PART B

1. Explain the basic MIPS implementation of instruction set
2. Explain the basic MIPS implementation with necessary multiplexers and control lines
3. What are control hazards? Explain the methods for dealing with the control hazards.
4. Discuss the influence of pipelining in detail
5. Explain how the instruction pipeline works. What are the various situations where an instruction pipeline can stall? What can be its resolution?
6. What is data hazard? How do you overcome it?What are its side effects?
7. Discuss the data and control path methods in pipelining
8. Explain dynamic branch prediction
9. How exceptions are handled in MIPS
10. Explain in detail about building a datapath
11. Explain in detail about control implementation scheme

UNIT IV
Parallelism

PART A

1. What is Instruction level parallelism?

ILP is a measure of how many of the operations in a computer program can be performed simultaneously. The potential overlap among instructions is called instruction level parallelism. There are two primary methods for increasing the potential amount of instruction-level parallelism. 1. Increasing the depth of the pipeline to overlap more instructions. 2. Multiple issue.

2. What is multiple issue? Write any two approaches.

Multiple issue is a scheme where multiple instructions are launched in one clock cycle. It is a method for increasing the potential amount of instruction-level parallelism. It is done by replicating the internal components of the computer so that it can launch multiple instructions in every pipeline stage.

The two approaches are:

1. Static multiple issue (at compile time)
2. Dynamic multiple issue (at run time)

3. Define Static multiple issue and Dynamic multiple issue.

Static multiple issue -An approach to implementing a multiple-issue processor where many decisions are made by the compiler before execution.

Dynamic multiple issue -An approach to implementing a multiple-issue processor where many decisions are made during execution by the processor.

4. What is Speculation?

An approach whereby the compiler or processor guesses the outcome of an instruction to remove it as a dependence in executing other instructions.

5. Define Issue Slots and Issue Packet

Issue slots are the positions from which instructions could be issued in a given clock cycle. By analogy, these correspond to positions at the starting blocks for a sprint. Issue packet is the set of instructions that issues together in one clock cycle; the packet may be determined statically by the compiler or dynamically by the processor.

6. Define VLIW

Very Long Instruction Word (VLIW) is a style of instruction set architecture that launches many operations that are defined to be independent in a single wide instruction, typically with many separate opcode fields.

7. Define Use latency.

Number of clock cycles between a load instruction and an instruction that can use the result of the load with-out stalling the pipeline.

8. What is Loop unrolling?

A technique to get more performance from loops that access arrays, in which multiple copies of the loop body are made and instructions from different iterations are scheduled together.

9. Define Register renaming.

The renaming of registers by the compiler or hardware to remove anti-dependences.

10. What is Superscalar and Dynamic pipeline schedule?

Superscalar-An advanced pipelining technique that enables the processor to execute more than one instruction per clock cycle by selecting them during execution.

Dynamic pipeline schedule-Hardware support for reordering the order of instruction execution so as to avoid stalls.

11. Define Commit unit.

The unit in a dynamic or out-of-order execution pipeline that decides when it is safe to release the result of an operation to programmer visible registers and memory.

12. What is Reservation station?

A buffer within a functional unit that holds the operands and the operation.

13. What is the use of reservation station and reorder buffer?

Reservation station is a buffer within a functional unit that holds the operands and the operation. Reorder buffer is the buffer that holds results in a dynamically scheduled processor until it is safe to store the results to memory or a register.

14. Differentiate in-order execution from out-of-order execution.

Out-of-order execution is a situation in pipelined execution when an instruction is blocked from executing does not cause the following instructions to wait. It preserves the data flow order of the program.

In-order execution requires the instruction fetch and decode unit to issue instructions in order, which allows dependences to be tracked, and requires the commit unit to write results to registers and memory in program fetch order. This conservative mode is called in-order commit.

15. What is In order commit?

A commit in which the results of pipelined execution are written to the programmer visible state in the same order that instructions are fetched.

16. Define Strong scaling and weak scaling.

Strong scaling: Speed-up achieved on a multi-processor without increasing the size of the problem.

Weak scaling: Speed-up achieved on a multi-processor while increasing the size of the problem proportionally to the increase in the number of processors.

17. Define Single Instruction, Single Data stream (SISD)

A sequential computer which exploits no parallelism in either the instruction or data streams. Single control unit (CU) fetches single Instruction Stream (IS) from memory. The CU then generates appropriate control signals to direct single processing element (PE) to operate on single Data Stream (DS) i.e. one operation at a time.
Examples of SISD architecture are the traditional uniprocessor machines like a PC

18. Define Single Instruction, Multiple Data streams (SIMD) and Multiple Instruction,Single Data stream (MISD)

Single Instruction, Multiple Data streams (SIMD):A computer which exploits multiple data streams against a single instruction stream to perform operations which may be naturally parallelized. For example, an array processor or GPU.

Multiple Instruction, Single Data stream (MISD): Multiple instructions operate on a single data stream. Uncommon architecture which is generally used for fault tolerance. Heterogeneous systems operate on the same data stream and must agree on the result. Examples include the Space Shuttle flight control computer.

19. Define Multiple Instruction, Multiple Data streams (MIMD) and Single program multiple data streams .

Multiple Instruction, Multiple Data streams (MIMD): Multiple autonomous processors simultaneously executing different instructions on different data. Distributed systems are generally recognized to be MIMD architectures; either exploiting a single shared memory space or a distributed memory space. A multi-coresuperscalar processor is an MIMD processor.

Single program multiple data streams : Multiple autonomous processors simultaneously executing the same program on different data.

20. Define multithreading.

Multiple threads to share the functional units of 1 processor via overlappingprocessor 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 tablememory shared through the virtual memory mechanisms, which already support multiple processes

21. What are Fine grained multithreading and Coarse grained multithreading?

Fine grained multithreading: Switches between threads on each instruction, causing the execution of multiples threads to be interleaved,Usually done in a round-robin fashion, skipping any stalled threads, CPU must be able to switch threads every clock

Coarse grained multithreading: Switches threads only on costly stalls, such as L2 cache misses

22. What is meant by hardware multithreading?

Hardware multithreading allows multiple threads to share the functional units of a single processor in an overlapping fashion to try to utilize the hardware resources efficiently. To permit this sharing, the processor must duplicate the independent state of each thread. It increases the utilization of a processor.

23. Define Multicore processors.

A multi-core processor is a processing system composed of two or more independent cores. The cores are typically integrated onto a single integrated circuit die or they may be integrated onto multiple dies in a single chip package.

24. What are symmetric multi-core processor and asymmetric multi-core processor?

A symmetric multi-core processoris one that has multiple cores on a single chip, and all of those cores are identical.Example: Intel Core 2.

In an asymmetric multi-core processor, the chip has multiple cores onboard, but the cores might be different designs.Each core will have different capabilities

25.What is SMT?

Simultaneous Multithreading (SMT) is a variation on hardware multithreading that uses the resources of a multiple-issue, dynamically scheduled pipelined processor to exploit thread-level parallelism. It also exploits instruction level parallelism.

26. Define SMP

Shared memory multiprocessor (SMP) is one that offers the programmer a single physical address space across all processors - which is nearly always the case for multicore chips. Processors communicate through shared variables in memory, with all processors capable of accessing any memory location via loads and stores

27. Differentiate UMA from NUMA.

Uniform memory access (UMA) is a multiprocessor in which latency to any word in main memory is about the same no matter which processor requests the access. Non uniform memory access (NUMA) is a type of single address space multiprocessor in which some memory accesses are much faster than others depending on which processor asks for which word.

PART-B

1. Explain Instruction level parallelism.
2. Explain challenges in parallel processing.
3. Explain in detail about Hardware multithreading.
4. Discuss in detail about Flynn's classification.
5. Explain SISD and SIMD with an example.
6. Explain MISD and MIMD with an example
7. Explain shared memory multiprocessor
8. Explain Multicore processors
9. . Explain the different types of multithreading

UNIT-V
Memory and I/O Systems

Part-A

1. What is principle of locality?

The principle of locality states that programs access a relatively small portion of their address space at any instant of time.

2. What are the temporal and spatial localities of references?

Temporal locality: The principle stating that a data location is referenced then it will tend to be referenced again soon.