Q1(i) Characteristics feature of cluster/wsc ?

* **Clusters or warehouse-scale computers look like individual computers connected by a network that do not share memory**

1)Clusters emphasize thread-level parallelism, WSCs emphasize request-level parallelism

2) Clusters have higher performance processors and network as compare to WSC.

X-----------------------------------X---------------------------------X-------------------------------X-----------------------X

Q1(ii) why benchmarks contain collection of programs?

To overcome the danger of placing too many eggs in one basket, benchmark suites are a popular measure of performance of processors with variety of applications. A key advantage of such suites is that the weakness of any 1 benchmark is lessened by the presence of other benchmark.

X-----------------------------------X---------------------------------X-------------------------------X-----------------------X

Q1 (b) The vector unit of a processor is enhanced to get a speedup of 10. What percentage of vectorization is required in applications to get an overall speedup of 2?

Overall speedup = 1 {(1−𝐹𝑒 )+ 𝐹𝑒 𝑆𝑝𝑒 }

2= 1 {(1−𝑥)+ 𝑥 10}

2 – 2x + 1 5 𝑥 = 1

9𝑥 5 = 1

𝑥 = 5 9 = 0.556

Or 56 %

X-----------------------------------X---------------------------------X-------------------------------X-----------------------X

Q1 (c)

Execution time of a program P1 on M1 is given as 5 m Sec. The clock frequency of M1 is 3.24 GHz. What are the total number of instructions in Pl for the following information available about P1:

Instruction type No. of cycles Percentage

Type A 2 22%

Type B 1 27%

Type C 4 17%

Type D 5 Remaining

f = 3.24×109 GHz

Execution time = 5×10-3 Sec

CPItotal = ?

CPItotal = (0.22 x 2) + (0.27 x 1) + (0.17 x 4) + ( 0.34 x 5) = 0.44 + 0.27+ 0.68+ 1.70

CPItotal = 3.09

CPU time (Execution time) = 𝐼𝐶 × 𝐶𝑃𝐼 𝑓𝑐𝑙𝑜𝑐𝑘

5 × 10−3 = 𝐼𝐶 × 3.09 3.24×109

IC = 5×3.24×109−3 3.09

IC = 5.24 × 106 Insctructions

X-----------------------------------X---------------------------------X-------------------------------X-----------------------X

Q2 (a)(i)

FIXED LENGTH INSTRUCTION is better than VARIABLE LENGTH INSTRUCTION?

A minor advantage of fixed length instructions compared to typical variable length encodings is that instruction addresses (and branch offsets) use fewer bits. This has been exploited in some ISAs to provide a small amount of extra storage for mode information

X-----------------------------------X---------------------------------X-------------------------------X-----------------------X

Q2 (a)(ii)

characteristics features of load/store architecture ?

Load/store architecture which fetches operands and results indirectly from main memory through a lot of scalar registers.

-Fixed length instructions

- Pipelined implementations

- compound instructions

X-----------------------------------X---------------------------------X-------------------------------X-----------------------X

Q2(b)

What do you mean by single core processor?

Definition of: **single core**. **single core**. A chip with one **CPU** (one processing unit). Microprocessors have been **single core** since their inception in the early 1970s. After the turn of the century, chips with two or more CPUs emerged.

What does multi core processor mean?

In consumer technologies, **multi**-**core** is usually the term used to describe two or more CPU’s working together on the same chip. Also called **multi core** technology, it **is a** type of architecture where a single physical processor contains the **core** logic of two or more processors.

homogeneous processors vs heterogeneous processors :

A system consisting of multiple **processors** which execute different instruction sets is called **heterogeneous**. On the contrary, if all the **processors** in a system execute the same instruction set, then it is **homogeneous.**

X-----------------------------------X---------------------------------X-------------------------------X-----------------------X

Q2(c) Methods use to specify branch conditions?

1)STATIC BRANCH PREDICTION:

Use profile information collected from earlier runs.

Advantage: Different runs lead to a small change in accuracy.

Disadvantage: Higher branch frequency limitation.

2)DYNAMIC BRANCH PREDICTION:

Memory contains a bit that says if branch was recently taken or not.

Advantage: Useful to reduce branch delay

Disadvantage: Don’t know if prediction is correct or not.

X-----------------------------------X---------------------------------X-------------------------------X-----------------------X

Q3(a)(i)

difference between user define and coerced exception?

**user**-requested **exceptions** are not really **exceptions**, since they are predictable. ... **Coerced exceptions** are harder to implement because they are not predictable

X-----------------------------------X---------------------------------X-------------------------------X-----------------------X

Q3(a)(ii)

### 3 DYNAMIC BRANCH TECHNIQUES:

### Dynamic branch prediction

Dynamic branch prediction uses information about taken or not taken branches gathered at run-time to predict the outcome of a branch.

### 1)Random branch prediction

### 2)Next line prediction

### 3)One-level branch prediction

X-----------------------------------X---------------------------------X-------------------------------X-----------------------X

Q3(b)

A processor comprises of five steps of execution having the following delays :S 1=32nSec•, S2 = 25nSec•, S3 = 20nSec; S4 = 25nSec; S5= 20nSec. If the processor is pipelined, what should be the clock frequency if the pipeline latch delay is ln Sec? Give the speedup of the pipelined processor over its non-pipelined counterpart.

GivenFive stage pipeline is used Delay of stages = 32, 25, 20,25 and 20 ns

Latch delay or delay due to each register =1 ns

Pipeline Cycle Time

Cycle time = Maximum delay due to any stage + Delay due to its register

= Max {32, 25, 20,25, 20 } + 1 ns

= 32 ns + 1 ns = 33 ns

Clock frequency = 1 𝐶𝑙𝑜𝑐𝑘 𝑐𝑦𝑐𝑙𝑒 Clock frequency = 1 33 × 10−9 = 30.303 × 106

𝑪𝒍𝒐𝒄𝒌 𝒇𝒓𝒆𝒒𝒖𝒆𝒏𝒄𝒚 = 𝟑𝟎. 𝟑𝟎𝟑 𝑴𝑯𝒁

Non-Pipeline Execution Time

Non-pipeline execution time for one instruction

= (32+25+ 20+25+ 20) ns = 122 ns

Speed Up RatioSpeed up = Non-pipeline execution time / Pipeline execution time

= 122 ns / 33 ns

Speed up = 3.697

X-----------------------------------X---------------------------------X-------------------------------X-----------------------X

Q3(c):

Precise Exception?

An exception for which the pipeline can be stopped, so instructions that preceded the faulting instruction can complete.

**precision exception** occurs if the result of ... which can happen only for a **floating**-**point** store, the inexact-result condition ... Because the integer **unit** and x87 FPU are separate execution **units**

X-----------------------------------X---------------------------------X-------------------------------X-----------------------X

Q4(a)(i):

There are three situations in which a **data hazard** can occur:

Data dependence (instruction) refers to the **data** of a preceding statement

(**RAW**), a **true** dependency.

**Name dependence**: when 2 instructions use same register or. memory location,

(WAR), an anti-dependency.

(WAW), an output dependency

X-----------------------------------X---------------------------------X-------------------------------X-----------------------X

Q4(a)(ii):

SEARCHING OF BLOCK IN A CACHE ?

To **search** a block in the **cache** ... In a **direct mapped cache** structure, the **cache** is organized into multiple sets .... The placement policy is slow as it **takes** time to iterate through all the lines.

X-----------------------------------X---------------------------------X-------------------------------X-----------------------X

Q5(a)(ii):

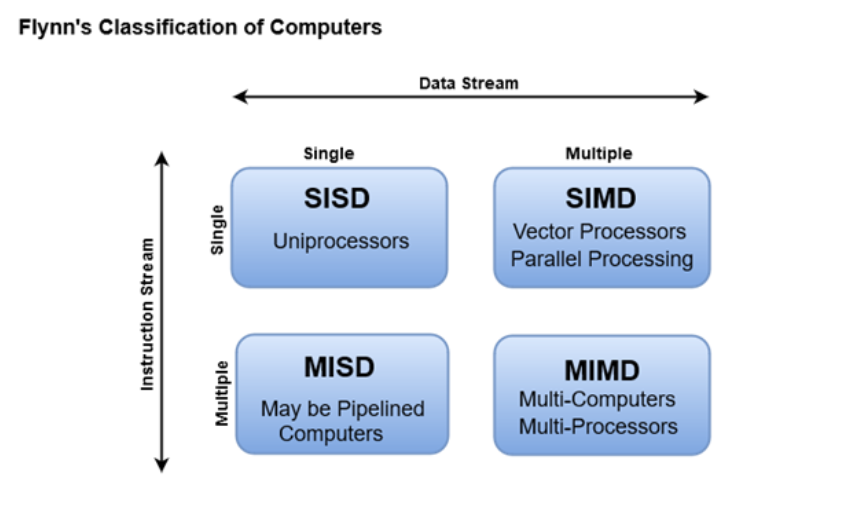
WRITE THROUGH AND WRITE BACK CACHE:

**Write-through cache** directs write I/O onto cache and through to underlying permanent storage before confirming I/O completion to the host. This ensures data updates are safely stored on, for example, a shared storage array, but has the disadvantage that I/O still experiences latency based on writing to that storage. Write-through cache is good for applications that write and then re-read data frequently as data is stored in cache and results in low read latency.

**Write-back cache**is where write I/O is directed to cache and completion is immediately confirmed to the host. This results in low latency and high throughput for write-intensive applications, but there is data availability exposure risk because the only copy of the written data is in cache. As we will discuss later, suppliers have added resiliency with products that duplicate writes. Users need to consider whether write-back cache offers enough protection as data is exposed until it is staged to external storage. Write-back cache is the best performer for mixed workloads as both read and write I/O have similar response time levels.

X-----------------------------------X---------------------------------X-------------------------------X-----------------------X

Q6(a)(i):



X-----------------------------------X---------------------------------X-------------------------------X-----------------------X

Q6(a)(ii):

# Challenges in Parallel Processing:

- Finding concurrency in a program

- The data locality problem

- Scalability support in hardware

- Scalability support in software

X-----------------------------------X---------------------------------X-------------------------------X-----------------------X

Q6(b)

CACHE OPTIMIZATION TECHNIQUES:

a **Non**-**blocking cache** will accept further requests and try to service those requests. ... A "hit-under-X-misses" **cache** will allow X number of misses to be outstanding in the **cache** before **blocking**. For example, a"hit-under-2-misses" **cache** will keep running if there are at most 2 misses that still not complete

**multi**-**level caches**, refers to a memory architecture which uses a hierarchy of memory stores based on varying access speeds to **cache** data

Hardware **prefetching** is a technique used by computer processors to boost execution performance by fetching instructions or data from their original storage in slower memory to a faster local memory before it is actually needed

**banked cache**, the **cache** is divided into a **cache** dedicated to instruction storage and a **cache** dedicated to data.

X-----------------------------------X---------------------------------X-------------------------------X-----------------------X

Differentiate between ILP (Instruction Level Parallelism) and DLP (Data Level Parallelism?

Instruction-level parallelism (ILP) is a measure of how many of the instructions in a computer program can be executed simultaneously.

Data Level Parallelism (DLP): A data parallel job on an array of 'n' elements can be divided equally among all the processors.

X-----------------------------------X---------------------------------X-------------------------------X-----------------------X

What are temporal and spatial locality observed in the behavior of programs?

Two different types of locality have been observed.

Temporal locality states that recently accessed items are likely to be accessed soon.

Spatial locality says that items whose addresses are near one another tend to be referenced close together in time

X-----------------------------------X---------------------------------X-------------------------------X-----------------------X

What are the characteristic features of servers?

First, availability is critical. Consider the servers running ATM machines for banks or airline reservation systems. 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.

A second key feature of server systems is scalability. Server systems often grow in response to an increasing demand for the services they support or an expansion 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.

Finally, 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 second—is 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.

X-----------------------------------X---------------------------------X-------------------------------X-----------------------X

. When does a phase ordering problem occur while using compiler optimization techniques?

Enforced ordering of some optimization introduce phase ordering problem.

X-----------------------------------X---------------------------------X-------------------------------X-----------------------X

What are the desirable instruction set properties that help a compiler writer?

Four properties in ISA

1. Regularity

2. Provide principle

3. Simplify tradeoffs among alternatives

4. Provide instructions that bind the quantities known at compile time as constants.

X-----------------------------------X---------------------------------X-------------------------------X-----------------------X

Synchronous exceptions: If the event occurs at the same place every time the program is executed with the same data and memory allocation, the event is synchronous.

Asynchronous exceptions: With the exception of hardware malfunctions, asynchronous events are caused by devices external to the processor and memory. Asynchronous events usually can be handled after the completion of the current instruction, which makes them easier to handle.

X-----------------------------------X---------------------------------X-------------------------------X-----------------------X

Which exceptions are the most difficult to handle in pipelined processors and why?

As in unpipelined implementations, the most difficult exceptions have two properties:

(1) they occur within instructions (that is, in the middle of the instruction execution corresponding to EX or MEM pipe stages)

(2) they must be restartable. In our RISC V pipeline, for example, a virtual memory page fault resulting from a data fetch cannot occur until sometime in the MEM stage of the instruction. By the time that fault is seen, several other instructions will be in execution. A page fault must be restartable and requires the intervention of another process, such as the operating system. Thus, the pipeline must be safely shut down and the state saved so that the instruction can be restarted in the correct state.

X-----------------------------------X---------------------------------X-------------------------------X-----------------------X

What do you understand by the latency and initiation interval of a pipelined functional unit?

We define latency the same way we defined it earlier: the number of intervening cycles between an instruction that produces a result and an instruction that uses the result. The initiation or repeat interval is the number of cycles that must elapse between issuing two operations of a given type.

X-----------------------------------X---------------------------------X-------------------------------X-----------------------X

What is a basic block? Suggest a compiler-based technique to improve the size of the basic block.

a basic block—a straight-line code sequence with no branches in except to the entry and no branches out except at the exit—is quite small. The simplest and most common way to increase the ILP is to exploit parallelism among iterations of a loop. This type of parallelism is often called loop-level parallelism. We will examine a number of techniques for converting such loop-level parallelism into instruction-level parallelism. Basically, such techniques work by unrolling the loop either statically by the compiler

X-----------------------------------X---------------------------------X-------------------------------X-----------------------X

What are the advantages of dynamic scheduling over static scheduling of pipelined processors?

dynamic scheduling, where the hardware rearranges the instruction execution to reduce the stalls. Dynamic scheduling offers several advantages:

• It enables handling some cases when dependencies are unknown at compile time (e.g., because they may involve a memory reference);

• It simplifies the compiler;

• It allows code that was compiled with one pipeline in mind to run efficiently on a different pipeline.

X-----------------------------------X---------------------------------X-------------------------------X-----------------------X

What is loop unrolling and how does it improve the performance of pipelined processors? What other techniques used to augment loop unrolling, in order to reduce pipeline stalls?

Loop unrolling is a compiler optimization applied to certain kinds of loops to reduce the frequency of branches and loop maintenance instructions. It is easily applied to sequential array processing loops where the number of iterations is known prior to execution of the loop. A simple scheme for increasing the number of instructions relative to the branch and overhead instructions is loop unrolling. Unrolling simply replicates the loop body multiple times, adjusting the loop termination code. Loop unrolling can also be used to improve scheduling. Because it eliminates the branch, it allows instructions from different iterations to be scheduled together.

X-----------------------------------X---------------------------------X-------------------------------X-----------------------X

The problems observed in Scoreboard technique of dynamic pipeline scheduling include the lack of internal forwarding and name dependences. Does Tomasulo's approach resolve these issues? How?

The primary difference is that Tomasulo’s algorithm handles antidependences and output dependences by effectively renaming the registers dynamically. In Tomasulo’s scheme, register renaming is provided by reservation stations, which buffer the operands of instructions waiting to issue and are associated with the functional units. The basic idea is that a reservation station fetches and buffers an operand as soon as it is available, eliminating the need to get the operand from a register. This bypassing is done with a common result bus that allows all units waiting for an operand to be loaded simultaneously (on the 360/91, this is called the common data bus, or CDB). In pipelines that issue multiple instructions per clock and also have multiple execution units, more than one result bus will be needed.

X-----------------------------------X---------------------------------X-------------------------------X-----------------------X