1. Which one of the following is not a Flynn's classification?

A. MISD

B. SISD

C. MIMD

**D. SMID**

2. In MIMD, each processor has a \_\_\_\_\_\_\_\_\_\_\_ program and an instruction stream is generated from \_\_\_\_\_\_\_\_\_ program.

A. same, same

**B. separate, each**

C. same, each

D. separate, separate

3. Which architecture have one control unit, one processor unit and single memory unit.

A. MISD

**B. SISD**

C. MIMD

D. SMID

4. A system with multiple CPUs “sharing” the \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ main memory is called Shared memory multiprocessor.

A. different

**B. same**

C. common

D. global

5. Which one of the following increases the utilization of a processor.

A. single thread

B. UMA

C. N- UMA

. **D. Hardware multithreading**

6. Which multithreading switches threads only on costly stalls, such as last-level cache miss.

**A. Coarse grained**

. B. Fine grained

C. Course grained

D. Tune grained

1. Instruction level Parallelism achieves more than one instruction at a time through
2. **Dynamic scheduling**
3. Priority scheduling
4. Time-slot scheduling
5. Double scheduling
6. Anti-dependence occurs when
7. **j writes a register/memory that i reads.**
8. j reads a register/memory that i reads.
9. j writes a register/memory that j reads.
10. j writes a register/memory that i writes.
11. I=i+1;

J=i+1

The above is example sequence logic of

1. Write after write (WAW)
2. Write after read (WAR)
3. **Read after write (RAW)**
4. Read after read (RAR)
5. he architecture in which multiple operations can be performed parallelly in a particular process, with its own set of resources is
6. Bit level parallelism
7. Task parallelism
8. Data parallelism
9. **Instruction level parallelism**
10. The decision on when to execute an operation depends largely on the
11. **Compiler**
12. Hardware
13. Software
14. OS
15. A processor assigned with a thread block that executes a code, which we usually call
16. Multithreaded DIMS Processor
17. **Multithreaded SIMD Processor**
18. Multithreaded queue
19. Multithreaded stack
20. dentify the way in which the performance is increased with pipelining
21. By decreasing instructionlatency
22. By eliminating datahazards

# By exploiting instruction levelparallelism

1. By decreasing the cache missrate
2. The objective of S/W and H/W techniques, is toexploit

# Parallelism

1. Scalability
2. Supervision
3. Compatibility
4. The \_\_\_\_\_\_\_\_\_\_\_\_\_do not have parallel processing capabilities

# Single Instruction stream, Single Data stream

# Single Instruction stream, Multiple Data stream

1. Multiple Instructionstream, Multiple Datastream
2. All of theabove
3. SIMD symbolizes anorganizationthat .
   1. refers to a computer system capable of processing several programs at the sametime.
   2. represents organization of single computer containing a control unit, processor unit and a memoryunit.
   3. **includes many processing units under the supervision of a common controlunit.**
   4. none of theabove
4. Types of parallelism in the applicationsare\_\_\_\_\_\_\_\_\_
5. Data-LevelParallelism
6. Task-LevelParallelism
7. Instruction-LevelParallelism
8. **All the above**
9. Data-level parallelism/task-level parallelism in a tightly coupled equipment which permits communication among parallel threads, are handledby
   1. Instruction-LevelParallelism
   2. Request-LevelParallelism

# Thread-LevelParallelism

* 1. Vector Architectures and Graphic ProcessorUnits

1. In\_\_\_\_\_\_\_\_\_\_\_\_\_\_, all the CPUs shares the common memory.
   1. distributed memory multiprocessors
   2. Uni-core systems
   3. SISD systems
   4. **shared memory multiprocessor**
2. Choose the incorrect statement about multiprocessors
   * 1. **Multiprocessors support distributed computing**
     2. Multiprocessors are fault tolerant
     3. Multiprocessors boost execution speed
     4. Construction of Multiprocessors is not cost effective
3. “Memory addresses in one processor do not map to another processor, so there is no concept of global address space across all processors.”- this statement describes
   1. Shared memory systems
   2. Symmetric Multiprocessing systems
   3. **Distributed memory systems**
   4. UMA- syatems
4. The method of updating the main memory as soon as the data modified at the cache memory is called as\_\_\_\_\_\_
   * 1. Write back
     2. **Write -through**
     3. Snoopy
     4. Write-around
5. . The \_\_\_\_\_\_\_\_\_\_\_\_systems have a shared logical address space, but physical memory is distributed among CPUs.
   * 1. **NUMA**
     2. COMA
     3. UMA
     4. CMA
6. \_\_\_\_\_\_\_\_\_ is not one of the states of a cache line.
7. Modified
8. Exclusive
9. **Inclusive**
10. Invalid
11. The MESI protocol supports \_\_\_\_\_\_\_\_\_\_\_ caches.
12. **Write-back**
13. Write through
14. Read-back
15. Read through
16. In MESI protocol, “M’ stands for
17. Memory
18. **Modified**
19. Multiplied
20. Manipulate
21. The requests from the buses are monitored by \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ operations
22. **Snooping**
23. Spoofing
24. Simulating
25. Swooping
26. Which of the following is not the bus side request?
27. BusRd
28. Flush
29. Flushopt
30. **PrRd**

PART-B

1. **Differentiate between data dependencies and name dependencies**

**Data dependencies**

Data dependence means that one instruction is dependent on another if there exists a chain of dependencies between them. . Compilers can be of great help in detecting and scheduling around these sorts of hazards; hardware can only resolve these dependencies with severe limitations.

**Name dependencies**

A name dependency occurs when two instructions use the same register or memory location, called a name, but there is no flow of data between them. • Anti-dependence occurs when j writes a register/memory that i reads. • Output dependence occurs when i and j write to the same register/memory location The name used in the instruction is changed so that they do not conflict. This technique is known as register renaming (uses temp registers).

1. **List out the limitations of ILP**

1. An instruction stream needs to be run on an ideal processor with no significant limitations.

2. The ideal processor always predicts branches correctly, has no structural hazards.

3. This eliminates all control and name dependencies. (only data dependencies)

4. Theoretically it is possible for the last dynamically executed instruction in the program to be scheduled on the first cycle.

3. **Briefly explain about the classification of ILP architectures.**

1. **Sequential Architecture :**

Here, program is not expected to explicitly convey any information regarding parallelism to hardware, like superscalar architecture.

1. **Dependence Architectures :**

Here, program explicitly mentions information regarding dependencies between operations like dataflow architecture.

1. **Independence Architecture :**

Here, program gives information regarding which operations are independent of each other so that they can be executed instead of the ‘nop’s.

**4. Explain Score boarding with an example**

Instructions to be issued when they are ready, not necessarily in order, hence out of order execution. To implement out-of-order issue we need to split the instruction decode phase into two:

1. Issue—decode instructions and check for structural hazards;

2. Read operands—wait until no data hazards obtain then read the operands and start executing.

It dynamically schedules the pipeline. instructions must pass through the issue phase in order; This method can stall or bypass each other, in the read operands phase and enter, or even, complete execution in out of order manner.

Example

CDC6600 used a scoreboard, the goal of a scoreboard is to maintain processor throughput of one instruction per clock cycle (no structural hazard). If the next instruction would stall, then store it on a queue and start with a later instruction and takes full responsibility for instruction issue and execution. It uses as many as 16 separate functional units.

5.Write short notes on FLYNN‘S CLASSIFICATION

6.What are the challenges in parallel processing?

7.What are the advantages of Parallel Computing over Serial Computing ?

8.What is Flynn's classification of computer? Explain.

9.Difference between fine-grained and coarse grained multithreading.

10.Write short notes on COMA model.

11.What is parallelism? Give the necessity of parallelism at hardware level.

12. Compare shared memory system and distributed memory systems.

13. Compare UMA, NUMA systems. Also describe about COMA.

14.Brief about multi-core systems with neat diagram.

15.Write the responses to the PrRd in the invalid state.

16. Define BusRd and BusRdX requests in MESI protocol.

17. List and brief about various states of cache line in MESI protocol.

Part-C

1. Explain briefly about the data hazards in ILP (6)

Explain in detail about the three methods used to overcome hazards in ILP (6)

Answer key for refernce

<https://www.srividyaengg.ac.in/coursematerial/ECE/106614.pdf>

1. Explain data level parallelism in detail.

Answer key for reference

<https://www.cs.umd.edu/~meesh/411/CA-online/chapter/exploiting-data-level-parallelism/index.html>

1. Explain in detail about Parallelism and its types.
2. The Parallel computer architecture adds a new dimension in the development of computer system by using more and more number of processors. In principle, performance achieved by utilizing large number of processors is higher than the performance of a single processor at a given point of time. Justify and describe the need of parallelism .compare which type of parallelism is need for the real time scenario.

5. Briefly explain four categories of Flynn's classification with neat diagram.

6. Write short notes on

(i) Memory in multiprocessor system

(ii) Hardware multithreading

7.Explain the two types of hardware (memory) level parallelism with their unique features and challenges.

8.What is cache coherence? Explain various methods adapted for maintaining the cache coherence problem? Discuss with examples.

9.Explain in detail about MESI protocol with the neat state diagram.

10.Consider the following stream of read/write operations on the same cache. Demonstrate how MESI work all operations on the same cache with necessary fields.

Operations: R1, W1, R2, W2, R3, R4, W5, R6, W6