LOVELY
PROFESSIONAL
UNIVERSITY

Lecture # 1

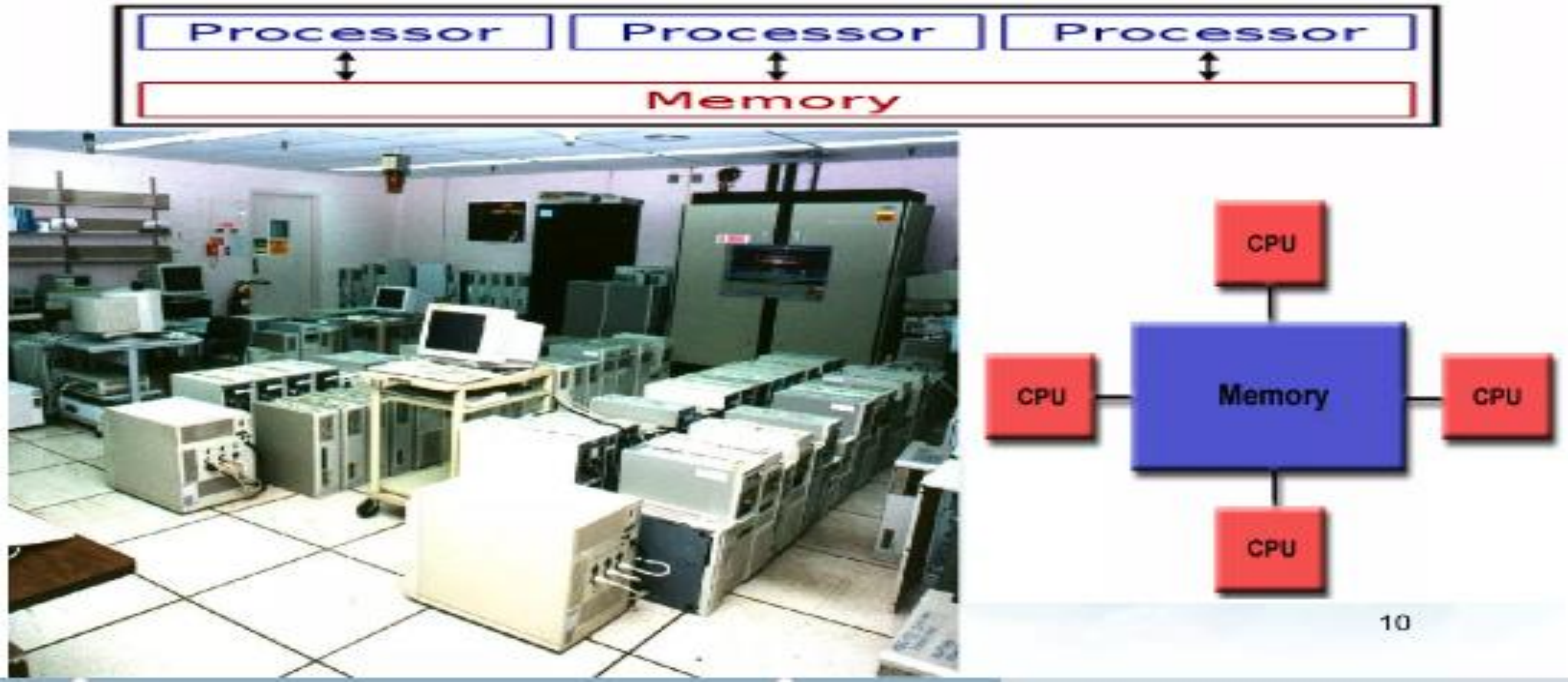
CSE 423

Parallel And Distributed Systems

Parallel Systems

- **DEFINATION:** A system is said to be a parallel system in which multiple processor have direct access to shared memory which forms a common address space.
- Usually tightly-coupled system are referred to as parallel system. In this systems there is a single system wide primary memory that is shared by all the processors.
- Parallel computing is the use of two or more processors in combination to solve a single problem.

Parallel System



Applications of Parallel System

- Manufacturing
- Life Sciences
- Finance
- Electronic design automation

Advantages of Parallel System

- Cost efficiency
- Power consumption
- High performance
- Improve utilization

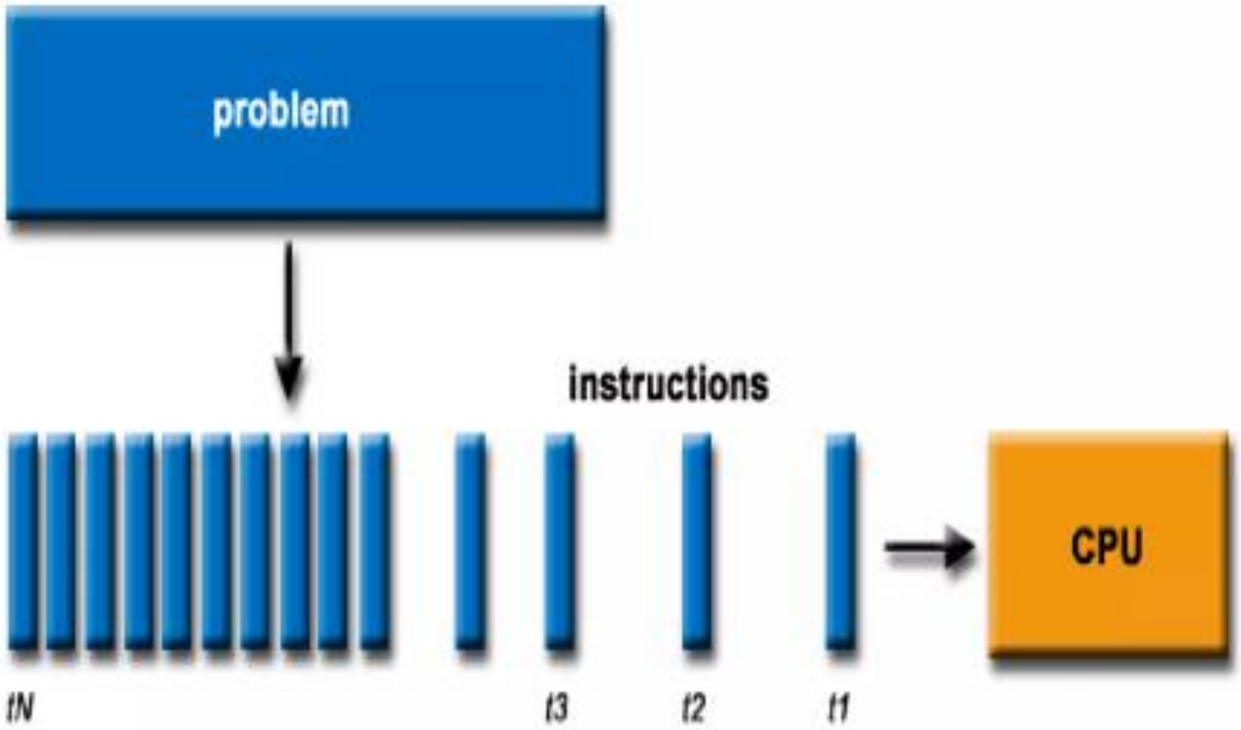
Disadvantages of Parallel System

- It addresses such as communication and synchronization between multiple sub-tasks and processes which is difficult to achieve.
- The algorithms must be managed in such a way that they can be handled in the parallel mechanism.
- The algorithms or program must have low coupling and high cohesion. But it’s difficult to create such programs.
- More technically skilled and expert programmers can code a parallelism based program well.

Why Parallel Computing ?

- Traditionally, software has been written for serial computation:
    - To be run on a single computer having a single CPU;
    - A problem is broken into a discrete series of instructions.
    - Instructions are executed one after another.
    - Only one instruction may execute at any moment of time.

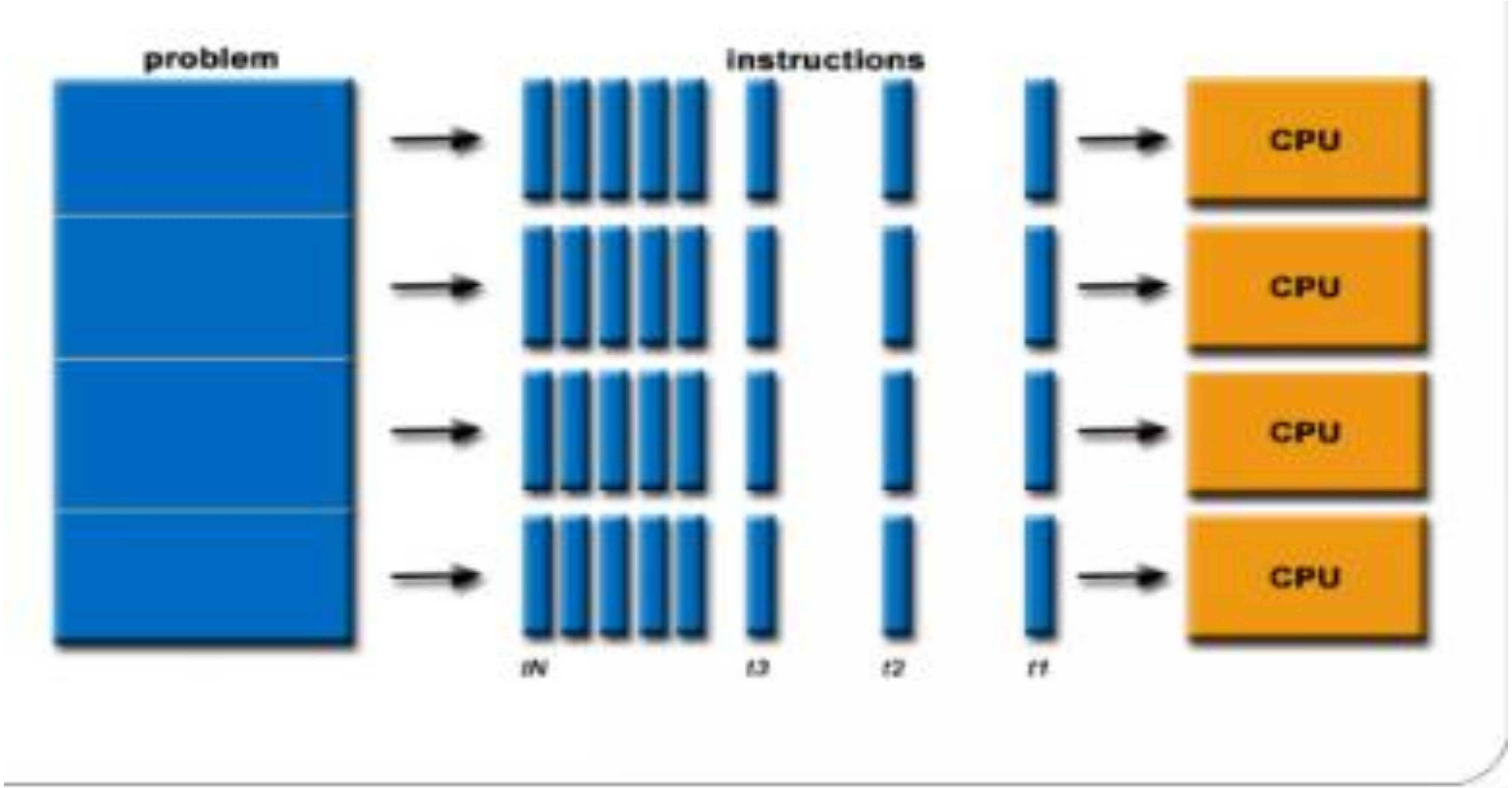
Serial Problem…



Parallel Computing…

- Parallel computing is a form of computation in which many calculations are carried out simultaneously.
- In the simplest sense, it is the simultaneous use of multiple compute resources to solve a computational problem:
  - To be run using multiple CPUs
  - A problem is broken into discrete parts that can be solved concurrently
  - Each part is further broken down to a series of instructions
  - Instructions from each part execute simultaneously on different CPUs.

Parallel Problem



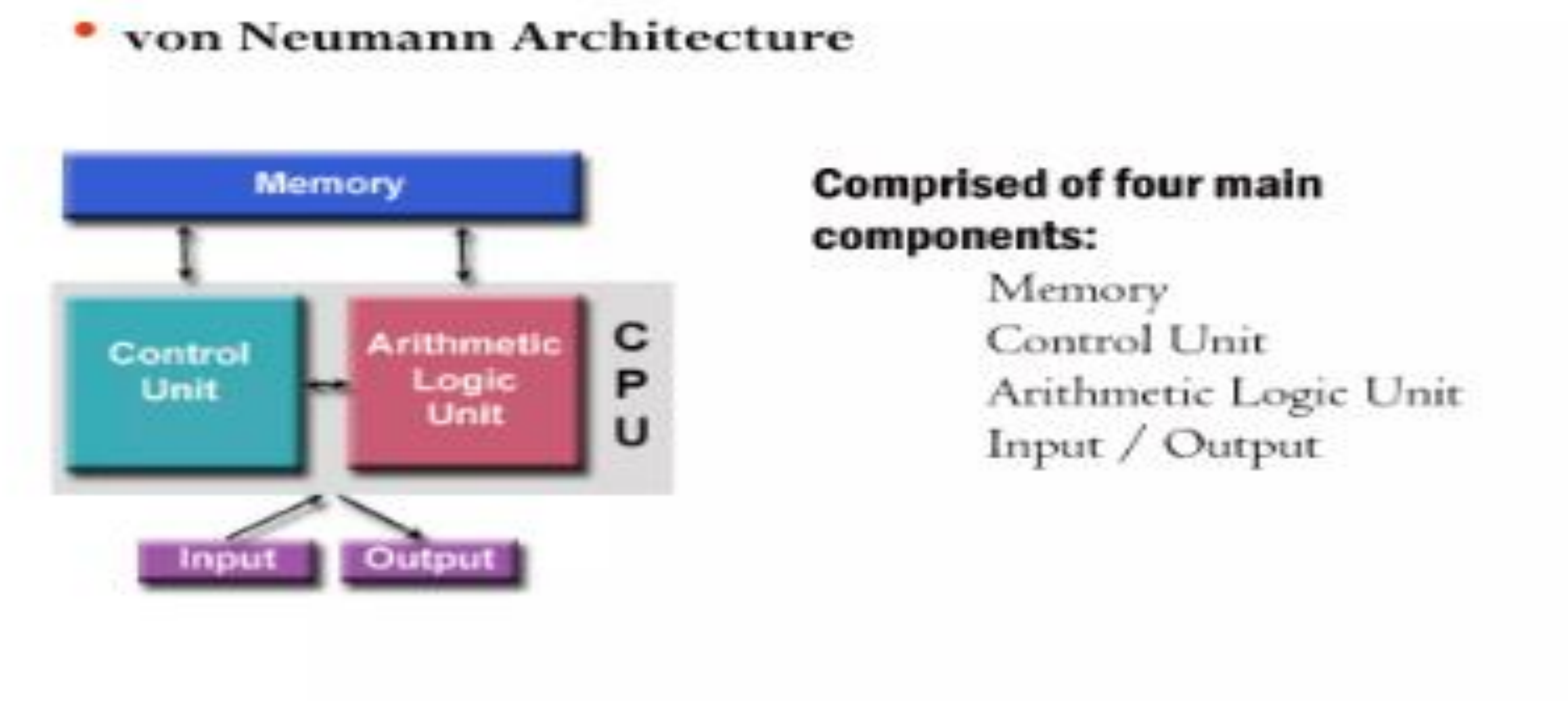
Why use Parallel Computing?

- Save time and/or money
- Solve large problems: e.g. – Web search engines/ databases processing millions of transactions per second.
- Provide concurrency;
- Use of non-local resources;

Why use Parallel Computing?…

- Limits to serial computing:
  - Transmission speeds
  - Limits to miniaturization
  - Economic limitations
- Current computer architecture are increasingly relying upon hardware level parallelism to improve performance:
  - Multiple execution units
  - Pipelined instructions
  - Multi- core

Parallel Computer Architecture



- von Neumann Architecture

Comprised of four main components:

Memory
Control Unit
Arithmetic Logic Unit
Input / Output

Parallel Computer Architecture…

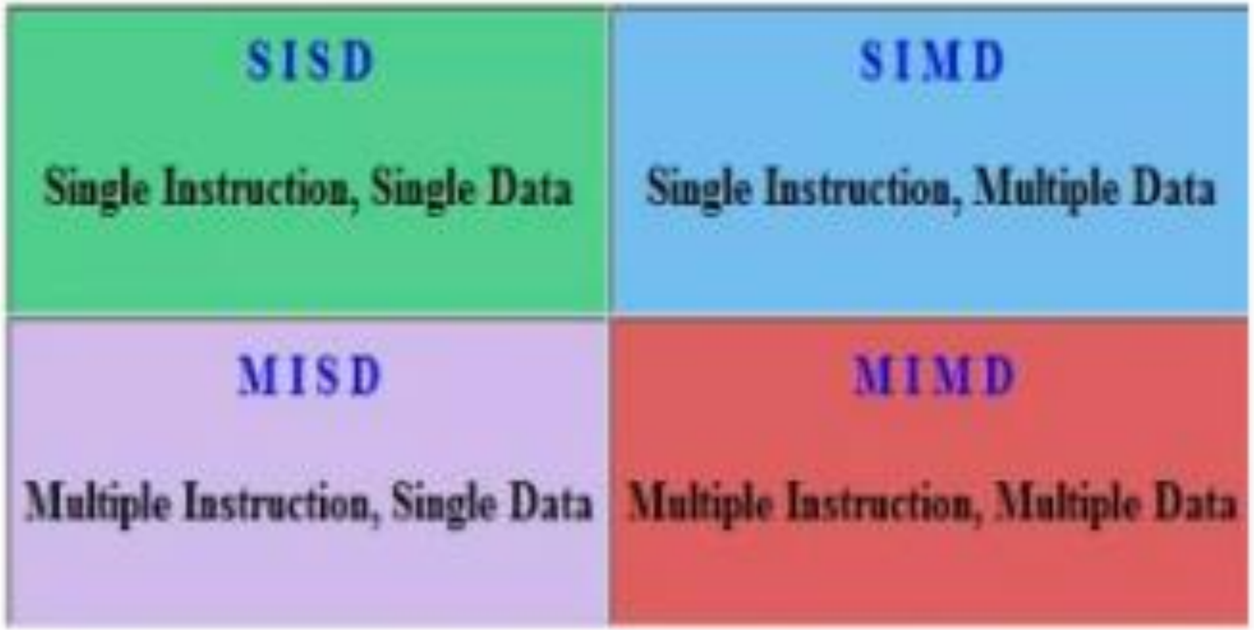
- **Memory** is used to store both program instructions and data
  - **Program instructions** are coded data which tell the computer to do something
  - **Data** is simply information to be used by the program
- **Control unit** fetches instructions/data from memory, decodes the instructions and then sequentially coordinates operations to accomplish the programmed task.

Parallel Computer Architecture…

- **Arithmetic unit** That performs basic arithmetic operations
- **Input/Output** is the interface to the human operator

Parallel Computer Architecture…

Flynn’s classical Taxonomy:



SISD Single Instruction, Single Data	SIMD Single Instruction, Multiple Data
MISD Multiple Instruction, Single Data	MIMD Multiple Instruction, Multiple Data

Single Instruction, Single Data (SISD):

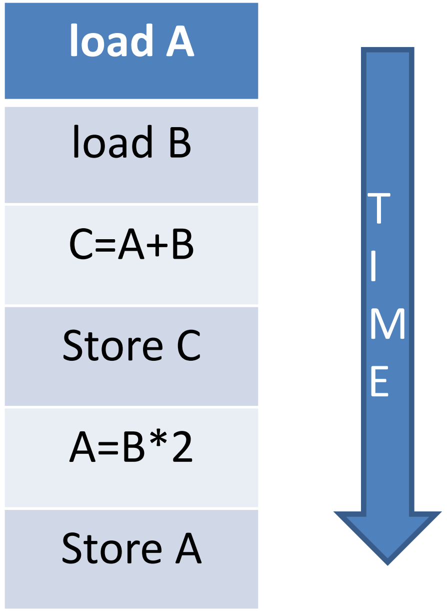
- A serial (non parallel) computer

Single Instruction:

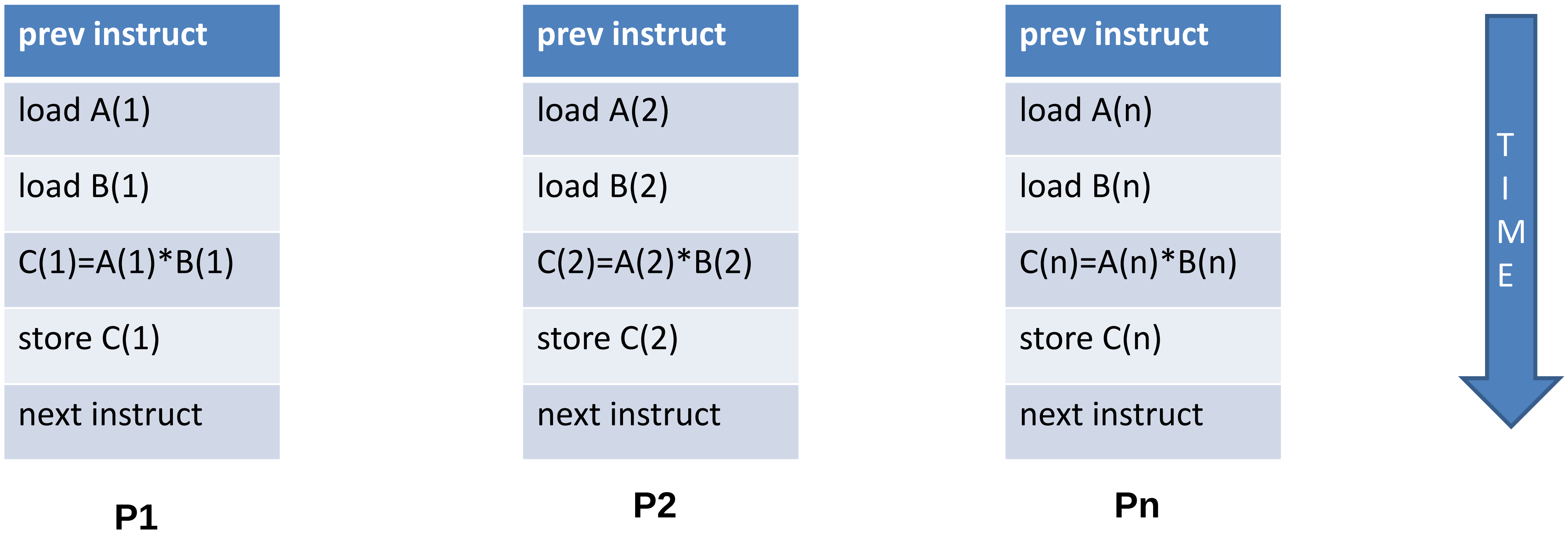
  - Only one instruction stream is being acted on by the CPU during anyone clock cycle.

Single Data :

- Only one data stream is being used as input during any one clock cycle.
- Deterministic execution.



Single Instruction, Multiple Data (SIMD):



P1	P2	Pn
prev instruct	prev instruct	prev instruct
load A(1)	load A(2)	load A(n)
load B(1)	load B(2)	load B(n)
C(1)=A(1)*B(1)	C(2)=A(2)*B(2)	C(n)=A(n)*B(n)
store C(1)	store C(2)	store C(n)
next instruct	next instruct	next instruct

Single Instruction, Multiple Data (SIMD):

A type of parallel computer

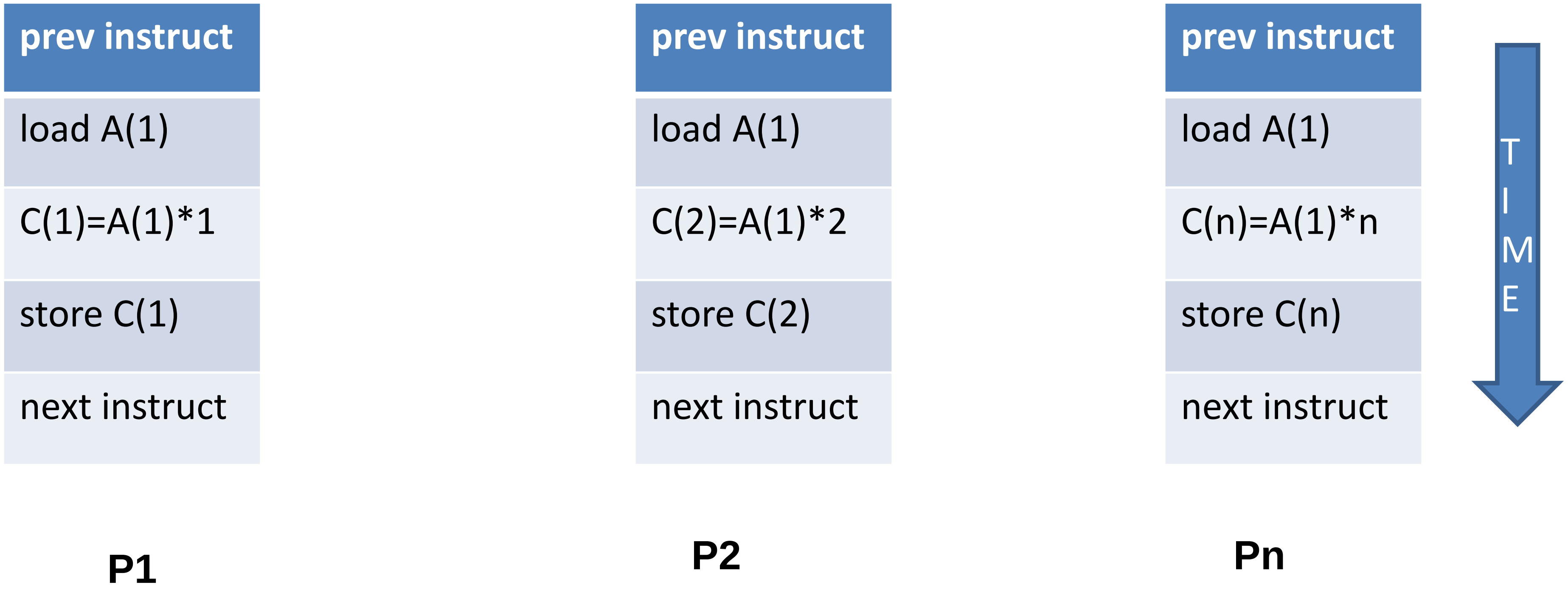
Single instruction:

- All processing units execute the same instruction at any given clock cycle

Multiple data:

- Each processing unit can operate on a different data element.
- Best suited for specialize problems characterized by a high degree of regularity, such as graphics/image processing.
- Two varieties processor arrays and vector pipelines.

Multiple Instruction, Single Data (MISD):

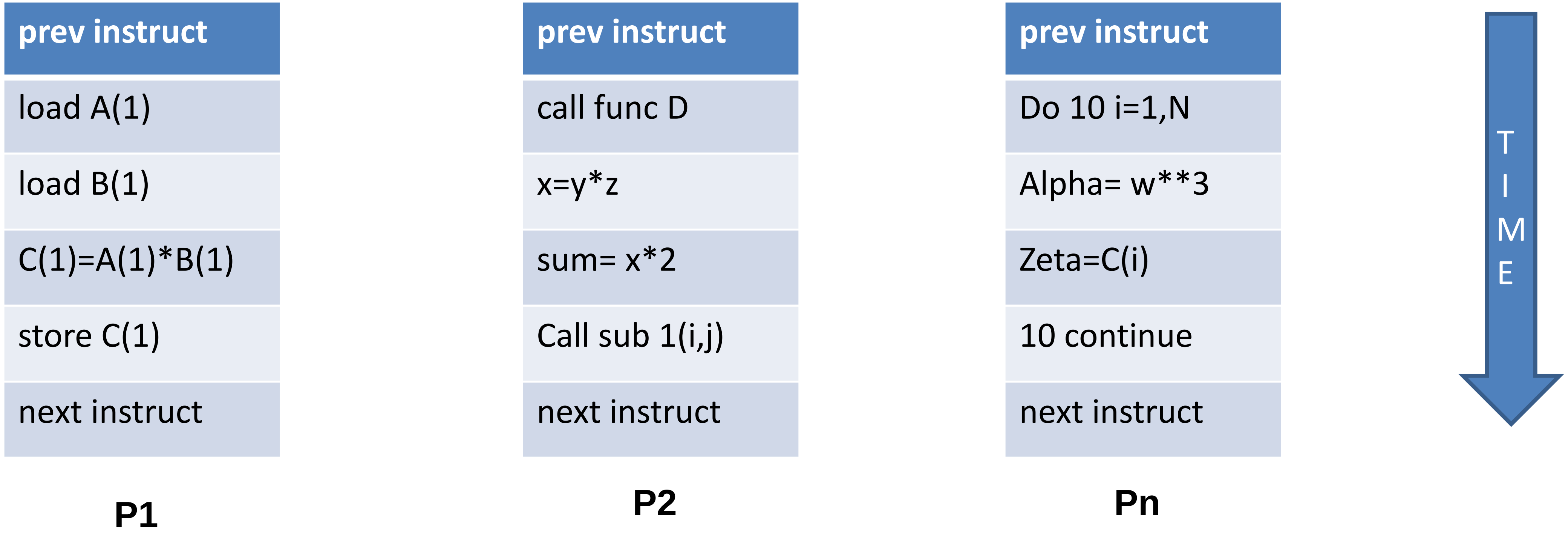


P1	P2	Pn
prev instruct	prev instruct	prev instruct
load A(1)	load A(1)	load A(1)
C(1)=A(1)*1	C(2)=A(1)*2	C(n)=A(1)*n
store C(1)	store C(2)	store C(n)
next instruct	next instruct	next instruct

Multiple Instruction, Single Data (MISD):

- A single data stream is fed into multiple processing units.
- Each processing unit operates on the data independently via independent instruction streams.
- Some conceivable usage might be:
  - Multiple frequency filters operating on a single signal stream

Multiple Instruction, Multiple Data (MIMD):



prev instruct	prev instruct	prev instruct
load A(1)	call func D	Do 10 i=1,N
load B(1)	x=y*z	Alpha= w**3
C(1)=A(1)*B(1)	sum= x*2	Zeta=C(i)
store C(1)	Call sub 1(i,j)	10 continue
next instruct	next instruct	next instruct
P1	**P2**	**Pn**

Multiple Instruction, Multiple Data (MIMD):

- **Multiple Instruction:** Every processor may be executing a different instruction stream.
- **Multiple Data:** Every processor may be working with a different data stream.
- **Execution** can be synchronous or asynchronous, deterministic or non- deterministic.

THANK YOU

&

ANY QUERIES?