1. List and explain the parallel processing mechanisms in uni processor computers.

A number of parallel processing mechanisms have been developed in uniprocessor computers. We identify them in the following six categories:

- **Multiplicity of functional units**
- **Parallelism and pipelining within the CPU**
- **Overlapped CPU and I/O operations**
- **Use of a hierarchical memory system**
- **Balancing of subsystem bandwidths**
- **Multiprogramming and time sharing**

Multiplicity of functional units: The early computer had only one arithmetic and logic unit in its CPU. Furthermore, the ALU could only perform one function at a time, a rather slow process for executing a long sequence of arithmetic logic instructions.

In practice, many of the functions of the ALU can be distributed to multiple and specialized functional units which can operate in parallel. The CDC-6600 (designed in 1964) has 10 functional units built into its CPU.

These 10 units are independent of each other and may operate simultaneously. A scoreboard is used to keep track of the availability of the functional units and registers being demanded. With 10 functional units and 24 registers available, the instruction issue rate can be significantly increased.

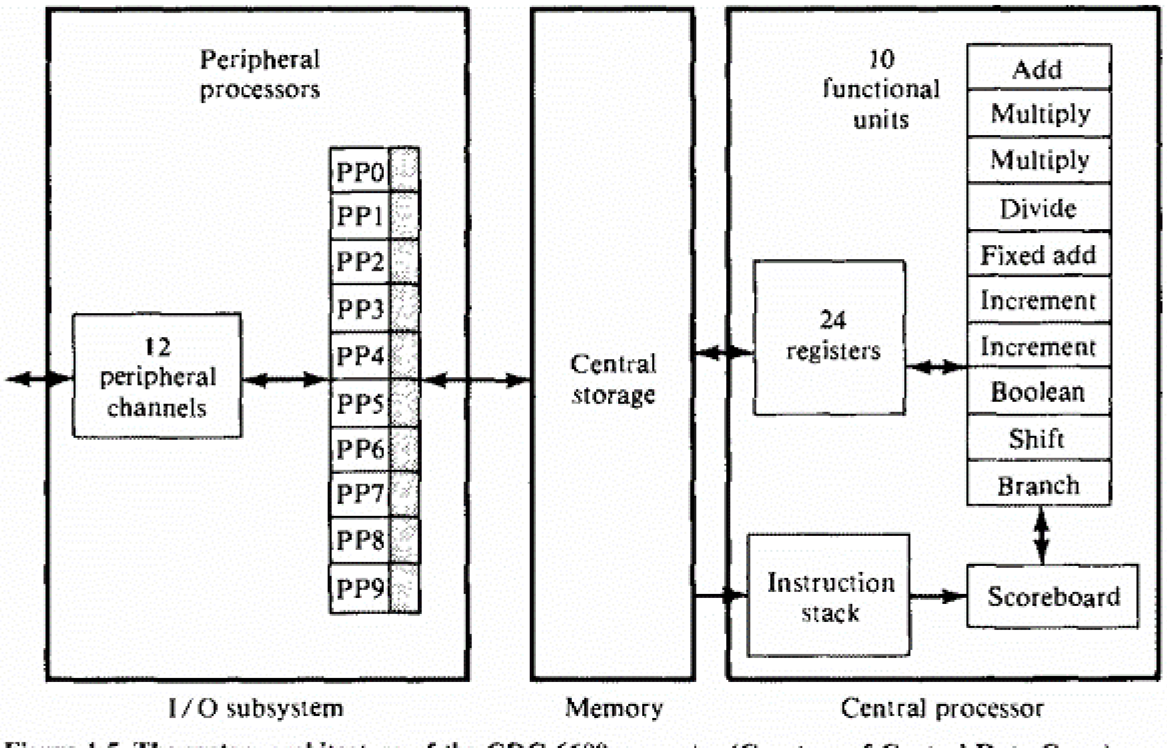


Figure 1.5 The system architecture of the CDC-6600 computer (Courtesy of Control Data Corp.).

Another good example of a multifunction uniprocessor is the IBM 360/91 (1968), which has two parallel execution units (E units): one for fixed-point arithmetic, and the other for floating-point arithmetic.

Within the floating-point E unit are two functional units: one for floating-point add-subtract and the other for floating-point multiply-divide.

IBM 360/91 is a highly pipelined, multifunction, scientific uniprocessor.

Parallelism and pipelining within the CPU: Parallel adders, using such techniques as carry-lookahead and carry-save, are now built into almost all ALUs. This is in contrast to the bit-serial adders used in the first-generation machines.

High-speed multiplier recoding and convergence division are techniques for exploring parallelism and the sharing of hardware resources for the functions of multiply and divide. The use of multiple functional units is a form of parallelism with the CPU.

Various phases of instruction executions are now pipelined, including instruction fetch, decode, operand fetch, arithmetic logic execution, and store result.

To facilitate overlapped instruction executions through the pipe, instruction prefetch and data buffering techniques have been developed. Most commercial uniprocessor systems are now pipelined in their CPU with a clock rate between 10 and 500 ns.

Overlapped CPU and I/O operations: I/O operations can be performed simultaneously with the CPU computations by using separate I/O controllers, channels, or I/O processors.

The direct-memory-access (DMA) channel can be used to provide direct information transfer between the I/O devices and the main memory. The DMA is conducted on a cycle-stealing basis, which is apparent to the CPU.

Furthermore, I/O multiprocessing, such as the use of the 10 1/0 processors in CDC-6600, can speed up data transfer between the CPU (or memory) and the outside world. Back-end database machines can be used to manage large databases stored on disks.

Use of hierarchical memory system: Usually, the CPU is about 1000 times faster than memory access. A hierarchical memory system can be used to close up the speed gap.

Computer memory hierarchy is conceptually illustrated in fig. below. The innermost level is the register files directly addressable by ALU. Cache memory can be used to serve as a buffer between the CPU and the main memory. Block access of the main memory can be achieved through multiway interleaving across parallel memory modules. Virtual memory space can be established with the use of disks and tape units at the outer levels.

All these techniques are intended to broaden the memory bandwidth to match that of the CPU.

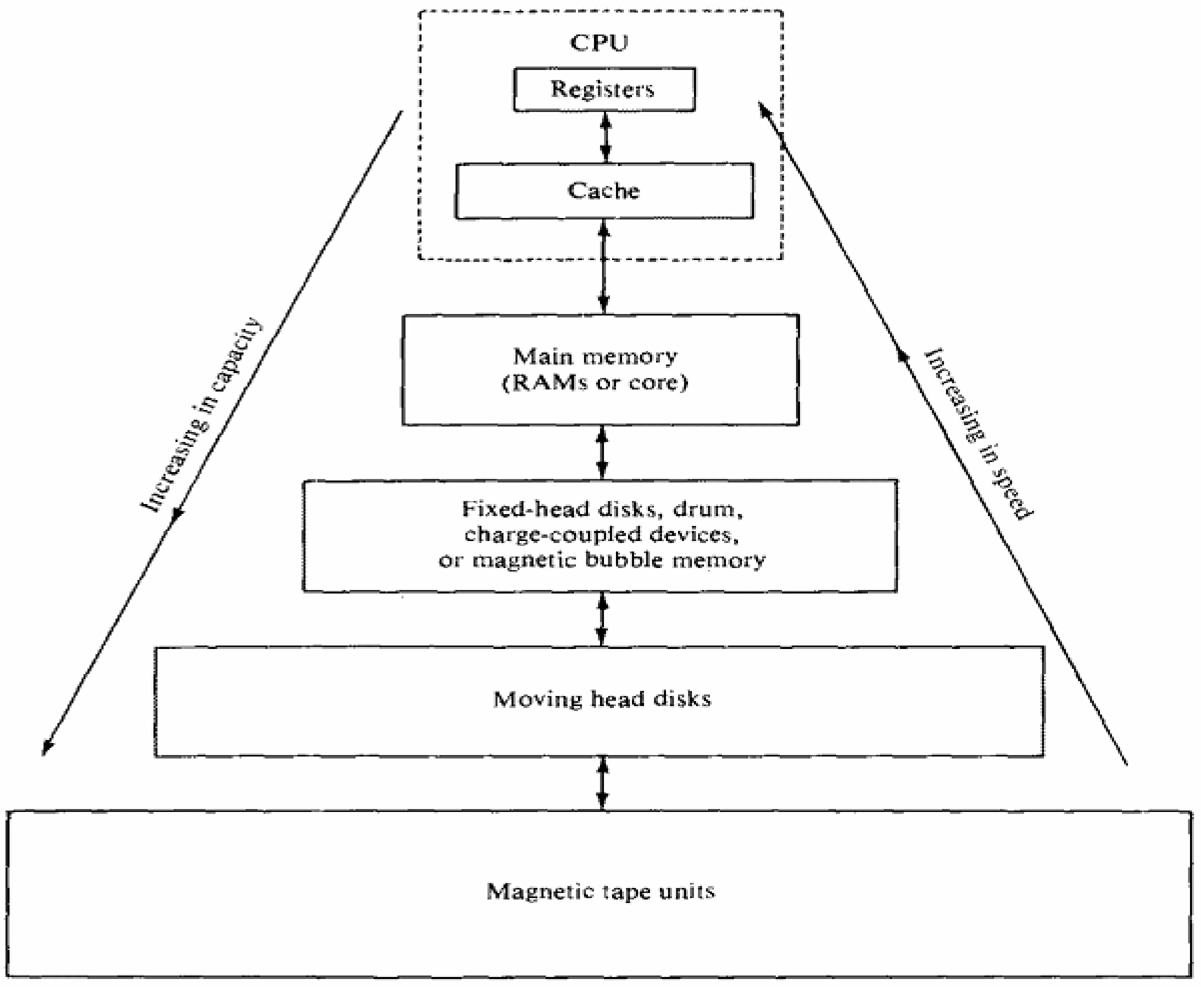


Figure 1.6 The classical memory hierarchy.

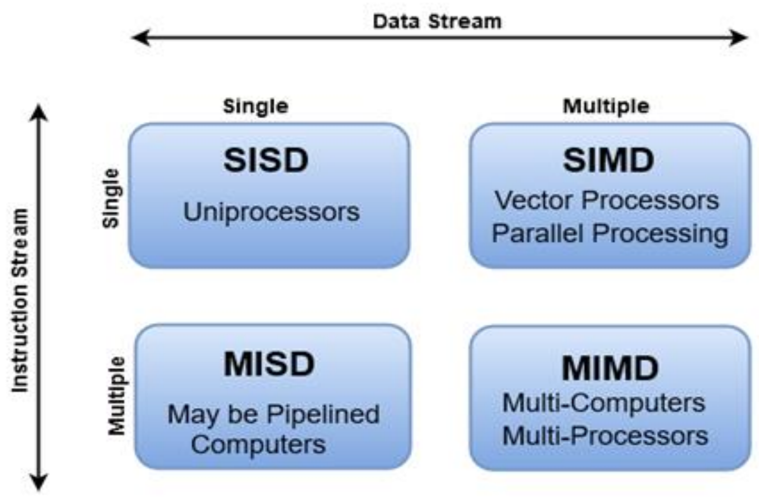
2. Explain the Flynn's classification of computer architectures with neat diagrams.

- In general, digital computers may be classified into four categories, according to the multiplicity of instruction and data streams. This scheme for classifying computer organizations was introduced by Michael J. Flynn.
- The essential computing process is the execution of a sequence of instructions on a set of data.
- The term stream is used here to denote a sequence of items (instructions or data) as executed or operated upon by a single processor.
- Instructions or data are defined with respect to a referenced machine. An instruction stream is a sequence of instructions as executed by the machine; a data stream is a sequence of data including input, partial, or temporary results, called for by the instruction stream.
- Computer organizations are characterized by the multiplicity of the hardware provided to service the instruction and data streams. Listed below are Flynn's four machine organizations:

1. <u>Single instruction stream, single data stream (SISD)</u>

2. Single instruction stream, multiple data stream (SIMD)
3. Multiple instruction stream, single data stream (MISD)
4. Multiple instruction stream, multiple data stream (MIMD)

Flynn's Classification of Computers



1. SISD

i. SISD stands for 'Single Instruction and Single Data Stream'. It represents the organization of a single computer containing a control unit, a processor unit, and a memory unit.

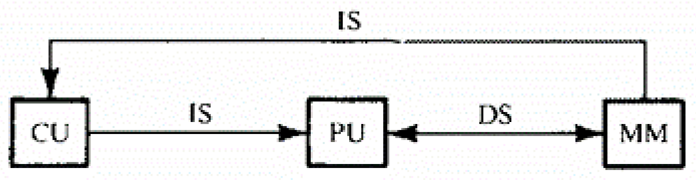
ii. An SISD computer may have more than one functional unit in it. All the functional units are under the supervision of one control unit

iii. Instructions are executed sequentially but may be overlapped in their execution stages (pipelining).

iv. Most SISD uniprocessor systems are pipelined.

v. Most conventional computers have SISD architecture like the traditional Von-Neumann computers.

vi. The speed of the processing element in the SISD model is limited(dependent) by the rate at which the computer can transfer information internally. Dominant representative SISD systems are IBM PC, workstations.



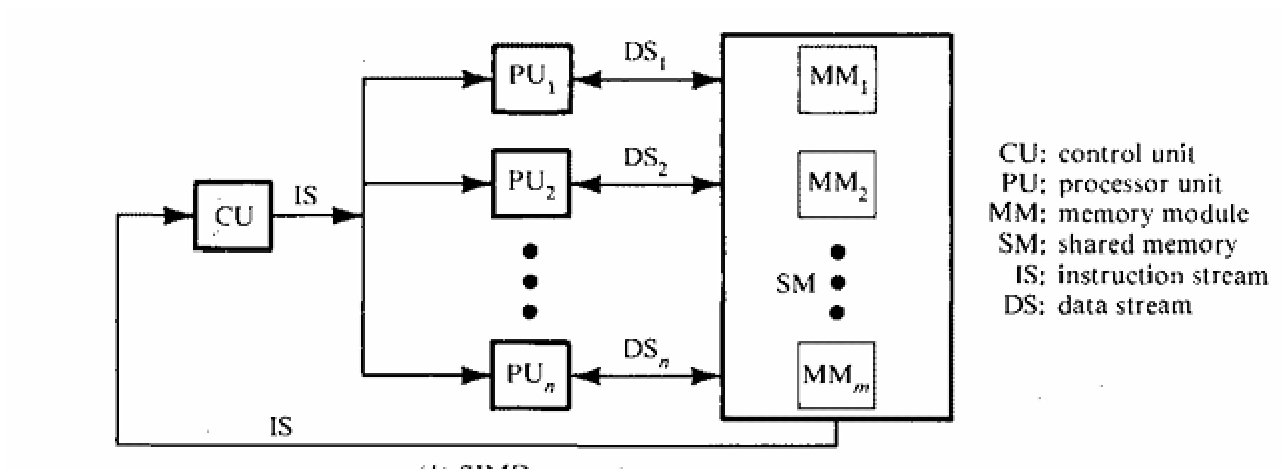
(a) SISD computer

vii. Instructions are decoded by the Control Unit and then the Control Unit sends the instructions to the processing units for execution.

viii. Data Stream flows between the processors and memory bi-directionally.

2. SIMD:

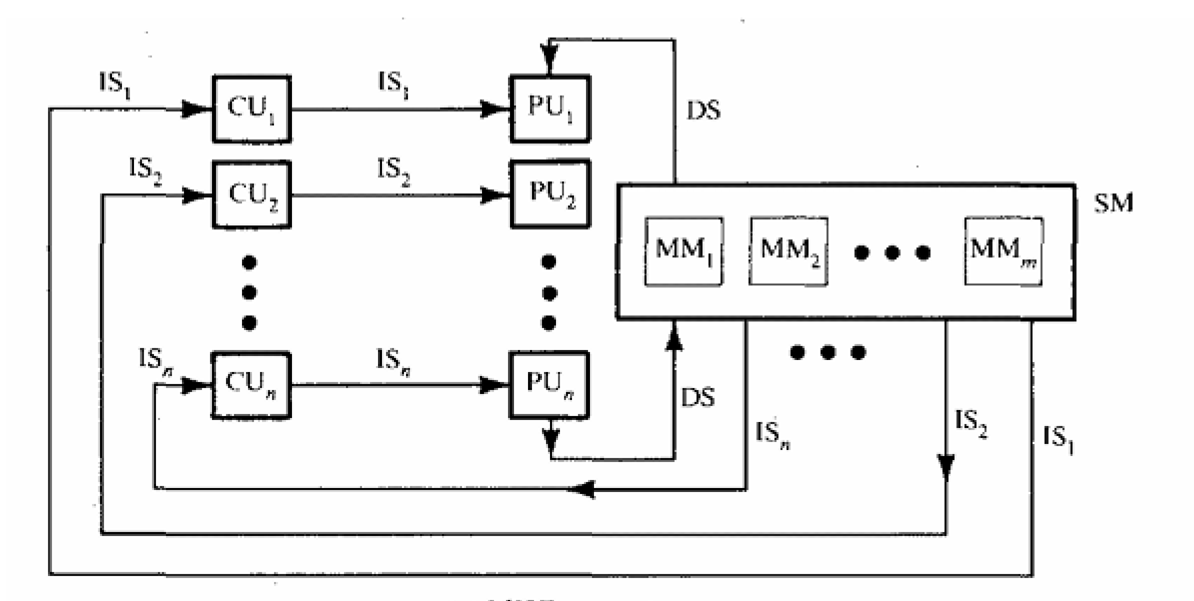
i. SIMD stands for 'Single Instruction and Multiple Data Stream'. It represents an organization that includes many processing units under the supervision of a common control unit.
ii. SIMD is mainly dedicated to array processing machines. However, vector processors can also be seen as a part of this group.
iii. All processors receive the same instruction from the control unit but operate on different items of data.
iv. The shared memory unit must contain multiple modules so that it can communicate with all the processors simultaneously.
v. The SIMD machines are further divided into word-slice versus bit-slice modes.



(*b*) SIMD computer

3. MISD

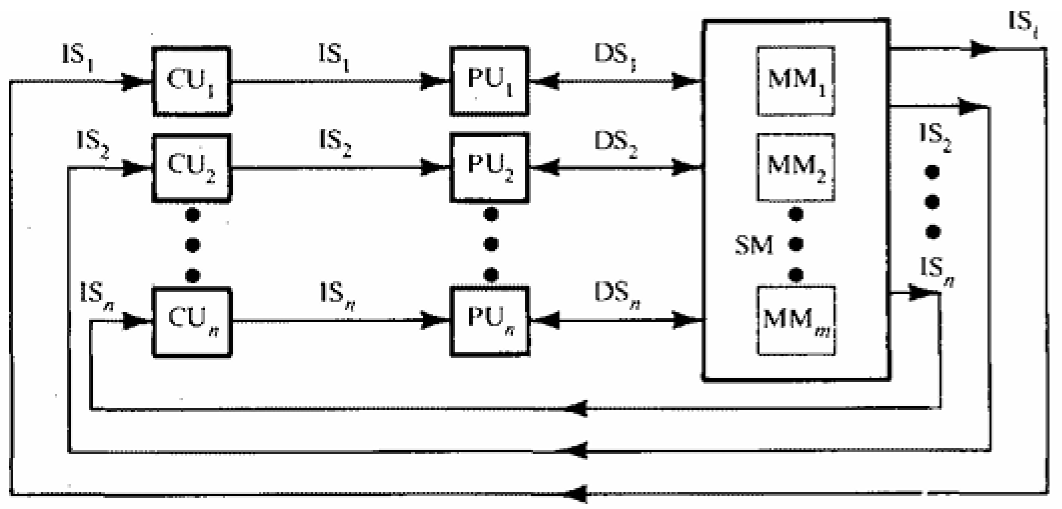
i. MISD stands for 'Multiple Instruction and Single Data stream'.
ii. There are n processor units, each receiving distinct instructions operating over the same data stream and its derivatives.
iii. The results (output) of one processor become the input (operands) of the next processor in the macropipe.
iv. This structure has received much less attention and has been challenged as impractical by some computer architects. No real embodiment of this class exists.
v. Example: The experimental Carnegie-Mellon C.mmp computer (1971)



(*c*) MISD computer

4. MIMD:

i. MIMD stands for 'Multiple Instruction and Multiple Data Stream'.
ii. Most multiprocessor systems and multiple computer systems can be classified in this category (Figure 1.16d). An intrinsic MIMD computer implies interactions among the n processors because all memory streams are derived from the same data space shared by all processors.
iii. If the n data streams were derived from disjointed subspaces of the shared memories, then we would have the so-called multiple SISD (MSISD) operation, which is nothing but a set of n independent SISD uniprocessor systems.
iv. An intrinsic MIMD computer is tightly coupled if the degree of interactions among the processors is high. Otherwise, we consider them loosely coupled. Most commercial MIMD computers are loosely coupled.
v. Examples: Cray T90, Cray T3E, IBM-SP2



(*d*) MIMD computer

3. Explain the functional structure of SIMD array processor.

The SIMD (Single Instruction, Multiple Data) array processor is a specialized type of array processor designed for parallel computation. Here’s an explanation of its functional structure based on the provided information:

Array Processor Overview: The SIMD array processor is a synchronous parallel computer featuring multiple Processing Elements (PEs), each equipped with an Arithmetic Logic Unit (ALU) and local memory. PEs operate in lock-step, executing the same instruction simultaneously on different data elements.

Parallelism and ALU Replication: Spatial parallelism is achieved through the replication of ALUs across PEs. This allows multiple data elements to be processed simultaneously, enhancing computational throughput.

Control and Instruction Execution:

- Scalar and control-type instructions are executed centrally by the Control Unit (CU), which coordinates the operation of all PEs.
- Vector instructions are broadcasted to all PEs, enabling distributed execution over operands fetched directly from local memories of each PE.

Data Routing and Interconnection:

- A data-routing network interconnects the PEs, facilitating communication and data exchange among them.
- The CU controls the interconnection pattern based on program requirements, directing data flow and synchronization among PEs.

Instruction Handling:

- Instruction fetch and decode are managed by the CU, which fetches instructions either from local memories or from a centralized control memory.
- PEs themselves are passive regarding instruction decoding, focusing solely on executing instructions synchronized with other PEs.

Interconnection Networks:
Different array processors employ various interconnection networks among PEs, such as mesh-structured networks (e.g., Illiac-IV) or crossbar networks (e.g., Burroughs Scientific Processor).

Applications and Performance:

- Array processors excel in executing parallel algorithms like matrix multiplication, merge sort, and FFT due to their ability to process large datasets simultaneously.
- Performance evaluation emphasizes resource optimization and efficient utilization of parallel computing capabilities.

Challenges and Enhancements:

- Programming array processors is more complex compared to sequential or pipeline architectures.
- Performance enhancements include techniques like skewed memory allocation, language extensions supporting vector-array processing, and ongoing architectural improvements.

The functional structure of a SIMD array processor revolves around synchronized PEs executing parallel instructions on multiple data elements, coordinated by a central control unit and interconnected via a data-routing network. This design optimizes parallel computation for tasks requiring high throughput and simultaneous data processing.

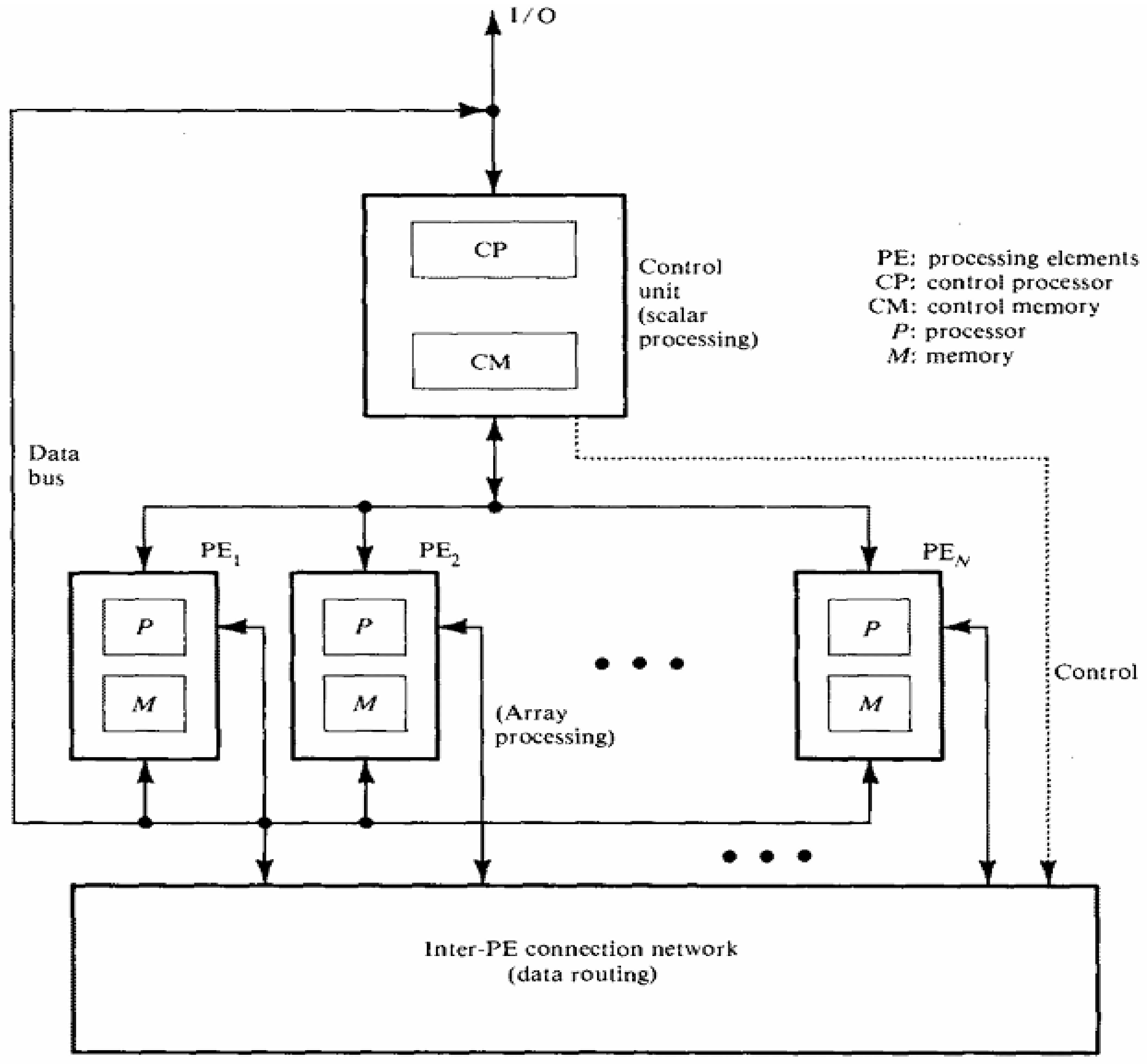


Figure 1.12 Functional structure of an SIMD array processor with concurrent scalar processing in the control unit.

4. What is Array Processor?

A processor which is used to perform different computations on a huge array of data is called an array processor. The other terms used for this processor are vector processors or multiprocessors. This processor performs only single instruction at a time on an array of data. These processors work with huge data sets to execute computations. So, they are mainly used for enhancing the performance of computers.

Array Processor Architecture
An array processor includes a number of ALUs (Arithmetic Logic Units) which allows all the array elements to be processed together. Each ALU in the processor is provided with local memory which is known as a Processing Element or PE. The architecture of this processor is shown below. By using this processor, a single instruction is issued through a control unit & that instruction is simply applied to a number of data sets simultaneously. By using a single instruction, a similar operation is performed on an array of data which makes it suitable for vector computations.

Array Processor Architecture
Array Processor Architecture
The array processing architecture is known as a 2-dimensional array or matrix. This architecture is implemented by the two-dimensional processor. In this processor, the CPU issues a single instruction & after that, it is applied to a no. of data simultaneously. This architecture mainly depends on the fact that all data sets work on similar instructions, however, if these data sets are reliant on each other, it is not achievable to apply parallel processing. Thus these processors contribute efficiently & enhance the processing speed as compared to the whole instructions.

Working of Array Processor
An array processor has an architecture mainly designed for processing arrays of numbers. This processor architecture contains a number of processors that works simultaneously, each handling one array element, so that a single operation is applied to all the array elements in parallel. To get the same effect within a conventional processor, the operation should be applied to every array element sequentially and much more slowly.

This processor is a self-contained unit connected to the main computer through an internal bus or an I/O port. This processor increases the overall speed of instruction processing. These processors operate asynchronously from the host CPU to improve the overall system capacity. This processor is a very powerful tool that handles troubles with a high level of parallelism.