Full Name: _____ NetID: _____

Midterm 10/17/2016

OAKLAND UNIVERSITY, School of Engineering and Computer Science
CSE 564: Computer Architecture, Fall 2016

Please write and/or mark your answers clearly and neatly; answers that are not readable are not considered. Please answer in appropriate details if needed and you may get partial points for the intermediate steps even if a final answer is not correct.

1. Name two great ideas in computer architecture and explain each with at most two sentences. (4 points)

Design for Moore's Law
Use abstraction to simplify design
Make the common case fast
Performance via parallelism
Performance via pipelining
Performance via prediction
Hierarchy of memories
Dependability via redundancy

2. Moore's Law: 1). What is Moore's Law about? 2). Why we have been used Moore's law to predict computer performance before multicore? 3). What limits the performance scaling based on Moore's law (6 points)

1). Moore's law: processor transistor density double every two years
2). The increased number of transistor had been effectively used for improving instruction level parallelism, thus the frequency of CPU double about every two years, thus we can use moore's law to predict performance
3). We are reaching the limit of ILP improvement and power/heating becomes an issue when we increase processor frequency.

3. Some microprocessors today are designed to have adjustable voltage, so a 25% reduction in voltage may result in a 20% reduction in frequency. What would be the impact on dynamic energy and on dynamic power? (8 points)

Answer Since the capacitance is unchanged, the answer for energy is the ratio of the voltages since the capacitance is unchanged:

$$\text{Energy}_{\text{new}} / \text{Energy}_{\text{old}} = (\text{Voltage} \cdot 0.75)^2 / \text{Voltage}^2 = 0.75^2 = 0.5625$$

thereby reducing energy to about 56.25% of the original. For power, we add the ratio of the frequencies

$$\text{Power}_{\text{new}} / \text{Power}_{\text{old}} = 0.5625 * (\text{Frequency switched} \cdot 0.80) / (\text{Frequency switched}) = 0.45$$

shrinking power to about 45% of the original.

4. Consider two different implementations of the same instruction set architecture. The instructions can be divided into four classes according to their CPI (class A, B, C, and D). P1 with a clock rate of 2.5 GHz and CPIs of 1, 2, 3, and 3, and P2 with a clock rate of 3 GHz and CPIs of 2, 2, 2, and 2. (10 points)

Given a program with a dynamic instruction count of 1.0E6 instructions divided into classes as follows: 10% class A, 20% class B, 50% class C, and 20% class D, which implementation is faster?

What is the global CPI for each implementation?

Class A: 10^5 instr. Class B: 2×10^5 instr. Class C: 5×10^5 instr. Class D: 2×10^5 instr.

Time = No. instr. × CPI/clock rate

Total time P1 = $(10^5 + 2 \times 10^5 \times 2 + 5 \times 10^5 \times 3 + 2 \times 10^5 \times 3)/(2.5 \times 10^9) = 10.4 \times 10^{-4}$ s

Total time P2 = $(10^5 \times 2 + 2 \times 10^5 \times 2 + 5 \times 10^5 \times 2 + 2 \times 10^5 \times 2)/(3 \times 10^9) = 6.66 \times 10^{-4}$ s

CPI(P1) = $10.4 \times 10^{-4} \times 2.5 \times 10^9/10^6 = 2.6$

CPI(P2) = $6.66 \times 10^{-4} \times 3 \times 10^9/10^6 = 2.0$

5. What are the basic five stages of the RISC pipeline, and explain the functions of each stage (3 points)

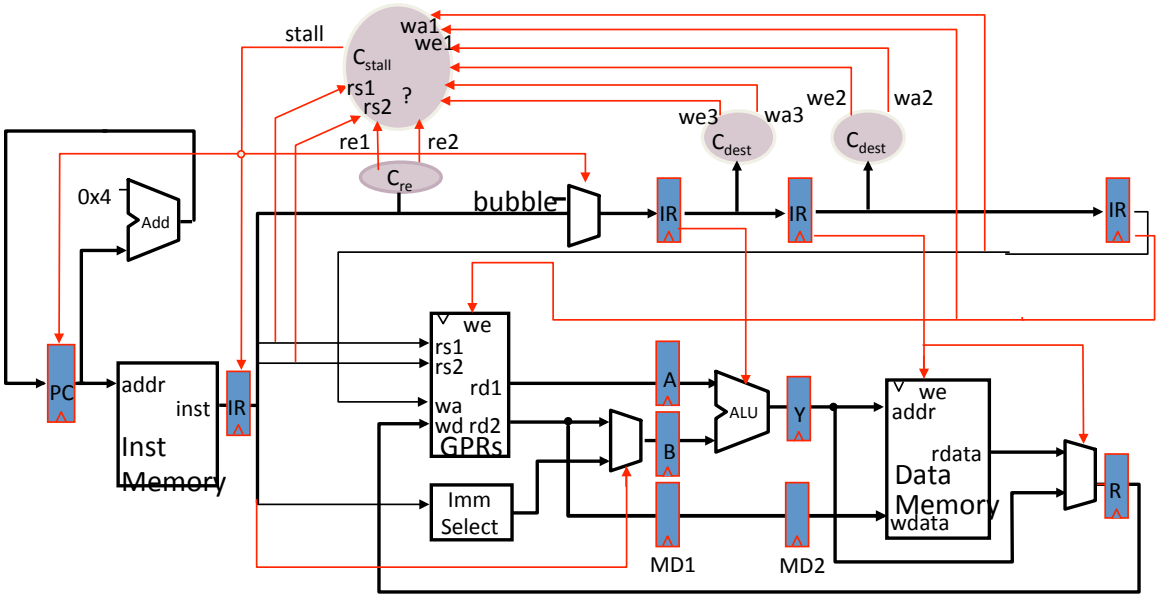
IF, ID, EXE, MEM, WB

6. Pipeline CPU implementation normally have separate instruction cache and data cache, what would be the problem if a pipeline implementation uses single-port unified cache? (3 points)

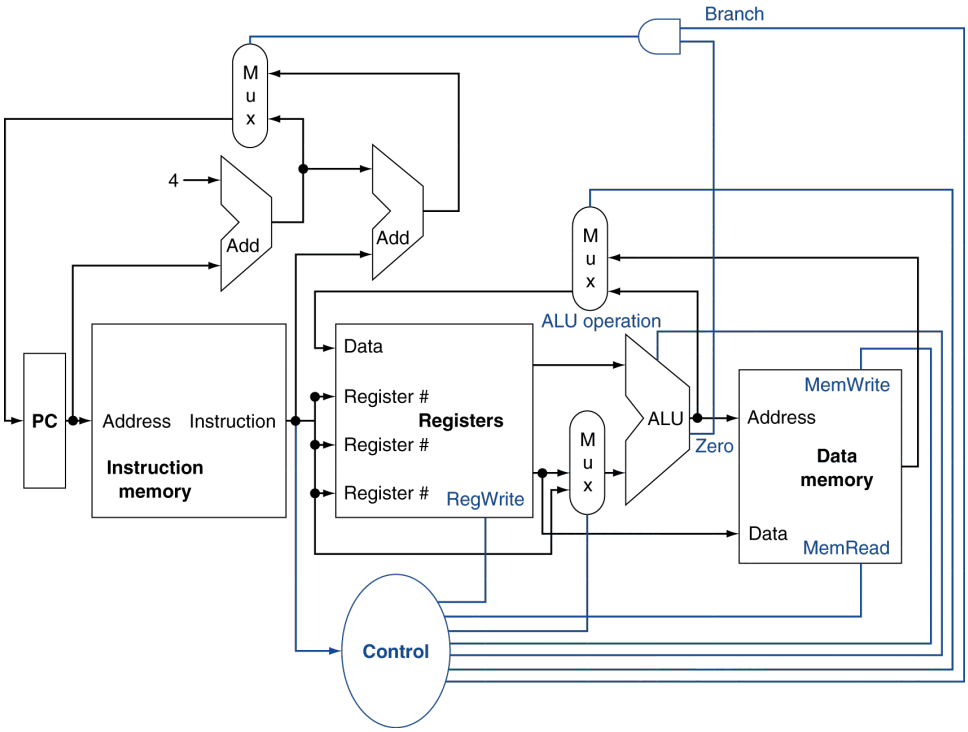
We will have structural hazards because of accessing the same cache in the IF and WB stage by two instructions

7. Pipeline registers are registers that are invisible to programmers (ISA) to stage input and output operands/results between pipeline stages, for example, register A and B in the figure below. We however also need multiple IR registers to store the instructions between pipeline stages. Explain why? (3 points)

Each stage of the pipeline is dealing with different instructions, thus we need to keep the instruction info for each stage of the pipeline

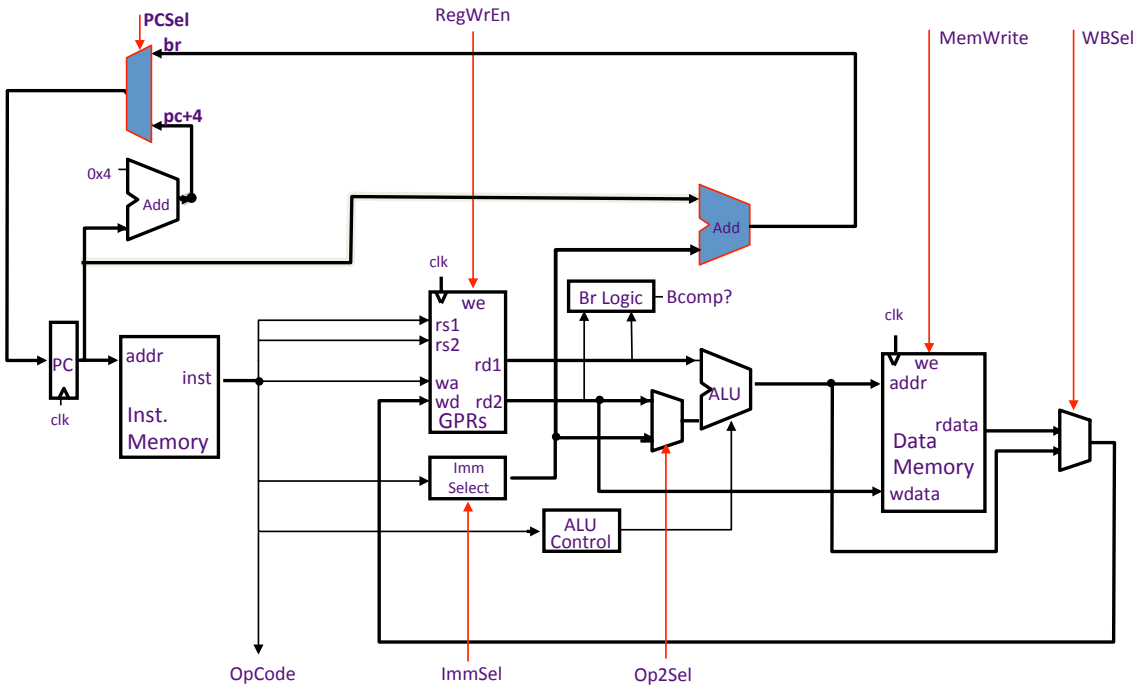


8. Control path are signals used to direct the correct input/output data path in pipeline execution. In the following figure, there are multiple control signals. Which instruction(s) would trigger MemWrite control? Which instruction(s) would trigger MemRead control? Which instruction(s) would trigger RegWrite control? (7 points)

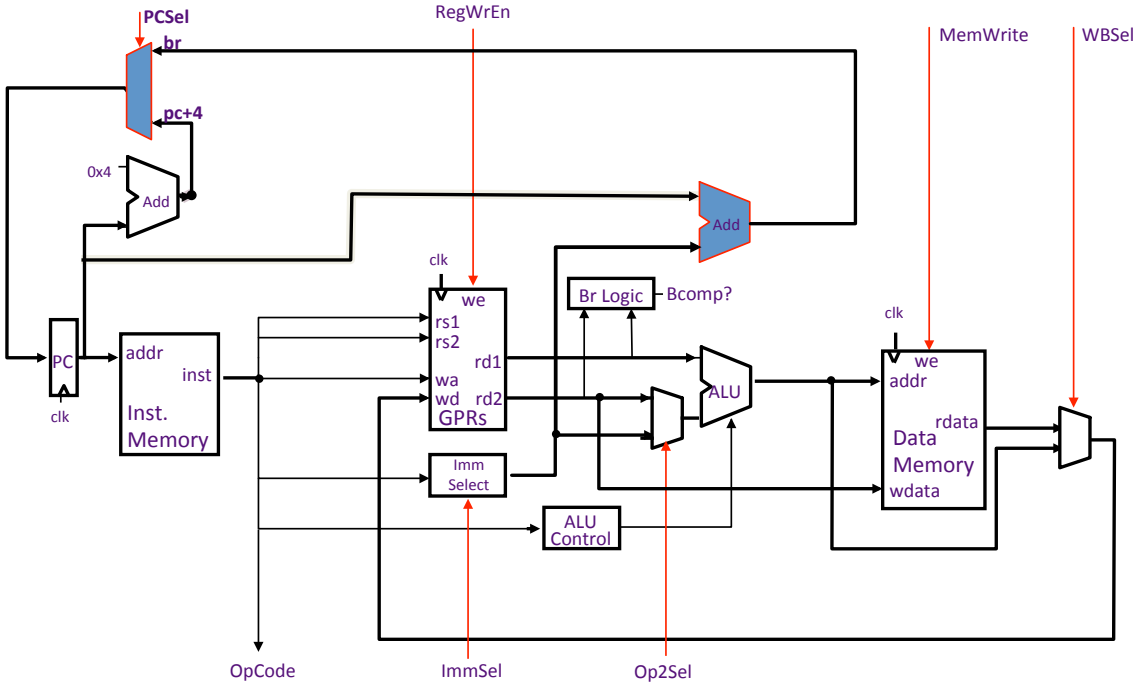


MemWrite: St
MemRead: ld
RegWrite: ALU, LD

9. Highlight the datapath used in the following diagram when “add x4 x5 x6” is being executed in the single-cycle RISC-V implementation. (6 points)



10. Highlight the datapath used in the following diagram when “ld x4 32(x5)” is being executed in the single-cycle RISC-V implementation. (6 points)



11. In the following RISC-V instruction sequence, draw the 5-stage pipeline execution figure and calculate the number of cycles to execute then in the following configuration (register files can be read/write in the same cycle): (12 points)

    a) full interlocking, i.e. no bypassing
    b) bypassing only between EXE-EXE stage
    c) bypassing between EXE-EXE and MEM-EXE stage
    d) full bypassing

1. ld x5 -32(x4)
2. ld x6 -16(x4)
3. add x6 x5 x6
4. add x6 x6 x6

a) 12 cycles, b) 10 cycles, c) 9 cycles, and d) 9 cycles

a)	1	2	3	4	5	6	7	8	9	10	11	12
ld x5 -32(x4)	IF	ID	EXE	MEM	WB							
ld x6 -16(x4)		IF	ID	EXE	MEM	WB						
add x6 x5 x6			IF	-	-	ID	EXE	MEM	WB			
add x6 x6 x6				IF	-	-	-	-	ID	EXE	MEM	WB

b)	1	2	3	4	5	6	7	8	9	10	11	12
ld x5 -32(x4)	IF	ID	EXE	MEM	WB							
ld x6 -16(x4)		IF	ID	EXE	MEM	WB						
add x6 x5 x6			IF	-	-	ID	EXE	MEM	WB			
add x6 x6 x6				IF	-	-	ID	EXE	MEM	WB		

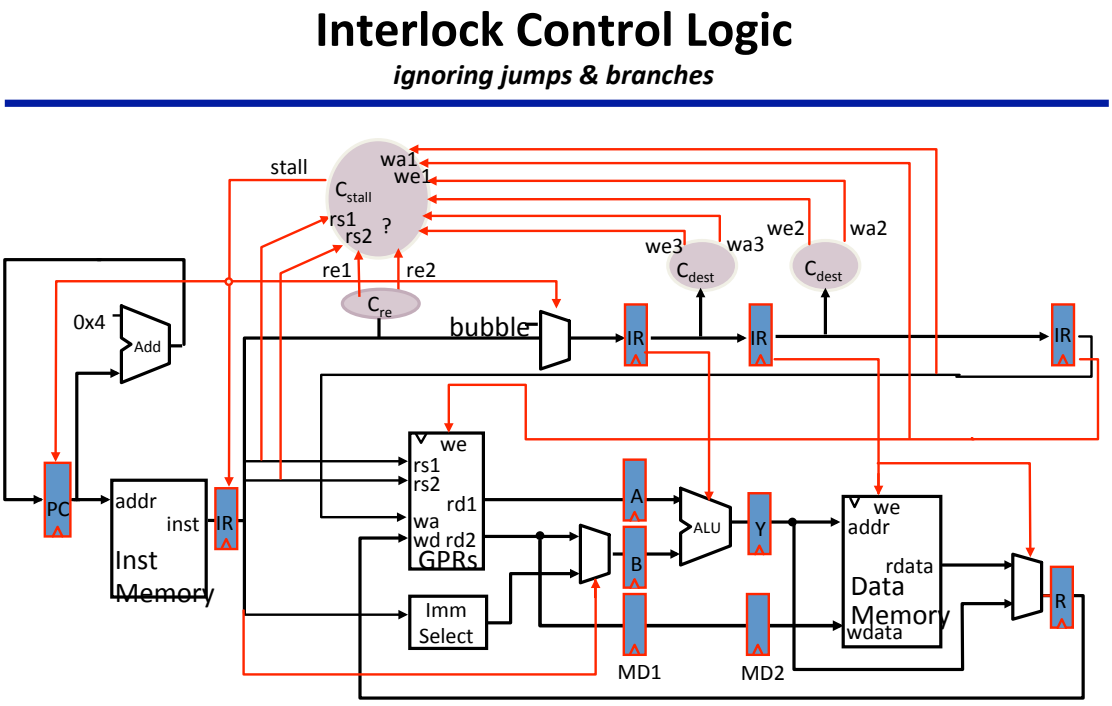
c)	1	2	3	4	5	6	7	8	9	10	11	12
ld x5 -32(x4)	IF	ID	EXE	MEM	WB							
ld x6 -16(x4)		IF	ID	EXE	MEM	WB						
add x6 x5 x6			IF	-	ID	EXE	MEM	WB				
add x6 x6 x6				IF	-	ID	EXE	MEM	WB			

d)	1	2	3	4	5	6	7	8	9	10	11	12
ld x5 -32(x4)	IF	ID	EXE	MEM	WB							
ld x6 -16(x4)		IF	ID	EXE	MEM	WB						
add x6 x5 x6			IF	-	ID	EXE	MEM	WB				
add x6 x6 x6				IF	-	ID	EXE	MEM	WB			

12. Interlock are control logic for data hazard detection, in the following diagram, the pseudo-code for RAW hazard detection between two instructions in ID/RF and EXE stages can be written as follows (x0 is not used): (6 points)

If ((rs1 == wa3 && re1 == true && we3 == true) || (rs2 == wa3 && re2 == true && we3 == true)) insert bubble for stall the instruction in ID/RF stage.

Write the similar pseudo-code for detecting the RAW hazard between two instructions in ID/RF and MEM stages.



If ((rs1 == wa2 && re1 == true && we2 == true) || (rs2 == wa2 && re2 == true && we2 == true)) insert bubble for stall the instruction in ID/RF stage.

13. The JALR instruction, which unconditional and indirect jumps to rs1+immediate, incurs 1 cycle delay for the pipeline, and conditional branch, which branch to PC+immediate if certain condition is met, incurs 2 cycles delay for the pipeline. Explain why condition jump has 2 cycle delay when branch is taken and unconditional jump only has 1 cycle delay for the pipeline. (4 points)

Because the condition branch needs ALU stage to compute the branch condition

14. Explain the relationships between memory wall, principle of locality and memory hierarchy. (4 points)

15. List all variable references that exhibit temporal locality and that exhibit spatial locality in the following code (4 points)

```
sum = 0;
for (i = 0; i < n; i++)
   sum += a[i];
return sum;
```

Temporal: sum, i
Spatial: a[i]

16. For 32-bit address to access 4-way set associative, 256 sets, and 1-word (4-bytes) cache line size, how many bits are used for block index and how many bits are used for tag field. (Answer: 8 and 22) (4 points)

17. CPU execution time and CPI is computed as follows when considering memory access stalls. (10 points)

$$\text{CPU execution time} = (\text{CPU clock cycles} + \text{Memory stall cycles}) \times \text{Clock cycle time}$$

$$\text{Memory stall cycles} = \text{IC} \times \frac{\text{Memory accesses}}{\text{Instrution}} \times \text{Miss rate} \times \text{Miss penalty}$$

Assume we have a computer where the clocks per instruction (CPI) is 1.0 when all memory accesses hit in the cache. The only data accesses are loads and stores, and these total 50% of the instructions. If the miss penalty is 25 clock cycles and the miss rate is 2% for I-Cache and 10% for D-cache, how much faster would the computer be if all instructions were cache hits?

CPU (all hit) = IC * 1.0
CPU (with miss) = IC * (1.0 + 1* 0.02 * 25 + 0.5*0.1*25) = IC * 2.75

So if all cache hit, it would be 2.75/1.0 = x2.75 faster

---------- The end of the Test ----------

Full Name: _____ NetID: _____