1. List & Explain Parallel Processing Mechanisms in Uniprocessor Computers

Parallel Processing in Uniprocessor Computers:

- **Definition:** Parallel processing allows a computer to execute multiple instructions at the same time. In a uniprocessor system, which has only one CPU, this is achieved through various methods that allow tasks to be processed concurrently.

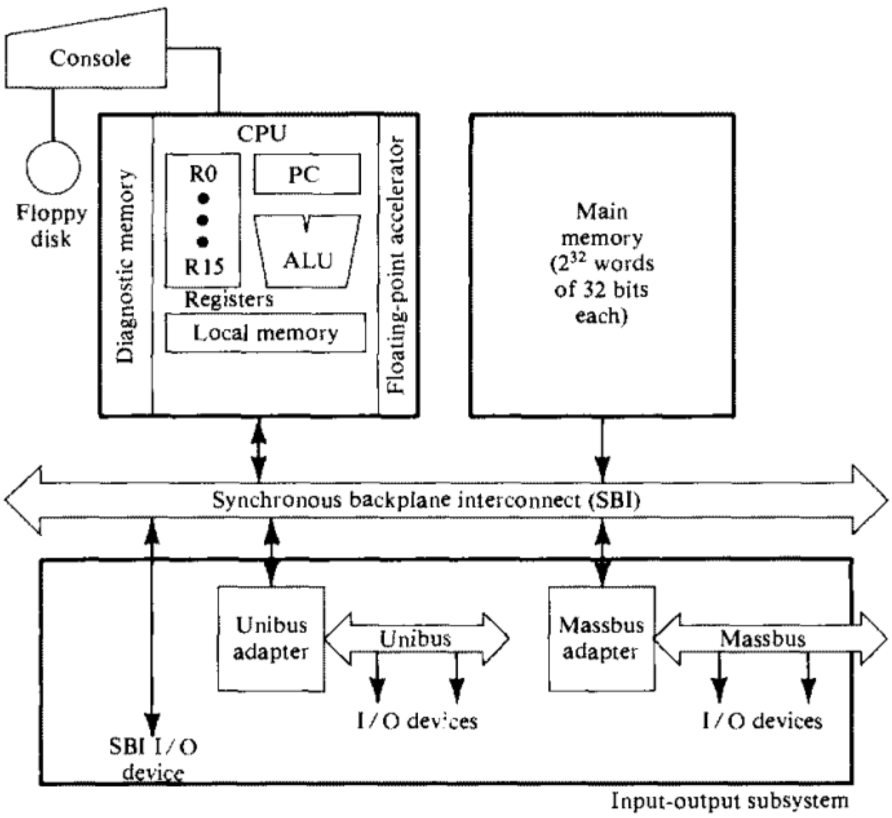


Figure 1.3 The system architecture of the supermini VAX-11/780 uniprocessor system (Courtesy of Digital Equipment Corporation).

-

Mechanisms:

1. **Pipelining:**
    - **Concept:** Pipelining breaks down a task into smaller stages, with each stage processed simultaneously.
    - **Explanation:** Imagine a conveyor belt in a factory where each worker performs a different task on the product as it moves along the belt. While one worker is painting, another is assembling, and so on. This simultaneous work speeds up the overall process.
    - **Advantages:** Increases CPU efficiency by allowing multiple instructions to be processed at different stages of completion.
2. **Multithreading:**

- **Concept:** Multithreading allows multiple threads (small tasks) to run in the CPU at the same time, with the CPU quickly switching between them.
- **Explanation:** Think of someone managing several tasks at once, like cooking and cleaning. They do a little bit of each task in turn, so both tasks get done faster than if they were done one after the other.
- **Advantages:** Keeps the CPU active and reduces idle time, leading to better performance.

3. **Instruction-Level Parallelism (ILP):**
    - **Concept:** ILP enables the CPU to execute multiple instructions that are not dependent on each other at the same time.
    - **Explanation:** Imagine reading a book while listening to music. Since these activities don't interfere with each other, you can do both simultaneously, just like how ILP allows independent instructions to be processed together.
    - **Advantages:** Improves performance by making use of the CPU's ability to handle multiple operations at once.
4. **Superscalar Execution:**
    - **Concept:** Superscalar execution allows the CPU to dispatch multiple instructions to different execution units within the processor during a single clock cycle.
    - **Explanation:** Think of a highway with multiple lanes where different cars (instructions) can travel side by side. This reduces congestion and allows more cars to reach their destination faster.
    - **Advantages:** Enhances the throughput of the processor by enabling it to execute several instructions in parallel.
5. **Simultaneous Multithreading (SMT):**
    - **Concept:** SMT is an advanced form of multithreading that allows multiple threads to execute on different parts of the CPU at the same time.
    - **Explanation:** Imagine two people working on the same task but each handling different parts of it simultaneously. This way, the task is completed faster because both are contributing at the same time.
    - **Advantages:** Improves CPU utilization by allowing more threads to run concurrently, leading to better overall performance.
6. **Branch Prediction:**
    - **Concept:** Branch prediction is a technique where the CPU guesses the direction of a branch (like an if-else decision) and begins executing instructions ahead of time.
    - **Explanation:** Think of planning your route while driving. If you predict correctly that you'll turn left, you can start slowing down and preparing. If you're wrong, you just correct and move on, but you've saved time if you're right.

   - **Advantages:** Reduces delays in instruction execution by allowing the CPU to prepare in advance, improving overall processing speed.

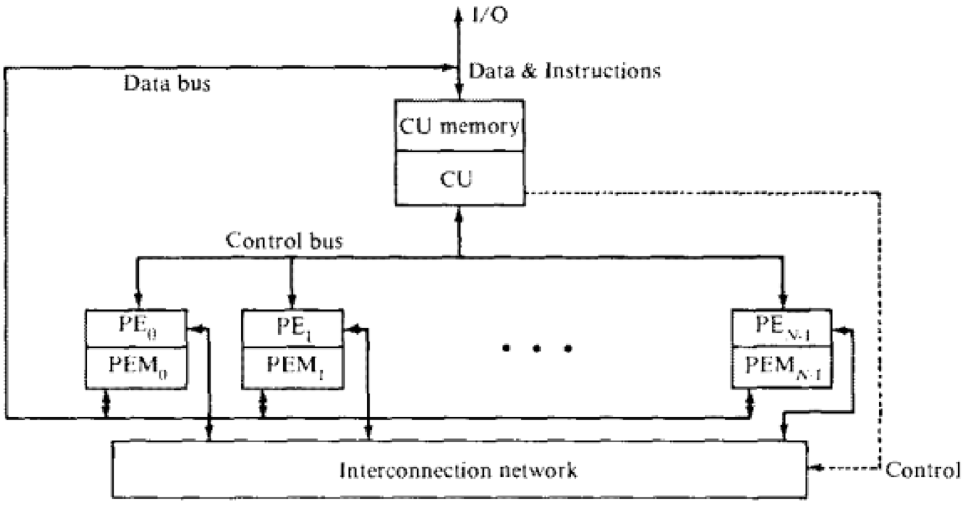
7. **Out-of-Order Execution:**
   - **Concept:** The CPU can execute instructions out of order, based on the availability of data and resources, rather than strictly following the original sequence.
   - **Explanation:** Picture a chef in a busy kitchen who starts cooking whichever dish has all the ingredients ready, rather than waiting for each dish to be prepared in the order they were ordered. This way, the kitchen works more efficiently.
   - **Advantages:** Increases the CPU's efficiency by allowing it to execute tasks as soon as the necessary resources are available, rather than waiting for previous tasks to complete.

8. **Cache Memory:**
   - **Concept:** Cache memory is a small, fast memory located close to the CPU, which stores frequently accessed data and instructions.
   - **Explanation:** Imagine you are studying and keep important notes within arm's reach. This saves time since you don't have to keep going to the bookshelf every time you need to refer to something. Similarly, cache memory reduces the time the CPU spends accessing data from slower main memory.
   - **Advantages:** Speeds up processing by reducing the time needed to fetch data, allowing the CPU to work faster.

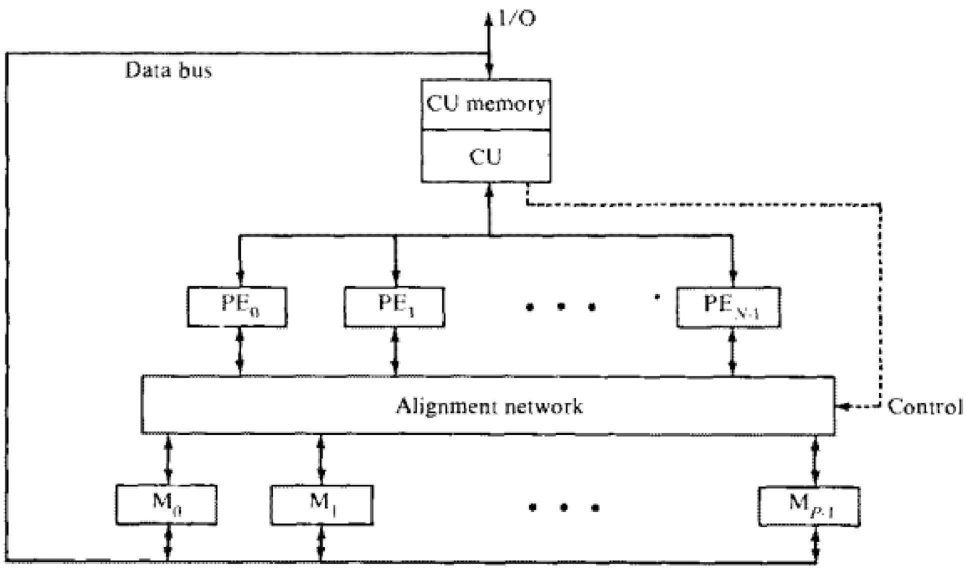
2. Functional Structure of SIMD (Single Instruction, Multiple Data) Array Processor

Definition:

SIMD (Single Instruction, Multiple Data) is a type of parallel computing where one instruction is applied to many data points at the same time. It's like doing the same task on multiple items in one go, making the process faster.



(a) Configuration I (Illiac IV)



(b) Configuration II (BSP)

Figure 5.1 Architectural configurations of SIMD array processors.

Functional Structure:

1. **Control Unit:**
   - **Role:** The control unit is like the boss. It gives out the same instruction to all the workers (Processing Elements or PEs) at the same time.
   - **Diagram Explanation:** Imagine a conductor guiding an orchestra. All the musicians play the same notes simultaneously, following the conductor's lead.
   - **Importance:** Ensures that all parts of the processor work together in harmony.
2. **Processing Elements (PEs):**
   - **Role:** Each PE is like a worker on a production line. They all perform the same task but on different pieces of data.

- **Diagram Explanation:** Think of workers on a farm, where each person picks apples from different trees, but they all follow the same method.
- **Importance:** Allows the processor to handle large amounts of data at once, making it efficient for tasks like image processing.

3. **Interconnection Network:**
    - **Role:** This network connects the control unit to all the PEs and lets them communicate with each other.
    - **Diagram Explanation:** Visualize a network of roads in a city. These roads connect different parts of the city, allowing people and goods (data) to move around efficiently.
    - **Importance:** Ensures smooth communication between different parts of the processor, making sure that all data gets to the right place.
4. **Memory Unit:**
    - **Role:** The memory unit stores the data and instructions that the PEs need to perform their tasks.
    - **Diagram Explanation:** Imagine a warehouse that holds all the supplies. Workers (PEs) come here to get what they need to do their jobs.
    - **Importance:** Provides quick access to data, which speeds up the overall processing time.

Advantages of SIMD:

1. **High Performance:**
    - SIMD is particularly good at handling large sets of data, like images or scientific calculations. It processes many data points in parallel, which makes tasks faster.
2. **Energy Efficiency:**
    - Since SIMD executes the same instruction on multiple data points, it can be more energy-efficient compared to other processing methods, reducing power consumption.
3. **Simplicity in Design:**
    - SIMD's design is straightforward, making it easier to implement in hardware, which can reduce manufacturing costs.
4. **Improved Throughput:**
    - SIMD can significantly increase the number of operations performed in a given time, leading to better overall system performance.

This structure allows SIMD processors to efficiently handle tasks that require the same operation to be performed on a large amount of data, such as processing images, videos, or sound.

3. What is Dependability? State Different Measures of Dependability

Dependability:

- **Definition:** Dependability in computing is the ability of a system to perform its intended functions under specified conditions for a certain period without failure. It measures how consistently and reliably a system operates.

Measures of Dependability:

1. **Reliability:**
   - **Definition:** Reliability is the likelihood that a system will operate without failure for a defined period.
   - **Example:** If a computer system operates smoothly without any crashes or malfunctions for an entire year, it is considered highly reliable.
   - **Diagram Explanation:** Imagine a car that starts and runs smoothly every day for a year. Its reliability is measured by how consistently it performs without breaking down.
2. **Availability:**
   - **Definition:** Availability refers to the proportion of time a system is operational and ready to use when needed.
   - **Example:** If a server is up and running 99.9% of the time, it means it has high availability.
   - **Diagram Explanation:** Picture a store that is open nearly every day of the year. High availability means it is accessible most of the time.
3. **Maintainability:**
   - **Definition:** Maintainability is how easily a system can be repaired or maintained after a failure.
   - **Example:** A computer that can be quickly fixed and restored to working order after a problem is said to have good maintainability.
   - **Diagram Explanation:** Think of a car that is easy to repair with readily available parts and straightforward procedures. It's easy to maintain.
4. **Safety:**
   - **Definition:** Safety is the ability of a system to avoid causing catastrophic consequences in the event of a failure.

- **Example:** An aircraft control system that ensures no unsafe conditions arise even if there is a malfunction is designed with safety in mind.
- **Diagram Explanation:** Consider a safety feature in a car that prevents accidents even if something goes wrong. Safety ensures that failures do not lead to dangerous situations.

5. **Integrity:**
   - **Definition:** Integrity is the resistance of a system to unauthorized modifications or destruction of data.
   - **Example:** A banking system that prevents any unauthorized changes to transaction records is designed to maintain data integrity.
   - **Diagram Explanation:** Imagine a locked vault where only authorized personnel can access and modify the contents. Integrity ensures that data remains accurate and secure.

How is MTBF Measured?

Example Assume a disk subsystem with the following components and MTTF:

- 10 disks, each rated at 1,000,000-hour MTTF
- 1 SCSI controller, 500,000-hour MTTF
- 1 power supply, 200,000-hour MTTF
- 1 fan, 200,000-hour MTTF
- 1 SCSI cable, 1,000,000-hour MTTF

Using the simplifying assumptions that the lifetimes are exponentially distributed and that failures are independent, compute the MTTF of the system as a whole.

1.7 Dependability ■ 27

Answer The sum of the failure rates is

$$\text{Failure rate}_{\text{system}} = 10 \times \frac{1}{1{,}000{,}000} + \frac{1}{500{,}000} + \frac{1}{200{,}000} + \frac{1}{200{,}000} + \frac{1}{1{,}000{,}000}$$

$$= \frac{10 + 2 + 5 + 5 + 1}{1{,}000{,}000 \text{ hours}} = \frac{23}{1{,}000{,}000} = \frac{23{,}000}{1{,}000{,}000{,}000 \text{ hours}}$$

or 23,000 FIT. The MTTF for the system is just the inverse of the failure rate:

$$\text{MTTF}_{\text{system}} = \frac{1}{\text{Failure rate}_{\text{system}}} = \frac{1{,}000{,}000{,}000 \text{ hours}}{23{,}000} = 43{,}500 \text{ hours}$$

or just under 5 years.

The primary way to cope with failure is redundancy, either in time (repeat the operation to see if it still is erroneous) or in resources (have other components to take over from the one that failed). Once the component is replaced and the system fully repaired, the dependability of the system is assumed to be as good as new. Let's quantify the benefits of redundancy with an example.

MTBF (Mean Time Between Failures):

- **Definition:** MTBF measures the average time between failures of a system during normal operation. It helps in understanding how reliable a system is.

How to Measure MTBF:

1. **Observation:**
   - **Method:** Track the operational time of the system and record each time a failure occurs.
   - **Example:** If a machine operates continuously for 1000 hours before it breaks down, that period is noted as its MTBF.

   - **Diagram Explanation:** Imagine tracking the time a vending machine works before it breaks. Recording these times helps determine how often it needs maintenance.

2. **Calculation:**
   - **Formula:** MTBF = Total operational time / Number of failures.
   - **Example:** If a system runs for 5000 hours and encounters 5 failures, its MTBF is calculated as 5000 hours / 5 failures = 1000 hours.
   - **Diagram Explanation:** Think of calculating the average time between car breakdowns over a year. If the car breaks down 5 times in 5000 hours, you divide 5000 by 5 to find the average time between failures.

Usage:

- **Purpose:** MTBF helps predict the reliability of a system and plan maintenance activities to minimize downtime and improve performance.
- **Example:** In a factory, knowing the MTBF of machinery helps schedule regular checks and prevent unexpected breakdowns, ensuring smooth operation.

By understanding these measures of dependability and how MTBF is calculated, you can assess and improve the reliability and performance of various systems.

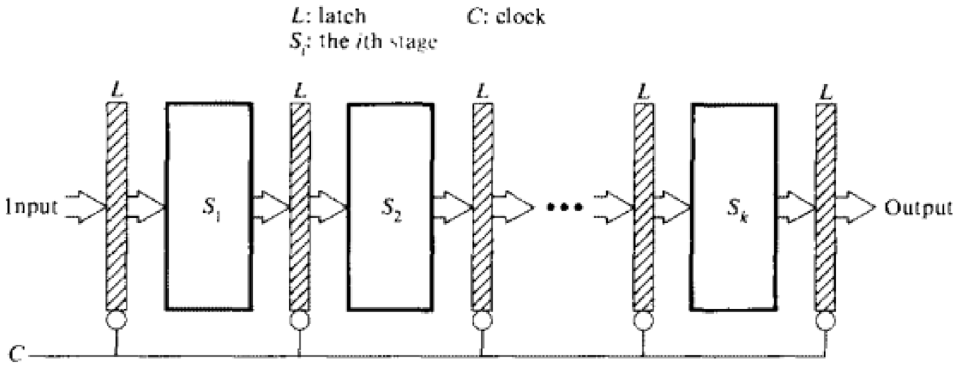
5. Explain with a Neat Diagram, Basic Structure of Pipelining. Compare Advantages & Disadvantages of Interleaved Memory Organization

Pipelining:

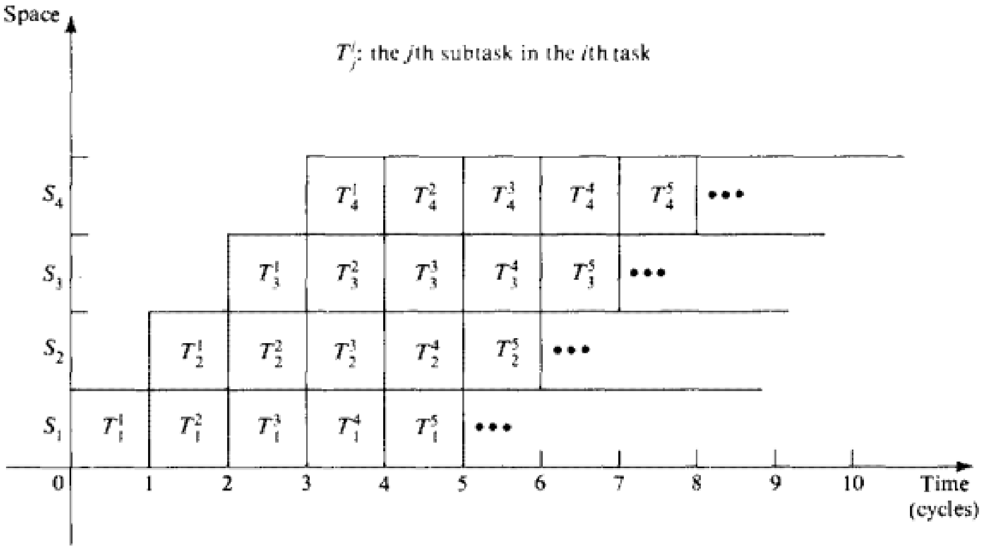
Definition: Pipelining is a technique used in CPUs to improve performance by overlapping the execution of multiple instructions. It works by breaking down the instruction execution process into separate stages, so different stages of different instructions can be processed simultaneously.

Basic Structure:

PRINCIPLES OF PIPELINING AND VECTOR PROCESSING 147



(a) Basic structure of a linear pipeline processor



(b) The space-time diagram depicting the overlapped operations

Figure 3.1 Linear pipeline processor for overlapped processing of multiple tasks.

Stages in Pipelining:

1. **Fetch:**
   - **Role:** Retrieve the instruction from memory.
   - **Diagram Explanation:** Imagine a conveyor belt where instructions are picked up one by one.
2. **Decode:**
   - **Role:** Interpret the instruction to understand what needs to be done.
   - **Diagram Explanation:** Think of this stage as a worker analyzing each instruction to determine the required action.
3. **Execute:**
   - **Role:** Perform the operation specified by the instruction.

- **Diagram Explanation:** This stage is like the worker executing the instruction, such as performing a calculation.

4. **Memory:**
   - **Role:** Access memory if the instruction requires data read or write operations.
   - **Diagram Explanation:** Visualize this as fetching or storing data to or from a storage area.
5. **Write-back:**
   - **Role:** Write the result of the execution back to the register or memory.
   - **Diagram Explanation:** This is where the result is recorded or saved, completing the task.

Diagram Explanation:

Imagine an assembly line where each station performs a different part of the task. As soon as one part is completed at a station, the next part starts at the next station. Similarly, in pipelining, different stages of multiple instructions are processed in parallel to increase the throughput.

Advantages of Pipelining:

1. **Increased Throughput:**
   - **Explanation:** More instructions are processed in a given time period because multiple instructions are at different stages of execution simultaneously.
   - **Diagram Explanation:** Picture a factory line where more products are completed in less time due to simultaneous work at different stages.
2. **Efficient CPU Utilization:**
   - **Explanation:** Keeps various parts of the CPU busy and maximizes resource usage.
   - **Diagram Explanation:** Like keeping every worker on a production line occupied with a task, improving overall efficiency.

Disadvantages of Pipelining:

1. **Complex Design:**
   - **Explanation:** Designing a pipelined CPU is more complicated due to the need to manage different stages and handle hazards.
   - **Diagram Explanation:** Imagine the complexity of coordinating multiple assembly line stages and ensuring they work together smoothly.
2. **Pipeline Hazards:**
   - **Explanation:** Issues such as data hazards (when instructions depend on the results of previous instructions), control hazards (branch instructions), and structural hazards (resource conflicts) can disrupt the pipeline.

  - **Diagram Explanation:** Picture traffic jams on an assembly line where certain parts get delayed, affecting the entire process.

Interleaved Memory Organization:

Advantages of Interleaved Memory Organizations

- **Improved Bandwidth: By dividing memory into multiple modules, interleaving can increase the effective bandwidth, allowing the processor to access data at a higher rate.**
- **Reduced Memory Access Latency: Interleaving can reduce the average memory access time, as multiple memory modules can be accessed simultaneously.**
- **Increased System Throughput: A higher bandwidth and reduced latency can lead to increased overall system throughput, especially for applications that are memory-bound.**
- **Better Utilization of Memory Bus: Interleaving can help to distribute memory traffic more evenly across the memory bus, reducing contention and improving performance.**

Disadvantages of Interleaved Memory Organizations

- **Increased Complexity: Implementing interleaved memory can be more complex than using a single, large memory module.**
- **Higher Cost: The additional hardware required for interleaving, such as address decoders and multiple memory modules, can increase the overall cost of the system.**
- **Potential for Contention: While interleaving can reduce contention, it is still possible for conflicts to occur, especially when multiple processors are accessing memory simultaneously.**
- **Limited Scalability: The benefits of interleaving may diminish as the number of memory modules increases, due to the increased complexity and potential for contention.**