?

Answer: In architectures referred to as single instruction stream, multiple data stream (SIMD), a single control unit dispatches instructions to each processing unit. In an SIMD parallel computer, the same instruction is executed synchronously by all processing units.
Computers in which each processing element is capable of executing a different program independent of the other processing elements are called multiple instruction stream, multiple data stream (MIMD) computers.
SIMD computers require less hardware than MIMD computers because they have only one global control unit. SIMD computers require less memory because only one copy of the program needs to be stored.
In contrast, MIMD computers store the program and operating system at each processor.

Differentiate between UMA and NUMA.

Answer: If the time taken by a processor to access any memory word in the system (global or local) is identical, the platform is classified as a uniform memory access (UMA) multicomputer.
On the other hand, if the time taken to access certain memory words is longer than others, the platform is called a non-uniform memory access (NUMA) multicomputer.

What is Cache Coherence?

Answer: In shared address space platform ensuring that concurrent operations on multiple copies of the same memory word have well-defined semantics is called cache coherence.

What is PRAM Model?

Answer: A Model of computation (the Random Access Machine, or RAM) consists of p processors and a global memory of unbounded size that is uniformly accessible to all processors. All processors access the same address space. Processors share a common clock but may execute different instructions in each cycle. This ideal model is also referred to as a parallel random access machine (PRAM).

What is Snoopy cache system?

Answer: Snoopy caches are typically associated with multiprocessor systems based on broadcast interconnection networks such as a bus or a ring. In such systems, all processors snoop on (monitor) the bus for transactions. This allows the processor to make state transitions for its cache-blocks.

What is MPI?

Answer: Message Passing Interface (**MPI**) is a standardized and portable message-passing system designed by a group of researchers from academia and industry to function on a wide variety of parallel computers. The standard defines the syntax and semantics of a core of library routines useful to a wide range of users writing portable message-passing programs in the C programming language.

1. Parallel computing uses _____ execution.

○ sequential

○ unique

○ simultaneous

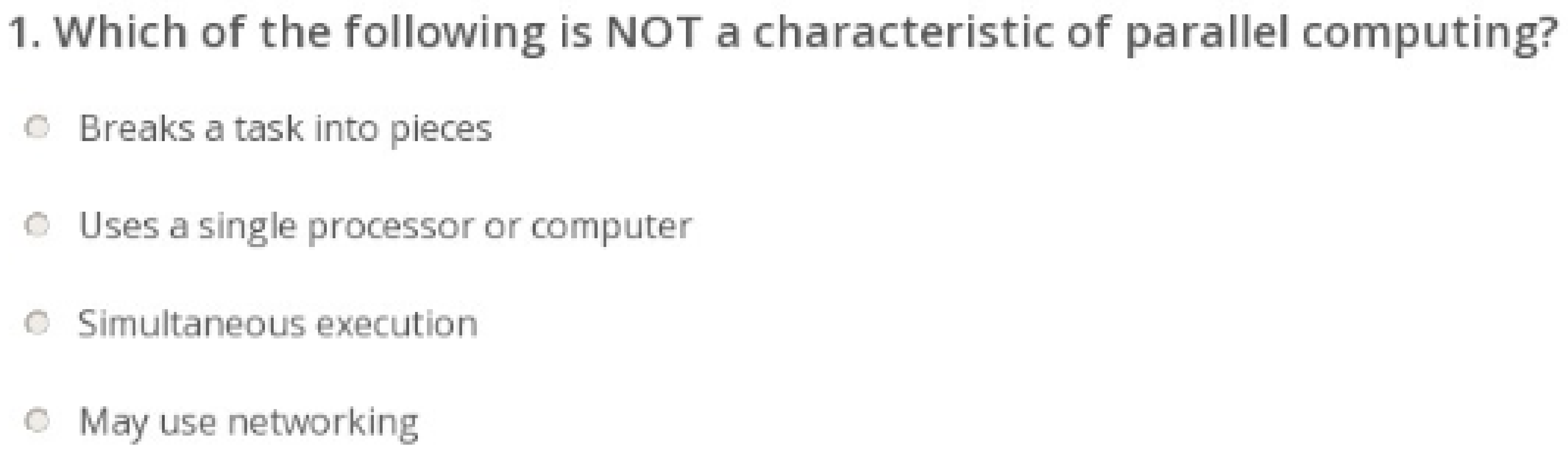
○ none of the answers is correct.

SCHOOL OF
HUMAN SCIENCES
& TECHNOLOGY

2. Which of the following is true about parallel computing performance?

- ○ Computations use multiple processors.
- ○ There is an increase in speed.
- ○ The increase in speed is loosely tied to the number of processor or computers used.
- ○ All of the answers are correct.

SCHOOL OF HUMAN SCIENCES & TECHNOLOGY



1. Which of the following is NOT a characteristic of parallel computing?

○ Breaks a task into pieces

○ Uses a single processor or computer

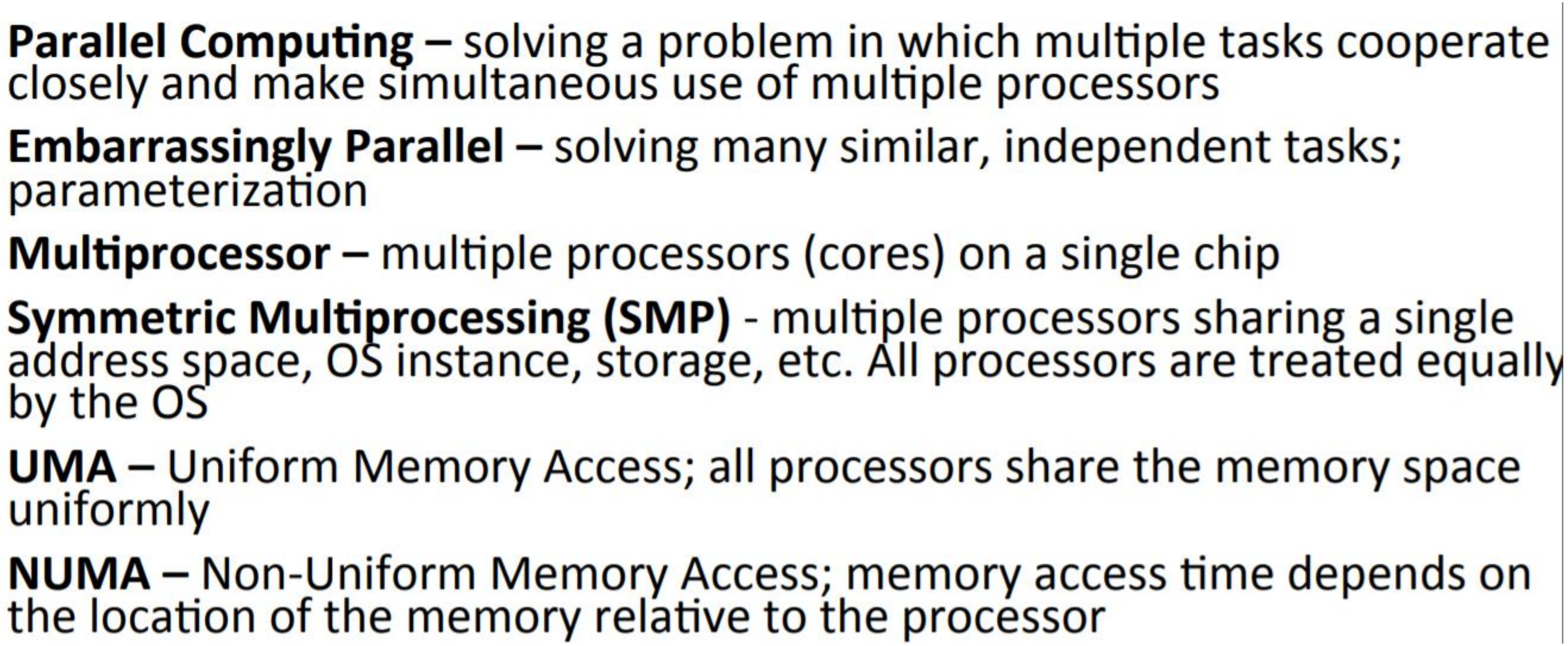
○ Simultaneous execution

○ May use networking

SCHOOL OF
HUMAN SCIENCES
& TECHNOLOGY

PARALLEL COMPUTING

Basic Terminology



Parallel Computing – solving a problem in which multiple tasks cooperate closely and make simultaneous use of multiple processors

Embarrassingly Parallel – solving many similar, independent tasks; parameterization

Multiprocessor – multiple processors (cores) on a single chip

Symmetric Multiprocessing (SMP) - multiple processors sharing a single address space, OS instance, storage, etc. All processors are treated equally by the OS

UMA – Uniform Memory Access; all processors share the memory space uniformly

NUMA – Non-Uniform Memory Access; memory access time depends on the location of the memory relative to the processor

Basic Terminology

Cluster Computing – A parallel system consisting of a combination of commodity units (processors or SMPs)

Capacity Cluster – A cluster designed to run a variety types of problems regardless of the size; the goal is to perform as much research as possible

Capability Cluster – The fastest, largest machines designed to solve large problems; the goal is to demonstrate capability of the parallel system

High Performance Computing – Solving large problems via a cluster of computers that consist of fast networks and massive storage

Other Terminology

Instruction-Level Parallelism (ILP) - Improves processor performance by having multiple processor components simultaneously executing instructions

Pipelining – Breaking a task into steps performed by different processors with inputs streaming through, much like an assembly line

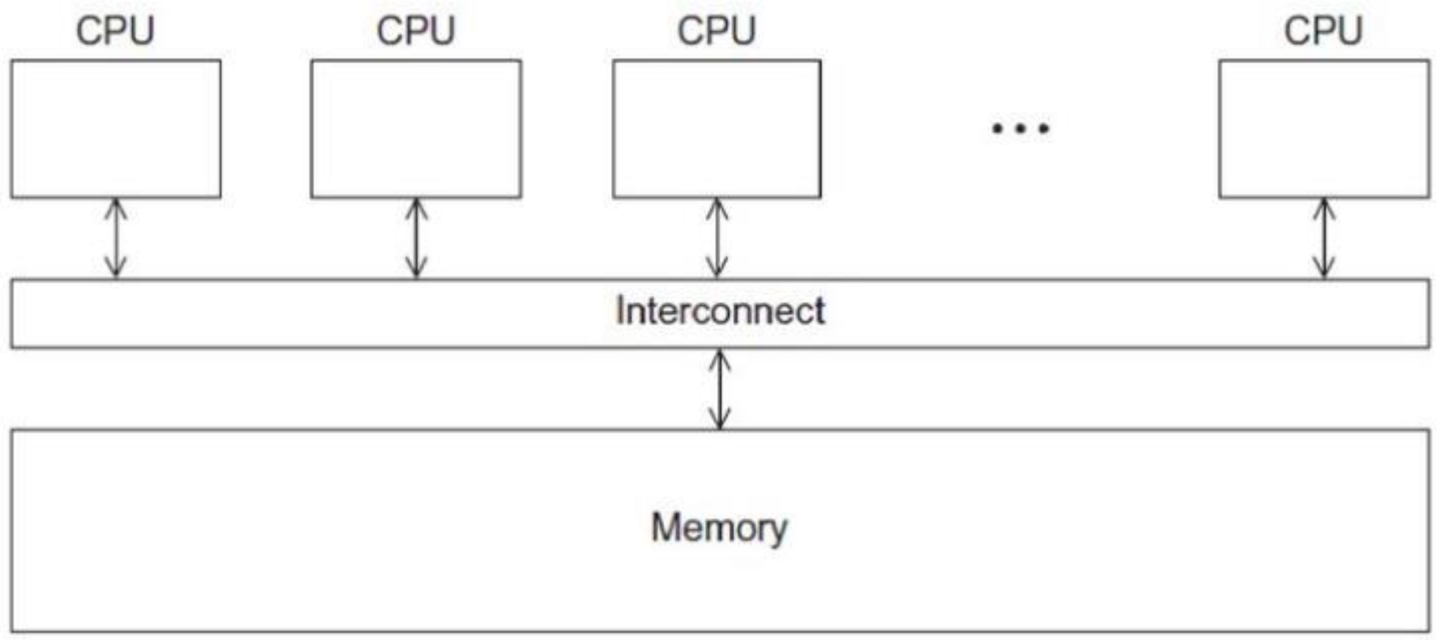
Superscalar Execution - The ability for a processor to execute more than one instruction during a clock cycle by simultaneously dispatching multiple instructions to different execution units on the processor

Cache Coherence - Synchronization between local caches for each processor

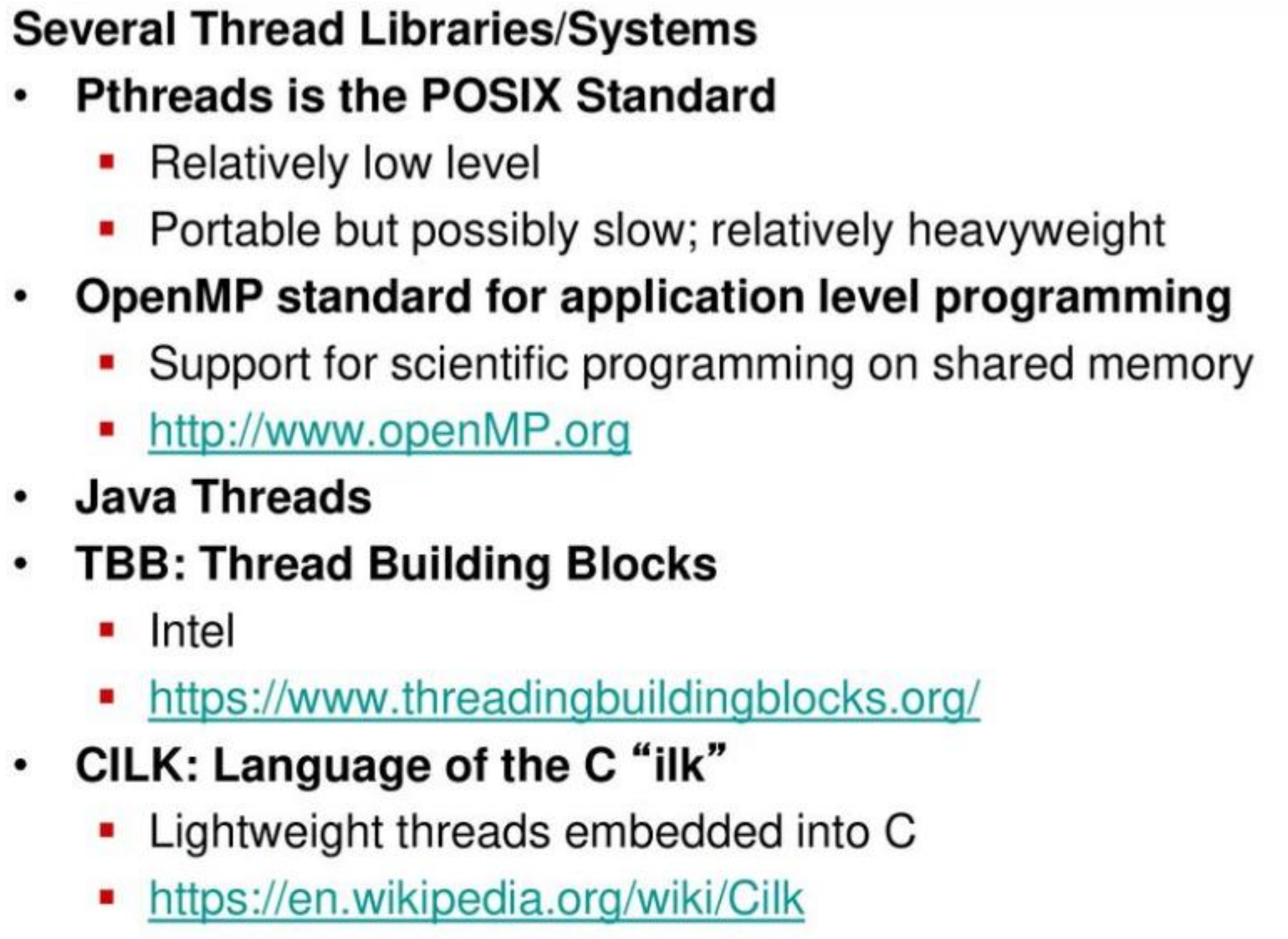
Models for Parallel Computation

- Shared memory:
  - Load
  - Store
  - lock
  - unlock
- Message Passing:
  - Send
  - Receive
  - Broadcast
  - …

Shared Memory Architecture



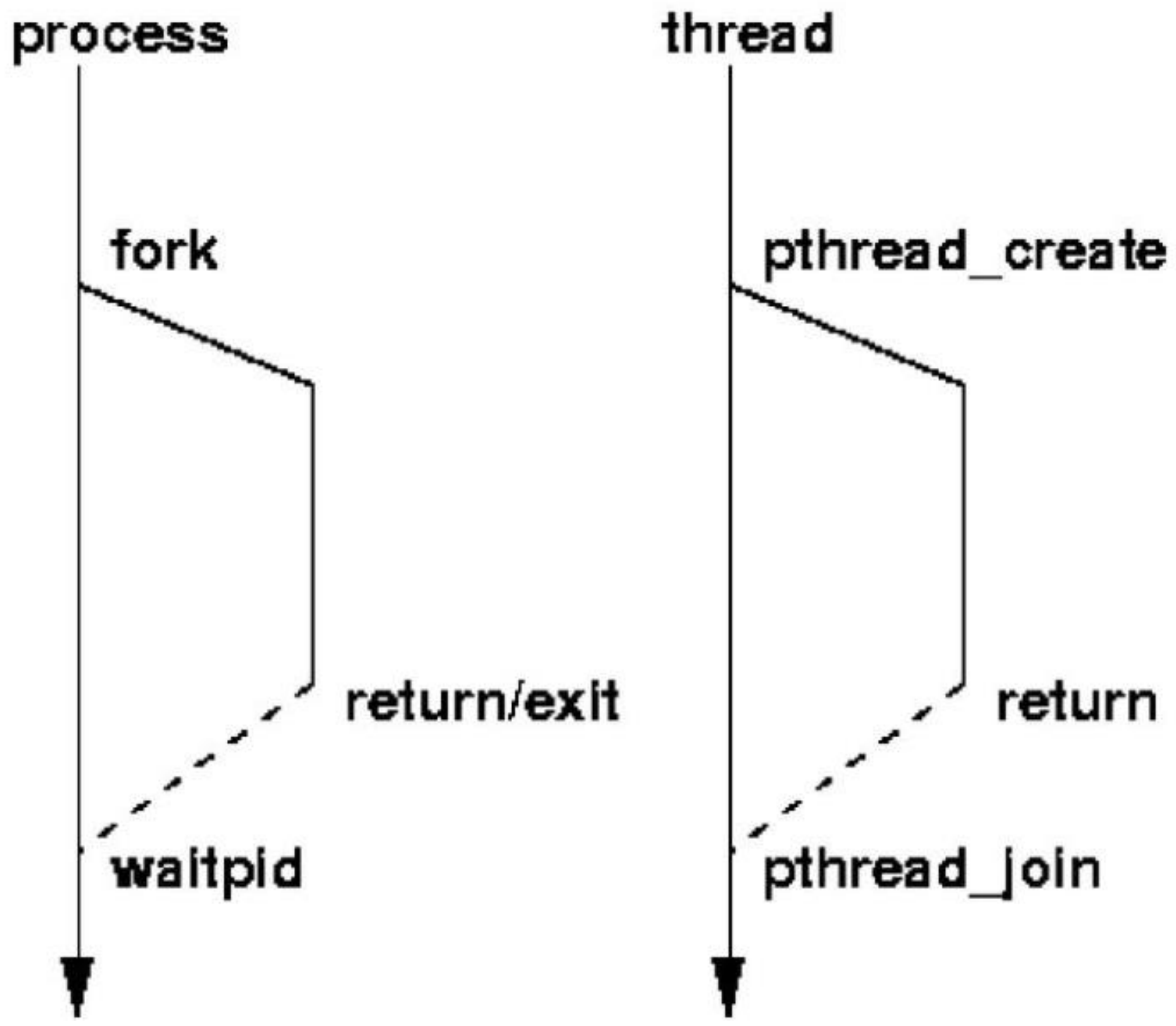
Shared Memory Programming



Several Thread Libraries/Systems

- **Pthreads is the POSIX Standard**
  - Relatively low level
  - Portable but possibly slow; relatively heavyweight
- **OpenMP standard for application level programming**
  - Support for scientific programming on shared memory
  - http://www.openMP.org
- **Java Threads**
- **TBB: Thread Building Blocks**
  - Intel
  - https://www.threadingbuildingblocks.org/
- **CILK: Language of the C "ilk"**
  - Lightweight threads embedded into C
  - https://en.wikipedia.org/wiki/Cilk

Unix Processes vs PThreads



OPENMP Goals

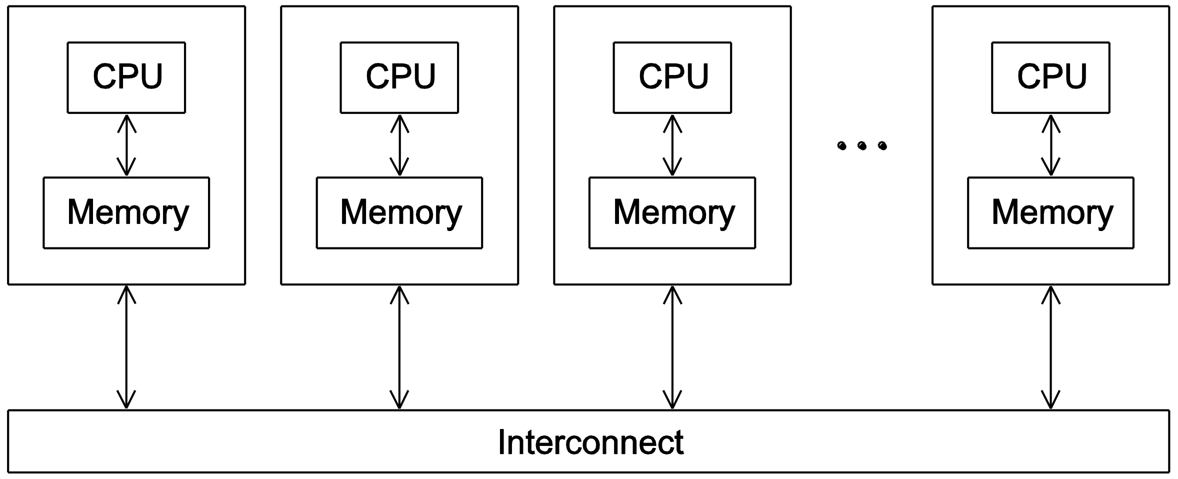
Standardization: standard among all shared memory architectures and hardware platforms

Lean: simple and limited set of compiler directives for shared memor programming. Often significant performance gains using just 4-6 directives in complex applications.

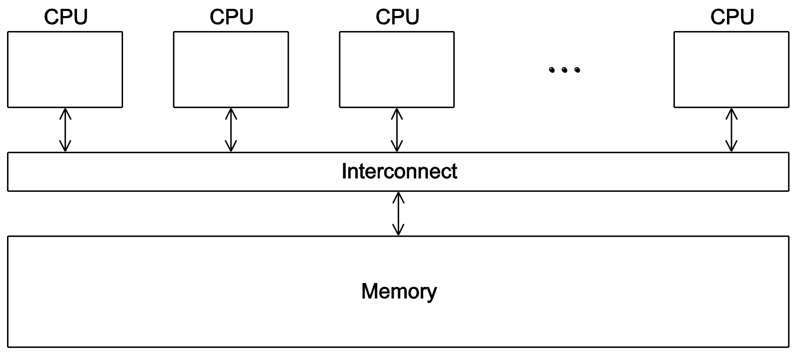
Ease of use: supports incremental parallelization of a serial program, not an all-or-nothing approach.

Portability: supports Fortran, C, and C++

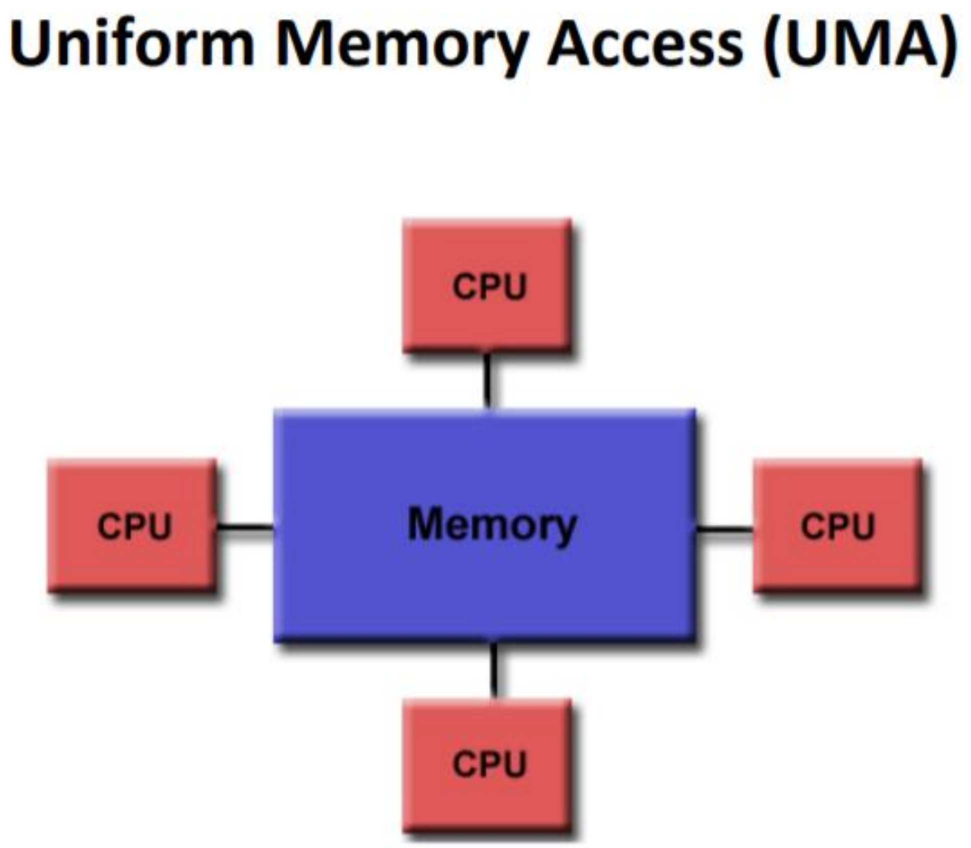
Mainly for distributed memory systems

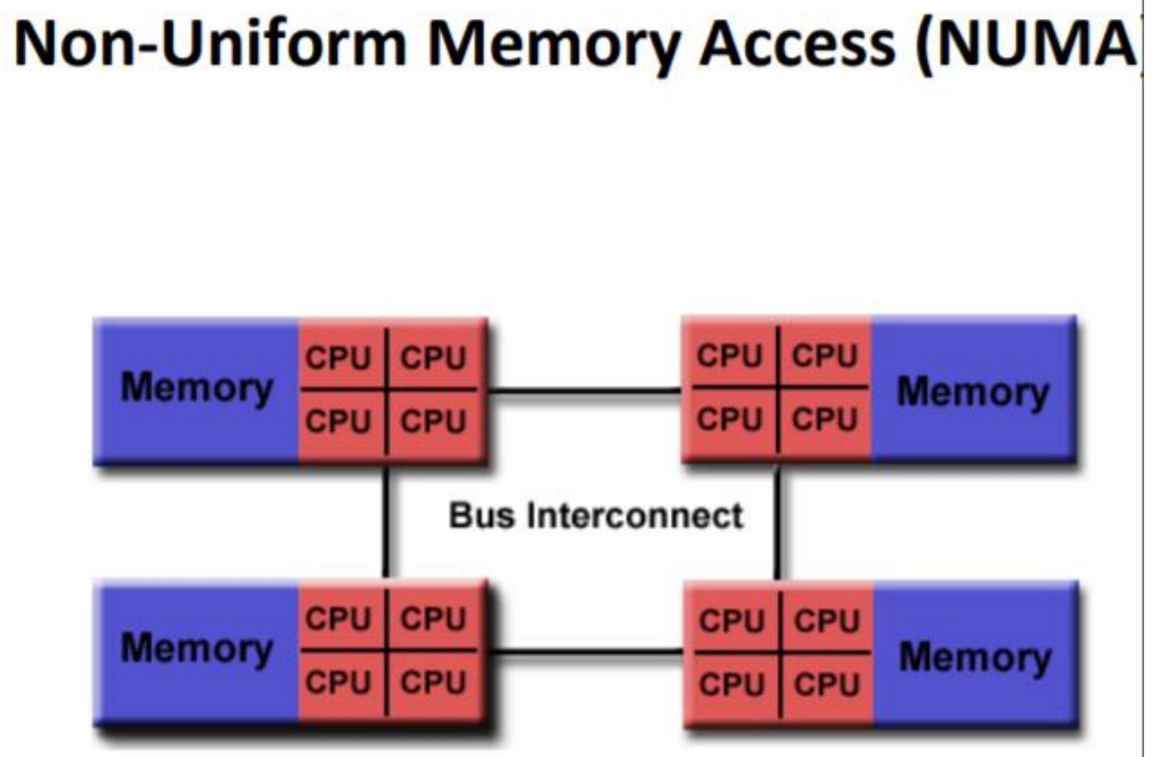


Not targeted for shared memory machines. But can work

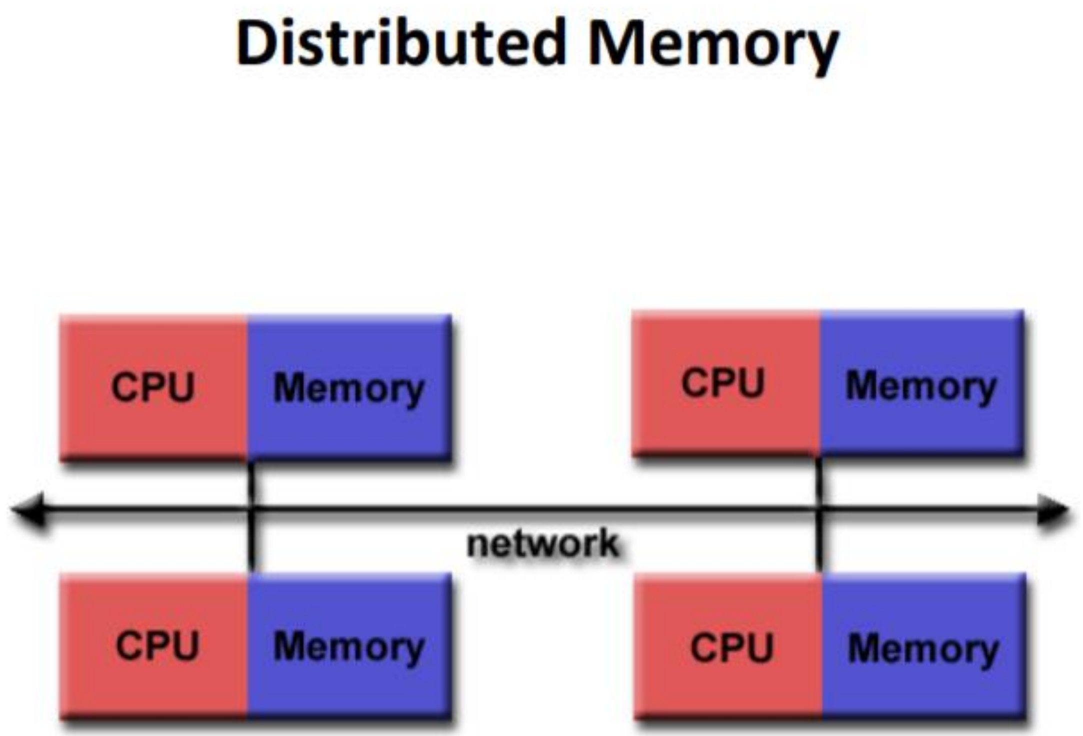


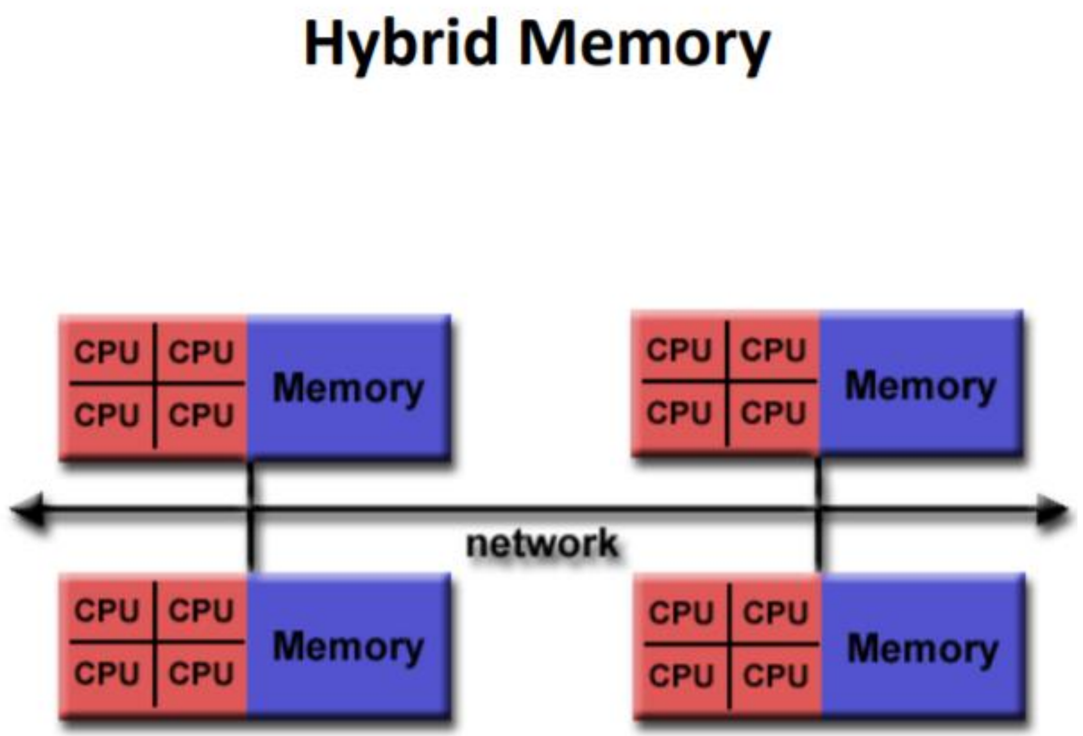
MIMD Architectures (Shared Memory)





More MIMD Architectures





MPI (Message Passing Interface)

- A standard message passing specification for the vendors to implement

- Context: distributed memory / parallel computers
  - Each processor has its own memory and cannot access the memory of other processors
  - Any data to be shared must be explicitly transmitted from one to another

- Most message passing programs use the *single program multiple data (SPMD)* model
  - Each processor executes the same set of instructions
  - Parallelization is achieved by letting each processor operate a different piece of data
  - MIMD (Multiple Instructions Multiple Data*)*

Why MPI

- Distributed memory (shared nothing) systems
  - With MPI, it is:
    - Common,
    - easier to build,
    - dominate high-end computing (most highest-performing applications)
    - etc
- Performance depends on managing memory use
  - Goal of many parallel programming models is to simplify programming by hiding details of memory locality and management (parallel programming for the masses)
- Support for modular programming

Why MPI?

- Small
  - Many programs can be written with only 6 basic functions
- Large
  - MPI’s extensive functionality from many functions
- Scalable
  - Point-to-point communication
- Flexible
  - Don’t need to rewrite parallel programs across platforms

What is MPI?

- A Message-Passing library specification
  - extended message-passing model
  - not a language or compiler specification
  - not a specific implementation or product
- Used For:
  - parallel computers,
  - clusters,
  - and heterogeneous networks
- Full-featured
  - Designed to provide access to advanced parallel hardware for
    - end users,
    - library writers,
    - and tool developers

Message Passing Features

- Parallel programs consist of separate processes, each with its own address space
    - It is the responsibility of the programmer to manage memory by placing data in a particular process
- Data Sent/Received explicitly between processes
    - Programmer manages memory motion
- Collective operations
    - On arbitrary set of processes work together to solve a problem
- Data distribution
    - also managed by programmer
        - Message passing model doesn’t get in the way
        - It doesn’t help either ☺

Message Passing Libraries

- MPI : Message Passing Interface,
  - now the industry standard, for C/C++ and other languages
- **Running as a <u>set of processes</u>. No shared variables**
- All communication, synchronization require subroutine calls (function call)
  - Enquiries
    - How many processes?
    - Which one am I?
    - Any messages waiting?
  - Communication
    - point-to-point: **`Send`** and **`Receive`**
    - Collectives such as broadcast
  - Synchronization
    - Barrier

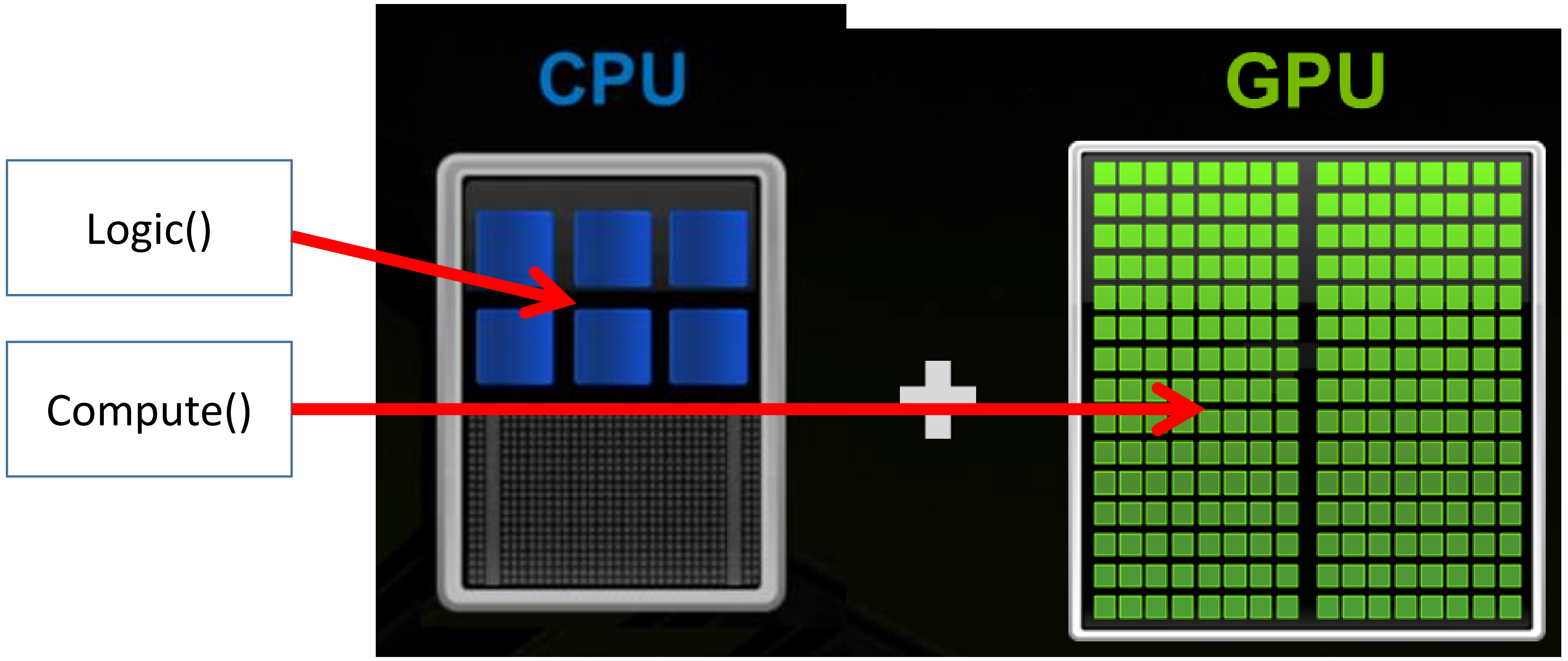
Advanced Features of MPI

- Communicators:
  - encapsulate communication spaces for library safety
- Datatypes:
  - reduce copying costs and permit heterogeneity
- Multiple communication modes
  - allow precise buffer management
- Extensive collective operations
  - allow for scalable global communication
- Process topologies
  - permit efficient process placement,
  - user views of process layout
- Profiling interface
  - encourages portable tools

Communicator

- An identifier associated with a group of processes
  - Each process has a unique rank within a specific communicator from 0 to (nprocesses-1)
  - Always required when initiating a communication by calling an MPI function
- Default: MPI_COMM_WORLD
  - Contains all processes
- Several communicators can co-exist
  - A process can belong to different communicators at the same time

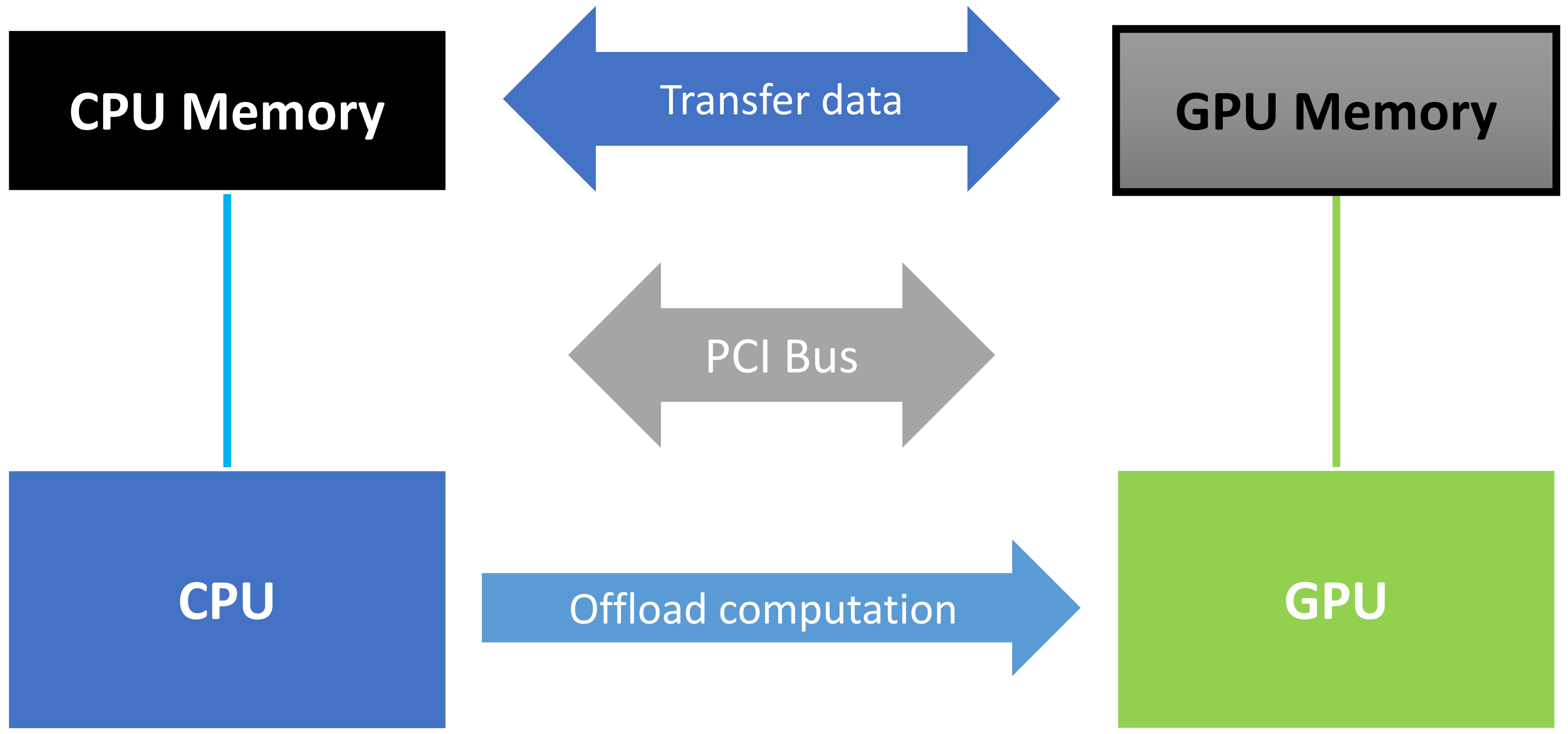
Heterogeneous Parallel Computing with GPU



Latency-Optimized
Fast Serial Processing

Throughput-Optimized
Fast Parallel Processing

Basic Concepts



GPU computing is all about 2 things:

- Transfer data between CPU-GPU
- Do parallel computing on GPU

Programming GPUs

Low-Level Programming

- Proprietary – NVidia's CUDA
- Portable – OpenCL

High-Level Programming

- OpenACC
- OpenMP 4.5
- Allows for a single code base that may be compiled on both multicore and GPUs

OpenACC and OpenMP

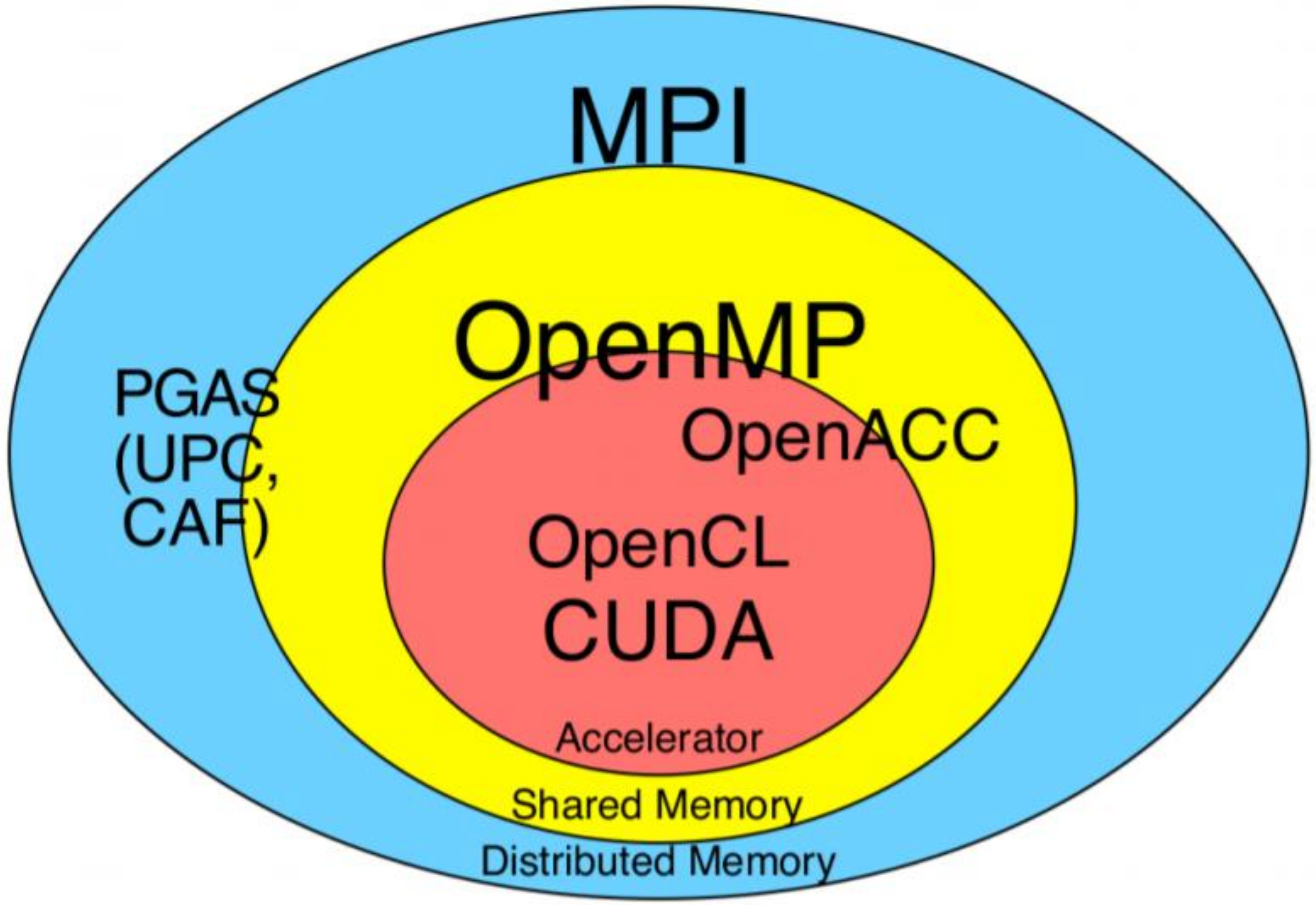
Share the same model of computation

Host-centric – host device offloads code regions and data to accelerators

Device – has independent memory and multiple threads

Mapping clause – Defines the relationship between memory on the host device and memory on the accelerator

Conclusions



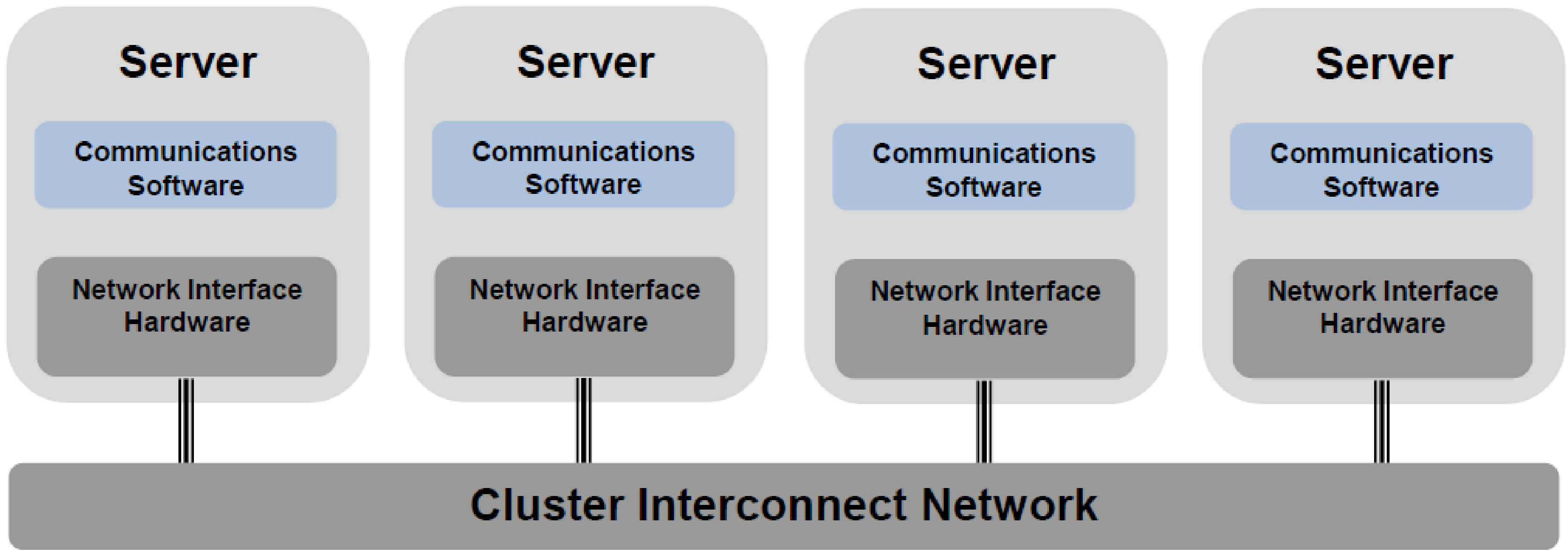
SCHOOL OF
HUMAN SCIENCES
& TECHNOLOGY

HPC

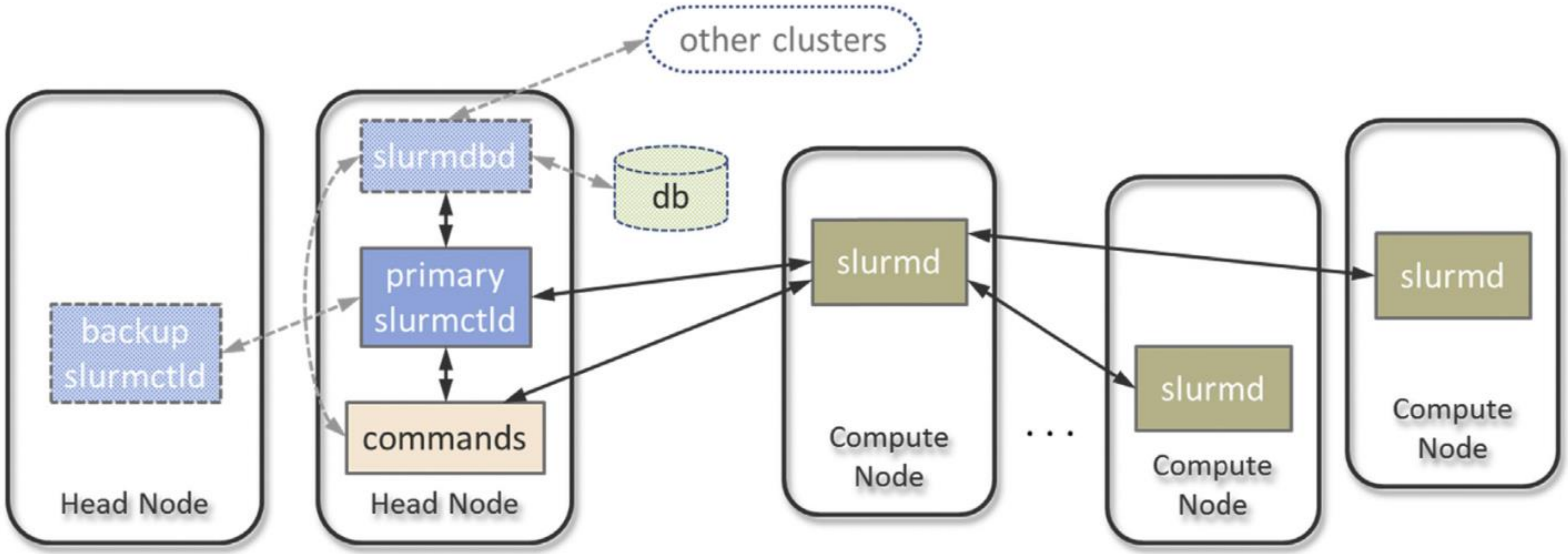
HPC Clusters: Affordable, Efficient, Scalable

- **Since the 1990s, there has been an increasing trend to move away from expensive /specialized proprietary parallel supercomputers to clusters of computers**
  - From specialized supercomputers to cost effective, general purpose systems
- **So What's So Different about Clusters?**
  - Commodity, standard, affordable, cost effective, scalable and reliable architecture

Cluster Architecture


Server
Communications Software
Network Interface Hardware
Server
Communications Software
Network Interface Hardware
Server
Communications Software
Network Interface Hardware
Server
Communications Software
Network Interface Hardware
Cluster Interconnect Network

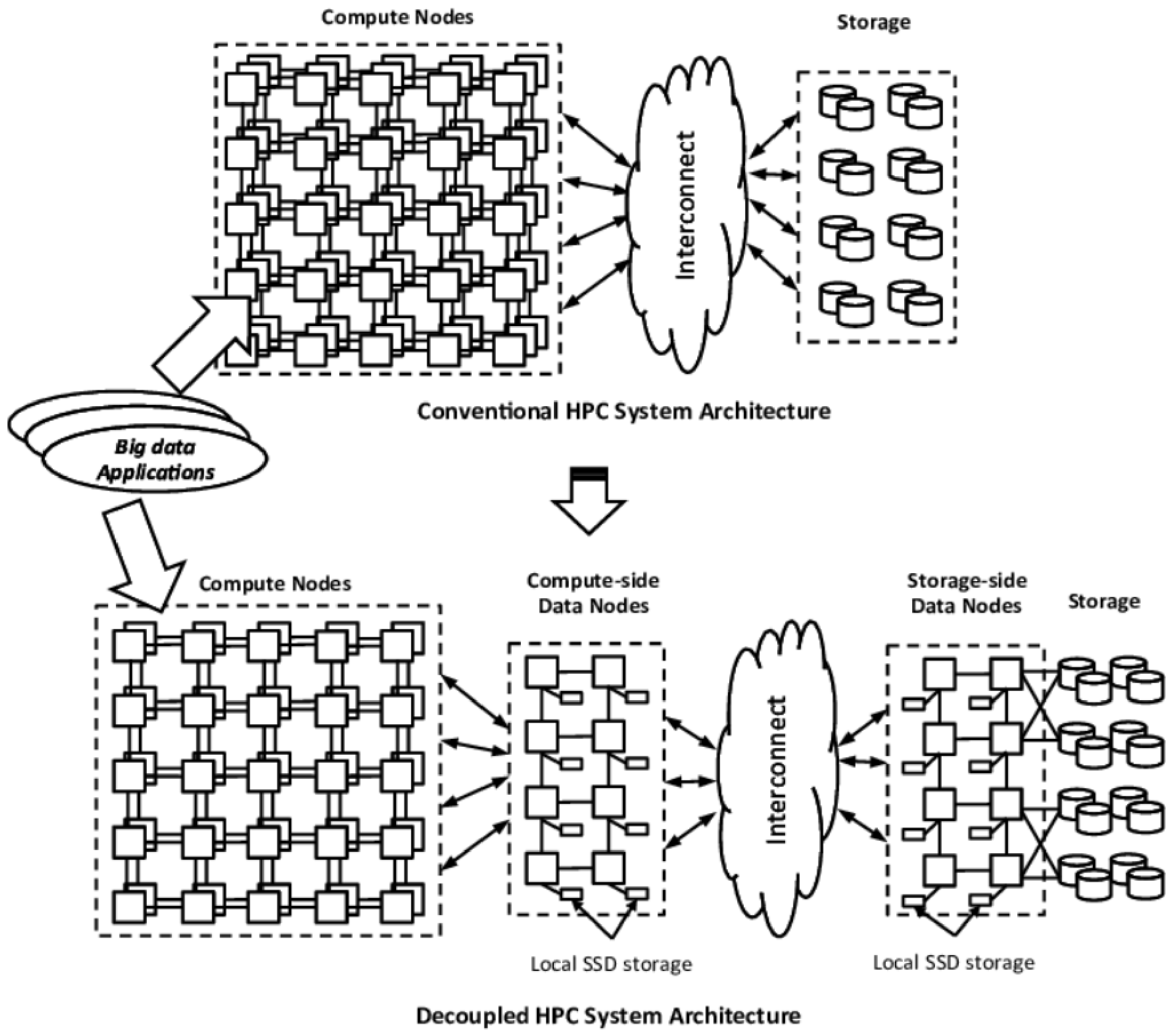
Simplified architecture of SLURM.



Components framed by dashed lines are optional.

SCHOOL OF HUMAN SCIENCES & TECHNOLOGY

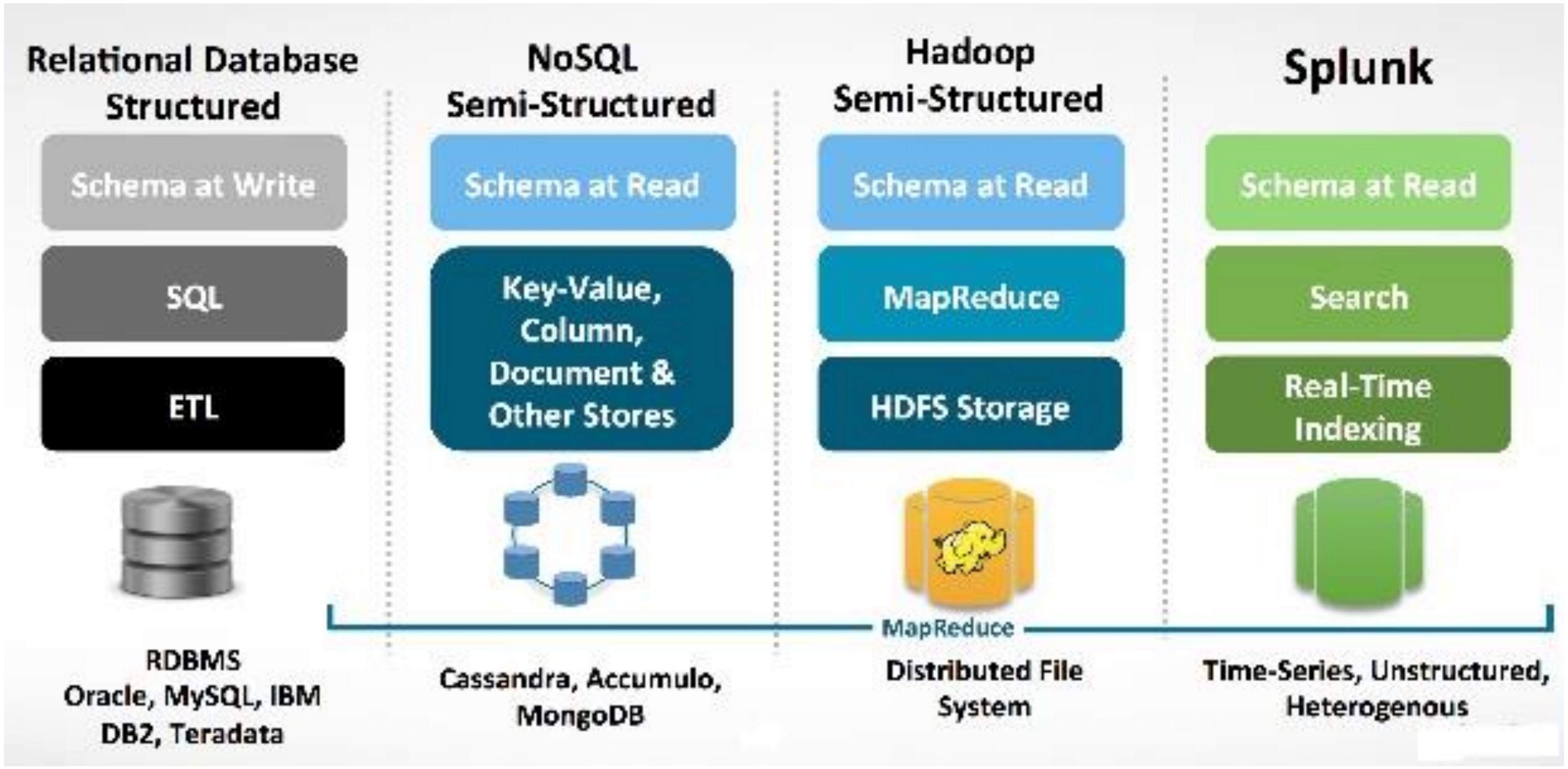
Decoupled High Performance Computing System Architecture



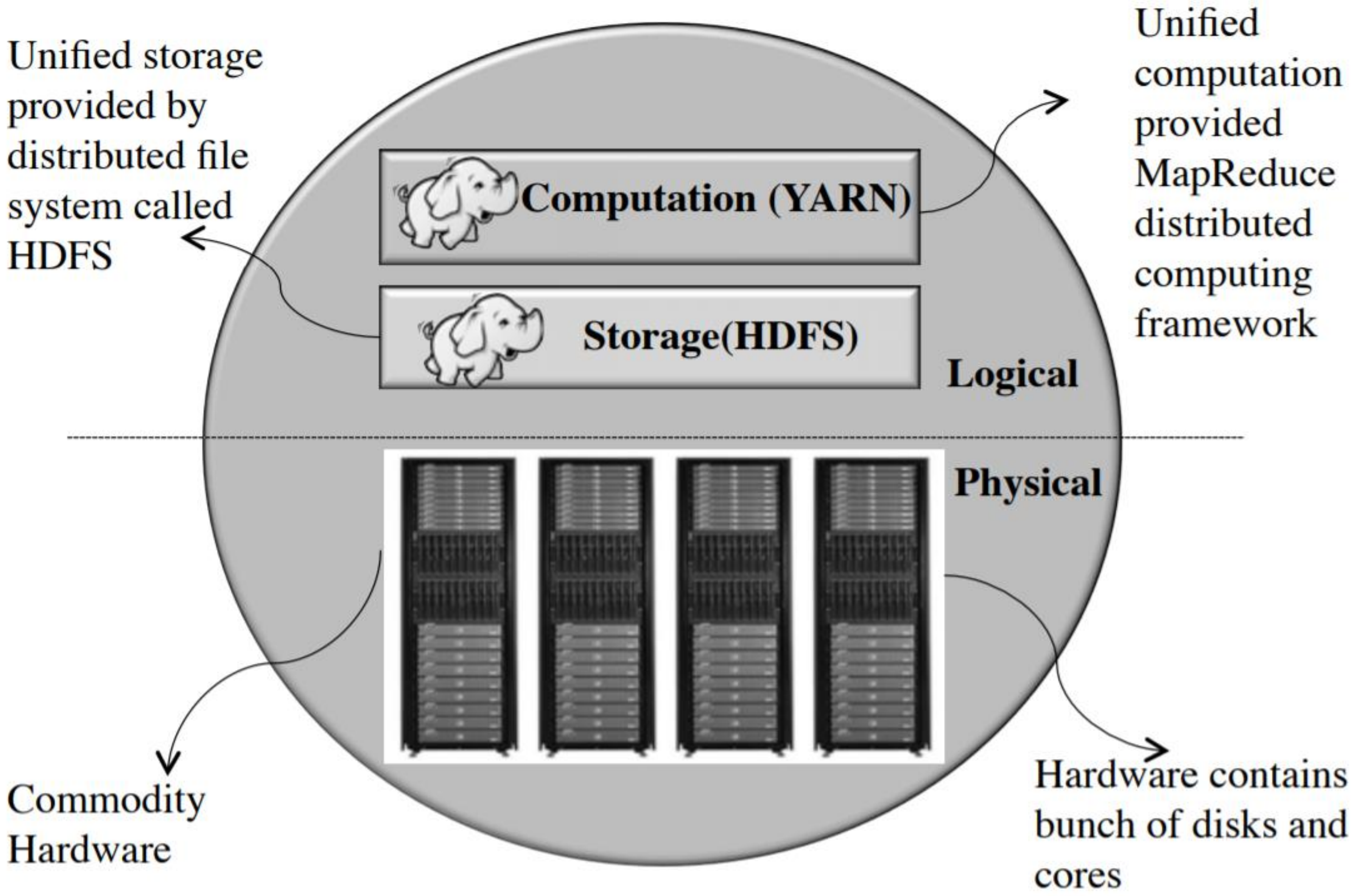
SCHOOL OF
HUMAN SCIENCES
& TECHNOLOGY

BigData

Big Data Technologies

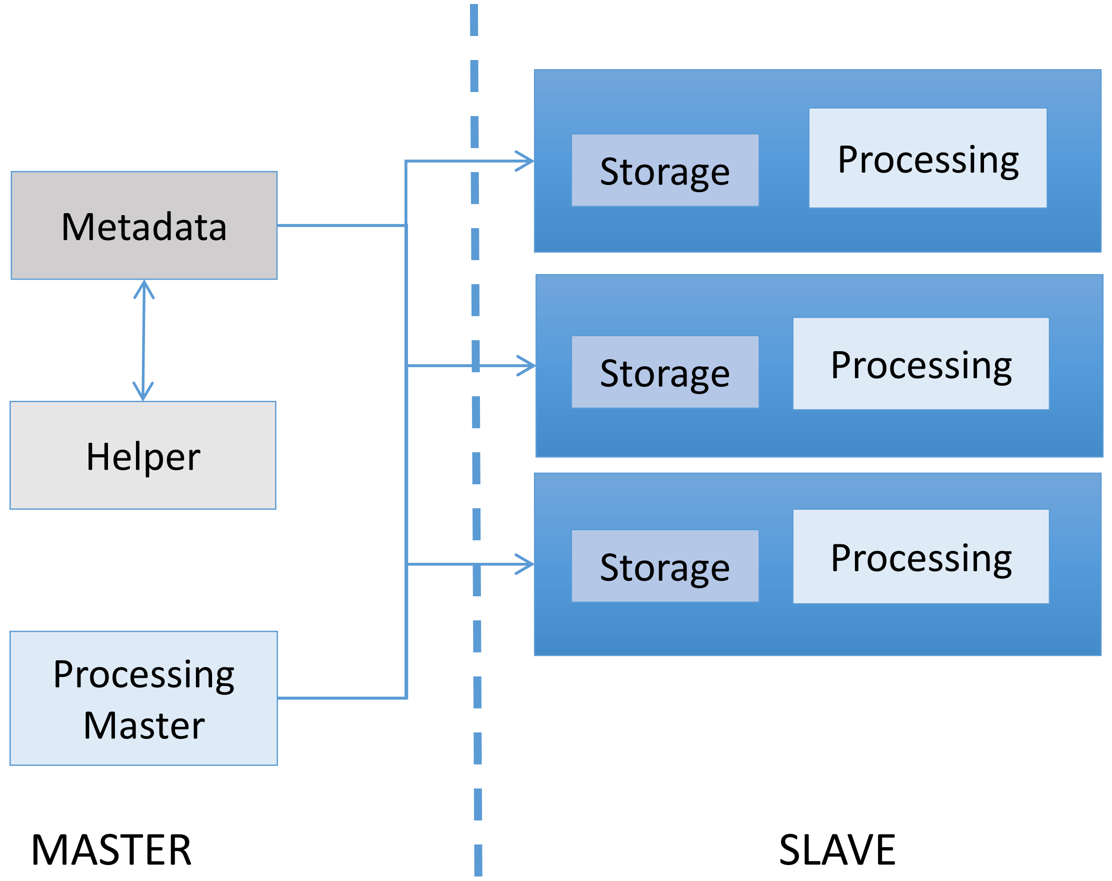


Hadoop – Big Picture


Unified storage provided by distributed file system called HDFS
Computation (YARN)
Storage(HDFS)
Unified computation provided MapReduce distributed computing framework
Logical
Physical
Commodity Hardware
Hardware contains bunch of disks and cores

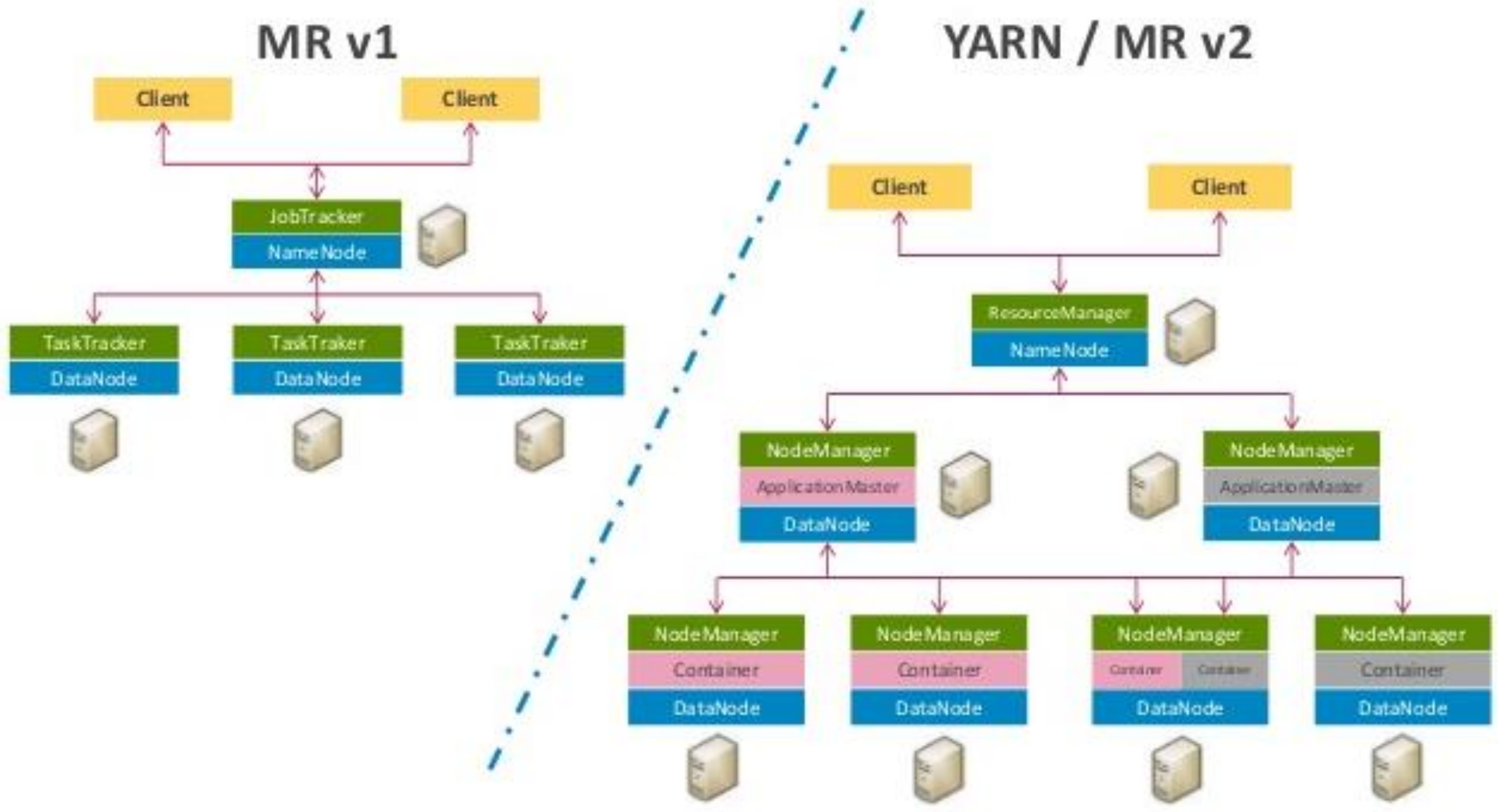
Hadoop Architecture

Master is a **high-end machine** where as slaves are inexpensive computers. The Big Data files get divided into the **number of blocks**. Hadoop stores these blocks in a distributed fashion on the cluster of slave nodes. On the master, we have **metadata stored**.

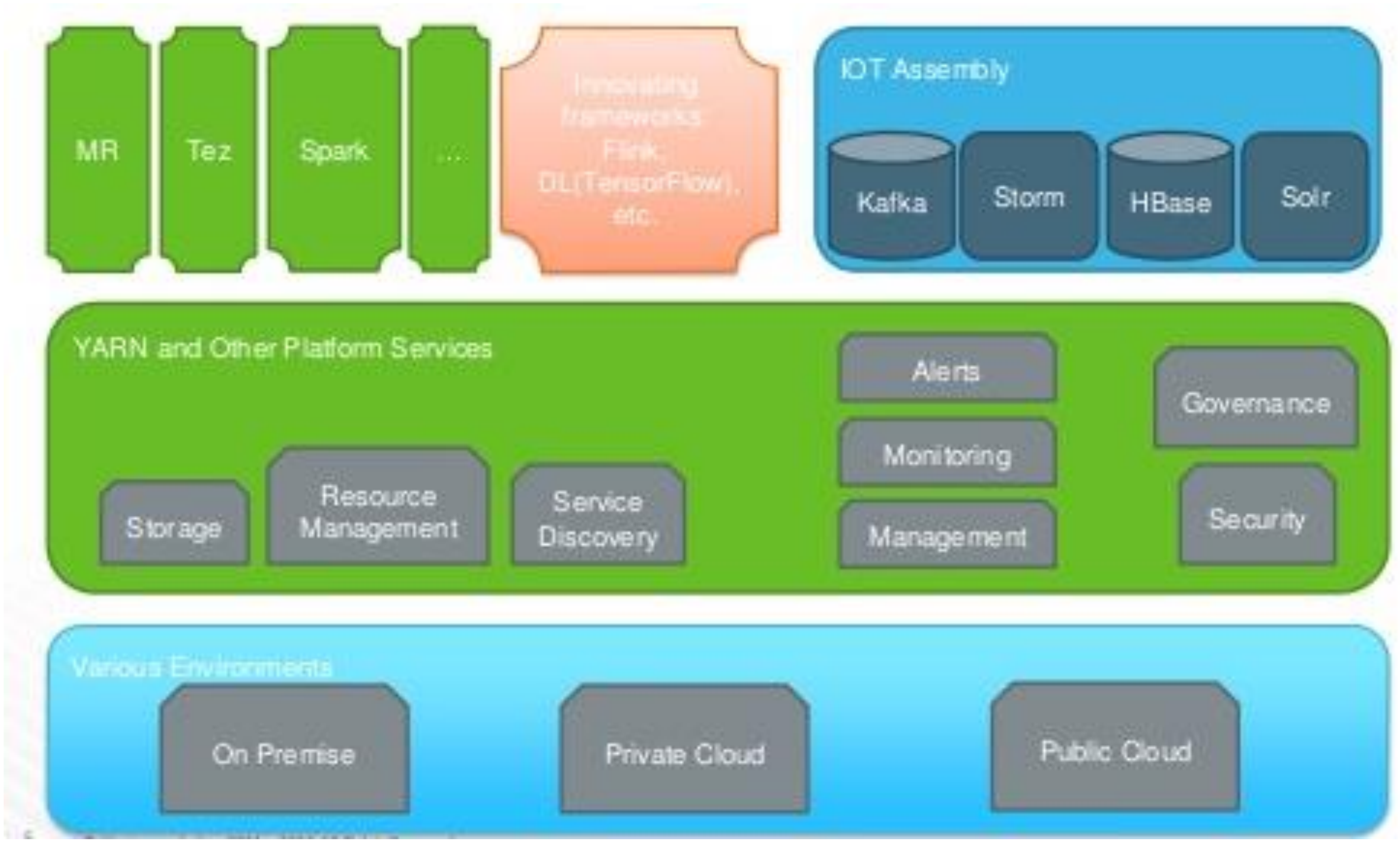


MR vs YARN Architecture

- YARN
  - Yet Another Resource Negotiator
- MR
  - MapReduce



Hadoop 3.0 Architecture



SCHOOL OF HUMAN SCIENCES & TECHNOLOGY

Quantum Computer

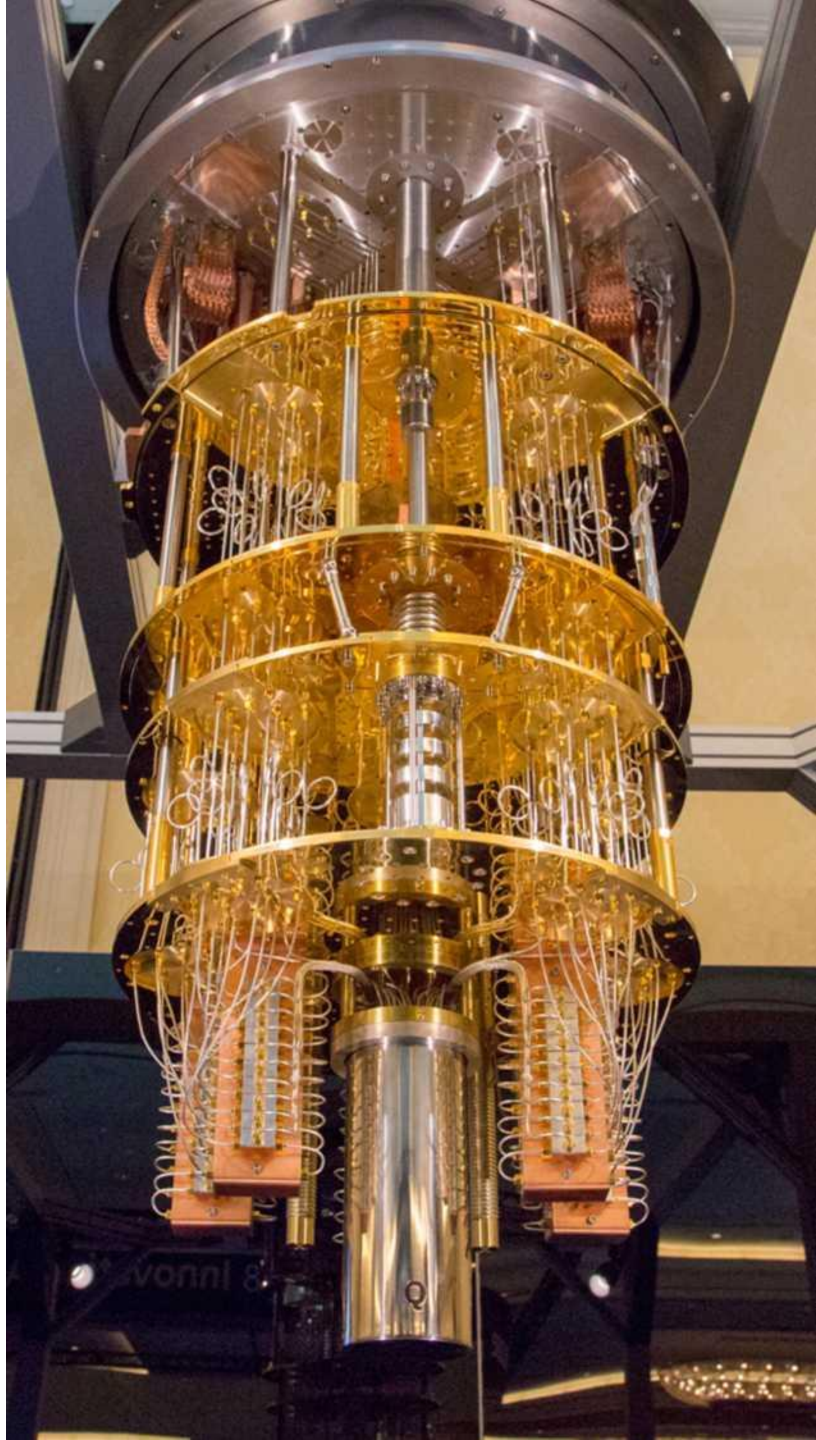
WHAT IS QUANTUM COMPUTING?

Quantum computer is not a faster computer!

In a quantum computer:

1. The information is stored in a quantum system
2. The computation is governed by quantum-mechanical phenomena.

It is fundamentally different from digital computers and .



SCHOOL OF HUMAN SCIENCES & TECHNOLOGY

What is a Quantum bit or Qubit?

- A Qubit is the quantum concept of a bit

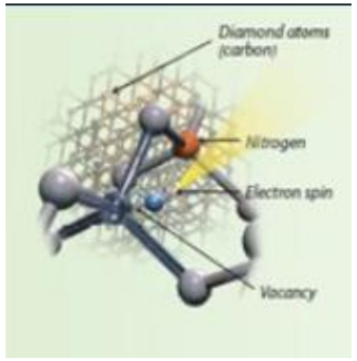
- It is either an element nor a device. A Qubit is a logical concept that can be implemented on a vast number of systems with a quantum behaviour.

- As a bit, the Qubit can be in two base states: 0 and 1.

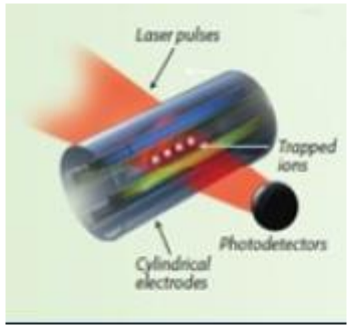
However, a Qubit can work with all the possible combinations that can be built with these base states 0 and 1

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$$

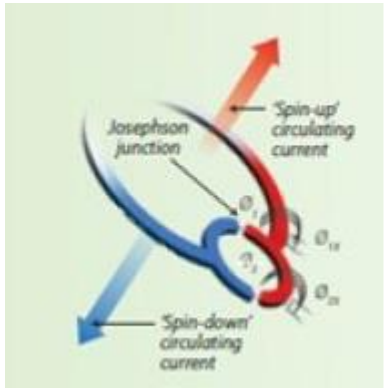
Type of Quantum Processors



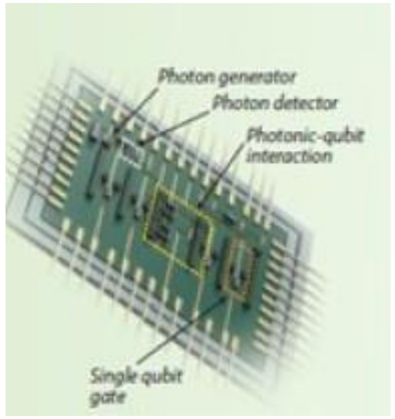
Spin Qubits – Electron or nuclear spins on a solid Substract.



Ion's Traps– Trap ions in electric fields

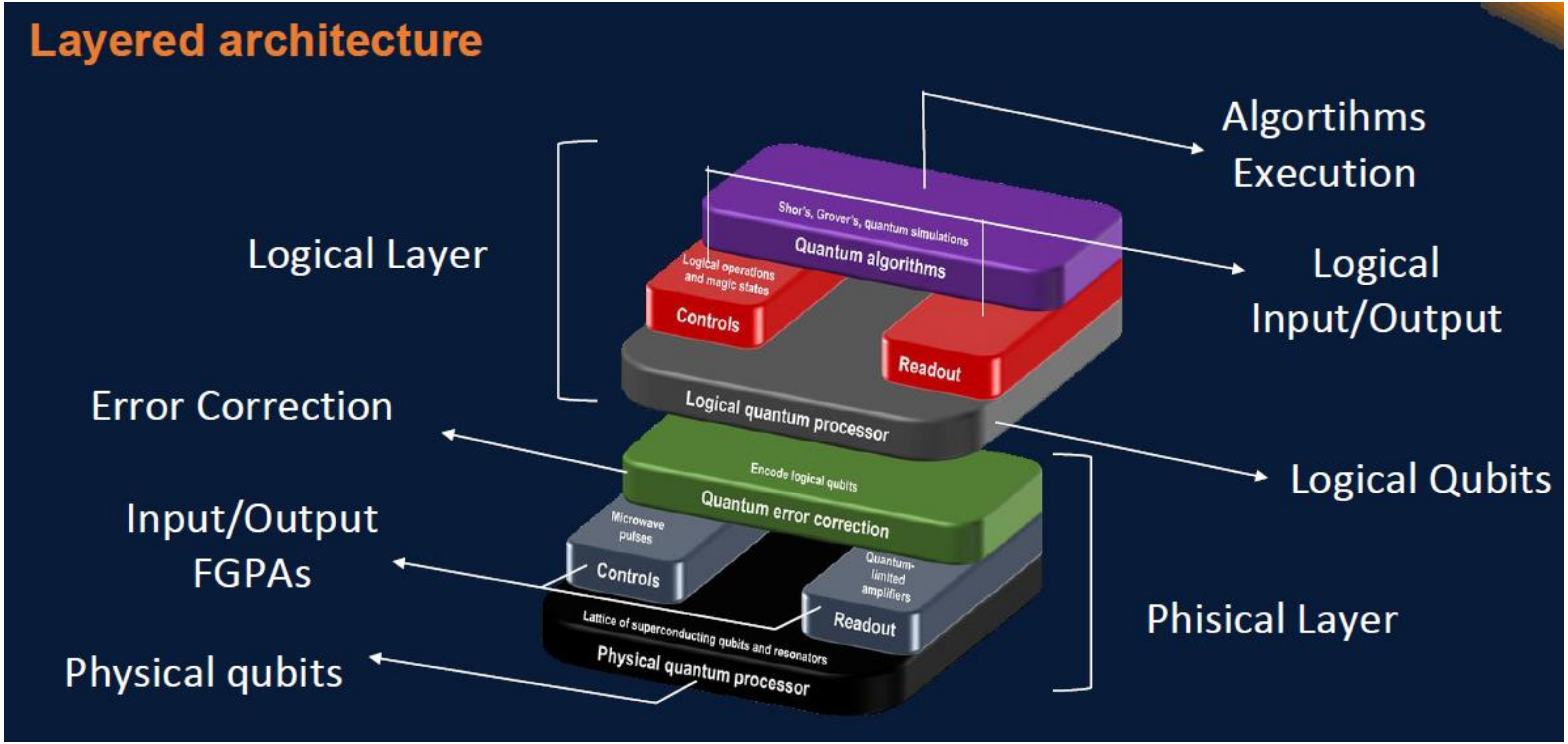

Josephson junction
'Spin-up' circulating current
'Spin-down' circulating current

Superconducting Circuits – currents superposition around a superconductor.



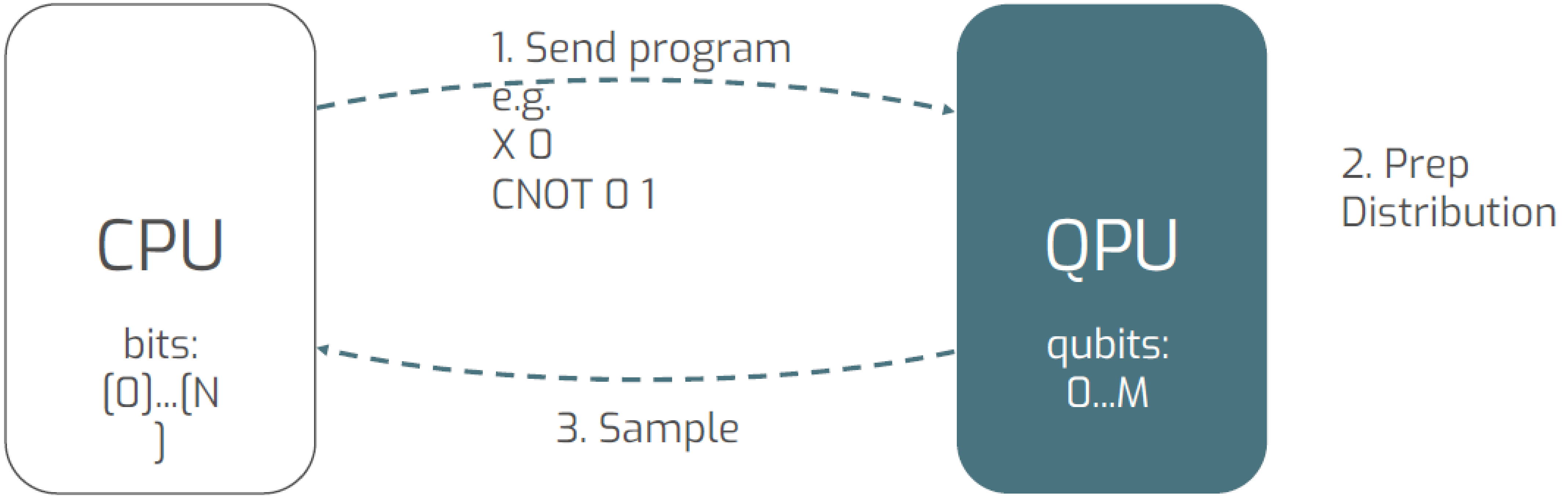
Photonic Circuits – The qubits are photons driven in in silicon circuits

Quantum Processor Architecture



Useful Quantum computation is Hybrid

Quantum programming is preparing and sampling from complicated distributions



SCHOOL OF
HUMAN SCIENCES
& TECHNOLOGY

Networking

RDMA for Extreme Performance

- Remote Direct Memory Access (RDMA) is the ability for one computer to access data from a remote computer without any OS or CPU involvement
  - Network card directly reads/writes memory with no extra copying or buffering and very low latency

- RDMA is an integral part of the Exadata high-performance architecture enabling:
  - High throughput and low-CPU usage for large data transfers
  - Unique Direct-to-Wire Protocol to deliver 3x faster inter-node OLTP cluster messaging
  - Unique Smart Fusion Block Transfer that eliminates log write on inter-node block move
  - Unique RDMA protocol to coordinate transactions between nodes

RoCE – RDMA Over Converged Ethernet

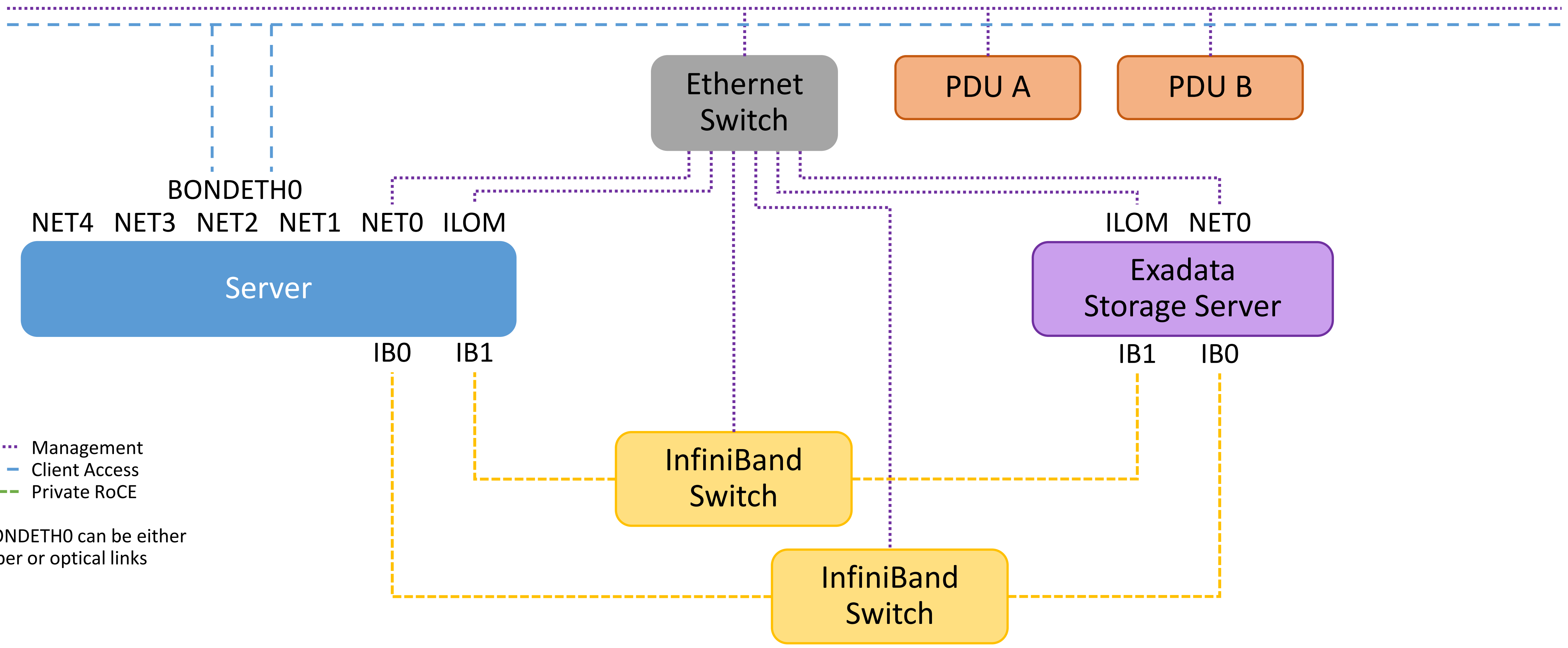
- RDMA over Converged Ethernet is a protocol that runs InfiniBand RDMA software on top of Ethernet
  - Same software at upper levels of network protocol stack
  - InfiniBand packets sent as ethernet UDP packets at low level
- RoCE on Exadata supports all Exadata RDMA optimizations
- RoCE enables scalability and volume of Ethernet with speed of RDMA
- Defined by an Open Consortium
  - InfiniBand Trade Association (IBTA)
  - Developed in open-source and maintained in upstream Linux
  - Supported by major network card and switch vendors

Layer	RoCE	InfiniBand
Application	User Application	User Application
	Transport (InfiniBand)	Transport (InfiniBand)
Network	IP Network	InfiniBand Network
Hardware	Ethernet	InfiniBand

New RoCE Internal Network Fabric

- InfiniBand was the only viable RDMA capable network at the inception of Exadata, but now Ethernet has caught up
- Exadata RoCE provides RDMA speed and reliability on Ethernet fabric
  - 100Gb throughout
  - Zero packet loss messaging
  - Prioritization of critical database messages
  - Latest KVM based virtualization

Network Architecture - InfiniBand

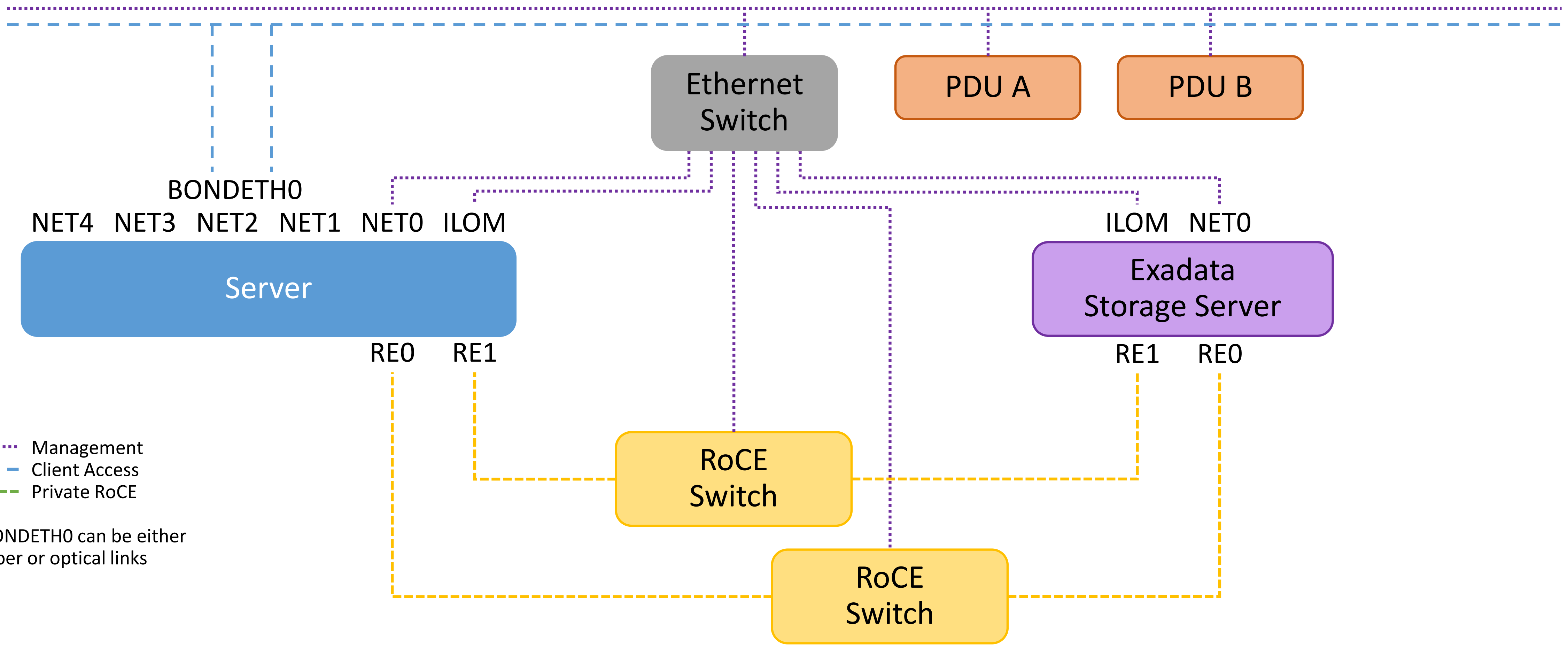


Key

- ⋯⋯ Management
- – – Client Access
- - - - Private RoCE

* BONDETH0 can be either copper or optical links

Network Architecture - RoCE

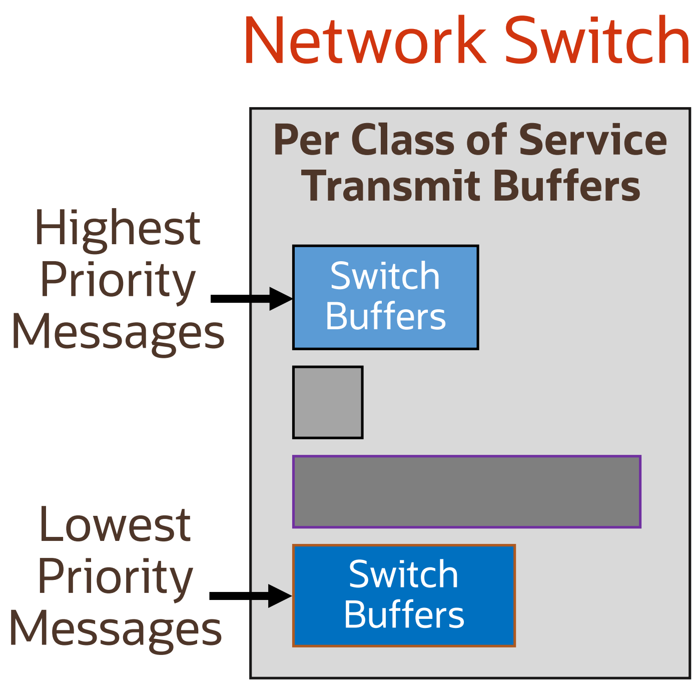


Key

- ⋯⋯ Management
- – – Client Access
- – – – Private RoCE

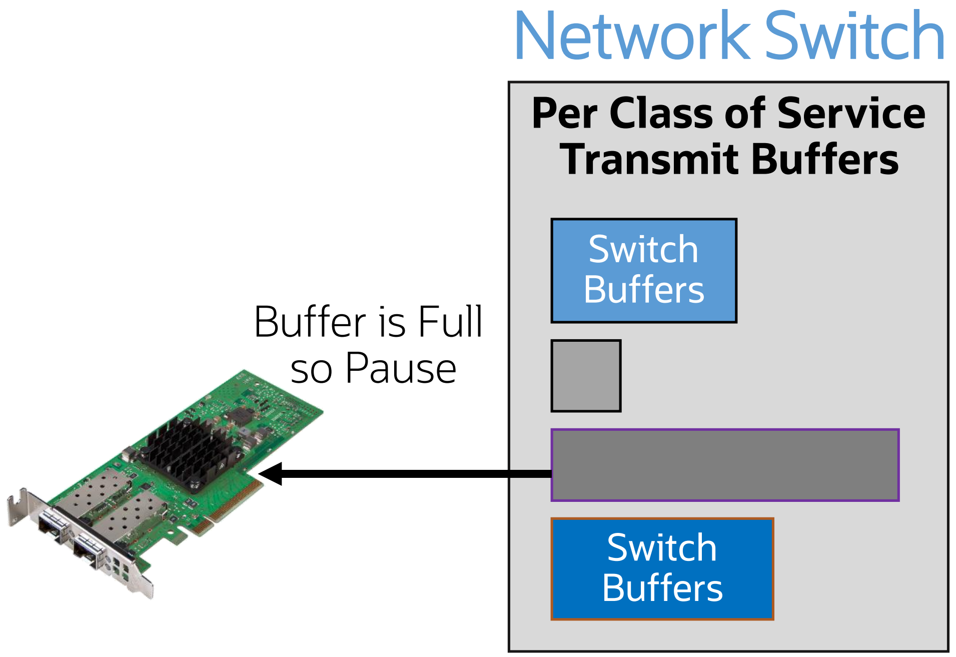
* BONDETH0 can be either copper or optical links

RoCE – High Priority Networking



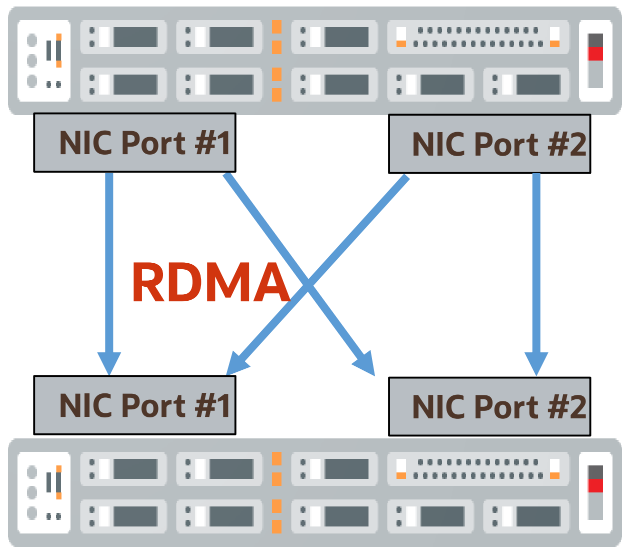
- Network prioritization for latency sensitive DB algorithms
  - Ensures that messages requiring low latency are not slowed by high throughput messages
    - Examples: cluster heartbeat, transaction commit, cache fusion
  - High throughput examples: backup, reporting, batch
- RoCE Class of Service (CoS)
  - Allows packets to be sent on multiple classes of service, each with separate network buffers for independence
- Server uniquely chooses the most optimal Class of Service for each message

RoCE – Avoiding Packet Loss



- Packet loss is usually caused by congestion
  - Packets sent faster than receiver or switches can process
  - Less common sources – switch failure, link failure
- Conventional Ethernet silently drops packets and expects retransmit if sends are too fast
  - Packet Drop causes huge hit to latency and throughput
- RoCE avoids packet drops using:
- RoCE Priority-based Flow Control (PFC)
  - RoCE switch tells sender to pause if switch buffer is full
- RoCE Explicit Congestion Notification (ECN)
  - RoCE switch marks packet flow as too fast, telling source to slow down packet sends

RoCE Instant Failure Detection



- Server uses frequent heartbeat messages between nodes to detect possible server failure
- Server failure detection normally requires long timeout to avoid false server evictions from cluster
  - Hard to quickly distinguish between slow response to heartbeat due to very high CPU load, and server failure
- Server uses RDMA to quickly confirm server failure
  - RDMA uses hardware, so remote ports respond even if software is slow
  - 4 RDMA Reads are sent to suspect server
  - Across all combinations of source and target ports
  - If all 4 RDMAs fail, server is evicted from cluster

InfiniBand vs RoCE

Feature	InfiniBand	RoCE
Fabric Management	Centralized using Subnet Manager	Decentralized Autonomous Fabric Management
Speed	40Gb/s	100Gb/s
Lossless Network	✓	✓
Multi-rack*	✓	✓
All Servers Performance Features	✓	✓
Server Virtualization	Xen o other	KVM
Instant Failure Detection	Via Subnet Manager Query	Via RDMA Queries

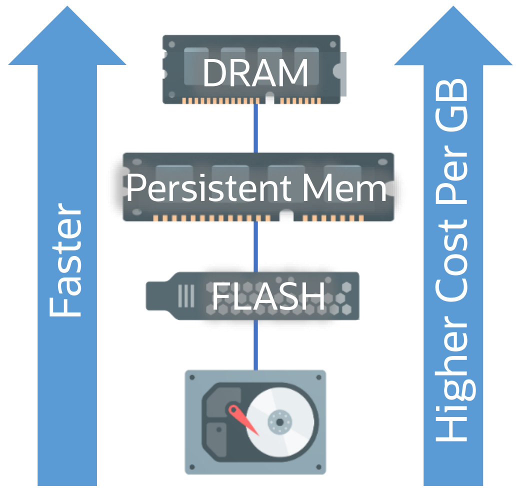
* Multi-racking between InfiniBand and RoCE is not possible

SCHOOL OF
HUMAN SCIENCES
& TECHNOLOGY

Memory

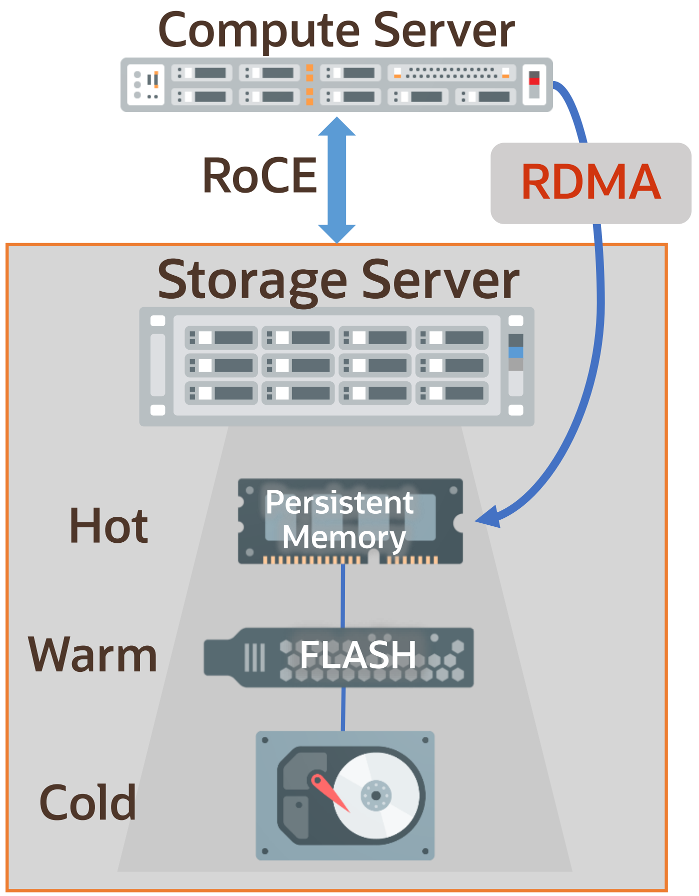
New Persistent Memory

- Persistent memory is a new silicon technology
  - Capacity, performance, and price are between DRAM and flash
- Intel® Optane™ DC Persistent Memory:
  - Reads at memory speed – much faster than flash
  - Writes survive power failure unlike DRAM
- Database Server implements sophisticated algorithms to maintain integrity of data on PMEM during failures
  - Call special instructions to flush data from CPU cache to PMEM
  - Complete or backout sequence of writes interrupted by a crash



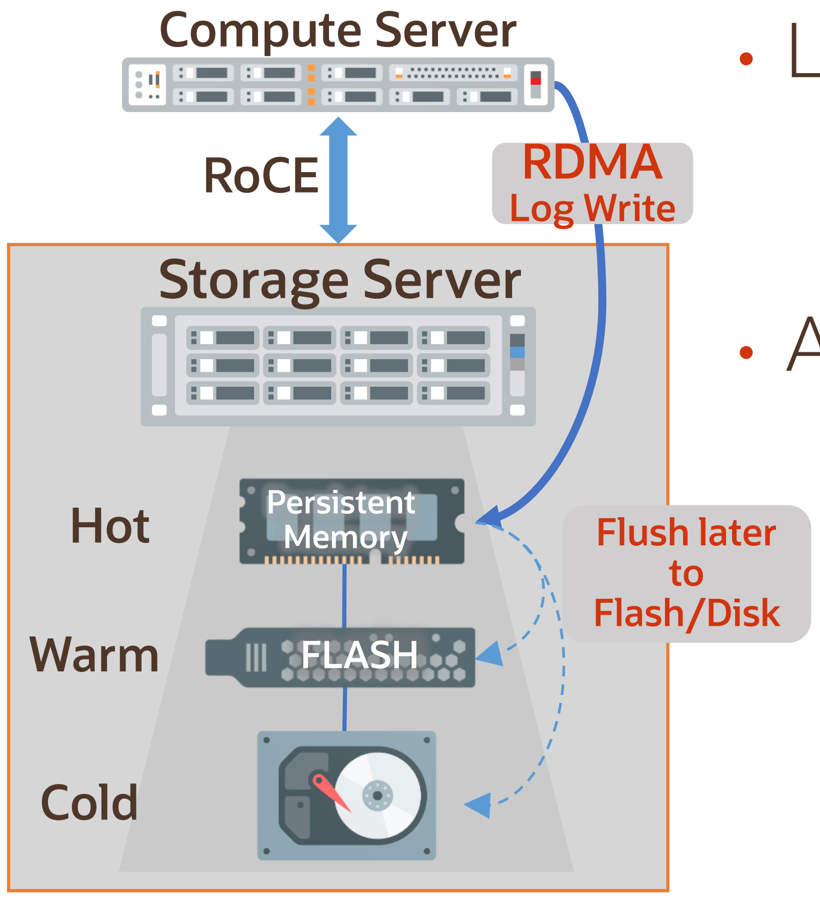
Persistent Memory Data Accelerator

Shared Persistent Memory Optimized for Database



- Storage Servers transparently add Persistent Memory Accelerator in front of Flash memory
  - 2.5X higher IOs per second than current – 16 Million IOPS
- Database uses RDMA instead of IO to read remote PMEM
  - Bypasses network and IO software, interrupts, context switches
  - 10X better latency - <19 µsec for 8K database read
- PMEM Automatically tiered and shared across DBs
  - Using as a cache for hottest data increases effective capacity 10x
- Persistent Memory mirrored automatically across storage servers for fault-tolerance

Persistent Memory Commit Accelerator



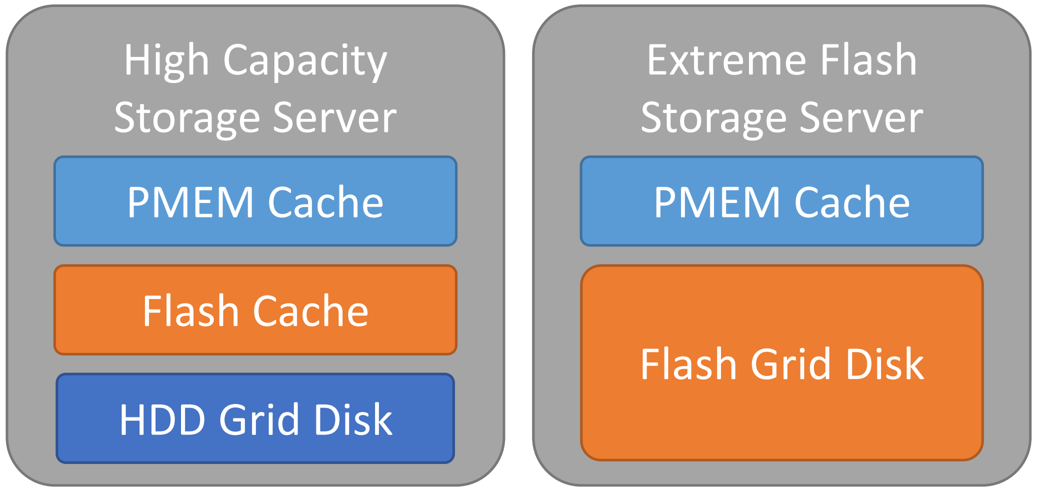
- Log Write latency is critical for OLTP performance
  - Faster log writes means faster commit times
  - Any log write slowdown stalls the whole database
- Automatic Commit Accelerator
  - Database issues one-way RDMA writes to PMEM on multiple Storage Servers
  - Bypasses network and IO software, interrupts, context switches, etc.
  - Up to 8x Faster Log Writes

PMEM in Shared Storage Advantages

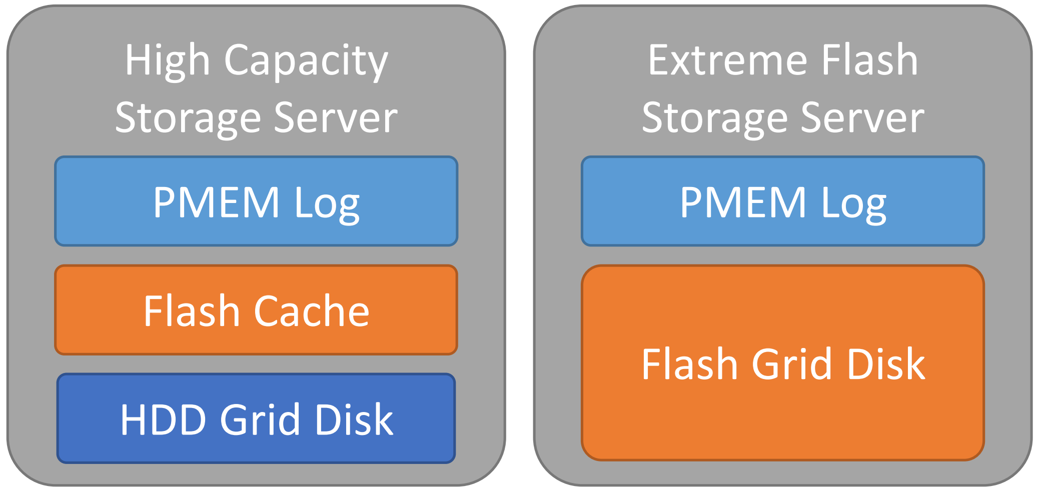
- **Better utilization of PMEM capacity** – only hottest data moved into PMEM
- **More adaptive** - aggregate performance of PMEM on all storage servers dynamically used by any server
- **End to end security** – PMEM only accessible to databases using database access controls, no OS or local access
- **Simpler** – all PMEM benefits with no extra management
- **More resilient** – automatic mirroring of PMEM across storage servers protects from PMEM failures
- **Much better latency** – RDMA from Database directly to PMEM

Database PMEM Cache and PMEM Log

- PMEM Cache
  - Implementation of Persistent Memory Data Accelerator
  - Database can near-instantly retrieve cached data from storage
  - Default Write Through, optional Write Back
  - Configured automatically by OEDA
  - No management required



- PMEM Log
  - Implementation of Persistent Memory Cache Accelerator
  - Database can near-instantly write commit records to storage
  - Configured automatically by OEDA
  - No management required



Database PMEM Management

- **No User Interaction or Management Required**
  - Installed and configured by OEDA
- ILOM performs fault management for PMEM
  - ILOM sends ASR traps for failed PMEM
  - MS alert history shows PMEM Failure and Replacement Alerts
- Secure Erase
  - Crypto Erase option
  - Drop CellDisk Erase=1pass/3pass/7pass does crypto erase on underlying PMEM DIMMs

SCHOOL OF
HUMAN SCIENCES
& TECHNOLOGY