CA notes:

- Semi-Conductor:
a material, like silicon, that has electrical conductivity b/w a conductor & an insulator. It is used in microchips, transistors, & Processors to control electrical signals in computers & electrical devices.

→ How to Improve Performance?
- Improvement in semiconductor technology like feature size (small) & clock speed (high)
- Improvements in CA's such as enabled by HLL compilers, Unix

& should lead to RISC arch.'s.

- These together have enabled lightweight computers & Productivity-based managed/ interpreted Prog. lang.'s.

→ Types of parallelism:

- Instruction-level } single Processor Per. imp. ended in 2003.
- Data-level
- Thread-level
- Request-level } new Models for Performance.

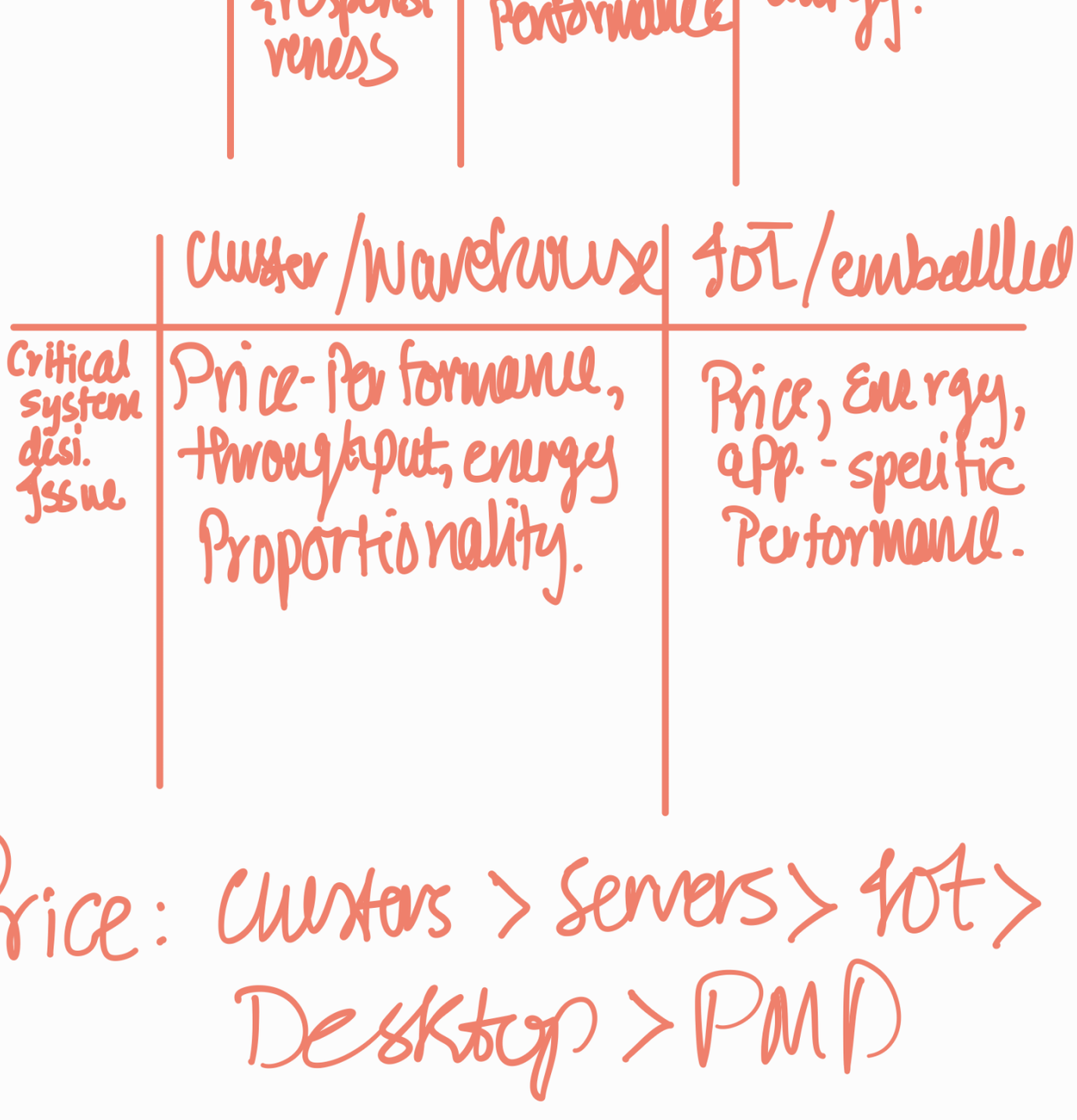
→ Classes of computers:

- Personal Mobile Device (PMD) emphasis on energy efficiency & real-time. eg. smart phones & tablets.
- Desktop Computing: Emphasis on price-perfor-

market.

- Servers:
  Emphasis on availability, scalability & throughput.
- Clusters/warehouse scale computers:
  Used for "Software as a Service (SaaS)", also emphasis on availability & Price-performance.
  Sub-class: supercomputers
  emphasis: floating-point Performance & fast internal networks.
- Internet of things/Embedded computers:
  Emphasis on price.

	PMD	Desktop	Servers
Critical system desi. Issue	Cost, Energy, Media Performa. & responsi-	Price-Performance, energy, graphics	Throughput, availability, scalability & energy



1. DLP (Data-Level Parallelism)

Data-Level Parallelism (DLP) occurs when the same operation is performed on large amounts of data simultaneously. It focuses on processing multiple data elements in parallel, where each data element can be operated on independently of the others.

Example:
Consider a scenario where you have a list of numbers, and you need to multiply each number by 2. Each multiplication operation is independent, so they can all be performed in

parallel. In this case, you are using DLP because the operations on each individual piece of data (each number) can be done at the same time.

2. TLP (Thread-Level Parallelism)

Thread-Level Parallelism (TLP) involves executing multiple threads (independent tasks) simultaneously. Each thread can be thought of as a separate task that can be processed independently. TLP is often achieved through multiple processor cores, where each core executes a different thread concurrently.

Example:
Imagine a program that performs two tasks: one task to download data from the internet and another to process the data. These tasks can be executed in parallel on different threads, improving performance by making use of multiple cores.

3. RLP (Request-Level Parallelism)

Request-Level Parallelism (RLP) occurs when multiple independent requests can be processed at the same time. These requests are typically handled by systems such as web servers, databases, or cloud services. Each request is independent of others, so they can be processed concurrently.

Example:
Consider a web server handling multiple users who are making different requests (e.g., loading different pages or

querying different databases). Each request is independent, and the server can process several requests at once, improving the efficiency of the server.

4. ILP (Instruction-Level Parallelism)

Instruction-Level Parallelism (ILP) refers to the ability of a processor to execute multiple instructions from a single thread in parallel. Modern processors can identify independent instructions within the same program and execute them simultaneously to speed up processing.

Example:
If a processor is running a program with instructions like "add two numbers" and "multiply two numbers," and these operations do not depend on each other, the processor can execute both instructions at the same time, using ILP to enhance performance.

- classes of Parallelism in appli.:
  - Data-level (DLP)
  - Task-level (TLP)
- classes of architectural parallelism:
  - Instruction-level (ILP)
  - GPUs / vector architectures
  - Thread-level

· Request - level.

Flynn's Taxonomy is a classification system for computer architectures based on the number of instructions and data streams they can handle simultaneously. It divides computer architectures into four categories:

- **SISD (Single Instruction, Single Data):**
- This is a traditional, single-processor architecture where one instruction operates on one data at a time.
- Example: A basic personal computer running a single program at once.
- **SIMD (Single Instruction, Multiple Data):**
- A single instruction is applied to multiple data elements simultaneously. This is useful for tasks like image processing or scientific computing.
- Example: Graphics processing units (GPUs) performing parallel operations on multiple pixels of an image at the same time.
- **MISD (Multiple Instruction, Single Data):**
- Multiple instructions operate on the same data simultaneously, but this type is rare in practice.
- Example: Systems in redundancy applications, where the same data is processed by different instructions to verify accuracy (like fault-tolerant computing).
- **MIMD (Multiple Instruction, Multiple Data):**
- Multiple processors execute different instructions on different data elements. This is common in modern multi-core processors.
- Example: A modern multi-core processor running different tasks (such as browsing, video editing, etc.) on

separate cores.

In essence, Flynn's Taxonomy helps to understand how computers process data in parallel or sequentially, impacting performance and application suitability.

→ Moore's law:

Moore's Law is the observation made by Gordon Moore, co-founder of Intel, in 1965 that the number of transistors on a microchip (and thus its processing power) doubles approximately every two years, leading to a rapid increase in computing performance and a decrease in relative cost.

- transistor density ↑ 35% / year
- Die size ↓ 10-20% / year
- Integration Overall ↑ 40-55% / year.

- DRAM capacity ↑ 25 - 40% year but is slowing down.
  - ↳ 8 GB in 2014
    16 GB in 2019 but 32 GB not soon.

- flash capacity ↑ 50 - 60% / year faster than DRAM capacity. Plus, it is 8-10X cheaper per bit compared to DRAM making it cost effective alternative.

- Magnetic Disk ↑ 5%/year & has slowed down significantly, no longer possible to make the storage larger by making the bits smaller but manufacturers can add more platter (7 to 9) to increase capacity.

Moreover, 8 - 10 times cheaper per Bit than Flash & 200 - 300 times cheaper than DRAM making it the most cost-efficient for large storage.

Bandwidth: The maximum amount of data that can be transmitted per second, such as gigabits per second for an internet connection.

Throughput is the total amount of work done in a given time, such as megabytes per second for a disk transfer.

Response time(or latency) is the time between the start and the completion of an event, such as milliseconds for a disk access.

Thermal Design Power (TDP) is the amount of heat a computer chip (like a CPU or GPU) is expected to generate under normal, sustained workloads.

- **It represents sustained power consumption**, meaning how much power the chip will regularly use over time.
- **It helps design cooling systems and power supplies**, ensuring the chip doesn't overheat or get too little power.
- **It's lower than peak power but higher than average power**, meaning the chip may use more power in short bursts (up to 1.5 times TDP), but its regular use is somewhere between the average and peak levels.

- clock rate can be reduced dynamically to limit Power Consumption.
- clock rate (freq;) ↓, Power ↓, not energy.

$$\text{Energy(dynamic)} = \frac{1}{2} \times \text{capacitive load} \times \text{voltage}^2$$

$$\text{Power (dynamic)} = \frac{1}{2} \times \text{capacitive load} \times \text{voltage}^2 \times \text{freq. switched.}$$

Here's a simple explanation of these power-saving techniques:

- **Do nothing well** – When the system is idle, it efficiently reduces power consumption instead of wasting energy.

- **Dynamic Voltage-Frequency Scaling (DVFS)** – Adjusts the CPU's voltage and clock speed based on workload to save power.
- **Low power state for DRAM, disks** – Puts memory and storage devices into a low-energy mode when not in use.
- **Overclocking, turning off cores** – Overclocking increases performance but uses more power; turning off unused CPU cores saves energy.

Power (static) = Current (static) × Voltage.

Power gating is a technique used to save energy by completely turning off power to parts of a chip (like CPU cores) when they are not in use. This reduces leakage power, which is the small amount of energy wasted even when a component is idle. It helps improve battery life and overall efficiency in devices.

- Power(static) is 25–50% of Total Power. It is proportional to the product of static current & voltage. It scales with no. of transistors & power gating can be used to reduce static Power.

- failure Rate (λ) = $\frac{1}{MTTF}$ (failure per hour)

- failure Rate (FIT) = $\frac{10^9}{MTTF}$ (failure Per 10^9 hour)

- if a CPU is 10 times faster than other (n = 10)
- computation in Parallel
- I/O in Serial

for Amelhal's law.