Very short answer questions. You must use 10 or fewer words. "True" and "False" are considered very short answers.

[1] Which is on average more effective, dynamic or static branch prediction?

[1] Is the peak performance of a processor a good indicator of what the observed performance will be?

[1] Predicting the direction of a branch is not enough. What else is necessary?

[1] Using a different mapping scheme will reduce which type of cache miss?

[1] Which type of cache miss may increase if using longer lines?

[1] Which type of cache miss may be increased by using shorter lines?

[1] Write down the average memory access time equation.

[1] What pipeline hazard can be avoided by "throwing money at the problem"?

[1] What pipeline hazard can be avoided using a technique known as value prediction?

[1] When we talk about the number of operands in an instruction (a 1-operand or a 2-operand instruction, for example), what do we mean?

[1] If we simply make transistors and wires smaller and smaller but make no other changes to the design, what happens to the power density?

[1] There are two main ways to define performance - what are they?

[1] What is the primary difference between Scoreboarding and Tomasulo's algorithm?

[1] What is the main difference between a commodity cluster and a custom cluster?

[1] Superscalar and VLIW are 2 techniques for increasing ILP. What is the primary difference between them?

[1] Give 1 advantage to using a VLIW.

[1] Give 1 disadvantage to using a VLIW.

[1] Give 1 advantage to using a superscalar. Your answer must be different than your VLIW answer.

[1] Give 1 disadvantage to using a superscalar. Your answer must be different than your VLIW answer.

[1] What principle makes Virtual Memory possible?

[2] MIMD parallel processors are very flexible and cost-effective. However, there are two major challenges to obtaining a substantial decrease in response time when using this approach. What are they?

[2] Area on the die used to be the most critical design constraint. That is no longer true - what is the most critical factor now? Why?

Short Answers (20 or fewer words)

[2] Writing to a cache is inherently slower than reading from a cache. Why?

[2] The power consumed by a chip has increased over time, but the clock rate has increased at a far greater relative rate. How was this possible? Explain how designers keep the chip from melting.

[2] Why is it so difficult for the processing elements on a CMOS-based chip to communicate with things that located off the chip?

[2] Does the design of the instruction set have a significant impact on the ability to pipeline a processor? If so, explain your answer. (Give an example)

[2] What is a benchmark program?

[2] Which is easier to write a program for, a shared memory machine or a message passing machine? Why?

[2] Which is more expensive to build - a shared memory machine, or a message passing machine? Why?

[2] Why do most pipelined machines avoid the use of condition codes?

[2] What does ROB stand for, and why is it used in modern advanced pipelines? (What necessary function does it help support?)

[1] What is the definition of a precise interrupt?

[1] Why is it important to support precise interrupts in modern pipelined processors?

[2] Processors have been built that were able to issue 8 instructions at a time using a fast clock. However, these processors are no longer being built - why not? Why would you choose a 3-issue machine over an 8-issue machine, if the clock rates were the same?

[3] What is the definition of a basic block? Why is there a desire to create a bigger one? Give one example of a way to create a bigger basic block.

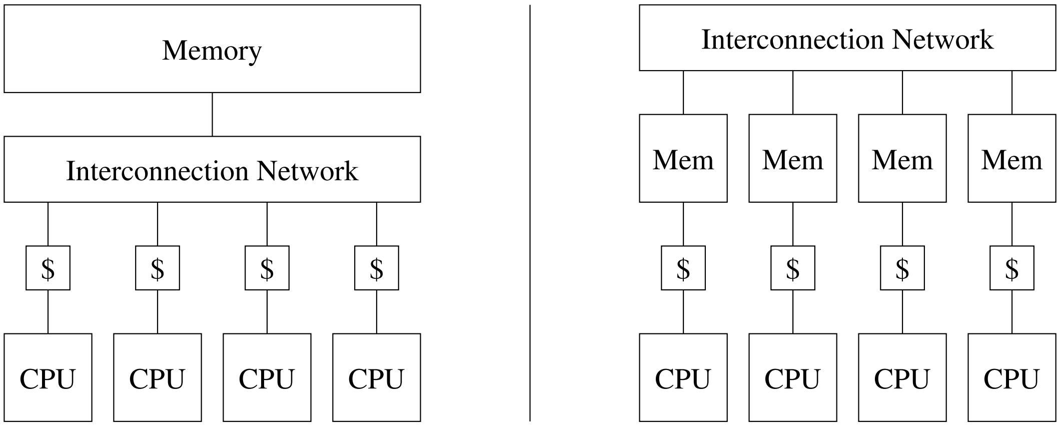
[2] Slow and wide architectures can be more power efficient than fast and narrow architectures. Explain why. Also, explain the underlying assumption that is being made, and why it is that we are still making narrow fast machines.

[2] What is "leakage" current? If Vdd is lowered, what happens to the amount of leakage current?

[2] An important program spends 20% of its time doing memory operations (loads and stores). By redesigning the memory hierarchy you can make the memory operations 90% faster (take 10% as long), or you can redesign the hardware to make the rest of the machine 20% faster (take 80% as long). Which should you do and why? (You must show your work to get full credit.)

[4] You are responsible for designing a new embedded processor, and for a variety of reasons you must use a fixed 24 bit instruction size and you must support at least 64 different opcodes. You would like to use a 3-operand instruction format, and have 128 registers. If it is possible to do this, draw what an instruction would look like. If it is not possible explain why, and give at least 2 different ways to solve the problem.

[2] Look at the figure below, then write below each machine if it is more likely to be a message passing or a shared memory design.



Design Approach A

Design Approach B

[10] For each technique in the following table, indicate how (on average) the 3 terms of the AMAT equation are affected (assume there is a single L1 cache). The first one is done as an example.
(Decreases = "-" or "V"; Increases = "+" or "/\"; leave blank if unaffected).

Technique	Hit Time	Miss Rate	Miss Penalty
Increasing Line/Block Size		**-**	**+**
Increasing Associativity			
Decreasing Cache Size			
Hardware Prefetching			
Compiler Optimizations (excluding prefetch)			
Non-blocking Cache			
Virtually Addressed Cache			
Victim Cache (victim cache and regular cache accessed in parallel)			

[12] You have been writing C programs for a simple, non-pipelined machine. You have recently received a promotion, and now your job is to write C programs for a heavily pipelined, high performance processor. This processor has an advanced tournament-style branch predictor, and a 2 billion entry return address stack. Your code will be compiled using a highly optimizing compiler, on the -O3 setting. Your programs must execute as fast as possible (the emphasis is on response time, not throughput).

a) Give an example of how you will change the way you write your C program. Explain in detail why you decided to make the change (what is the problem you are overcoming?)

b) Give another example of how you will change the way you write your C program. Explain in detail why you decided to make the change (what is the problem you are overcoming?)

c) Which of your two changes is likely to make the biggest difference in performance, and why?

[7] The MIPS implementation we used in class has a 5-stage pipeline, writes to the register file during the first half of the cycle and reads during the second half, and uses both a branch delay slot and a load delay slot. If the machine is redesigned to be an 8-stage pipeline, with the following stages:

F D1 D2 E1 E2 M1 M2 WB

a) Assuming this machine has a branch predictor and the branch condition is calculated by the end of the D2 stage, how big is the branch penalty (measured in cycles) when the prediction is incorrect? What if the branch condition is not calculated until the end of E2?

b) How many load delay slots would this machine need (assuming it has forwarding logic and you are forwarding to E1) assuming the memory returns the value by the end of M2? M1?

c) What type of data hazard does the above pipeline need to worry about?

d) If the above pipeline were modified to support out of order completion, what new data hazard would be introduced?

e) If in addition to completing out of order, instructions were allowed to issue out of order, what new data hazard would be introduced?

---

[3] In your first design of a 5-stage pipeline (F,D,E,M,W) F takes 23 time units, D takes 26, E takes 27, M takes 48 and W takes 26.

a) What will the clock cycle time be for this pipeline?

b) Is it a balanced pipeline? If not, explain what you could do to make it more balanced. What would the cycle time be now?

[2] Assuming a 25-bit address and a 1k-byte Direct Mapped cache with a linesize=4, show how an address is partitioned/interpreted by the cache.

[3] Assuming a 25-bit address and a 160-byte 5-way SA cache with a linesize=8, show how an address is partitioned/interpreted by the cache.

[2] Assuming a 20-bit address and a 536-byte FA cache with a linesize=2, show how an address is partitioned/interpreted by the cache.

[2] Given a 4 Megabyte physical memory, a 31 bit Virtual address, and a page size of 1K bytes, write down the number of entries in the Page Table, and the width of each entry.

[4] Given a 1 Gigabyte physical memory, a 44 bit Virtual address, and a page size of 2K bytes, write down the number of entries in the Page Table, and the width of each entry. Is there a problem with this configuration? If so, how can you fix it?

[8] Here is a code sequence.

```
store   R4, 200(R7)

and     R5, R7, R4

load    R2, 100(R5)

add     R3, R2, R6

sub     R1, R7, R1

add     R1, R4, R7
```

Assuming a standard 5-stage pipeline that does not support hazard detection and does no forwarding,

a) Indicate all dependencies (draw lines/arrows between them, and write beside each line/arrow which hazard is involved).

b) Insert as many No Operations (NOPS) as required in order to ensure this code runs correctly. (Remember, writes to the register file occur on the first half of the cycle, and reads occur during the second half).

c) Circle the NOPs that can be removed if forwarding and hazard detection logic is implemented.

d) There are things the static scheduler does not know which makes guaranteeing correctness difficult. What if, at execution time, R5=400 and R7=300? Will your dependency graph and ability to schedule this code change? If so, explain/show how.

[10] Here is a code sequence.

```
add     R1, R2, R3

sub     R1, R5, R1

lw      R4, 0(R7)

xor     R11, R12, R13

add     R6, R6, R4

and     R4, R3, R5
```

Assuming a 7-stage pipeline (F,D,E1,E2,M1,M2,WB) that does not support hazard detection, does no forwarding, and does not use a branch delay slot,

a) Indicate all dependencies (draw lines/arrows between them, and write beside each line/arrow which hazard is involved).

b) Insert as many No Operations (NOPS) as required in order to ensure this code runs correctly. (Remember, writes to the register file occur on the first half of the cycle, and reads occur during the second half).

c) Circle the NOPs that can be removed if forwarding and hazard detection logic is implemented. Assume data on a read returns at the end of M2, and is needed at the beginning of E1. Also assume that on an arithmetic instruction, data is ready at the end of E2, but not at the end of E1.

[8] Assume our machine has 8 logical and 16 physical registers. On the left below is the code you are dealing with. On the right below is the register mapping upon entering the code sequence.

```
lw      R4, 0(R7)

sub     R6, R6, R4

and     R4, R3, R5
```

BEFORE	
Logical	Physical
0	1
1	2
2	3
3	4
4	5
5	6
6	7
7	8

Free Pool: 9,10,11,12,13,14,15,0

(a) Indicate all the dependencies on the code segment above.

(b) Rewrite the code sequence below using the actual physical register names instead of the logical ones.

```
lw      P___, 0(P___)

sub     P___, P___, P___

and     P___, P___, P___
```

(c) Now indicate all the dependencies in the above renamed code.

[5] Here is a code sequence. This is (obviously) generic code.

```
       INSTRUCTION 1

       NOP

       INSTRUCTION 2

       INSTRUCTION 3

       INSTRUCTION 4

Label: INSTRUCTION 5

       NOP

       NOP

       INSTRUCTION 6

       INSTRUCTION 7

       INSTRUCTION 8

       breq   CONDITION,Label        ; branch to Label if CONDITION
```

Assume there are the following dependencies in this code:

RAW between "INSTRUCTION 1" and "INSTRUCTION 2"
RAW between "INSTRUCTION 5" and "INSTRUCTION 6"
RAW between "INSTRUCTION 7" and "breq"

WAW between "INSTRUCTION 1" and "INSTRUCTION 2"
WAW between "INSTRUCTION 5" and "INSTRUCTION 7"

WAR between "INSTRUCTION 6" and "INSTRUCTION 7"

Draw in the arrows, and then indicate how you would schedule the code to remove the maximum number of stalls. How many are left when you are done?

[6] Suppose I have a 3-issue multithreaded machine, and there are 3 threads - A, B, and C. Assume:

The number of independent instructions Thread A can find (in order): 1, then 3, then 0
The number of independent instructions Thread B can find (in order): 3, then 0, then 3
The number of independent instructions Thread C can find (in order): 2, then 1, then 1

Fill in the following table assuming fine-grained scheduling is being used.

Time	**Slot1**	**Slot2**	**Slot3**
0			
1			
2			
3			

Fill in the following table assuming the use of course-grained scheduling.

Time	**Slot 1**	**Slot2**	**Slot3**
0			
1			
2			
3			

Now, repeat the process assuming the use of simultaneous multithreading on a 5-issue multithreaded machine with 4 threads - A, B, C and D. Assume:

The number of independent instructions Thread A can find (in order): 1, then 2, then 0, then 1
The number of independent instructions Thread B can find (in order): 3, then 2, then 1, then 2
The number of independent instructions Thread C can find (in order): 0, then 1, then 3, then 1
The number of independent instructions Thread D can find (in order): 1, then 0, then 1, then 1

Time	**Slot1**	**Slot2**	**Slot3**	**Slot4**	**Slot 5**
0					
1					
2					
3					

---

[2] Processor A requires 250 instructions to execute a given program, uses 4 cycles per instruction, and has a cycle time of 6 ns. Processor B requires 2 cycles per instruction, and requires 300 instructions to do the same program. What must the cycle time of Processor B be in order to give the same CPU time as Processor A?

[5] In class, we talked about the cycle by cycle steps that occur on different interrupts. For example, here is what happens if there is an illegal operand interrupt generated by instruction i (note this machine has a 7 stage pipeline and supports imprecise interrupts. RR stands for Register Read):

	1	2	3	4	5	6	7	8	9	10	11	12
i	IF	ID	RR	EX	M1	M2	WB	<- Interrupt detected				
i+1		IF	ID	RR	EX	M1	M2	WB	<- Instruction Squashed			
i+2			IF	ID	RR	EX	M1	M2	WB	<- Trap Handler fetched		
i+3				IF	ID	RR	EX	M1	M2	WB		
i+4					IF	ID	RR	EX	M1	M2	WB	

Fill out the following table if for the same machine, instruction i+1 experiences a fault in the M1 stage (page fault, for example):

	1	2	3	4	5	6	7	8	9	10				
i	IF	ID	RR	EX	M1	M2	WB							
i+1		IF	ID	RR	EX	M1	M2	WB						
i+2			IF	ID	RR	EX	M1	M2	WB					
i+3				IF	ID	RR	EX	M1	M2	WB				
i+4					IF	ID	RR	EX	M1	M2	WB			
i+5						IF	ID	RR	EX	M1	M2	WB		
i+6							IF	ID	RR	EX	M1	M2	WB	
i+7								IF	ID	RR	EX	M1	M2	WB

Assuming precise interrupts are being supported, what happens in this case?

	1	2	3	4	5	6	7	8	9	10				
i	IF	ID	RR	EX	M1	M2	WB	<- EX stage has page fault						
i+1		IF	ID	RR	EX	M1	M2	WB	<- Inst Decode has Illegal Instruction					
i+2			IF	ID	RR	EX	M1	M2	WB					
i+3				IF	ID	RR	EX	M1	M2	WB	<- IF stage has page fault			
i+4					IF	ID	RR	EX	M1	M2	WB			
i+5						IF	ID	RR	EX	M1	M2	WB		
i+6							IF	ID	RR	EX	M1	M2	WB	
i+7								IF	ID	RR	EX	M1	M2	WB

What is the maximum number of exceptions that could happen at a single time in the above machine (assuming no hardware errors)? Explain how you got your answer.