ST. ANNE'S
COLLEGE OF ENGINEERING AND TECHNOLOGY
(Approved by AICTE, New Delhi. Affiliated to Anna University, Chennai)
(An ISO 9001: 2015 Certified Institution)
ANGUCHETTYPALAYAM, PANRUTI – 607 106.

QUESTION BANK

PERIOD: JULY - NOV 2019 BATCH: 2016– 2020

BRANCH: ECE YEAR/SEM: IV/VII

SUB CODE/NAME: EC6009 – ADVANCED COMPUTER ARCHITECTURE

UNIT I FUNDAMENTALS OF COMPUTER DESIGN

PART – A

1. How the server adapts the dependability characteristics?[ID]
   Dependability is critical. Consider the servers running Google, taking orders for Cisco, or running auctions on eBay. Failure of such server systems is far more catastrophic than failure of a single desktop, since these servers must operate seven days a week, 24 hours a day. It estimates revenue costs of downtime as of 2000. To bring costs up-to-date, Amazon.com had $2.98 billion in sales in the fall quarter of 2005. As there were about 2200 hours in that quarter, the average revenue per hour was $1.35 million. During a peak hour for Christmas shopping, the potential loss would be many times higher.
2. Mention the cost of an integrated circuit.[D][Nov/Dec 2016]
   The costs of integrated circuits have dropped exponentially, the basic process of silicon manufacture is unchanged. A wafer is still tested and chopped into dies that are packaged. Thus the cost of a packaged integrated circuit is

$$\text{Cost of integrated circuit} = \frac{\text{Cost of die} + \text{Cost of testing die} + \text{Cost of packaging and final test}}{\text{Final test yield}}$$

3. How the server adapts the scalability characteristics? (U)
   The key feature of server systems is scalability. Server systems often grow in response to an increasing demand for the services they support or an increase in functional requirements. Thus, the ability to scale up the computing capacity, the memory, the storage, and the I/O bandwidth of a server is crucial. Lastly, servers are designed for efficient throughput. That is, the overall performance of the server in terms of transactions per minute or Web pages served per seconds what is crucial. Responsiveness to an individual request remains important, but overall efficiency and cost-effectiveness, as determined by how many requests can be handled in a unit time, are the key metrics for most servers.
4. What are the five trends in computer architecture.[D][Nov/Dec 2016]
   Integrated circuit logic technology
   Semiconductor DRAM
   Semiconductor FLASH
   Magnetic disk technology
   Network technology
5. Define Amdahl's Law.[D][Nov/Dec 2017]][A/M 2019]
   Amdahl's law is often used in parallel computing to predict the theoretical speedup when using multiple processors.

$$\text{Speedup} = \frac{\text{Performance for entire task using the enhancement when possible}}{\text{Performance for entire task without using the enhancement}}$$

Alternatively,

$$\text{Speedup} = \frac{\text{Execution time for entire task without using the enhancement}}{\text{Execution time for entire task using the enhancement when possible}}$$

6. If a 20% reduction in voltage leads to 15% reduction in frequency, what would be the impact on dynamic energy and power? .[D][Nov/Dec 2017]

   Since the capacitance is unchanged, the answer is the ratios of the voltages and frequencies:

$$\frac{\text{Power}_{\text{new}}}{\text{Power}_{\text{old}}} = \frac{(\text{Voltage} \times 0.85)^2 \times (\text{Frequency switched} \times 0.85)}{\text{Voltage}^2 \times \text{Frequency switched}} = 0.85^3 = 0.61$$

   There by reducing power to about 60% of the original.

7. Define throughput.[D][Apr/May 2018]
   The total amount of work done in a given time is called throughput.
8. Some microprocessors today are designed to have adjustable voltage. So that a 20% reduction in voltage leads to 15% reduction in frequency, what would be the impact on dynamic power? .[ID][Apr/May 2018]

   Since the capacitance is unchanged, the answer is the ratios of the voltages and frequencies:

$$\frac{\text{Power}_{\text{new}}}{\text{Power}_{\text{old}}} = \frac{(\text{Voltage} \times 0.85)^2 \times (\text{Frequency switched} \times 0.85)}{\text{Voltage}^2 \times \text{Frequency switched}} = 0.85^3 = 0.61$$

   There by reducing power to about 60% of the original.
9. Define bandwidth and latency. [D]
   Bandwidth or throughput is the total amount of work done in a given time, such as megabytes per second for a disk transfer. In contrast, latency or response time is the time between the start and the completion of an event, such as milliseconds for a disk access. Clearly, bandwidth improves much more rapidly than latency.
10. How to predict the number of good chips per wafer? .[ID]
    The number of dies per wafer is approximately the area of the wafer divided by the area of the die. It can be more accurately estimated by

$$\text{Dies per wafer} = \frac{\pi \times (\text{Wafer diameter}/2)^2}{\text{Die area}} - \frac{\pi \times \text{Wafer diameter}}{\sqrt{2 \times \text{Die area}}}$$

    The first term is the ratio of wafer area (πr^2) to die area. The second compensates for the "square peg in a round hole" problem—rectangular dies near the periphery of round wafers. Dividing the circumference (πd) by the diagonal of a square die is approximately the number of dies along the edge.
11. How the performance of the benchmark is measured in real time applications .[ID]
    The performance of the benchmark is measured in real time applications such as a compiler. Attempts at running programs that are much simpler than a real application have led to performance pitfalls. Examples include
    - Kernels, which are small, key pieces of real applications;
    - Toy programs, which are 100-line programs from beginning programming assignments, such as Quicksort

- Synthetic benchmarks, which are fake programs invented to try to match the profile and behavior of real applications, such as Dhrystone.

12. How the transistor performance is increased even if it is complex? . [D]

The increase in transistor performance is based on devices shrink quadratically in the horizontal dimension and also shrinks in the vertical dimension. The shrink in the vertical dimension requires a reduction in operating voltage to maintain correct operation and reliability of the transistors. This combination of scaling factors leads to a complex interrelationship between transistor performance and process feature size. To a first approximation, transistor performance improves linearly with decreasing feature size.

13. Specify the standards required by the market place. .[ID]

Standards	*Certain standards may be required by marketplace*
Floating point	Format and arithmetic: IEEE 754 standard (App. I), special arithmetic for graphics or signal processing
I/O interfaces	For I/O devices: Serial ATA, Serial Attach SCSI, PCI Express (Ch. 6, App. E)
Operating systems	UNIX, Windows, Linux, CISCO IOS
Networks	Support required for different networks: Ethernet, Infiniband (App. E)
Programming languages	Languages (ANSI C, C++, Java, FORTRAN) affect instruction set (App. B)

14. Define Module reliability. [D]][A/M 2019]

It is a measure of the continuous service accomplishment (or, equivalently, of the time to failure) from a reference initial instant. Hence, the mean time to failure (MTTF) is a reliability measure. The reciprocal of MTTF is a rate of failures, generally reported as failures per billion hours of operation, or FIT (for failures in time).Thus, an MTTF of 1,000,000 hours equals 109⁄106 or 1000 FIT.

15. In what way the volume issues affect the cost wise in super computers .[ID]

Volume is a second key factor in determining cost. Increasing volumes affect cost in several ways. First, they decrease the time needed to get down the learning curve, which is partly proportional to the number of systems (or chips) manufactured. Second, volume decreases cost, since it increases purchasing and manufacturing efficiency. As a rule of thumb, some designers have estimated that cost decreases about 10% for each doubling of volume. Moreover, volume decreases the amount of development cost that must be amortized by each computer, thus allowing cost and selling price to be closer.

16. Write about commodities. [D]

Commodities are products that are sold by multiple vendors in large volumes and are essentially identical. Virtually all the products sold on the shelves of grocery stores are commodities, as are standard DRAMs, disks, monitors, and keyboards. In the past 15 years, much of the low end of the computer business has become a commodity business focused on building desktop and laptop computers running Microsoft Windows.

17. Define Service Level Agreements or Service Level Objectives. [D][Nov/Dec 2015]

It guarantee their networking or power service would be dependable. For example, they would pay the customer a penalty if they did not meet an agreement more than some hours per month. Thus, an SLA could be used to decide whether the system was up or down. Systems alternate between two states of service with respect to an SLA:

1. Service accomplishment, where the service is delivered as specified

2. Service interruption, where the delivered service is different from the SLA Transitions between these two states are caused by failures (from state 1 to state 2) or restorations (2 to 1). Quantifying these transitions leads to the two main measures of dependability.

18. What are the three approaches of source code modifications? [D]

The three different approaches are

1. No source code modifications are allowed.

2. Source code modifications are allowed, but are essentially impossible. For example, database benchmarks rely on standard database programs that are tens of millions of lines of code. The

database companies are highly unlikely to make changes to enhance the performance for one particular computer.

3. Source modifications are allowed, as long as the modified version produces the same output.

19. Discuss about SPEC (Standard Performance Evaluation Corporation[D][Nov/Dec 2015

SPEC benchmark report requires an extensive description of the computer and the compiler flags, as well as the publication of both the baseline and optimized results. In addition to hardware, software, and baseline tuning parameter descriptions, a SPEC report contains the actual performance times, shown both in tabular form and as a graph. A TPC benchmark report is even more complete, since it must include results of a benchmarking audit and cost information. These reports are excellent sources for finding the real cost of computing systems, since manufacturers compete on high performance and cost-performance

20. Define SPEC ratio [D]

SPEC uses this approach, called the ratio the SPECRatio. It has a particularly useful property that it matches the way we compare computer performance throughout this text—namely, comparing performance ratios. For example, suppose that the SPECRatio of computer A on a benchmark was 1.25 times higher than computer B; then you would know

$$1.25 = \frac{\text{SPECRatio}_A}{\text{SPECRatio}_B} = \frac{\dfrac{\text{Execution time}_{\text{reference}}}{\text{Execution time}_A}}{\dfrac{\text{Execution time}_{\text{reference}}}{\text{Execution time}_B}} = \frac{\text{Execution time}_B}{\text{Execution time}_A} = \frac{\text{Performance}_A}{\text{Performance}_B}$$

21. Define geometric mean in SPECR. [D]

SPEC Ratio is a ratio rather than an absolute execution time, the mean must be computed using the geometric mean. Since SPEC Ratios have no units, comparing SPEC Ratios arithmetically is meaningless.) The formula is

$$\text{Geometric mean} = \sqrt[n]{\prod_{i=1}^{n} sample_i}$$

22. Mention the two important properties of geometric mean. [D]

The geometric mean ensures two important properties:

1. The geometric mean of the ratios is the same as the ratio of the geometric means.
2. The ratio of the geometric means is equal to the geometric mean of the performance ratios, which implies that the choice of the reference computer is irrelevant.

Hence, the motivations to use the geometric mean are substantial, especially when we use performance ratios to make comparisons.

23. Define geometric mean and geometric standard deviation. [D]

$$\text{Geometric mean} = \exp\left(\frac{1}{n} \times \sum_{i=1}^{n} \ln(sample_i)\right)$$

$$\text{gstdev} = \exp\left(\sqrt{\frac{\sum_{i=1}^{n} (\ln(sample_i) - \ln(\text{Geometric mean}))^2}{n}}\right)$$

24. Define Speedup. [D]

Speedup tells us how much faster a task will run using the computer with the enhancement as opposed to the original computer.

$$\text{Speedup} = \frac{\text{Performance for entire task using the enhancement when possible}}{\text{Performance for entire task without using the enhancement}}$$

Alternatively,

$$\text{Speedup} = \frac{\text{Execution time for entire task without using the enhancement}}{\text{Execution time for entire task using the enhancement when possible}}$$

25. What are the two factors of Amdahl's Law to find the speedup? [D]

1. The fraction of the computation time in the original computer that can be converted to take advantage of the enhancement. For example, if 20 seconds of the execution time of a program that takes 60 seconds in total Can use an enhancement, the fraction is 20/60. This value, which we will call Fraction enhanced, is always less than or equal to 1.
2. The improvement gained by the enhanced execution mode; that is, how much faster the task would run if the enhanced mode were used for the entire program This value is the time of the original mode over the time of the enhanced mode. If the enhanced mode takes, say, 2 seconds for a portion of the program, while it is 5 seconds in the original mode, the improvement is 5/2. We will call this value, which is always greater than 1, Speedup enhanced.

26. Write the processor performance equation [D]

Essentially all computers are constructed using a clock running at a constant rate. These discrete time events are called ticks, clock ticks, clock periods, clocks,cycles,or clock cycles.

Computer designers refer to the time of a clock period by its duration (e.g., 1 ns) or by its rate (e.g., 1 GHz). CPU time for a program can then be expressed two ways:

$$\text{CPU time} = \text{CPU clock cycles for a program} \times \text{Clock cycle time}$$

or

$$\text{CPU time} = \frac{\text{CPU clock cycles for a program}}{\text{Clock rate}}$$

27. Define Service interruption.[D]

Service interruption is measured as mean time to repair (MTTR). Mean time between failures (MTBF) is simply the sum of MTTF + MTTR. Although MTBF is widely used, MTTF is often the more appropriate term. If a collection of modules have exponentially distributed lifetimes meaning that the age of a module is not important in probability of failure the overall failure rate of the collection is the sum of the failure rates of the modules

27. Define Module availability. [D][A/M 2019]

Module availability is a measure of the service accomplishment with respect to the alternation between the two states of accomplishment and interruption. For non redundant systems with repair, module availability is

$$\text{Module availability} = \frac{\text{MTTF}}{(\text{MTTF} + \text{MTTR})}$$

Reliability and availability are now quantifiable metrics, rather than synonyms for dependability. From these we can estimate reliability of a system quantitatively if we make some assumptions about the reliability of components and that failures are independent.

28. Compare the performance of two different computers.(6) [ID][NOV/DEC2018]

The performance of two different computers are X and Y. The phrase "X is faster than Y" is used here to mean that the response time or execution time is lower on X than on Y for the given task. In particular, "X is *n* times faster than Y" will mean

$$\frac{\text{Execution time}_Y}{\text{Execution time}_X} = n$$

Since execution time is the reciprocal of performance, the following relationship holds:

$$n = \frac{\text{Execution time}_Y}{\text{Execution time}_X} = \frac{\frac{1}{\text{Performance}_Y}}{\frac{1}{\text{Performance}_X}} = \frac{\text{Performance}_X}{\text{Performance}_Y}$$

29. *Find the die yield for dies that are 1.5 cm on a side and 1.0 cm on a side, assuming a defect density of 0.4 per cm^2 and α is 4.*

The total die areas are 2.25 cm^2 and 1.00 cm^2. For the larger die, the yield is

$$\text{Die yield} = \left(1 + \frac{0.4 \times 2.25}{4.0}\right)^{-4} = 0.44$$

For the smaller die, it is $\text{Die yield} = \left(1 + \frac{0.4 \times 1.00}{4.0}\right)^{-4} = 0.68$

That is, less than half of the entire large die is good but more than two-thirds of the small die are good.

30. *In what way design of super computers are specifically cost-sensitive.*

The cost of a manufactured computer component decreases over time even without major improvements in the basic implementation technology. The underlying principle that drives costs down is the learning curve manufacturing costs decrease over time. The learning curve itself is best measured by change in yield the percentage of manufactured devices that survives the testing procedure. Whether it is a chip, a board, or a system, designs that have twice the yield will have half the cost.

$$n = \frac{\text{Execution time}_Y}{\text{Execution time}_X} = \frac{\frac{1}{\text{Performance}_Y}}{\frac{1}{\text{Performance}_X}} = \frac{\text{Performance}_X}{\text{Performance}_Y}$$

31. *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 failure are independent, compute the MTTF of the system as a whole. [D][OCT/NOV 2018]

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.

PART – B

[First Half]

[Review of fundamentals of CPU, MEMORY and IO]

1. Explain briefly about the fundamentals of the processor (8)
2. Explain how management of Input/output interface improve the performance of a computing processor. (16). [D][Apr/May-2019]
3. Discuss the factors that need to be considered while designing the instruction set architecture of a processor(10) .[D][Nov/Dec 2017]
4. Explain the recent trends and cost in computer technology. (16)
5. Write short notes on power and energy consumption in a microprocessor. (16)
6. Explain the fundamentals of CPU in modern computer(16)[D]

[Trends in technology, energy, power and cost]

7. Write short notes on power and energy consumption in a microprocessor. (16)[D][Nov/Dec2016]
8. Explain the recent trends and cost in computer technology. (16)
9. Appraise the major factors that influence the cost of a computer and outline how these factors are changing over time(16)[D][Apr/May-2018]
10. What are the major concerns that should be addressed by system architecture with respect to power and energy? Discuss the techniques that commonly employed to address these issues.(10) ID][Nov/Dec 2017]
11. Discuss about the cost and commodities in super computers.(8)
12. With an example explain briefly about the quantitative principles (the benchmark) of computer design(13) [D][Apr/May-2019]
13. Explain briefly about the dependability of the processor.

[Second Half]

[Dependability, performance evaluation]

14. Discuss the performance evaluation methods of different computers(16) [D][Nov/Dec2016]
15. How to measure the performance of a desk top system using SPEC benchmark?(5) [D][Apr/May-2019]
16. Appraise module reliability and module availability with an example(8)[D][Apr/May-2018]
17. What are the various performance measures available to evaluate the performance processor? How do each of these relate to the ultimate measure of performance? Explain.(10)[ID][Nov/Dec 2017]
18. How to relate the performance of two different computers say, X and Y? Appraise with an example.(8)[D][Apr/May-2018]
19. What are the different classes of parallelism and parallel architectures that are available? (6)[D][Apr/May-2017]

20. Find the number of dies per 300mm(30)cm wafer for a die that is 1.5cm on a side and for a die is 1.0cm on a side.(7)[D]
21. Discuss the factors that need to be considered while designing the Instruction Set Architecture of a processor. (10)[D]][Nov/Dec 2017]

UNIT II-INSTRUCTION LEVEL PARALLISM
PART – A

1. What is Instruction Level Parallelism?)[D][Nov/Dec 2011])(May/June 2012)
   Instruction level parallelism- All processors use pipelining to overlap the execution of instructions and improve performance. This potential overlap among instructions is called instruction-level parallelism (ILP), since the instructions can be evaluated in parallel. The amount of parallelism that is available within a basic block is quite small. The basic block is a block of code with no branches into the code except at the start and no branches out of the code except at the end

2. What is loop –level parallelism?[D]
   Loop –level parallelism-The simplest and most common way to increase the ILP is to exploit parallel- ism among iterations of a loop. This type of parallelism is often called loop-level parallelism.
   for (i=1; i<=1000; i=i+1)
   x[i] = x[i] + y[i];
   This is a parallel loop and every iteration of the loop can overlap with any other iteration, although within each loop iteration there is little or no opportunity for overlap.

3. What is loop unrolling? and what are the advantages of loop unrolling?)[D][Nov/Dec 2011]).
   Loop unrolling is a simple but useful method for increasing the size of straight-line code fragments that can be scheduled effectively. This transformation is useful in a variety of processors.
   Advantages:
   - Unrolling improves the performance of the loop by eliminating overhead instructions.
   - Loop unrolling can also be used to improve scheduling. Because it eliminates the branch, it allows instructions from different iterations to be scheduled together.

4. what is dynamic scheduling.[D] May/June 2013)
   Dynamic scheduling: The CPU rearranges the instructions to reduce stalls while preserving dependences. It uses a hardware based mechanism to rearrange instruction execution order to reduce stalls at runtime and enables handling cases where dependences areas unknown at compile time. A dynamically scheduled processor cannot remove true data dependences but tries to avoid or reduce stalls
5. list the advantages of dynamic scheduling.[D](Nov/Dec 2012)(May/June 2012)

   Dynamic scheduling advantages:
   - It allows code that was compiled with one pipeline in mind to run efficiently on a different pipeline, eliminating the need to have multiple binaries and recompile for a different microarchitecture.
   - It enables handling some cases when dependences are unknown at compile time (for example, because they may involve a memory reference), and it simplifies the compiler.
   - It allows the processor to tolerate unpredictable delays such as cache misses, by executing other code while waiting for the miss to resolve.
   - Hardware speculation, a technique with significant performance advantages, which builds on dynamic scheduling.
   - A dynamically scheduled processor cannot change the data flow, it tries to avoid stalling when dependences are present.

Static pipeline scheduling tries to minimize stalls by separating dependent instructions so that they will not lead to hazards. Compiler pipeline scheduling can also be used on code destined to run on a processor with a dynamically scheduled pipeline.

6. Write the 5 levels of branch prediction and define them.[D] (May/June 2013)
   The five levels of branch prediction are,
   - Perfect-> all branches and jumps are perfectly predicted at the start of execution.
   - Tournament-based branch predictor-> the prediction scheme uses a correlating 2-bit predictor and a non correlating 2-bit predictor together with a selector, which chooses the best predictor each branch.
   - Standard 2-bit predictor with 512 2-bit entries-> In addition, we assume a 16-entry buffer to predict returns.
   - Profile-based-> a static predictor uses the profile history of the program and predicts that the branch is always taken or always not taken based on the profile.
   - None -> No branch prediction is used, though jumps are still predicted. Parallelism is largely limited to within a basic block.

7. Write the 7 fields of Reservation stations.[D]
   Reservation stations 7 fields:
   - Op the operation to perform on source operands S1 and S2.
   - Qj, Qk the reservation stations that will produce the corresponding source operand; a value of zero indicates that the source operand is already available in Vj or Vk, or is unnecessary. (The IBM 360/91 calls these SINK unit and SOURCE unit.)
   - Vj, Vk the value of the source operands. Note that only one of the V field or the Q field is valid for each operand. For loads, the Vk field is used to the offset from the instruction.
   - fields are called SINK and SOURCE on the IBM 360/91.)
   - A used to hold information for the memory address calculation for a load or store. Initially, the immediate field of the instruction is stored here; after the address calculation, the effective address is stored here.
   - Busy indicates that this reservation station and its accompanying functional unit are occupied.
   - Qi the number of the reservation station that contains the operation whose result should be stored into this register.

8. List the limitations of loop unrolling[D]

9. What is an imprecise exception?[D]
   Imprecise exception- An exception is imprecise if the processor state when an exception is raised does not look exactly as if the instructions were executed sequentially in strict program order. Imprecise exceptions can occur because of two possibilities:
   - The pipeline may have already completed instructions that are later in pro- gram order than the instruction causing the exception.
   - The pipeline may have not yet completed some instructions that are earlier in program order than the instruction causing the exception.

10. What is Branch prediction buffer / Branch history table?[D]
    Branch prediction buffer- The simplest dynamic branch-prediction scheme is a branch-prediction buffer or branch history table. A branch-prediction buffer is a small memory indexed by the lower portion of the address of the branch instruction. The memory contains a bit that says whether the branch was recently taken or not. This scheme is the simplest sort of buffer. It has no tags and is useful only to reduce the branch delay when it is longer than the time to compute the possible target PCs.

11. List the data hazards in ILP?[D]
    ILP data hazards: consider two instructions i and j, with I preceding j in program order. The possible data hazards are
    - RAW (Read after Write) - j tries to read a source before i writes it, so j incorrectly gets the old

value.

- WAW (Write After Write)- j tries to write an operand before it is written by i. The writes end up being performed in the wrong order, leaving the value written by i rather than the value written by j in the destination.
- WAR (Write after Read) - j tries to write a destination before it is read by i, so i incorrectly gets the new value.

12. What is Register Renaming?[D]
    Register renaming- A name dependence is not a true dependence, instructions involved in a name dependence can execute simultaneously or be reordered, if the name (register number or memory location) used in the instructions is changed so the instructions do not conflict. This renaming can be more easily done for register operands, where it is called register renaming. Register renaming can be done either statically by a compiler or dynamically by the
13. Write the merits and demerits of software pipelining and loop rolling [D]. (Nov/Dec2012).
    Software pipelining ADV:
    - The advantage of software pipelining is that optimal performance can be achieved with compact object code.
    - A drawback of software pipelining is its complexity; the problem of finding an optimal schedule is NP-complete. (This can be shown by transforming the problem of resource constrained scheduling problem [141 to the software pipelining problem).
    - There Significant gains can be realized if the reduction in executed instructions compensates for any performance reduction caused by any increase in the size of the program.
    - Branch penalty is minimized.

    - DIS ADV: Increased program code size, which can be undesirable, particularly for embedded applications. Can also cause an increase in instruction cache misses, which may adversely affect performance.
    - Unless performed transparently by an optimizing compiler, the code may become less readable.
    - If the code in the body of the loop involves function calls, it may not be possible to combine unrolling with in lining, since the increase in code size might be excessive. Thus there can be a trade-off between the two optimizations.
    - Possible increased register usage in a single iteration to store temporary variables which may reduce performance, though much will depend on possible optimizations.[6]
    - Apart from very small and simple codes, unrolled loops that contain branches are even slower than recursions.
14. What are the advantages and disadvantages of trace scheduling method?[D](may/June 2012).

    ADV

    - Trace scheduling is an optimization technique developed by Josh Fisher used in compilers for computer programs.[1]
    - A compiler often can, by rearranging its generated machine instructions for faster execution, improve program performance. It increases ILP (Instruction Level Parallelism) along the important execution path by statically predicting frequent execution path. Trace scheduling is one of many known techniques for doing so.
    - A trace is a sequence of instructions, including branches but not including loops, that is executed for some input data. Trace scheduling uses a basic block scheduling method to schedule the instructions in each entire trace, beginning with the trace with the highest frequency. It then adds compensation code at the entry and exit of each trace to compensate for any effects that out of order execution may have had.

    DIS ADC

- This can result in large increases in code sizes and poor or erratic performance if program's behavior varies significantly with the input.
- Trace scheduling was originally developed for Very Long Instruction Word, or VLIW machines, and is a form of global code motion. It works by converting a loop to long straight-line code sequence using loop unrolling and static branch prediction. This process separates out "unlikely" code and adds handlers for exits from trace. The goal is to have the most common case executed as a sequential set of instructions without branches.

15. Define Software Pipelining (or) Symbolic loop unrolling (May/June 2012).
With software pipelining a reorganized loop contains instructions from different iterations of the original loop. Sometimes called symbolic loop unrolling.

16. What is pipeline CPI? (Nov/Dec 2013)
The value of the CPI (cycles per instruction) for a pipelined processor is the sum of the base CPI and all contributions from stalls:
Pipeline CPI = Ideal pipeline CPI + Structural stalls + Data hazard stalls + Control stalls
The ideal pipeline CPI is a measure of the maximum performance attainable by the implementation.
.

17. What are the limitations of VLIW?[D](Nov/Dec 2011).
They need for a power ful compiler.
Limitation due to Lock step operations
Larger memory BW and register file BW

18. What is the impact of speculation on energy efficiency? [D](Nov/Dec2012).
Two ways
    - The instruction that were speculated and whose results were not needed generated excess work for the processor, wasting energy.
    - Undoing the speculation and restoring the state of the processor to continue execution at the appropriate address consumes additional energy that would not be needed without speculation.

    Certainly speculation will raise the power consumption and, if we could control speculation, it would be possible to measure the cost.

19. Define Loop carried dependence with the example.[D](May/ June 2013).

A loop-carried dependence exists However, if a statement in one iteration of a loop depends only on a statement in the same iteration of the loop, this creates a loop independent dependence.

example

```
for(i = 0; i < 4; i++)
  S1: b[i] = 8;
  S2: a[i] = b[i-1] + 10;
```

In this example, code block 1 shows loop-dependent dependence between statement S2 iteration i and statement S1 iteration i-1. This is to say that statement S2 cannot proceed until statement S1 in the previous iteration finishes.

20. Write a note on trace scheduling?[D](May/June 2012).
Trace scheduling is a way to organize the global code motion process, so as to simplify the code scheduling by incurring the costs of possible code motion on the less frequent paths. There are two steps to trace scheduling.
- The first step, called trace selection, tries to find a likely sequence of basic blocks whose operations will be put together into a smaller number of instructions; this sequence is called a trace.

- Once a trace is selected, the second process, called trace compaction, tries to squeeze the trace into a small number of wide instructions. Trace compaction is code scheduling; hence, it attempts to move operations as early as it can in a sequence, packing the operations into as few wide instructions as possible.

21. What is recorder buffer?[D][Nov/Dec 2017]).

A re-order buffer (*ROB*) is used in a Tomasulo algorithm for out-of-order instruction execution. It allows instructions to be committed in-order.

Normally, there are three stages of instructions: "Issue", "Execute", "Write Result". In Tomasulo algorithm, there is an additional stage "Commit". In this stage, the results of instructions will be stored in a register or memory. In the "Write Result" stage, the results are just put in the re-order buffer. All contents in this buffer can then be used when executing other instructions depending on these.

22. What is meant by delayed branching?[D][Nov/Dec 2017]).

A technique for minimizing the effect of control dependencies is to separate the point where the branch operation takes effect from the branch tests. The branch instruction performs a test on a branch condition. If the test succeeds, the PC is modified, but the modification does not take effect immediately. This *delayed branch* allows one or more instructions following the branch to be executed in the pipeline *whether the branch is taken or not.*

23. What is pipelining?[D][Nov/Dec 2017]).

Pipeline- pipelining is the technique of splitting a sequential process into sub-operations with each sub-operation being executed in a dedicated segment that operates concurrently with all other segments. It improves the system performance. A pipeline may be visualized as a collection of processing segments called pipe stages through which binary information flows. Each segment performs partial processing as decided by the task.

24. Outline the limitations of Instruction Level Parallelism?[D][Apr/May-2018]

Register renaming
Branch prediction
Jump prediction
Memory-address alias analysis

25. Explain the idea behind dynamic scheduling. ?[D](Nov/Dec 2016).

- Dynamic instruction scheduling attempts to exploit instruction level parallelism by allowing instructions to execute as early as possible when:
- There is an available functional unit to avoid structural hazards and WAW hazards
- Source operands are available to avoid RAW hazards and to write results when destination registers are no longer needed to avoid WAR hazards

26. Give an example for data dependence? [D](Nov/Dec 2016).

Data dependency means that one/more attribute uniquely identifies other attributes of a relation. in more simple terms we can say that some data values are dependent on other data values in order to get recognized.
Example : roll no→ name

27. What is loop unrolling? [D]

Loop unrolling is a loop transformation technique that helps to optimize the execution time of a program. We basically remove or reduce iterations. Loop unrolling increases the program's speed by eliminating loop control instruction and loop test instructions.

28. What are the types of hazards in pipeline? [D]

Structural hazards arise from resource conflicts when the hardware cannot support all possible combinations of instructions simultaneously in overlapped execution.
Data hazards arise when an instruction depends on the results of a previous instruction in a that is exposed by the overlapping of instructions in the pipeline.
Control hazards arise from the pipelining of branches and other instructions that change the PC

29. What is Register Renaming?

Register renaming- A name dependence is not a true dependence, instructions involved in a name dependence can execute simultaneously or be reordered, if the name (register number or memory location) used in the instructions is changed so the instructions do not conflict. This renaming can be more easily done for register operands, where it is called register renaming. Register renaming can be done either statically by a compiler or dynamically by the hardware

30. What is score boarding?[D]

Scoreboard- scoreboard is a hardware mechanism that maintains an execution rate of one instruction per cycle by executing an instruction per cycle by executing an instruction as soon as its operands are made available and no hazard conditions prevent it, from execution. It works by replacing the three pipeline stages of ID, EX, WB with four stages ID1, ID2, EX, WB. Every instruction goes through the scoreboard where a record of data dependencies is constructed.

31. What is meant by dynamic branch prediction?[D][Apr/May-2019]

Dynamic predictors are sometimes used in processors where the expectation is that branch behavior is highly predictable at run time. The simplest dynamic branch-prediction scheme is a branch-prediction buffer or branch history table

32. How many bits are in the (0,2) branch predictor with 4K entries? How many entries are in a (2,2) predictor with the same number of bits?[ID][Apr/May-2019]

PART-B

[First half]

[ILP concept, pipelining overview, compiler techniques for exposing ILP]

1. Explain the types of dependencies in ILP(8)[D] [Nov/dec-2016]

2. Explain the compilation techniques that can be used to express instruction level parallelism(8) [D] [Nov/dec-2016]

[Dynamic branch prediction, Dynamic Scheduling]

3. Explain in Dynamic Scheduling. Explain how it is used to reduce data hazards(8)[D] [Nov/dec-2016]

4. How data hazards can be overcome with Dynamic Scheduling? Appraise with an example [ID][Apr/May-2018] Apr/May-2019]

5. Briefly explain how to overcome data hazards with dynamic scheduling using Tomasulo's approach.(16)[D] (NOV/DEC 2011, MAY/JUNE 2012, NOV/DEC 2013)

6. Explain the static and dynamic branch prediction schemes in detail(8)[D](NOV/DEC 2011,MAY/JUNE 2012, NOV/DEC 2012,MAY/JUNE 2014) [Apr/May-2019]

7. Explain how to Reduce Branch Costs with Dynamic Hardware Prediction. (16) [D](MAY/JUN2013,APR/MAY 2013)

8. Consider the following code:

```
LOOP:   L.D      F2,0(R1)
        MUL,D    F4,F2,F0
        L.D      F6,0(R2)
        ADD.D    F6,F4,F6
        S.D      0(R2).F6
        DADDIU   R1,R1,#8
        DADDIU   R2,R2,#8
        CMPI     R3,R1,#800
        BEQZ     R3,LOOP
```

Assume that there are separate functional units for effective address calculations, for ALU operations and for branch conditions evaluation. Assume latencies for add and 5 for multiply, assume that loads and stores access memory one clock cycle after the effective address calculations. Show the working of this code for two iterations of the loop when executed on a single issue Tomasulo processor that supports speculation.(10)[ID] [Nov/dec-2017]

[Second Half]

[Multiple instruction issue]

9. Write short notes on the different types of multiple issue processors(6)[D]Nov/dec-2017]
10. Discuss the various hardware techniques used for handling control hazards (10) [D][Apr/May-2017]
11. What is meant by loop unrolling? Discuss the advantages and disadvantages(6) [D][Apr/May-2017]
12. Briefly describe any techniques to reduce the control hazard stalls.(16)[D] (APR/MAY 2011)
13. Appraise with an example the use of simple compiler technology to enhance a processor ability to exploit instruction level parallelism(16) [D][Apr/May-2018]
14. Describe how the compiler technology can be used to improve the performance of instruction level parallelism (16) [D](APR/MAY 2011,MAY/JUNE 2012)
15. Briefly explain what are the steps involved in instruction level parallelism.(8)[D]
    [Hardware based speculation, static scheduling, multithreading]
16. Define Multithreading. Explain how ILP is achieved using Multithreading with an example. (8)[D] [Nov/dec-2016]
17. Explain hardware based speculation to overcome the control dependencies(16)[D] (NOV/DEC 2012) (16)
18. Explain the static and dynamic branch prediction schemes in detail(8)[D] (NOV/DEC 2011,MAY/JUNE 2012, NOV/DEC 2012,MAY/JUNE 2014)
19. Briefly describe any techniques to reduce the control hazard stalls.(16)[D] (APR/MAY 2011)
    [Limitations of ILP]
20. Explain the basic compiler techniques for exploiting ILP(9) [Apr/May-2019]

UNIT III DATA LEVEL PARALLELISM

PART A

1. Define data level parallelism. [D][Apr/May-2018]

   Data parallelism is a form of parallelization across multiple processors in parallel computing environments. It focuses on distributing the data across different nodes, which operate on the data in parallel. It can be applied on regular data structures like arrays and matrices by working on each element in parallel. Data-Level Parallelism (DLP) arises because there are many data items that can be operated on at the same time.

2. How did single core architectures exploit data level parallelism)[ID] (NOV/DEC2017]
   Single core architectures that exploit data level parallelism, i.e. SIMD style of architectures. SIMD architectures can exploit significant data-level parallelism for not only matrix-oriented scientific computing, but also for media-oriented image and sound processing, which are very popular these days. Additionally, SIMD is more energy efficient than MIMD, as we need to fetch only one instruction per data operation. This makes SIMD attractive for personal mobile devices also.
3. SIMD is preferred over MIMD. Justify it.[ID]
   SIMD is typically used for problems requiring lots of computations with processors performing the same operation in parallel. MIMD is frequently used for problems that break down algorithms into

separate and independent parts, with each part assigned to a different processor for simultaneous solution.

4. What are the types of vector processor?[D]
   a. Memory-to-Memory Architecture (Traditional)
      - For all vector operation, operands are fetched directly from main memory, then routed to the functional unit.
      - Results are written back to main memory.
      - Includes early vector machines through mid 1980s:
      - Advanced Scientific Computer (TI), Cyber 200 & ETA-1.
   b. Register-to-Register Architecture (Modern)

      All vector operations occur between vector registers.
      - SIMD processors are based on this architecture.
      - If necessary, operands are fetched from main memory into a set of vector registers (load-store unit).
      - Includes all vector machines since the late 1980s:
      - Convex, Cray, Fujitsu, Hitachi, NEC

5. What are the advantages and disadvantages of vector instruction?[D]

   Advantages of vector processing
   - increase in code density
   - 2.decrease of total number of instructions
   - 3.data is organised in patterns which is easier for the hardware to computer
   - 4.simple loops are replaced with vector instructions, hence decrease in overhead
   - 5. scalable

   Disadvantageous of scalar processor
   - 1.It consumes at 64 per clock, so cannot chain together
   - 2.It exposes memory latency
   - It is expensive
   - 4.one instruction fetch, decode? dispatch per operation.
   - 5.loop in unrolling and software pipeline for high performance increase instruction cache footprint

6. *Advantages of Vector instruction[*D]

The power wall leads architects to value architectures that can deliver high performance without the energy and design complexity costs of highly outof- order superscalar processors. Vector instructions are a natural match to this trend, since architects can use them to increase performance of simple in-order scalar processors without greatly increasing energy demands and design complexity. In practice, developers can express many of the programs that ran well on complex out-of-order designs more efficiently as data-level parallelism in the form of vector instructions.

Disadvantages of Vector instruction

With a vector instruction, the system can perform the operations on the vector data elements in many ways, including operating on many elements simultaneously. This flexibility lets vector designs use slow but wide execution units to achieve high performance at low power.

7. Define the term Vectorized or Vectorizable.[D]
   - Vectorizable means Capable of being vectorized.

   The vector units in the early Crays were pipeline units where the time to update a set of data n long was, Time for vector = startup time (length of pipeline) + n * result rate

   The most efficient way to execute a Vectorizable application is a vector processor. Vector architectures grab sets of data elements scattered about memory, place them into large, sequential register files, operate on data in those register files, and then disperse the results back into memory.

8. Differentiate MIPS vs VMIPS.[D]

MIPS	VMIPS
The important difference between MIPS and VMIPS is the frequency of pipeline	
In the straightforward MIPS code, every ADD.D must wait for a MUL.D, and every S.D must wait for the ADD.D.	On the vector processor, each vector instruction will only stall for the first element in each vector, and then subsequent elements will flow smoothly down the
Thus, pipeline stalls are required only once per vector instruction, rather than once per vector element.	
The most dramatic difference is that the vector processor greatly reduces the dynamic instruction bandwidth, executing only 6 instructions versus almost 600 for MIPS	This reduction occurs because the vector operations work on 64 elements and the overhead instructions that constitute nearly half the loop on MIPS are not present in the VMIPS code

9. Write short notes on convoy.[D]
   - •Convoy denotes vector execution and vector performance.
   - •Convoy is the set of vector instructions that could potentially execute together.
   - •Instruction in convoy must not contain any structural hazards.
   - •Sequences with read after write dependency hazards should be in different Convoys. However it can be in the same convoy via chaining.
10. Write code for strip-mined version of the DAXPY loop.[ID]

    Strip-mined version of the DAXPY loop in C,

```
low = 0;
VL = (n % MVL); /*find odd-size piece using modulo op % */
for (j = 0; j <= (n/MVL); j=j+1) { /*outer loop*/
 for (i = low; i < (low+VL); i=i+1) /*runs for length
  VL*/ Y[i] = a * X[i] + Y[i] ; /*main operation*/
  low = low + VL; /*start of next vector*/
  VL = MVL; /*reset the length to maximum vector length*/
```

11. Write short notes on memory banks.[D]

    A memory bank is a logical unit of storage in electronics, which is hardware-dependent. In a computer, the memory bank may be determined by the memory controller along with physical organization of the hardware memory slots.

    A memory bank is primarily used for storing cached data, or data that helps a computer access data much more quickly than standard memory locations. Typically, a memory bank is created and organized by the memory access controller and the actual physical architecture of the memory module. In SDRAM and DDR RAM, the memory bank can consist of multiple columns and rows of storage units spread across several chips. Each memory module can have two or more memory banks for program and data storage.

12. State the condition for stalling.[D]

    The VMIPS instruction LVWS (load vector with stride) fetches the vector into a vector register. Likewise, when storing a non-unit stride vector, use the instruction SVWS (store vector with stride). Supporting strides greater than one complicates the memory system. Once non-unit strides are introduced, it becomes possible to request accesses from the same bank frequently.

    When multiple accesses contend for a bank, a memory bank conflict occurs, thereby stalling one access. A bank conflict and, hence, a stall will occur if

13. Write short notes on SIMD multimedia extensions. [D][Apr/May-2018]

    SIMD Multimedia Extensions started with the simple observation that many media applications operate on narrower data types than the 32-bit processors were optimized for. Many graphics

systems used 8 bits to represent each of the three primary colors plus 8 bits for transparency. Depending on the application, audio samples are usually represented with 8 or 16 bits. By partitioning the carry chains within, say, a 256-bit adder, a processor could perform simultaneous operations on short vectors of thirty-two 8-bit operands, sixteen 16-bit operands, eight 32-bit operands, or four 64-bit operands.

14. Differentiate multimedia SIMD and Vector architecture.[D]

Multimedia SIMD Architecture	Vector Architecture
Like vector instructions, a SIMD instruction specifies the same operation on vectors of data. Unlike vector machines with large register files such as the VMIPS vector register, which can hold as many as sixty-four 64-bit elements in each of 8 vector registers, SIMD instructions tend to specify fewer operands and hence use much smaller register files.	
Multimedia SIMD extensions fix the number of data operands in the opcode, which has led to the addition of hundreds of instructions in the MMX, SSE, and AVX extensions of the x86 architecture. In contrast to vector architectures, which offer an elegant instruction set that is intended to be the target of a vectorizing compiler, SIMD extensions have three major	Vector architectures have a vector length register that specifies the number of operands for the current operation. These variable- length vector registers easily accommodate programs that naturally have shorter vectors than the maximum size the architecture supports. Moreover, vector architectures have an implicit maximum vector length in the architecture, which combined with the vector length register
Multimedia SIMD does not offer the more sophisticated addressing modes of vector architectures, namely strided accesses and gather-scatter accesses.	These features increase the number of programs that a vector compiler can successfully vectorizer.
Multimedia SIMD usually does not offer the mask registers to support conditional execution of elements as in vector architectures.	vector architectures offer the mask registers to support conditional execution of elements.

15. Write short notes on roofline visual performance model.[D]

One visual, intuitive way to compare potential floating-point performance of variations of SIMD architectures is the Roofline model. It ties together floating-point performance, memory performance, and arithmetic intensity in a two-dimensional graph. Peak floating-point performance can be found using the hardware specifications. Many of the kernels in this case study do not fit in on-chip caches, so peak memory performance is defined by the memory system behind the caches.

16. CUDA will simplify scheduling by hardware justify.[ID]

To simplify scheduling by the hardware, CUDA requires that thread blocks be able to execute independently and in any order. Different thread blocks cannot communicate directly, although they can coordinate using atomic memory operations in Global Memory. Many GPU hardware concepts are not obvious in CUDA. That is a good thing from a programmer productivity perspective, but most programmers are using GPUs instead of CPUs to get performance.

17. List out the hardware scheduler of GPU.[D]

GPU hardware has two levels of hardware schedulers:
i. The Thread Block Scheduler that assigns Thread Blocks (bodies of vectorized loops) to multithreaded SIMD Processors, which ensures that thread blocks are assigned to the processors whose local memories have the corresponding data

ii. The SIMD Thread Scheduler within a SIMD Processor, which schedules when threads of SIMD instructions should run.

18. What is mean by warp?[D]
19. Write short notes on Fermi GPU architecture.[D]

Fermi introduces several innovations to bring GPUs much closer to mainstream system processors than Tesla and previous generations of GPU architectures. The multithreaded SIMD *The Warp* is a traditional thread, but it contains just SIMD instructions that are executed on a multithreaded SIMD Processor. Results stored depending on a per-element mask. A Warp is a set of parallel CUDA Threads (e.g., 32) that execute the same instruction together in a multithreaded SIMT/SIMD processor. A streaming multiprocessor (SM) is a multithreaded SIMT/SIMD Processor that executes warps of CUDA Threads.

Processor of Fermi is more complicated than the simplified Version. To increase hardware utilization, each SIMD Processor has two SIMD Thread Schedulers and two instruction dispatch units. The dual SIMD Thread Scheduler selects two threads of SIMD instructions and issues one instruction from each to two sets of 16 SIMD Lanes, 16 load/store units, or 4 special function units. Thus, two threads of SIMD instructions are scheduled every two clock cycles to any of these collections. Since the threads are independent, there is no need to check for data dependences in the instruction stream. This innovation would be analogous to a multithreaded vector processor that can issue vector instructions from two independent threads.In Fermi's Dual SIMD Thread Scheduler, Each SIMD Lane has a pipelined floating-point unit, a pipelined integer unit, some logic for dispatching instructions and operands to these units, and a queue for holding results. The four Special Function units (SFUs) calculate functions such as square roots, reciprocals, sines, and cosines.

20. What is mean by error correcting codes? [D

Error Correcting Codes are used to to detect and correct errors in memory and registers

- To make long-running applications dependable on thousands of servers, ECC is the norm in the datacenter Dynamic errors can be detected by parity bits and detected and fixed by the use of Error Correcting Codes (ECCs). Because instruction caches are read-only, parity suffices. In larger data caches and in main memory, ECC is used to allow errors to be both detected and corrected.

21. Define arithmetic intensity.[D]

Arithmetic intensity is the ratio of floating-point operations per byte of memory accessed. It can be calculated by taking the total number of floating-point operations for a program divided by the total number of data bytes transferred to main memory during program execution. Some kernels have an arithmetic intensity that scales with problem size, such as dense matrix, but there are many kernels with arithmetic intensities independent of problem size.

22. Differentiate GPU and CPU)[D] [Nov/dec-2016]

GPUs and CPUs do not go back in computer architecture genealogy to a common ancestor; there is no Missing Link that explains both. The primary ancestors of GPUs are graphics accelerators, as doing graphics well is the reason why GPUs exist. While GPUs are moving toward mainstream computing, they can't abandon their responsibility to continue to excel at graphics.

CPU	GPU
A Graphics Processing Unit (GPU) is a special purpose processor, optimized for calculations commonly (and repeatedly) required for Computer Graphics, particularly SIMD operations.	A Central Processing Unit (CPU) is a general purpose processor - it can in principle do any computation, but not necessarily in an optimal fashion for any given computation.
Architecturally, the CPU is composed of just few cores with lots of cache memory that can handle a few software threads at a time.	A GPU is composed of hundreds of cores that can handle thousands of threads simultaneously.
The ability of a GPU with 100+ cores to process thousands of threads can accelerate some software by 100x over a CPU alone. What's more, the GPU achieves this acceleration while being more power- and cost-efficient than a CPU.	

23. List out the primary components of instruction set architecture of VMIPS.)[D] [Nov/dec-2016]
24. Differentiate vector architecture and GPUs.[D]

A SIMD Processor is like a vector processor. The multiple SIMD Processors in GPUs act as independent MIMD cores, just as many vector computers have multiple vector processors. This view would consider the NVIDIA GTX 480 as a 15-core machine with hardware support for multithreading, where each core has 16 lanes. The biggest difference is multithreading, which is fundamental to GPUs and missing from most vector processors.

Similarities to vector machines:

–Works well with data-level parallel problems
–Scatter-gather transfers
–Mask registers

25. What is mean by warp scheduler?[D]

A Warp Scheduler in a Streaming Multiprocessor schedules Warps for execution when their next instruction is ready to execute. Warp Scheduler is a hardware unit that schedules and issues threads of SIMD instructions when they are ready to execute; includes a scoreboard to track SIMD Thread execution.

26. Write short notes on SIMD lane and Thread processor.[D]

SIMD lane

The Hardware SIMD Lane that executes the operations in a thread of SIMD instructions on a single element. Results are stored depending on mask. OpenCL calls it a "processing element." AMD name is also "SIMD Lane".Registers in a single SIMD Lane allocated across body of vectorized loop. AMD also calls them "Registers".

Thread processor

A thread processor is a datapath and register file portion of a streaming multiprocessor that executes operations for one or more lanes of a warp. NVIDIA uses SIMT, singleinstruction multiple-thread, rather than SIMD, to describe a streaming multiprocessor.

27. List out the primary components of instruction set architecture of VMIPS.[D] [NOV/DEC-2016]
28. Use GCD test to identify if loop-carried dependency exists for the following

loop:for(i=1;i<=100;i++]
{A[3*i+5]=A[5*i+8]*7;}[ID] [NOV/DEC2017]

29. What is SIMD?[D][Apr/May-2018]
30. What are the properties of vector processor?[D]

PART-B

[First half]

[Vector architecture]

1. Explain about VIMPS vector architecture.[D] (16)
2. Discuss similarities and difference between vector architecture and GPUs.(16)[D][Nov/Dec-16]
3. Explain VMIPS vector instruction set in detail.16 [D]
4. Discuss the salient features of vector processor(10)[D] [NOV/DEC2017]
5. How do you determine the execution time of a sequence of vector operations? explain with example.(6)[ID] [NOV/DEC2017]
6. What is meant by gather-scatter? How it handles sparse matrices in vector architecture?(8) [ID]
7. Explain memory for supplying bandwidth for vector load/store units. (16)[D]
8. Explain with a diagram the basic structure of a vector processor.(16)[ID] [NOV/DEC2017]

9. With an example explain how vector length register handles a loop when it is not equal to 64.(8)[ID]
10. Explain in detail about roofline visual performance model.(8)[D]

[SIMD extensions]

11. Explain multithreaded SIMD processor. (8)
    Identify the true dependences, output dependences and anti dependences in the code given below and eliminate the name dependences through register renaming.(6)[ID][NOV/DEC2017]

```
For(i=0;i<100;i=i+1){
Y[i]=X[i]/c*S1*/
X[i]= X[i]+c;/*S2*/
Z[i] = X[i]+c;/*S3*/
Y[i]=c-Y[i];/ *S4*/
}
```

[Second half]

[Graphic processing units]

12. Present an outline of graphical processing units(8) [D][Apr/May-2018]
13. What are the architectural features that distinguish a GPU from a normal CPU? Discuss with a case study.(10)[ID] NOV/DEC2017]
14. Define GPU. Explain how programming can be performed in GPU with an example. (8)[ID]
15. Explain PTX GPU instruction set in detail. (16)[D]
16. Discuss about GPU memory structure. (10)[D]
17. Explain in detail about Fermi GPU structure.(6)
    [Loop level parallelism]
18. Explain detecting and enhancing loop level parallelism in detail.(16)[D] (Nov/Dec-2016)
19. Explain loop level parallelism with an example.(8) [D][Apr/May-2018]

UNIT IV THREAD LEVEL PARALLELISM
PART A

1. Define Thread-level parallelism (TLP)?[D]

   Thread-Level Parallelism, TLP is a software capability that allows high-end programs such as a database or web application to work with multiple threads at the same time. Programs that support this ability can do a lot more even under high levels of workloads.

   Thread-level parallelism (TLP) is the parallelism inherent in an application that runs multiple threads at once. This type of parallelism is found largely in applications written for commercial servers such as databases. By running many threads at once, these applications are able to tolerate the high amounts of I/O and memory system latency their workloads can incur - while one thread is delayed waiting for a memory or disk access, other threads can do useful work.

   TLP Use multiple instruction streams to improve

   - Throughput of computers that run many programs.
   - Execution time of multi-threaded programs.

2. What are the two different types of parallel structure in thread level parallelism?[D]
   - The processor must duplicate independent state of each thread e.g., a separate

copy of register file, a separate PC, and for running independent programs, a separate page table.

- memory shared through virtual memory mechanisms, which already support multiple processes.

3. Thread-level parallelism is an important alternative to instruction-level parallelism- Justify.[ID]

Yes, Thread-level parallelism is an important alternative to instruction-level parallelism. There are many techniques for converting such loop-level parallelism into instruction-level parallelism. Basically, such techniques work by unrolling the loop. An important alternative method for exploiting loop-level parallelism is the use of vector instructions on a vector processor.

• ILP exploits implicit parallel operations within a loop or straight-line code segment
• TLP explicitly represented by the use of multiple threads of execution that are inherently parallel.

4. Why MIMD architecture is preferred for Multiprocessors?[ID]

MIMD (multiple instruction, multiple data) is a technique employed to achieve parallelism. Machines using MIMD have a number of processors that function asynchronously and independently. At any time, different processors may be executing different instructions on different pieces of data. MIMD architectures may be used in a number of application areas such as computer-aided design/computer-aided manufacturing, simulation, modeling, and as communication switches. MIMD machines can be of either shared memory or distributed memory categories. So, MIMD architecture is preferred for Multiprocessors.

5. List the major MIMD Styles. What are the advantages of MIMD multiprocessor?[D] (Nov/Dec-13)

Major MIMD Styles:

- Centralized shared memory ("Uniform Memory Access" time or "Shared Memory Processor")
- Decentralized memory (memory modules associated with CPUs)

Advantages of MIMD multiprocessor

From a programmer's point of view, this memory model is better understood than the distributed memory model. Another advantage is that memory coherence is managed by the operating system and not the written program.

6. Define process and thread in the context of multi-processors. [D][Apr/May-2018]

With an MIMD, each processor is executing its own instruction stream. In many cases, each processor executes a different process. A *process* is a segment of code that may be run independently; the state of the process contains all the information necessary to execute that program on a processor. In a multiprogram med environment, where the processors may be running independent tasks, each process is typically independent of other processes. It is also useful to be able to have multiple processors executing a single program and sharing the code and most of their address space. When multiple processes share code and data in this way, they are often called threads. Today, the term *thread* is often used in a casual way to refer to multiple loci of execution that may run on different processors, even when they do not share an address space. For example, a multithreaded architecture allows the simultaneous execution of multiple processes, with potentially separate address spaces, as well as multiple threads that share the same address space.

7. Suppose you want to achieve a speedup of 80 with 100 processors. What fraction of the original computation can be sequential? [ID](Nov/Dec-14)

Amdahl's law is

$$\text{Speedup} = \frac{1}{\frac{\text{Fraction}_{\text{enhanced}}}{\text{Speedup}_{\text{enhanced}}} + (1 - \text{Fraction}_{\text{enhanced}})}$$

Assume that the program operates in only two modes: parallel with all processors fully used, which is the enhanced mode, or serial with only one processor in use. With

$$80 = \frac{1}{\frac{\text{Fraction}_{\text{parallel}}}{100} + (1 - \text{Fraction}_{\text{parallel}})}$$

Simplifying this equation yields:

$$\begin{aligned} 0.8 \times \text{Fraction}_{\text{parallel}} + 80 \times (1 - \text{Fraction}_{\text{parallel}}) &= 1 \\ 80 - 79.2 \times \text{Fraction}_{\text{parallel}} &= 1 \\ \text{Fraction}_{\text{parallel}} &= \frac{80 - 1}{79.2} \\ \text{Fraction}_{\text{parallel}} &= 0.9975 \end{aligned}$$

this simplification, the speedup in enhanced mode is simply the number of processors, while the fraction of enhanced mode is the time spent in parallel mode. Substituting into the previous equation: Thus, to achieve a speedup of 80 with 100 processors, only 0.25% of the original computation can be sequential. Of course, to achieve linear speedup (speedup of n with n processors), the entire program must usually be parallel with no serial portions

8. What are the advantages and disadvantages of using symmetric shared memory? [D](May/June-13)

Symmetric shared-memory machines usually support the caching of both shared and private data. Private data are used by a single processor, while shared data are used by multiple processors, essentially providing communication among the processors through reads and writes of the shared data.

Advantages:

Multiple processors are connected to multiple memory modules such that each processor can access any other processor's memory module. This multiprocessor employs a shared address space (also known as a single address space).

Communication is implicit with loads and stores – there is no explicit recipient of a shared memory access.

Processors may communicate without necessarily being aware of one another.

A single image of the operating system runs across all the processors.

Disadvantages:

Caches hold recently referenced data, as well as data near the recently referenced data. This can lead to performance increases by reducing the need to access main memory on every reference

9. Why Symmetric Shared memory architecture is called as UMA?[ID]

SMP architectures are also sometimes called uniform memory access (UMA) multiprocessors, arising from the fact that all processors have a uniform latency from memory, even if the memory is organized into multiple banks.

Uniform memory access (UMA), as in SMP:

- All processors have access to all parts of memory.
- Access time to all regions of memory is the same.
- Access time for different processors is the same.

10. How will you define the term shared memory?[D]

A shared memory is an extra piece of memory that is attached to some address spaces for their owners to use. As a result, all of these processes share the same memory segment and have access to it. Consequently, race conditions may occur if memory accesses are not handled properly. The following figure shows two processes and their address spaces.

11. What are the two difficulties used to make parallel processing challenges?[D]

Thread-level parallelism through two different software models.

- The first is the execution of a tightly coupled set of threads collaborating on a single task, which is typically called parallel processing.
- The second is the execution of multiple, relatively independent processes that may originate from one or more users, which is a form of request level parallelism.

Request-level parallelism may be exploited by a single application running on multiple processors.

12. Define multiprocessor cache processor. [D](May/June-12) (Apr/May-15)

A computer system which has two or more processors working simultaneously and sharing the same hard disk, memory and other memory devices such as cache can be defined as multiprocessor cache processor.

Advantages:

- Reduced Cost: Multiple processors share the same resources (like power supply and mother board).
- Increased Reliability: The failure of one processor does not affect the other processors though it will slow down the machine provided there is no master and slave processor.
- Increased Throughput: An increase in the number of processes completes the work in less time

13. What do you understand by Cache coherence Problem? Give an example. [ID]

In computer architecture, cache coherence is the uniformity of shared resource data that ends up stored in multiple local caches. When clients in a system maintain caches of a common memory resource, problems may arise with incoherent data, which is particularly the case with CPUs in a multiprocessing system.

- When two or more copies of a given datum exist in different processors' memories, it may lead to different processors having different values for the same variable.

  For example,

Time	Event	Cache contents for processor A	Cache contents for processor B	Memory contents for location X
0				1
1	Processor A reads X	1		1
2	Processor B reads X	1	1	1
3	Processor A stores 0 into X	0	1	0

This figure illustrates the problem and shows how two different processors can have two different values for the same location. This difficulty is generally referred to as the *cache coherence problem.*

14. State the reasons for Cache Coherence Problem. [ID]

Problem of inconsistency between a cached copy and the shared memory or between cached copies themselves due to the existence of multiple cached copies of data.

- Since all the processors share the same address space, it is possible for more than one processor to cache an address at the same time.(coherence issue).
- If one processor updates the data item without informing the other processor, inconsistency
- may result and cause incorrect execution.(consistency issue).

data between processors may cost anywhere from 50 clock cycles (for multicores) to over 1000 clock cycles (for large-scale multiprocessors), depending on the communication mechanism, the type of interconnection network, and the scale of the multiprocessor. The effect of long communication delays is clearly substantial.

15. When can we say that the memory is coherent in a multi-processor system?[D] (May/June-14)

In a multi-core processing system, a so-called memory coherence protocol notifies all the processing elements of changes to shared values, thereby ensuring that all copies of the data remain consistent. When multiple processors with on-chip caches are placed on a common bus sharing a common memory, then it's necessary to ensure that the caches are kept in a coherent state.

The exact nature and meaning of the memory coherency is determined by the consistency model that the coherence protocol implements. In order to write correct concurrent programs, programmers must be aware of the exact consistency model that is employed by their systems. When implemented in hardware, the coherency protocol can, e.g., be directory based or employ snooping (also called sniffing). Examples of specific protocols are the MSI protocol and its derivatives MESI, MOSI and MOESI.

16. What is known as coherence missing?[D]

The exact nature and meaning of the *memory coherency* is determined by the consistency model that the coherence protocol implements. In order to write correct concurrent programs, programmers must be aware of the exact consistency model that is employed by their systems.

The misses that arise from inter-processor communication, which are often called coherence misses, can be broken into two separate sources. The first source is the so-called true sharing misses that arise from the communication of data through the cache coherence mechanism. The second effect, called false sharing, arises from the use of an invalidation based coherence algorithm with a single valid bit per cache block.

17. Write a note on false sharing. [D] [Nov/dec-2017]

- False sharing arises from the use of an invalidation based coherence algorithm with a single valid bit per cache block.
- False sharing occurs when a block is invalidated because some word in the block, other than the one being read, is written into. If the word written into is actually used by the processor that received the invalidate.
- The word being written and the word read are different and the invalidation does not cause a new value to be communicated, but only causes an extra cache miss.

18. List the two protocols used to track the status of the shared data block. How the status is maintained in both the schemes? [D]

Directory-based coherence is a mechanism to handle Cache coherence problem in

Distributed shared memory (DSM) - Non-Uniform Memory Access (NUMA). Another popular way is to use a special type of computer bus between all the nodes as a "shared bus" (System bus). Directory-based coherence uses a special directory to serve instead of the shared bus in the bus-based coherence protocols. Both designs use the corresponding medium (i.e. directory or bus) as tool to facilitate the communication between different nodes, and to guarantee that the coherence protocol is working properly along all the communicating nodes.

Snooping: Each CPU (cache system) 'snoops' (i.e. watches continually) for write activity concerned with data addresses which it has cached. This assumes a bus structure which is 'global', i.e all communication can be seen by all. More scalable solution: 'directory based' coherence schemes.

19. Why do we need synchronization?[ID](May/June-14)

The key ability we require to implement synchronization in a multiprocessor is a set of hardware primitives with the ability to atomically read and modify a memory location. Without such a capability, the cost of building basic synchronization primitives will be too high and will increase as the processor count increases. One typical operation for building synchronization operations is the atomic exchange, which interchanges a value in a register for a value in memory.

20. How will you implement spin locks? [ID]

Spin locks is defined as a lock that a processor continuously tries to acquire, spinning around a loop until it succeeds. To implement spin locks, the coherence mechanisms of a multiprocessor can be used. Spin locks are used when programmers expect the lock to be held for a very short amount of time and when they want the process of locking to be low latency when the lock is available. Because spin locks tie up the processor, waiting in a loop for the lock to become free, they are inappropriate in some circumstances.

21. What is meant by consistency? [D](May/June-13)

Consistency states that data cannot be written that would violate the database's own rules for valid data. If a certain transaction occurs that attempts to introduce inconsistent data, the entire transaction is rolled back and an error returned to the user.

- A simple rule of consistency may state that the 'Gender' column of a database may only have the values 'Male' , 'Female' or 'Unknown'.

22. What is the importance of memory consistency model? [D](Nov/ Dec-13,APL/MAY2019]

The shared memory behaviour - not anything else related to the programs. We aren't interested in control flow within the programs, data manipulations within the programs, or behavior related to local (in the sense of non-shared) variables. There will be a line for each processor in the system, and time proceeds from left to right. Each shared-memory operation performed will appear on the processor's line.

The two main operations are Read and Write, which are

expressed as W(var)value
which means "write value to shared variable
var", and R(var)value
which means "Read value to shared variable var",

23. What is sequential consistency? ([D] [Nov/dec-2016] [D] [Nov/dec-2017]

Sequential consistency requires that the result of any execution be the same as if the memory accesses executed by each processor were kept in order.

- The accesses among different processors were arbitrarily interleaved.
- Sequential consistency eliminates the possibility of some nonobvious execution.
- Sequential consistency is to require a processor to delay the completion of any memory access until all the invalidations caused by that access are completed.

24. Define consistency memory model? [D](Nov/Dec-12)

Consistent memory must be seems simple. Here are two code Segments from processes P1 and P2, shown side by side:

```
P1:     A = 0;              P2:     B = 0;
        .....                       .....
        A = 1;                      B = 1;
L1:     if (B == 0)...      L2:     if (A == 0)...
```

- Assume that the processes are running on different processors, Locations
- A and B are originally cached by both processors with the initial value of 0.

25. List the methods for providing synchronization in threads. [D] (Nov/Dec-16)

There are number of other atomic primitives that can be used to implement synchronization. One operation, present in many older multiprocessors, is test-and-set, which tests a value and sets it if the value passes the test. For example, we could define an operation that tested for 0 and set the value to 1, which can be used in a fashion similar to how we used atomic exchange. Another atomic synchronization primitive is fetch-and-increment: It returns the value of a memory location and atomically increments it. By using the value 0 to indicate that the synchronization variable is unclaimed.

26. What is simultaneous multithreading (SMT)? [D] (Nov/Dec -12) (Nov/Dec-14)

Simultaneous multithreading support (to four threads per core) and the number of cores per chip to eight.

- Simultaneous multithreading is a variation on fine-grained Multithreading that arises naturally when fine-grained multithreading is Implemented on top of a multiple-issue, dynamically scheduled processor.
- SMT uses thread-level parallelism to hide long-latency events in a processor, thereby increasing the usage of the functional units.

27. Enlist the features of SMT architecture. [D] (Apr/May-15)

- SMT Architecture features distinct register files, holding both active and inactive threads, that allow both single-instruction-per-thread issue, and multiple instruction issue.
- The processor features standby stations associated with each functional unit which hold dependence-free instructions

. Scheduling of instructions onto standby stations/functional units uses a distributed mechanism.

28. Write the benefits of multicore architecture. [D] (May/June-14) (U)

- The good processing speed of the multicore processors is due to the multiple cores which operate simultaneously on instructions, at lower frequency than the single core.
- At the same clock frequency, the multicore processor will process more data than the single core processor.
- Multicore processors deliver high performance and handle complex tasks at a comparatively lower energy or lower power as compared with a single core,

29. Why design issues of SMT and CMT architectures are important? [ID] (May/June-14)

The issue slots usage limitations and its issues also determine the performance. In Multithreading, ILP is exhausted, as TLP is in. Large performance gap between MEMORY and PROCESSOR. Too many transistors on chip. More existing MT applications today. Multiprocessors on a single chip. Long network latency, too.

30. Write the advantages of CMP Architecture?[D] (MAY/JUN'12) (NOV/DEC'12)

- CMPs are more area and energy efficient than single processor solutions. CMPs

achieve greater throughput than single processor solutions as more work can be done concurrently.

- Custom multi-processor systems can now be designed and simulated from the ground up using software solutions from several companies.
  CMPs have several advantages over single processor solutions
  - Energy and silicon area efficiency
  - By Incorporating smaller less complex cores onto a single chip
  - Dynamically switching between cores and powering down unused cores [5]
  - Increased throughput performance by exploiting parallelism
  - Multiple computing resources can take better advantage of instruction, thread, and process level parallelism.

31. List the methods for providing synchronization in threads [D] [Nov/dec-2016]

Amount of computation assigned to each thread. = grain size. ▫ Threads can be used for data-level parallelism, but the overheads may ... Distributed shared memory. (DSM). ▫ Memory. Shift to distributed memory to support the bandwidth .

PART-B

[First half]

[Symmetric and distributed shared memory architectures]

1. Described distributed shared memory architecture in detail(16) [D] [Nov/dec-2016]
2. How to measure the performance of symmetric shared multiprocessor? Explain.(13)[ID] Apr/May -19).
3. What is multiprocessor? Explain with a diagram the basic structure of a symmetric shared memory multiprocessor.(8)[D][Apr/May2018]
4. With neat block diagram explain the centralized shared memory multiprocessor architecture. (16)[D] [Nov/Dec 2012)
5. With neat block diagram explain the centralized shared memory multiprocessor architecture.(16)[D](Nov/Dec 2012)
6. Explain the directory–based cache coherence protocol in distributed shared memory architecture.(16)[D] (Nov/Dec-13,16)
7. Discuss the performance of Symmetric Shared-Memory Multiprocessors for a multi-programmed workload consisting of both user activity and OS activity.(16)[D] (May/June-12) (Apr/May -15).
8. Discuss the various cache-coherence protocols used in symmetric shared memory architecture.(8)[D] (May/June-12).
9. Explain the concept of simultaneous Multi Threading(6)[D][Nov/Dec2017]

[Performance issues , Synchronization]

10. What are the hardware primitives available to resolve synchronization issues in a multiprocessor environment? Give examples.(8)[D] (Nov/Dec-13) (May/June-12).
11. Explain how to implement synchronization in a multiprocessor using a set of hardware primitives with the ability to automatically read and modify a memory locatin. (8)[D][May/Jun2018]

[Models of memory consistency]

12. Explain various memory consistency models in detail.(16)[D] (May/June-12, 13) (Nov/Dec-13, 14, 16) (Apr/May-15) Apr/May -19).
13. What do you mean by snooping protocol? Explain how it is used to maintain the coherence. (8) [D]
14. Explain the snoop based cache coherence protocol with a state diagram.(10) [D][Nov/Dec2017]
15. How are spin locks implemented using cache coherence?(6)[D][Nov/Dec2017]

[Case studies Intel i7 processor, SMT and CMP processors]

16. Discuss the design challenges of SMT architecture.(8)[D] (Nov/Dec-13) (May/June-13).
17. Explain Intel multi core architecture with its benefits. (8)[D]
18. Explain in detail about the CMP architecture and its performance.(16)[D] (Nov/Dec- 13-14).
19. Compare and contrast Intel multi core architecture and SUN CMP architecture.(16)[D](Apr/May-15).
20. Discuss the silent features of the Intel i7 processor(10)[D][Nov/Dec2017]

UNIT 5 MEMORY AND I/O

PART A

1. List the basic optimization techniques of cache.[D][Nov/Dec2016]

    - Larger block size reduces miss rate
    - ii) Bigger caches too reduce miss rate (but expensive)
    - iii) Higher associativity to reduce miss rate (Associativity to be explained later)
    - iv) Multilevel caches can reduce miss penalty
    - v) Priority to read misses over writes can reduce miss penalty
    - vi) Avoiding address translation (involving virtual memory) during indexing of cache to reduce hit tim

2. What are the types of storage devices?[D][Nov/Dec2016]
    - -Primary Magnetic Storage devices
    - –Magnetic Disks Optical Disks Magnetic Tape
    - Automated Tape Libraries Flash Memory Primary Optical Storage
    - Data Transfer Rate
3. Define the term reliability and availability.[D][Nov/Dec2017]
    - Reliability – measured by the mean time to failure (MTTF). Service interruption is measured by mean time to repair (MTTR)
    - Availability – a measure of service accomplishment
    - Availability = MTTF/(MTTF + MTTR)

4. Point out one simple techniques used to reduce each of the three ‘C’ misses in cache memories.[ID][Nov/Dec2017] –
    - Reduce Misses via Larger Block Size
    - -Reduce Misses via Higher Associatively
    - -Compiler Based generation
    - -Advantages Loop Interchanges
        - Effective Reuse of Data Cache
        - Loop Unrolling
        - Loop Fusion/Fission
        - Pre fetching –
    - Floating Point Division
5. Outline the difference between volatile memory and non volatile memory.[ID][May/Jun2018]
6. Define cache Hit and cache Miss[D][May/Jun2018] .[D] (Nov/Dec 2011) (May/June 2013)
   A cache miss, generally, is when something is looked up in the cache and is not found. The cache did not contain the item being looked up
7. What do you mean by write through cache and write back cache?[ID](April/May 2011)

Write through cache: The information is written to both the block in the cache and to the block in the lower level memory.

Write Back Cache: The information is written only to the block in the cache. The• modified cache block is written to main memory only when it is replaced.

8. What are the categories of cache organization based on placing a block?[D] (Nov/Dec 2013)
9. State the principle of locality.[D] (May/June 2014)
10. List the types of locality[D] (May/June 2014)
11. Write about memory hierarchy.[D]
12. Show the multilevel memory hierarchy with speed and size.[ID]
13. What is branch straightening?[D]
14. Define i) local miss rate ii) global miss rate[D]
15. Define the terms: access time, bandwidth (May/June 2014)[D]
16. Compare SRAM with DRAM[D]

SRAM DRAM 1.

1. SRAM does not need to be refreshed as the transistors inside would continue to hold the data as long as the power supply is not cut off. This behavior leads to a few advantages, not the least of which is the much faster speed that data can be written and read.

1. DRAM requires the data to be refreshed periodically in order to retain the data

2. SRAM needs a lot more transistors in order to store a certain amount of memory. A DRAM module only needs a transistor and a capacitor for every bit of data where SRAM needs 6 transistors.

2. DRAM has become the mainstream in computer main memory despite being slower and more power hungry compared to SRAM

17. How to improve the memory Bandwidth?[D]
18. What is flash memory?[D]
19. Give the comparison between flash and disk.[D]
20. What is the average time to read or write a 512 byte sector for a disk? The advertised average seek time is 5ms, the transfer rate is 40 MB/second, it rotates at 10000 RPM, and the controller overhead is 0.1ms. Assume the disk is idle so that there is no queuing delay.[ID] (May/June 2012)
21. What is the bus master?[D] (May/June 2013)

The device that is allowed to initiate data transfers on the bus at any given time is called bus master.

22. What is CPU- memory bus?[D]
23. Compare software and hardware RAID.[D](Nov/Dec 2012)
24. Define MTTR ii) MTTF.[D]
25. What are the standard levels of RAID?[D](Nov/Dec 2011) Apr/May -19).
26. How the row diagonal parity works?[D]
27. Define dependability.[D]
28. Define faults, errors, failures. Give example.ID]
29. Give the relation between faults, errors and failures.[D]
30. Differentiate between throughput and response time?[D] Apr/May -19).

PART B

[FIRST PART]

[Cache Performance , Reducing Cache Miss penalty and Miss Rate , Redusing Hit time]

1. Explain the categories of misses and how will you reduce cache miss rate(16)[DN][Nov/Dec2016]

2. Explain the various techniques available for reducing cache miss rate.(16)[D] (May/June 2012)(Nov/Dec 2013]
3. Explain the categories of misses and how will you reduce cache miss rate. (16)[D](Nov/Dec 2016).
4. Discuss any five advanced cache coherence protocol with a stated diagram(10)[D][Nov/Dec 2018]
5. Suppose that in 1000 memory reference there are 50 misses in the first level cache and 20 misses in the second level cache. What are the various miss rate? Assume the miss penalty from the L2 cache to memory is 100 clock cycles the hit time of the L2 cache is 10 clock cycles. The hit time of L1 is1 clock cycle and there are 2 memory references per instruction. What is the average memory access time?(6)[ID][Nov/Dec2017]
6. Discuss the various techniques available for reducing cache miss penalty.(16)[D] (May/June2012,14,16)
7. Explain the various hit time reduction technique.(16)[D] (Nov/Dec 2013)
8. Explain with an example any two techniques for improving the cache performance by reducing the miss rate(16)[D][May/Jun2018]

[SECOND PART]

[Main memory and performance, Memory technology, Types of storage devices]

9. Discuss the concept of virtual memory and explain how a virtual memory system is implemented , pointing out the hardware and software support(10)[ID][Nov/Dec2017]
10. Explain in detail about main memory and its performance. (16)[D]
11. Briefly explain the types of storage devices(8).[D] (May/June 2014)
12. Discuss in detail about Virtual memory. (8)[D]
13. Explain in detail about memory technology.(8)[D]

[Buses , RAID – Reliability, Availability and Dependability]

14. Explain the various levels of RAID.(16)[D] (May/June 2013,14) (Nov/Dec 2012,13,]
15. Explain about reliability, availability and dependability. (8)
16. Describe about bus. (8)[D]
17. Discuss the different levels of RAID technology , listing their advantages and disadvantages . (6)[D] [Nov/Dec2017]
18. Explain the Bus standards and the interfaces, with timing diagrams. Explain the read and write operations occurring in a typical bus. (16)[D]
    [I/O Performance Measures]
19. Discuss in detail about various I/O performance measures.(16)[D] (May/June 2013, 14) (Nov/Dec 2016).
20. Explain the various of measure I/O performance(8)[D][Nov/Dec2016]

****