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

[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?

[3] It is difficult for the internal processing elements on a chip to communicate with the outside world (things that are on other chips, for example). Explain why.

[2] What do we mean when we say something is an N-operand machine?

[3] What is a benchmark program? Do benchmark programs remain valid indefinitely? Why or why not?

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

[3] You have a 4-stage pipeline (F,D,E,W.) where F takes 20 time units, D and E take 25, and W takes 30. What will the clock rate be for this pipeline? Is it a balanced? If not, what would you try to do to fix it?

[2] Which data hazard occurs when instructions are allowed to complete out of order? Which one occurs when instructions are allowed to issue out of order?

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

[2] There are two approaches to prediction - static and dynamic. What is the difference? Which approach is on average more accurate?

[3] What is a precise interrupt? Why is supporting precise interrupts in machines that allow out of order completion is a challenge?

[1] One way to deal with control hazards is to predict if they are taken or not. What else is needed in addition to a prediction?

[2] Give a one-word definition of coherence, and a one-word definition of consistency.

[4] What is the primary difference between superscalar and VLIW processors? Give two advantages to using a superscalar processor, and 1 advantage to using a VLIW.

[2] Speculation is a very useful technique for improving performance. However, it is not being used as extensively as it once was - why not?

[2] What is the definition of a basic block? Why is there a desire to create a bigger one?

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

[2] Prefetching is one technique for reducing the Miss Rate. List 2 others. (You cannot reuse any from the previous question.)

[2] Giving reads priority over writes is one technique for reducing the cache miss penalty. List 2 others. (You cannot reuse any from the previous two questions.)

[2] What are the two biggest challenges to obtaining a substantial decrease in response time when using a MIMD parallel processor?

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

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

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

[2] Compilers have a hard time guaranteeing program correctness when doing static scheduling. Why is that? Give an example of a case where something might go wrong. (Hint - think Memory System)

[3] The designer has the choice of using a physically addressed cache or a virtually addressed cache. Explain the difference, and give 1 disadvantage for each.

[4] Understanding the hardware can influence how you write programs. Give 2 examples of things you might avoid using or things you would do differently if you are writing software for a heavily pipelined machine verses a non-pipelined one.

[10] 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 RR E M1 M2 WB (where **RR** stands for Register Read)

a. Assuming this machine has a branch predictor and the branch condition is calculated by the end of the E stage, how big is the branch penalty (measured in cycles) when the prediction is incorrect?

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

Is it possible to have a WAW hazard in the above 8-stage pipeline? Why or why not?

[4] You are responsible for designing a new processor, and for a variety of reasons you must use a fixed 16 bit instruction size. You would like to support 64 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 give at least 2 different ways to solve the problem.

[3] Assuming a 25-bit address and a 512-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 96-byte 3-way SA cache with a linesize=8, show how an address is partitioned/interpreted by the cache.

[2] Assuming a 25-bit address and a 230-byte FA cache with a linesize=2, show how an address is partitioned/interpreted by the cache.

[2] Given a 1 Megabyte physical memory, a 30 bit Virtual address, and a page size of 8K 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?

[10] Here is a code sequence.

```
sw      R1, 4(R10)

lw      R2, 0(R10)

lw      R3, 8(R1)

add     R4, R3, R2

sw      R1, 20(R4)

sw      R4, 24(R0)

lw      R1, 4(R0)

add     R5, R1, R4
```

a) Assuming a standard 5-stage pipeline that does not support hazard detection and does no forwarding, insert as many 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).

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

c) Reorder the code to remove as many stalls as possible (assuming there is forwarding logic). How many NOPs are left (if any)?

In this question, we are going to wire up a 12-bit processor. The machine is word-addressable, where a word is 12 bits. Immediates are sign extended, Offset is not. The machine has 3 different instruction formats: R, I, and J. Memory takes a single cycle to return a value.

R-type:	*(Arithmetic and logical:*				*rd = rs1 OP rs2)*
Opcode	rd	rs1	rs2	funct	
11-8	7-6	5-4	3-2	1-0	

I-type:	*(Arithmetic and logical:*			*rd = rs1 OP Immediate)*
	(Load:			*rd = mem[rs1+Immediate])*
	(Store:			*mem[rs1+Immediate]=rd)*
Opcode	rd	rs1	Immediate	
11-8	7-6	5-4	3-0	

J-type:	*(*	*PC = Offset)*
Opcode	Offset	
11-8	7-0	

The ALU can perform 4 functions, written this way: OP [ALU0 ALU1]
Add [10], AND [01], OR [11], NOT [00]

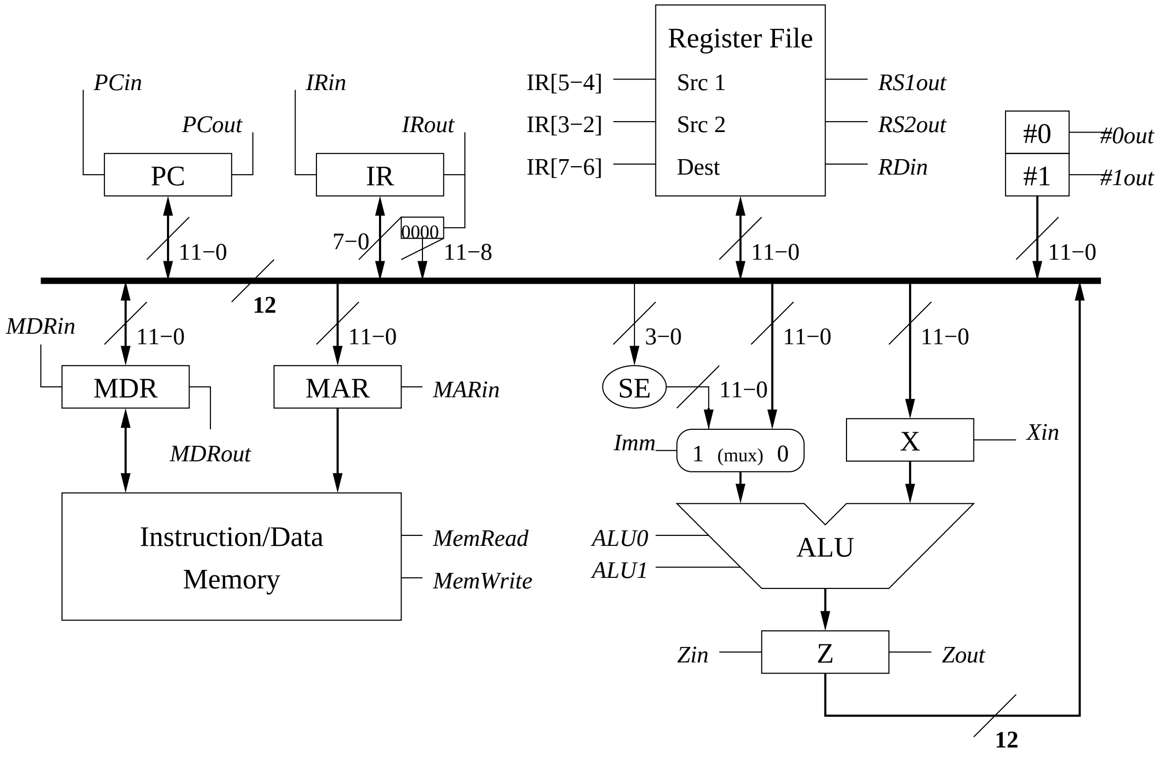
Here are some of the instructions that have been defined:

Name	**Opcode(Funct)**	**Name**	**Opcode(Funct)**	**Name**	**Opcode(Funct)**
NOP	0000(00)	lw	0001(xx)	sw	0011(xx)
NOT	1000(00)	beqz	0100(xx)	j	0101(xx)
AND	1000(01)	AND Imm	1001(xx)	ADD	1100(01)
OR	1000(10)	OR Imm	1010(xx)	ADD Imm	1101(xx)
XOR	1000(11)	XOR Imm	1011(xx)	SUB	1100(01)

Here are the 21 control signals.

PCin	PCout	IRin	IRout	MDRin	MDRout	MARin
Zin	Zout	RDin	RDout	MemRead	MmWrite	Imm
RS1out	RS2out	#0out	#1out	ALU0	ALU1	Xin

Here is a diagram of the machine.



[8] Fill in the microcode steps necessary to perform an instruction fetch (incrementing the PC is considered part of fetch).

Step	IRin	PCin	MARin	MDRin	RDin	Xin	Zin	IRout	PCout	MDRout	Zout	RDout	RS1out	RS2out	#0out	#1out	ALU0	ALU1	Mread	Mwrite	Imm
0																					
1																					
2																					
3																					
4																					
5																					

[3] Write down the binary bit pattern (the bit pattern that will be in the IR after you have done the instruction fetch) for the instruction: **OR Immd R0, R1, 2**

[6] Now that you have done the instruction fetch, fill in the microcode steps necessary to perform the following instruction: **OR Immd R0, R1, 2**

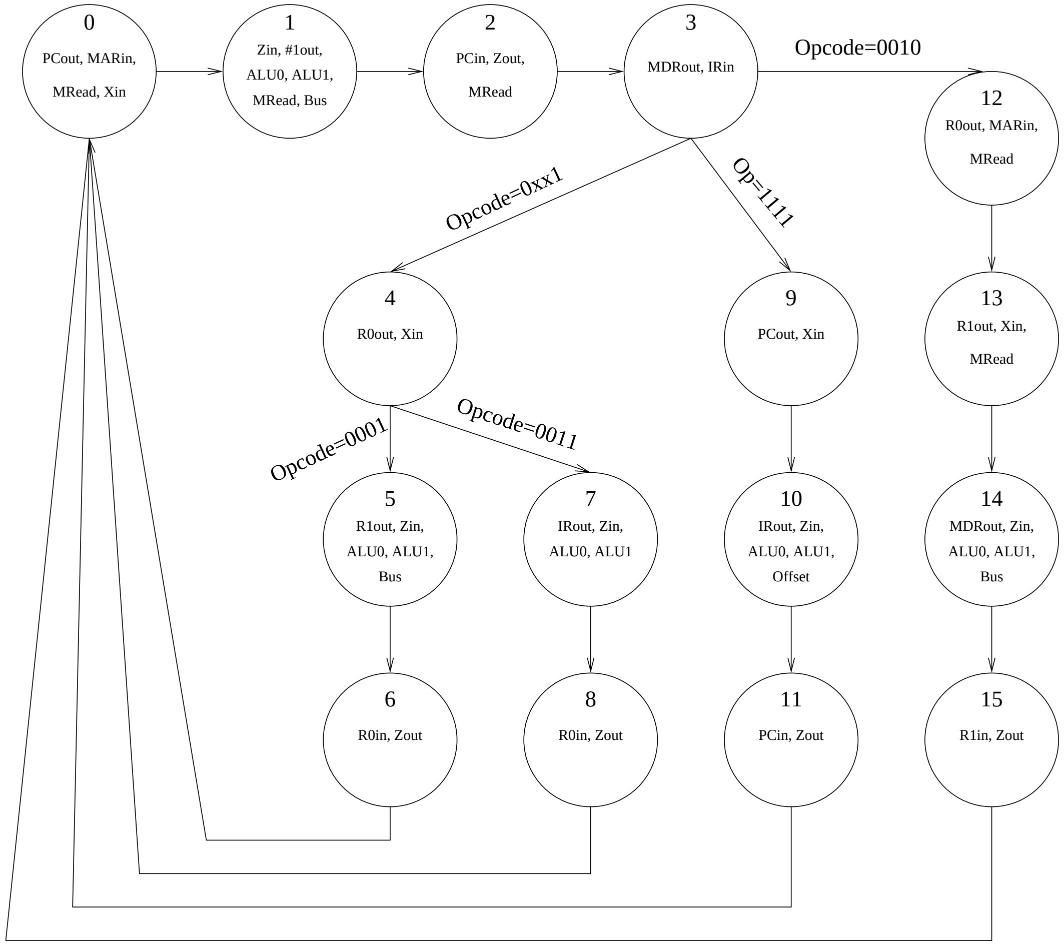
Step	IRin	PCin	MARin	MDRin	RDin	Xin	Zin	IRout	PCout	MDRout	Zout	RDout	RS1out	RS2out	#0out	#1out	ALU0	ALU1	Mread	Mwrite	Imm
0																					
1																					
2																					
3																					
4																					
5																					

[12] In the example below, assume the caches are write-back caches, and the coherence scheme uses an invalidation protocol on a snooping bus for a single cache block (X). Both caches are initially empty, and the value of X is 0. The entries in the table will contain the information after the processor and bus activity have both completed. A blank indicates no activity (or that nothing new was cached).

You are to fill out the table below with the missing values. The first entry has been done for you.

Processor Activity	Bus Activity (Coherence information)	Contents of CPU A's Cache	Contents of CPU B's Cache	Contents of Memory Location X
				0
CPU A Reads X	**Cache Miss for X**	**0**		**0**
CPU B Reads X				
CPU A Writes a 1 to X				
CPU B Reads X				

Here is the state diagram for a random machine:



[4] Assuming there are 4 state variables (Y3-Y0), that State0=!Y3*!Y2*!Y1*!Y0 (0000) and State15=Y3*Y2*Y1*Y0 (1111), write down the exact boolean equation for the **PCin** signal.

[2] Assuming the same situation as in the previous question, write down the exact boolean equation for **NextState7**.

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

The number of independent instructions Thread A can find (in order): 1, then 2, then 0, then 2
The number of independent instructions Thread B can find (in order): 3, then 0, then 2, then 3
The number of independent instructions Thread C can find (in order): 2, then 4, then 3, then 1

Fill in the following table if coarse grained scheduling is being used.

Time	Slot1	Slot2	Slot3	Slot4	Slot5	Slot 6
0						
1						
2						
3						

Now fill in the following table assuming the use of fine-grained scheduling.

Time	Slot1	Slot2	Slot3	Slot4	Slot5	Slot 6
0						
1						
2						
3						

Now, repeat the process assuming simultaneous multithreading is being used.

Time	Slot1	Slot2	Slot3	Slot4	Slot5	Slot 6
0						
1						
2						
3						

---

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

[1] Which type of cache miss can be reduced by using longer lines?

[1] Which type of cache miss can be reduced by using shorter lines?

[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 (note this machine has a 7 stage pipeline):

	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 instruction i+1 experiences a fault in the EX stage (Overflow, 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

What happens in this case?

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

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