Very Short Answer:

(1) (1) Do benchmarks remain valid indefinitely?

(2) (2) Issuing multiple instructions per cycle puts tremendous pressure on what two parts of the machine?

(3) (2) What is a "poison" bit?

(4) (1) Out of Order completion makes supporting what very difficult?

(5) (1) Are wire delays or transistors more likely to be the most significant limit on clock frequency in the future?

(6) (1) Is MIPS an accurate measure for comparing performance among computers?

(7) (2) Standard decoupled architectures split a program into two streams. What are they?

Short Answers:

(10) (4) Most silicon dies are fairly small. Why are they not bigger?

(11) (4) Describe what the Geometric Mean is, and the biggest drawback to using it.

(12) (4) Write down the 3-term CPU performance equation developed in class.

(13) (4) What are the advantages and disadvantages of stack machines?

(14) (4)What is a predicated instruction? What are the advantages to using predicated instructions? When would you *not* want to use one?

(15) (4) What is the definition of a basic block? Why is there a desire to create larger ones?

(16) (8) Compare and contrast Superscalar with VLIW. Describe each, and list the advantages and disadvantages of each approach.

(17) (6) Why is branch prediction important? What performance enhancing techniques have made it so? List 3 examples of existing Branch Prediction strategies in order of their (average) increasing effectiveness.

(18) (12) On the following page is the table used to describe scoreboarding. The contents of the table represent the state of the scoreboard at time 5. Your job is to fill in the table at time 8. (Assume at time 8 the 2nd LF completes.)

Time = 5

Instruction Status					
Instruction		Issue Inst	Read Operands	Execute Inst	Write Result
LF	F6,34(R2)	x	x	x	x
LF	F2,34(R3)	x	**X**		
MULTF	F0,F2,F4	**X**			
SUBF	F8,F6,F2				
DIVF	F10,F0,F6				
ADDF	F6,F8,F2				

Functional Unit Status										
FU	Name	Busy	Fm	Fi	Fj	Fk	Qj	Qk	Rj	Rk
1	Int/LS	**Yes**	**Load**	**F2**	**R3**					
2	Mult1	**Yes**	**Mult**	**F0**	**F2**	**F4**	**1**		**No**	**Yes**
3	Mult2									
4	Add									
5	Div									

Register Result Status									
	F0	F2	F4	F6	F8	F10	12	...	F30
FU#	**2**	**1**							

Time = 8

Instruction Status					
Instruction		Issue Inst	Read Operands	Execute Inst	Write Result
LF	F6,34(R2)				
LF	F2,34(R3)				
MULTF	F0,F2,F4				
SUBF	F8,F6,F2				
DIVF	F10,F0,F6				
ADDF	F6,F8,F2				

Functional Unit Status										
FU	Name	Busy	Fm	Fi	Fj	Fk	Qj	Qk	Rj	Rk
1	Int/LS									
2	Mult1									
3	Mult2									
4	Add									
5	Div									

Register Result Status									
	F0	F2	F4	F6	F8	F10	12	...	F30
FU#									

(19) (14) You are given the following code sequence:

```
LF      F0,34(R2)
LF      F1, 45(R3)
MULTF   F2,F1,F4
SUB     F3,F0,F1
DIVF    F4,F2,F0
ADDF    F0,F3,F1
```

Assume there are 8 logical and 16 physical registers. On the left below is the register mapping upon entering the code sequence. Your job is to fill in the mappings after the execution of the ADDF instruction, including the free list. of

BEFORE	
Logical	Physical
0	2
1	3
2	5
3	4
4	9
5	7
6	6
7	8

Free Pool: 0,1,10,11,12,13,14,15

AFTER	
Logical	Physical
0	
1	
2	
3	
4	
5	
6	
7	

Free Pool:

Now, rewrite the code sequence below using the actual physical register names instead of the logical ones.

```
LF      P__,34(R2)
LF      P__,45(R3)
MULTF   P__,P__,P__
SUBF    P__,P__,P__
DIVF    P__,P__,P__
ADDF    P__,P__,P__
```

(20) (14) Suppose we are considering two alternatives for our conditional branch instructions, as follows:

M1 - A compare is included in the branch.
M2 - A condition code is set by a compare instruction and followed by a branch that tests the condition

On both machines, the conditional branch instruction takes 2 cycles, and all other instructions take 1 cycle. On M2, 25% of all instructions executed are conditional branches. Because M2 does not have the compare included in the branch, the clock cycle is 1.25 times faster than M1.

a. Which design is faster?

b. What if M2 was only 1.2 times faster than M1?

(21) (12) 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:

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

Fill out the following table if instruction i causes a page fault in the MEM stage:

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

Fill out the following table if instruction i+2 experiences a fault in the execution stage (divide by zero, for example):

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

What happens in this case?

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