Very short answer questions. "True" and "False" are considered very short answers. (2015)

[1] Does an average program's locality behavior stay the same while it is executing, or does it vary?

[1] Does peak performance track observed performance?

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

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

[1] What is a balanced pipeline?

[2] How do two processes communicate when running on a shared memory machine?

[2] Clock rates have grown by a factor of 1000 while power consumed has only grown by a factor of 30. How was this accomplished, and why did it work?

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

[2] Does reducing the minimum feature size affect power density? If so, in what way?

[2] Do benchmarks remain valid indefinitely? Why or why not?

[2] Why are there multiple dies per silicon wafer? Why not just fabricate one huge die per wafer?

[2] Prefetching is one technique for reducing the Miss Rate. List 2 others.

[2] Lowering the associativity is one technique for reducing the Hit Time. List 2 others.

[2] A merging write buffer is one technique for reducing the Miss Penalty. List 2 others.

[2] What is the relationship between speculation and power consumption?

[2] Using less than 20 words, explain the difference between coherence and consistency.

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

[2] Superscalar and VLIW are 2 ways to increasing ILP. What is the primary difference between them?

[2] What two things are keeping parallel processors from being the dominant architecture? (In other words, what are the two biggest challenges in parallel processing?)

[3] What is the goal of the memory heirarchy, as stated in class? What two principles make it work? (Two types of the same principle, actually).

Short Answers:

[3] Does the instruction set architecture have a significant impact on the ability to pipeline a processor? If so, explain your answer.

[2] What is a benchmark program? ." What is the **perfect** benchmark?

[3] What makes one instruction set harder to write a virtual machine monitor for than another one?

[3] Why do CMOS processors avoid accessing items that are located off-chip?

[3] What are the two main ways to define performance? How do they differ? Give an example task for each.

[3] Why is it difficult to come up with good benchmarks for parallel processors?

[8] What do the following acronyms stand for?

SMT	**MIMD**	**UMA**
WAW	**DSM**	**ILP**
VMM	**VLIW**	

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

[2] Why is there a desire to create larger basic blocks? Give one example of a way to create a bigger basic block.

[3] Supporting precise interrupts in machines that allow out of order completion is a challenge. Briefly explain why.

[4] Processor clock rates have stopped increasing, even though feature sizes have continued to decrease. Explain why.

[3] List 2 advantages and 1 disadvantage of Superscalar processors.

[4] Why is branch prediction important? List 3 dynamic Branch Prediction strategies in order of (average) increasing effectiveness.

[3] Snooping is one approach to providing coherence - state what the other main approach is, and briefly outline how each of them work.

[3] What is it about the memory system that makes it hard for compilers to optimize code, and also for execution units to achieve maximal performance? (This is not a question about technology, it's a higher-level question).

[4] 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. These new programs must execute as fast as possible (the emphasis is on response time, not throughput). Give at least 2 examples of things you should do differently now, and be sure to explain in detail why (what is the problem you are overcoming?) (Note - make sure you are describing things that you can do in a high level language, not things that are done at the assembly language level.)

[3] Assuming a 16-bit address and a 128-byte Direct Mapped cache with a linesize=8, show how an address is partitioned/interpreted by the cache.

[3] Assuming a 16-bit address and a 160-byte 5-way SA cache with a linesize=4, show how an address is partitioned/interpreted by the cache.

[2] Given a 1 Megabyte physical memory, a 22 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.

[4] Given a 1 Megabyte physical memory, a 32 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?

[17] Here is a code sequence.

```
load    R1, 0(R10)

load    R2, 4(R1)

add     R3, R2, R1

store   R3, 20(R9)

sub     R3,R7,R8

load    R11, 4(R6)

load    R5, 8(R10)

add     R6, R5, R4
```

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) Assuming the pipeline has been modified so that it does support hazard detection and forwarding, schedule the code to remove as many stalls as possible. How many NOPS are left?

[3] Find the basic blocks in this code (if there are any), and circle them.

```
            lw      R1, 4(R10)
            lw      R2, 0(R10)
label1:     lw      R3, 8(R1)
            add     R4, R3, R2
            sw      R4, 4(R10)
label2:     sw      R4, 20(R4)
            bnez    R4, label1
label3:     lw      R1, 4(R0)
            add     R5, R1, R4
            add     R6, R1, R5
            beq     R5, R6, label2
```

[3] An important program spends 70% of its time doing Integer operations, and 30% of its time doing floating point arithmetic. By redesigning the hardware you can either make the Floating Point unit 100 times faster or the integer unit twice as fast. Which should you do, and why?

[4] You are responsible for designing a new embedded processor, and for a variety of reasons you must use a fixed 15 bit instruction size. You would like to support 32 different operations, use a 3-operand instruction format, and have 16 registers. If it is possible to do this, draw what an instruction would look like. If it is not possible, explain why, and show what you would do to fix the problem.

[7] The standard MIPS has a 5-stage pipeline, and uses a load and a branch delay slot. If the machine is redesigned to be an 8-stage pipeline, with the following stages:

F D RR E1 E2 M1 M2 WB (where **RR** stands for Register Read)

a) Assuming the machine has bypass/forwarding logic, how many load delay slots will this new design require if the memory returns the value at the end of M2? At the end of M1?

b) What is the branch penalty in cycles, assuming there is no branch delay slot and the branch condition is calculated and available at the end of E1? At the end of E2?

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?

[6] In an MOS device, there is a gate, drain, and source. Briefly explain how this device works. Pictures are welcome.

[6] 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+1 (Note this machine has a 6-stage pipeline):

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

Fill out the following table if instruction i experiences a page fault in the MEM stage (assume precise interrupts are not supported):

	1	2	3	4	5	6	7	8	9	10	11
i	IF	ID	RR	EX	MEM	WB					
i+1		IF	ID	RR	EX	MEM	WB				
i+2			IF	ID	RR	EX	MEM	WB			
i+3				IF	ID	RR	EX	MEM	WB		
i+4					IF	ID	RR	EX	MEM	WB	
i+5						IF	ID	RR	EX	MEM	WB

What happens in this case?

	1	2	3	4	5	6	7	8	9	10	11	
i	IF	ID	RR	EX	MEM	WB	<- Data write causes Page Fault					
i+1		IF	ID	RR	EX	MEM	WB	<- Instruction Fetch causes Page Fault				
i+2			IF	ID	RR	EX	MEM	WB				
i+3				IF	ID	RR	EX	MEM	WB			
i+4					IF	ID	RR	EX	MEM	WB		
i+5						IF	ID	RR	EX	MEM	WB	
i+6							IF	ID	RR	EX	MEM	WB

[2] What is the maximum number of exceptions that could happen at one time in the above machine? Why?