



Parallel and Distributed Computing

Dr. Ali Sayyed
Department of Computer Science
National University of Computer & Emerging Sciences



Course Details

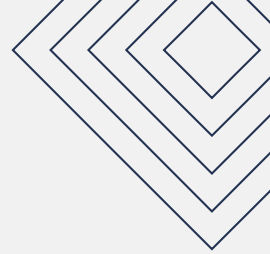
- Course Title: Parallel and Distributed Computing
- Credits: 03
- Instructor: Dr. Ali Sayyed
- Email: ali dot sayyed at nu dot edu dot pk
- GCR Code: **b4k34sz**



Course Overview

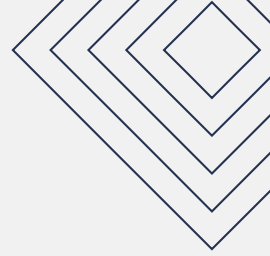
- This course covers a broad range of topics related to parallel and distributed computing.
- We'll examine the principles of parallelism and techniques for breaking down complex tasks into smaller, manageable parts that can be executed simultaneously.
- The course will also cover distributed systems, discussing the coordination and communication among interconnected machines to achieve collective objectives.
- Students will be able to write programs for Concurrent and Distributed System.





Course Assessments

#	Evaluation	Weightage %
1	Quizzes	10
2	Assignments	10
3	Sessional-I	15
4	Sessional-II	15
5	Final Term Exam	50
Total		100



Learning Outcomes

- **CLO1** Demonstrate understanding of various concepts involved in parallel and distributed systems.
- **CLO2:** Implement different parallel and distributed programming paradigms and algorithms using Message-Passing Interface (MPI), OpenMP and CUDA.
- **CLO3:** Perform analytical modelling, dependence, and performance analysis of parallel algorithms and programs.



Course Policies



1. Students are expected to attend all physical classes and actively participate in online discussions through Google Classroom.
2. Assignments are to be submitted via Google Classroom by the specified deadlines. All submissions will undergo plagiarism and AI contents checks.
3. All course material and additional resources, including announcements, will be shared through Google Classroom.
4. There will be no announced quizzes.
5. Quiz in a different section will not be allowed
6. Make sure to turn off, or place in silent mode, your cell phone.





Outline



- Introduction
- Computer Architecture and Boosting Processor Performance
- Hardware Models
- Incorporating Parallelism
- Decomposition Techniques
- Flynn's and Johnson Taxonomy of Architecture
- Memory Access and Memory Hierarchy
- Synchronization, Coordination and Consistency Models
- Shared Memory Programming Models: OpenMP
- Distributed Memory Programming Model: OpenMPI
- Vector Programming: CUDA
- Hadoop



Computer Architecture

Focusing on the computer processor architecture only





Overview of CPU



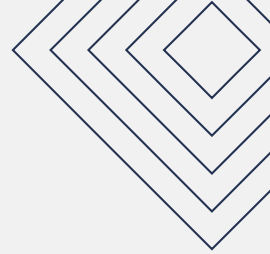
- The central processing unit (CPU, or processor) executes **program instructions on program data**.
- Program instructions and data are stored in the computer's **RAM**.
- Each processor implements a specific **instruction set architecture** (ISA), which defines the set of instructions and their binary encoding and the set of CPU registers.
- There are many **different ISAs**, including SPARC, IA-32 and IA-64, MIPS, ARM, ARC, PowerPC, and x86.



RISC VS CISC



- Two approaches are used to implement ISAs
 - Reduced Instruction Set Computer (RISC)
 - Complex Instruction Set Computer (CISC).
- RISC ISAs have a **small set** of instructions that **execute quickly** (in about a **single processor clock cycle**). The compilers combine several basic RISC instructions to implement higher-level functionality.
- In contrast, a CISC ISA's have a **larger set** of instructions and attempts to **minimize the number of instructions per program**, sacrificing the number of cycles per instruction.
- Intel's x86 ISAs are CISC-based. RISC ISAs are more commonly seen in high-end servers (e.g. SPARC) and in mobile devices (e.g. ARM) due to their low power requirements.

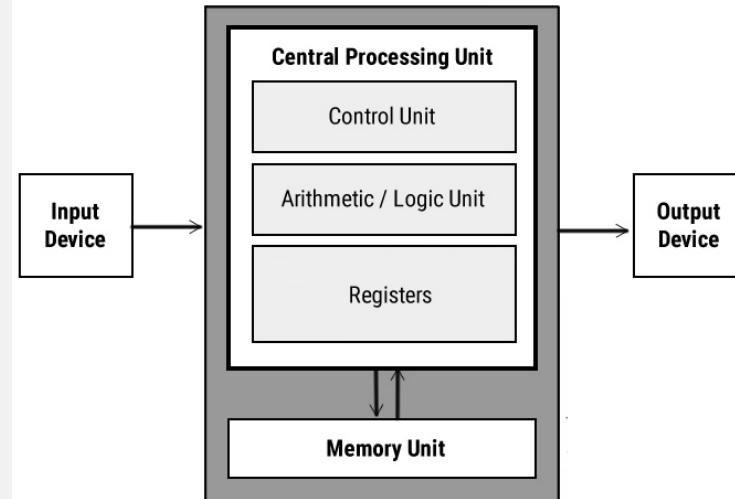


von Neumann Architecture

- In 1945, a Hungarian mathematician John von Neumann proposed the first architecture on which modern computers are based.
- The **general-purpose design** of the von Neumann architecture allows it to **execute any type of program**.
- It uses a **stored-program model**, meaning that the **program instructions reside in computer memory along with program data**, and both are inputs to the processor.

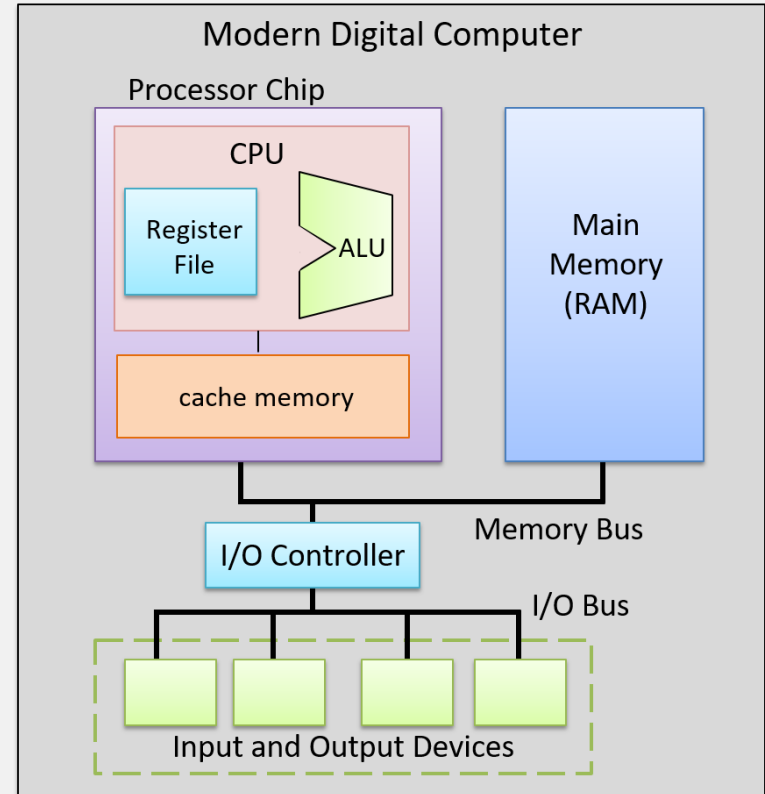
von Neumann Architecture

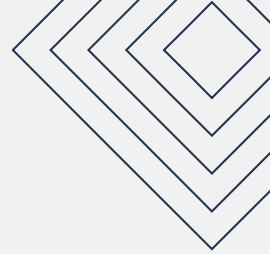
- The **ALU** allows arithmetic (add, subtract etc) and logic (AND, OR, NOT etc) operations to be carried out.
- The **Control Unit** fetches instructions/data from memory, decodes the instructions and then sequentially coordinates operations to accomplish the programmed task.
- The **Memory Unit** stores program data and instructions.
- **Input/Output** is the interface to the human operator.
- **Registers** are high speed storage areas in the CPU. All data must be stored in a register before it can be processed.



Today's Processors

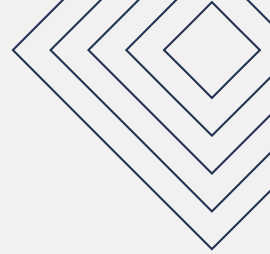
- Today's processors are implemented as digital circuits on a **silicon chips**.
- It also includes some fast **on-chip cache memory**, used to store copies of recently used program data and instructions close to the processor.
- This is an example of a processor in the context of a complete modern computer, whose components together implement the von Neumann architecture.





Clock-Driven Execution

- A **clock drives the CPU's execution of instructions**, triggering the start of each stage.
- The clock cycle is considered the **basic unit of measuring how fast** an instruction can be executed by the computer processor.
- A processor's clock speed (or clock rate) is typically measured in **megahertz** (MHz) or **gigahertz** (GHz). A 1-GHz has one billion clock ticks per second.
- The clock rate is a measure of how fast the CPU can run and is an estimate of the maximum number of instructions per second a CPU can execute. For example, a 2-GHz processor might achieve a maximum instruction execution rate of two billion instructions per second.



Increasing the Clock Speed

- **Increasing the clock rate on a single machine** will improve its speed and performance, clock rate alone is **not a meaningful metric for comparing the performance of different processors**.
- For example, RISC require fewer stages to execute instructions than CISC.
- In RISC, a slower clock may yield the same number of instructions completed per second as on CISC.
- For a specific microprocessor, however, doubling its clock speed will roughly double its instruction execution speed.

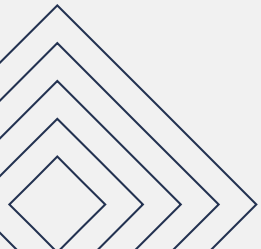
Boosting Performance

Speeding up the computation

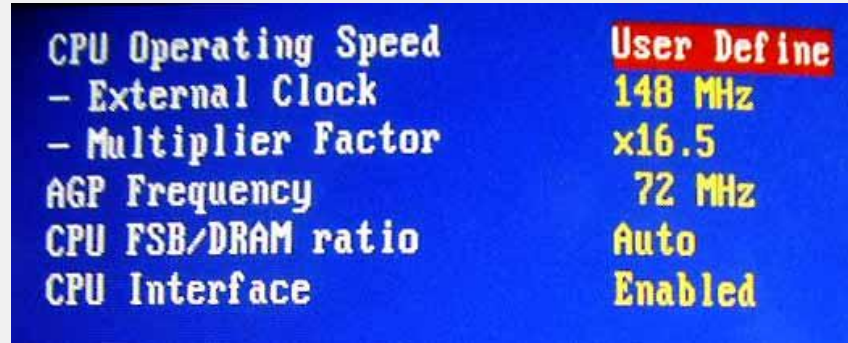


Speeding up the computation

- Overclocking
- Increasing the Clock Speed
- Multithreading
- Hardware Multithreading
- Multicore Processors
- Heterogeneous Computing or Multiprocessors System



Overclocking



CPU Operating Speed	User Define
– External Clock	148 MHz
– Multiplier Factor	x16.5
AGP Frequency	72 MHz
CPU FSB/DRAM ratio	Auto
CPU Interface	Enabled

- Increasing the clock speed beyond the manufacturer's specifications (overclocking) can provide a performance boost.
- An overclocked device may be unreliable or fail completely if the additional heat load is not controlled, or power delivery components cannot meet increased power demands.
- It should be done cautiously to avoid overheating and instability.



Increasing the Clock Speed



- Historically, increasing the clock rate has been a **very effective way** for computer architects to improve processor performance.
- For example, in 1974, the Intel 8080 CPU ran at 2 MHz. The clock rate of the Intel Pentium Pro, introduced in 1995, was 150 MHz, and the clock rate of the Intel Pentium 4, introduced in 2000, was 1.3 GHz.
- Clock rates peaked in the mid to late 2000s with processors like the IBM z10, which had a clock rate of 4.4 GHz.
- The highest clock rate on a production processor is the i9-14900KS, clocked at 6.2 GHz, which was released in 2024
- Today, however, CPU clock rates have reached their limit...





Essentials

Product Collection	Intel® Core™ i9 processors (14th gen)
Code Name	Products formerly Raptor Lake
Vertical Segment	Desktop
Processor Number ?	i9-14900KS
Lithography ?	Intel 7
Recommended Customer Price ?	\$689.00-\$699.00

CPU Specifications

Total Cores ?	24
# of Performance-cores	8
# of Efficient-cores	16
Total Threads ?	32
Max Turbo Frequency ?	6.2 GHz
Intel® Thermal Velocity Boost Frequency ?	6.2 GHz
Intel® Turbo Boost Max Technology 3.0 Frequency ¹ ?	5.9 GHz
Performance-core Max Turbo Frequency ?	5.7 GHz
Efficient-core Max Turbo Frequency ?	4.5 GHz
Performance-core Base Frequency ?	3.2 GHz
Efficient-core Base Frequency ?	2.4 GHz
Cache ?	36 MB Intel® Smart Cache
Total L2 Cache	32 MB
Processor Base Power ?	150 W
Maximum Turbo Power ?	253 W





Intel breaks Guinness World Record for fastest processor frequency

Intel i9-14900KF overclocker clinches CPU frequency world record at 9.12 GHz — Wytix joins Elmor as the only person to push a CPU past 9 GHz

News

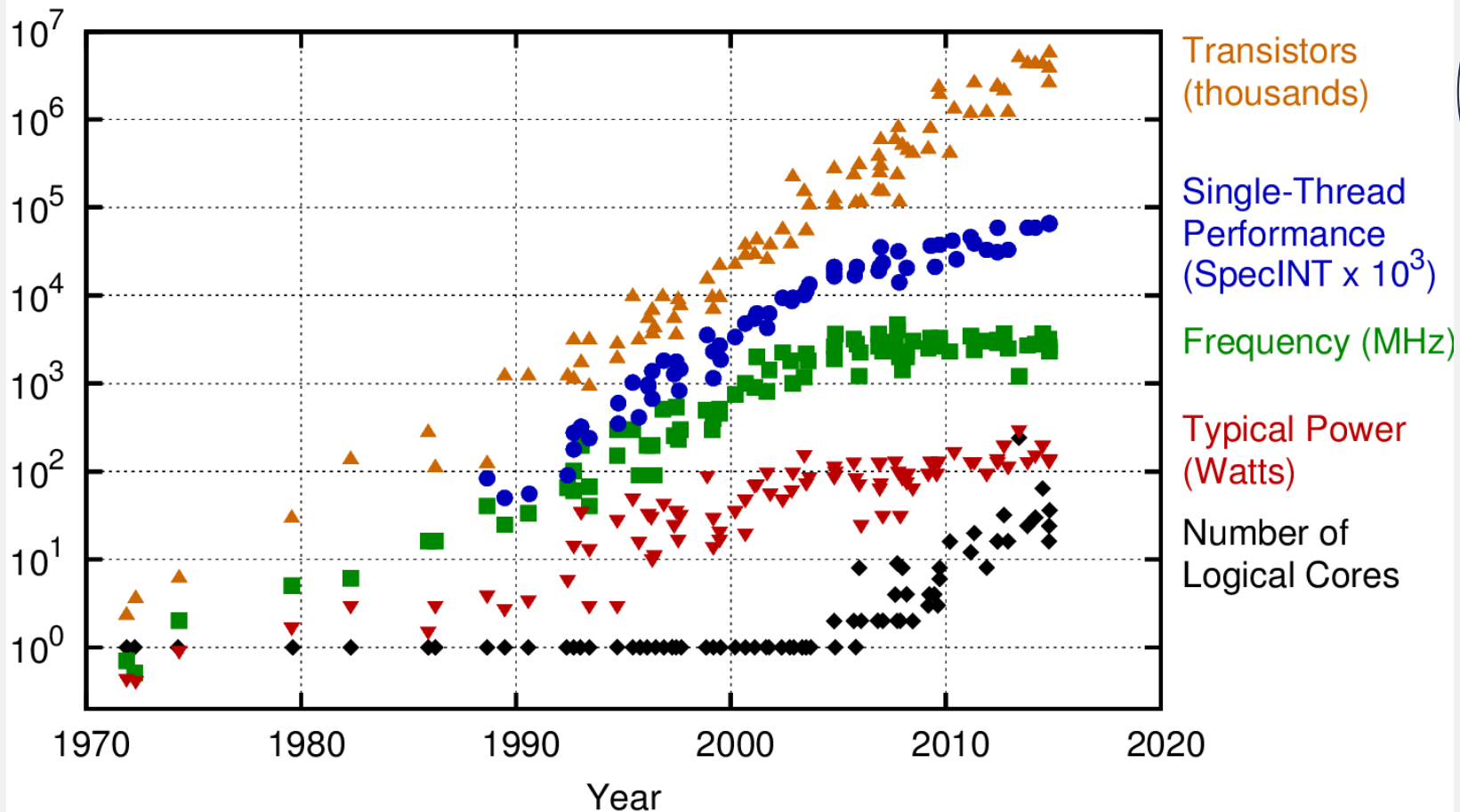
By [Hassam Nasir](#) published January 12, 2025

Elmor finally gets dethroned, but the Core i9 remains the frequency king.



Comments (7)

40 Years of Microprocessor Trend Data



Original data up to the year 2010 collected and plotted by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond, and C. Batten
New plot and data collected for 2010-2015 by K. Rupp



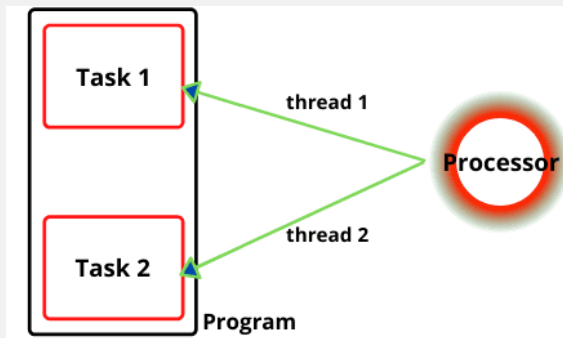
Increasing the Clock Speed

- Can we get a 10 GHz clock frequency? What will happen if we reach a 10 GHz clock frequency?
- **Potential consequences**
 - Heat Dissipation
 - Power Consumption
 - Electromagnetic Interference
 - Signal Distortion : As clock frequencies increase, maintaining signal integrity becomes more challenging.
 - Manufacturing Challenges




Multithreading

- A thread is an **independent** stream of execution.
- Multithreading is a feature in OS that allows a program to **do several tasks** at the same time.
- Think of it like having **multiple hands** working together to complete different parts of a job.
- On a **single core** system, the threads of a multithreaded applications **don't execute in parallel**. Instead, they share a single processor core.



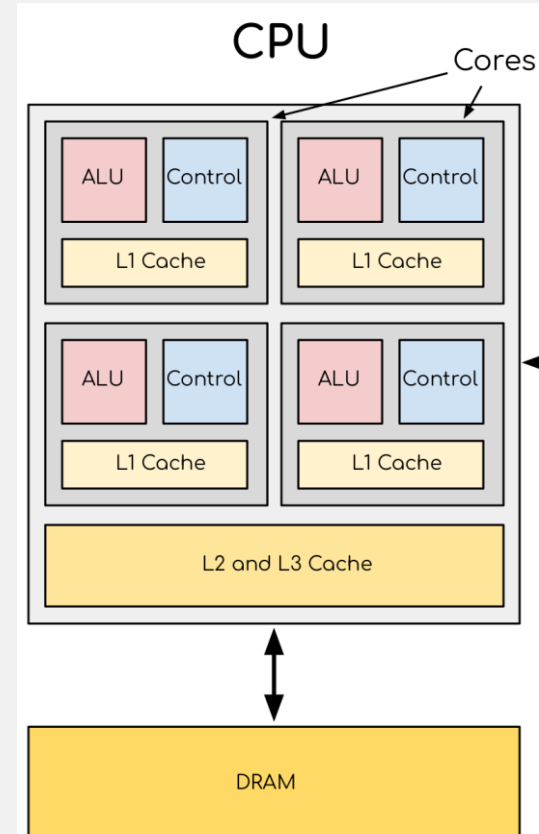


Hardware Multithreading

- Hardware multithreading (Intel calls this hyper-threading) is a **single-processor design that supports executing multiple hardware threads**. This techniques enable one physical core to function as **two logical cores**.
 - These threads of execution could then be **scheduled by the operating system** to run "at the same time" on a multithreaded processor.
 - The processor is designed to **quickly switch** between executing instructions.
 - **This usually results in a speed-up** of their execution as a whole as compared to their execution on a singly threaded processor.
- 

Multicore Processors


- Multicore processors contain **multiple** complete CPU **cores**.
- Each core contains its **own** complete and separate **functionality to execute** program instructions.
- A multicore processor contains replicas of these CPU cores with some additional hardware support for the cores to share cached data.





Multicore Processors



- Multicore microprocessor design is **the primary way** in which the performance of processor can **continue to keep pace with Moore's Law without** increasing the processor clock rate.
 - A multicore computer can simultaneously run several programs, the OS scheduling each core with a different program's instruction stream.
 - It **can speed up execution** of a single program if the program is written as an explicitly multithreaded (software-level threads) program.
- 



Heterogeneous Computing



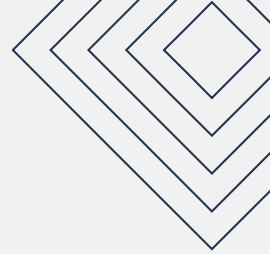
- Refers to a system that uses **more than one type of computing units**.
- Their processing units have **different ISAs**.
- With support for parallel computing using the computer's **CPU cores and one or more Hardware Accelerator Units**
- It is increasingly common for developers to implement heterogeneous computing solutions to **large, data-intensive and computation-intensive problems**.
- By making use of the processing capabilities of both the CPU and the accelerator units, **a programmer can increase the degree of parallel execution in their application**, resulting in improved performance and scalability.



Hardware Accelerators



- In addition to the CPU, **computers have other processing units** that are designed to perform specific tasks.
- These units are **not general-purpose processing units** like the CPU, but are special-purpose hardware that is optimized to implement functionality that is specific to certain devices or that is used to perform specialized types of processing in the system.
- GPU is a common example of a Hardware Accelerator Unit.

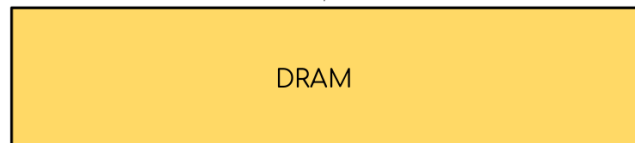
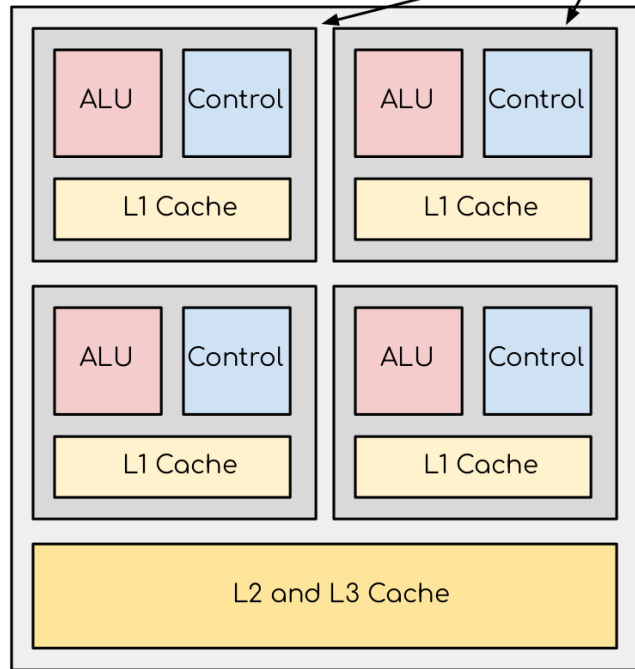


Graphics Processing Units (GPUs)

- A graphics processing unit is a specialized circuit initially designed to accelerate computer graphics and image processing.
- It consists of thousands of special-purpose processors.
- GPUs were found to be useful for non-graphic calculations involving parallel problems due to their parallel structure.
- General Purpose GPU (GPGPU) computing applies special-purpose GPU processors to general-purpose parallel computing tasks.

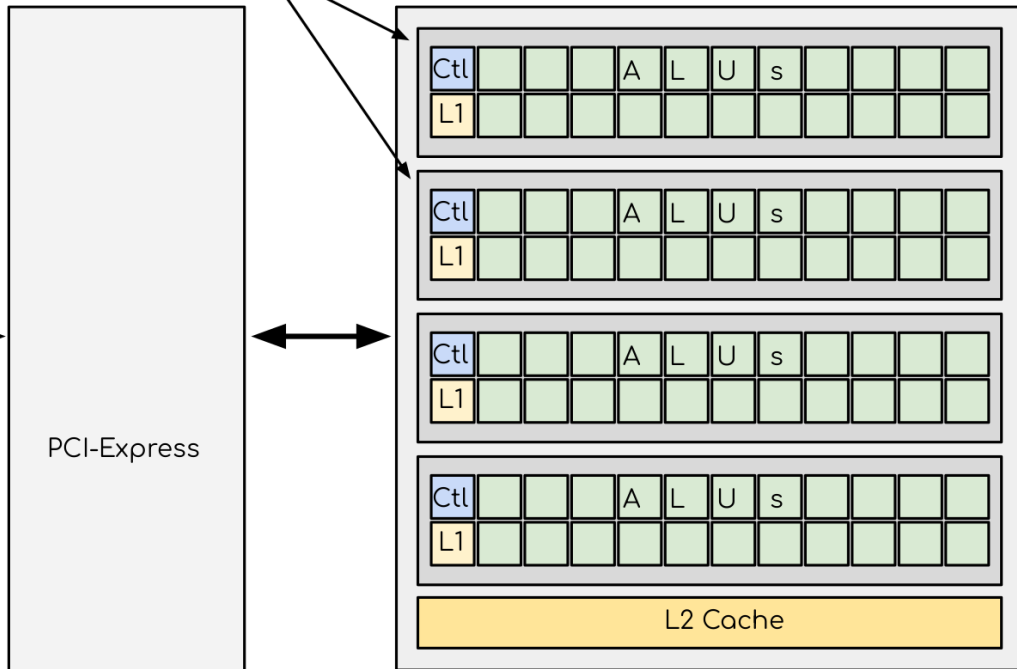
CPU

Cores



Streaming multiprocessors

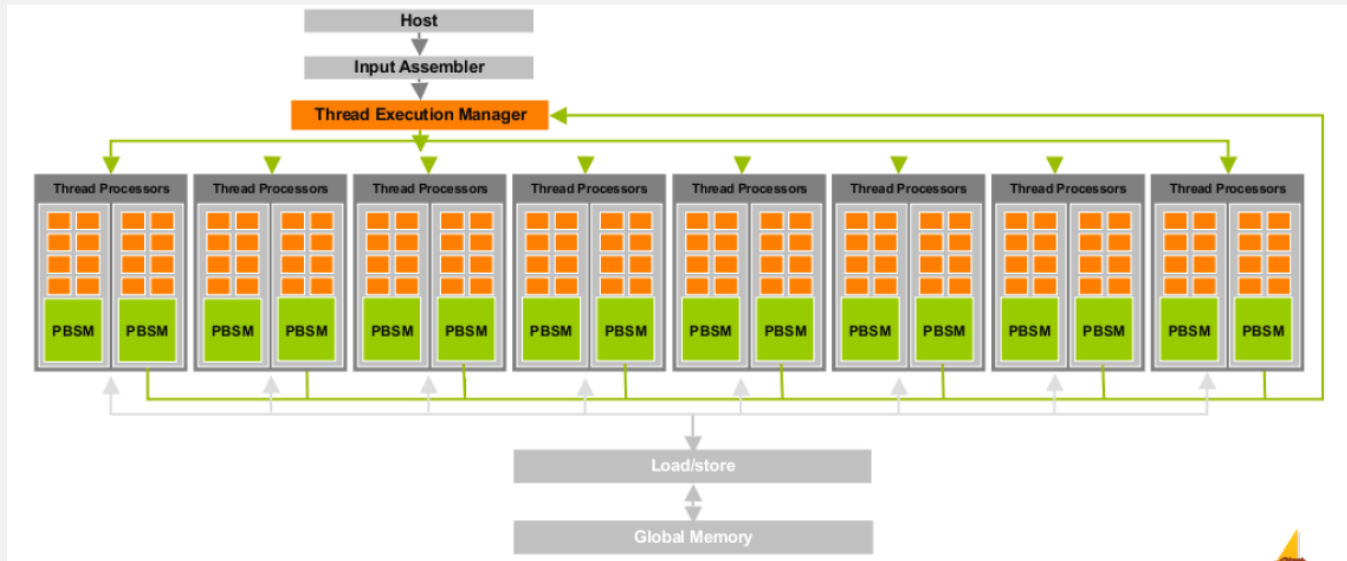
GPU



PCI-Express

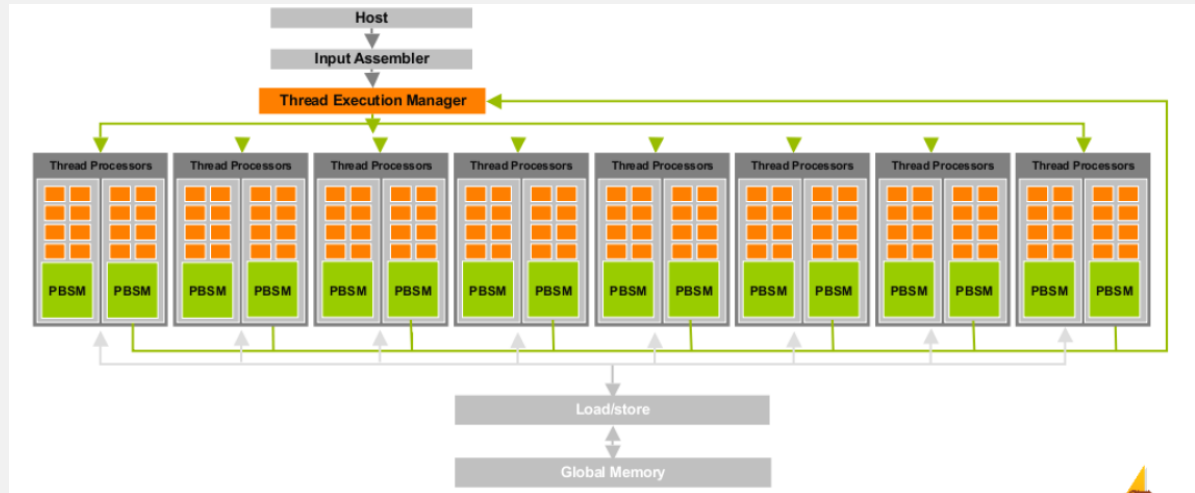
Graphics processing units (GPUs)

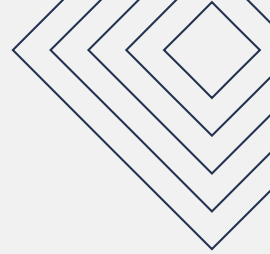
- A GPU consists of several streaming multiprocessors (SMs) with a large shared memory. In addition, each SM has a local (PBSM - per block shared memory) memory.



Graphics processing units (GPUs)

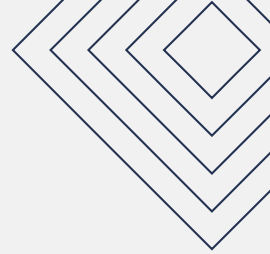
- In a GPU, the small local memories have much smaller access time than the large shared memory.
- Thus, as much as possible, cores access data in the **local memories** while the **shared memory** should essentially be used for data exchange between SMs.





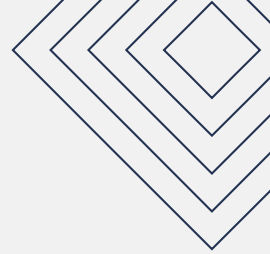
CPU vs GPU

CPU	GPU
Central Processing Unit	Graphics Processing Unit
4-8 Cores	100s or 1000s of Cores
Low Latency	High Throughput
Good for Serial Processing	Good for Parallel Processing
Quickly Process Tasks That Require Interactivity	Breaks Jobs Into Separate Tasks To Process Simultaneously
Traditional Programming Are Written For CPU Sequential Execution	Requires Additional Software To Convert CPU Functions to GPU Functions for Parallel Execution



Tensor Processing Unit

- A TPU (Tensor Processing Unit) is a specialized hardware accelerator (called AI Accelerator) designed by Google to perform high-speed computations for machine learning tasks, particularly those using neural networks.
- TPUs are optimized for frameworks like TensorFlow and are widely used for artificial intelligence applications such as natural language processing, image recognition, and recommendation systems.



Key Features of TPUs

- 1. Custom-Designed for Machine Learning:**

TPUs are specifically built to accelerate matrix operations, which are the core of machine learning algorithms, especially in deep learning.

- 2. High Performance:**

They offer extremely high throughput and efficiency for tasks like training and inference in neural networks.

- 3. Power Efficiency:**

TPUs consume less power compared to GPUs for specific workloads, making them more cost-effective for large-scale AI models.

- 4. Specialization in Tensor Operations:**

They excel at performing tensor computations such as matrix multiplication and convolutional operations, which are common in AI tasks.

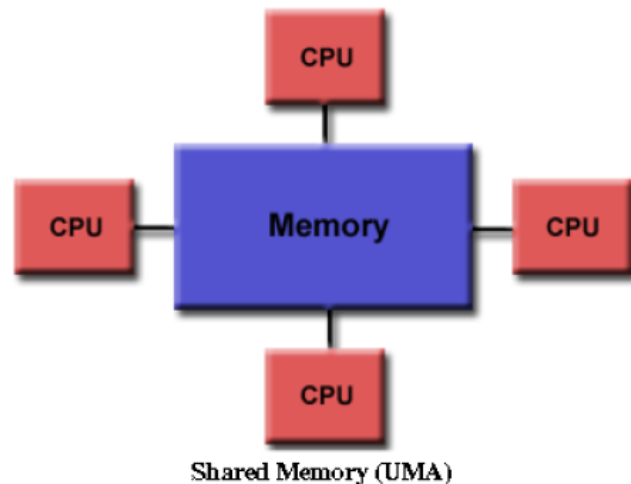


Multiprocessors Systems

- Most computers today (including tablets, smartphones, etc.) are equipped with several CPUs.
- Various characteristics determine the types of computations:
 - shared memory vs distributed memory
- Historically, shared memory machines have been classified as UMA and NUMA, based upon memory access times.

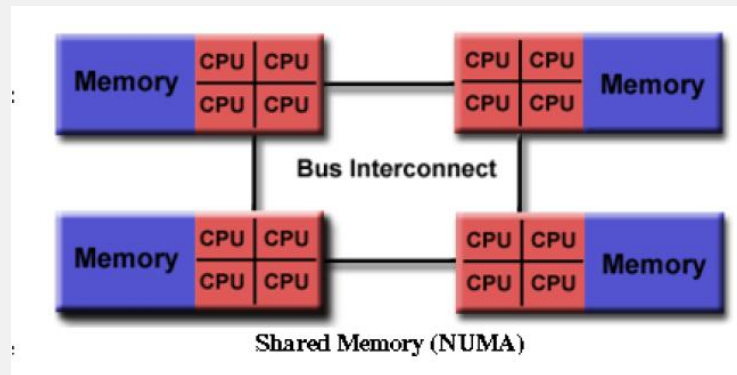
Uniform Memory Access (UMA)

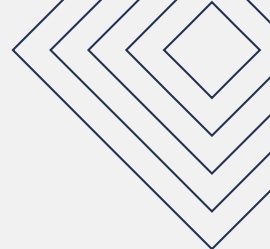
- Identical processors, equal access and access times to memory.
- if one processor updates a location in shared memory, then all the other processors know about the update.
- Multicore processors follow the same architecture and, in addition, integrate the cores onto a single circuit die.
- Easy to Implement
- Low Latency and Cost
- Limited Scalability



Non-uniform Memory Access (NUMA)

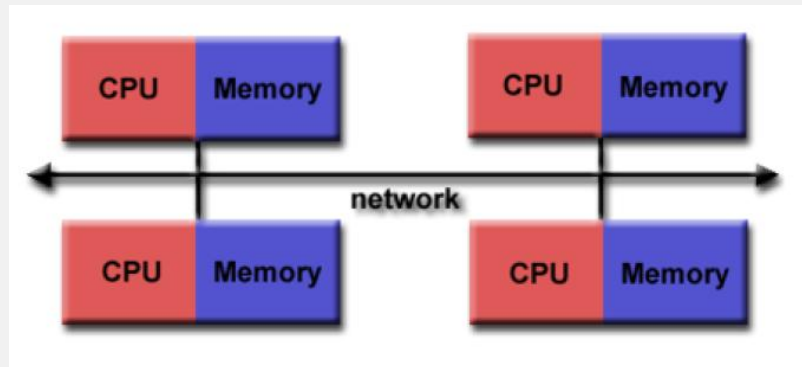
- Often made by physically **linking two or more** multicore processors.
- **Global address space** provides a user-friendly programming perspective to memory; it **feels like there is a single large memory** where all data reside.
- However, **not all processors have equal access time** to all memories, since memory access across link is slower.
- Improved performance and scalability
- Complexity, Performance variability and Higher Cost





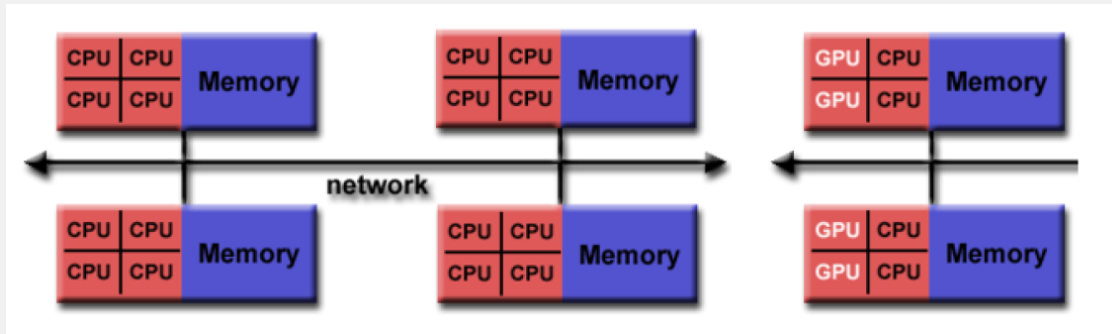
Distributed Memory

- Distributed memory systems **require a communication network** to connect inter-processor memory.
- Processors **have their own local memory and operate independently.**
- Data exchange between processors is managed by the **programmer , not by the hardware.**



Hybrid Distributed-Shared Memory

- The largest and fastest computers in the world today employ **both shared and distributed memory** architectures.
- Current trends seem to indicate that this type of memory architecture will continue to prevail.
- While this model allows for applications to scale, it increases the complexity of writing computer programs.





Issues in Multicore and Multiprocessor Systems



- **Multicore:**
 - The addition of additional cores **doesn't automatically improve** computer performance. The **OS and applications must** recognize and use the multiple cores.
 - Second, the performance benefit of additional cores **is not a direct multiple**. That is, adding a second core does not double the processor's performance.
 - This is due to the other shared resources, such as memory, caches, external buses etc.
- **Multiprocessor:**
 - The **OS and applications must** recognize and use the multiple cores.

