The University of Alabama in Huntsville
Electrical & Computer Engineering Department
CPE 431 01
Final Exam Solution
Fall 2008

1. (1 point) A _wafer_ is a thin disk sliced from a silicon crystal ingot.
2. (1 point) A _cluster_ is a set of computers connected over a local area network that function as a single large multiprocessor.
3. (1 point) A _loader_ is a systems program that places an object program in main memory so that it is ready to execute.
4. (1 point) The _program counter (PC)_ is the register containing the address of the instruction in the program being executed.
5. (1 point) _True_(True or False) The clock frequency of a pipelined processor is determined by the slowest stage.
6. (10 points) The table below shows the number of floating-point operations executed in two different programs and the runtime for those programs on three different machines:

Program	Floating-point operations	Execution time in seconds		
		Computer A	Computer B	Computer C
1	10,000,000	1	10	20
2	100,000,000	1000	100	20

One user has told you that the two programs above constitute the bulk of his workload, but he does not run them equally. The user wants to determine how the three machines compare when the workload consists of different mixes of these programs. Suppose the total number of FLOPS executed in the workload is equally divided among the two programs. Find which machine is fastest for this workload and by how much.

To equalize the number of FLOPS, program 1 must be run 10 times for every 1 run of program 2, or $w_1 = 10/11$ and $w_2 = 1/11$.

$ET_A = (10/11)*1 + (1/11)*1000 = 1010/11$

$ET_B = (10/11)*10 + (1/11)*100 = 200/11$

$ET_C = (10/11)*20 + (1/11)*20 = 220/11$

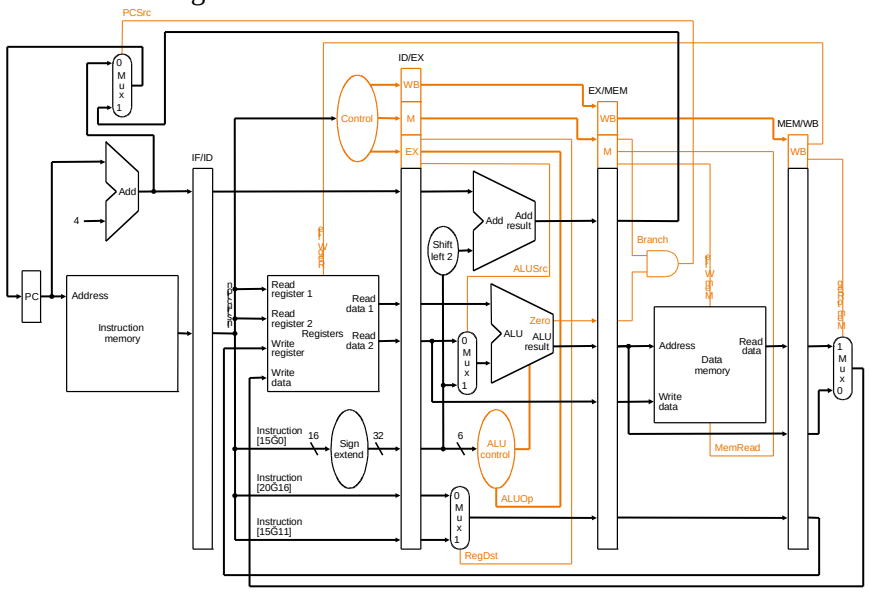
$$\frac{P_B}{P_A} = \frac{ET_A}{ET_B} = \frac{1010/11}{200/11} = 5.05 \qquad \frac{P_B}{P_C} = \frac{ET_C}{ET_B} = \frac{220/11}{200/11} = 1.1$$

7. (5 points) Write a minimal sequence of actual MIPS instructions to accomplish the samething as the following pseudoinstruction:

```
clear $t5
```

```
add $t5, $zero, $zero    or    and $t5, $t5, $zero      or
sub $t5, $t5, $t5        or    sll $t5, 32              or
addi $t5, $zero, 0       or    andi $t5, $t5, 0
```

8. (10 points) Consider executing the following code on a pipelined datapath like the one shown which has no forwarding.




```
sort:    addi $sp, $sp, -20
         sw   $ra, 16($sp)
         sw   $s3, 12($sp)
         sw   $s2, 8($sp)
         sw   $s1, 4($sp)
         sw   $s0, 0($sp)
400      add  $s2, $a0, $zero
404      add  $s3, $a1, $zero
408      add  $s0, $zero, $zero
for1tst: slt  $t0, $s0, $s3
416      beq  $t0, $zero, exit1
420      addi $s1, $s0, -1
for2tst: slt  $t0, $s1, $zero
428      bne  $t0, $zero, exit2
432      add  $t1, $s1, $s1
436      add  $t1, $t1, $t1
440      add  $t2, $s2, $t1
444      lw   $t3, 0($t2)
448      lw   $t4, 4($t2)
452      slt  $t0, $t4, $t3
456      beq  $t0, $zero, exit2
460      add  $a0, $s2, $zero
464      add  $a1, $s1, $zero
468      jal  swap
472      addi $s1, $s1, -1
476      j    for2tst
exit2:   addi $s0, $s0, 1
         j    for1tst
exit1:   lw   $s0, 0($sp)
         lw   $s1, 4($sp)
         lw   $s2, 8($sp)
         lw   $s3, 12($sp)
         lw   $ra, 16($sp)
         addi $sp, $sp, 20
         jr   $ra
```

If the `add $s2` instruction three instructions before the `for1tst` label begins executing in cycle 1 and the `beq $t0, $zero, exit1` is not taken, what are the values stored in the following fields of the IF/ID pipeline register in the 15th cycle? Assume that before the instructions are executed, the state of the machine was as follows:

The PC has the value 400_{10}, the address of the `add $s2` instruction
Every register has the initial value 30_{10} plus the register number.

Every memory word accessed as data has the initial value 1000_{10} plus the byte address of the word.

Fill in all of the fields, even if the current instruction in that stage is not using them.

IF/ID.Instruction = 0001 0101 0000 0000 0000 0000 0000 1100_2 = $1500000C_{16}$
IF/ID.PCInc = 428 (the address of bne) + 4 = 432_{10}

Cycle	IF	ID	EX	MEM	WB
1	add $s2				
2	add $s3	add $s2			
3	add $s0	add $s3	add $s2		
4	slt $t0	add $s0	add $s3	add $s2	
5	beq $t0	slt $t0	add $s0	add $s3	add $s2
6	beq $t0	slt $t0	bubble	add $s0	add $s3
7	beq $t0	slt $t0	bubble	bubble	add $s0
8	addi $s1	beq $t0	slt $t0	bubble	bubble
9	addi $s1	beq $t0	bubble	slt $t0	bubble
10	addi $s1	beq $t0	bubble	bubble	slt $t0
11	slt $t0	addi $s1	beq $t0	bubble	bubble
12	bne $t0	slt $t0	addi $s1	beq $t0	bubble
13	bne $t0	slt $t0	bubble	addi $s1	beq $t0
14	bne $t0	slt $t0	bubble	bubble	addi $s1
15	add $t1	bne $t0	slt $t0	bubble	bubble

The instruction of interest is `bne $t0`.

opcode = 000101_2 , rs = 01000_2, rt = 00000_2,
offset = (480 – 432)/4 = 48/4 = 12_{10} = 000000001100_2

9. (10 points) Using the MIPS program shown, determine the instruction format for each instruction and the decimal values of each instruction field.

```
      addi  $v0, $zero, 0     # Initialize count
loop: lw    $v1, 0($a0)       # Read next word from source
      sw    $v1, 0($a1)       # Write to destination
      addi  $a0, $a0, 4       # Advance pointer to next source
      addi  $a1, $a1, 4       # Advance pointer to next destination
      beq   $v1, $zero, loop  # Loop if word copied != zero
```

Instruction	Format	opcode	rs	rt	immediate/offest
`addi $v0, $zero, 0`	I-type	8	2	0	0
`lw $v1, 0($a0)`	I-type	35	4	3	0
`sw $v1, 0($a1)`	I-type	43	5	3	0
`addi $a0, $a0, 4`	I-type	8	4	4	4
`addi $a1, $a1, 4`	I-type	8	5	5	4
`beq $v1, $zero, loop`	I-type	4	3	0	-5

10. (5 points) For an m-stage pipeline, how many cycles does it take to execute n instructions if the pipeline is empty when these instructions begin to execute? _m + n - 1_

11. (10 points) Here is a series of address references given as word addresses: 1, 4, 8, 5, 20, 17, 19, 56, 9, 11, 4, 43, 5, 6. Assuming a direct-mapped cache with four-word blocks and a total size of 16 words that is initially empty, (a) label each reference in the list as a hit or a miss and (b) show the final contents of the cache.

$$16 words \times \frac{1 block}{4 words} \times \frac{1 set}{1 block} = 4 sets$$

	Tag	Index	Block Offset	hit/miss
1	000	00	01	miss
4	000	01	00	miss
8	000	10	00	miss
5	000	01	01	hit
20	001	01	00	miss
17	001	00	01	miss
19	001	00	11	hit
56	011	10	00	miss
9	000	10	01	miss
11	000	10	11	hit
4	000	01	00	miss
43	010	10	11	miss
5	000	01	01	hit
6	000	01	10	hit

0	~~M[0],~~ M[16]	~~M[1],~~ M[17]	~~M[2],~~ M[18]	~~M[3],~~ M[19]
1	~~M[4],~~ ~~M[20],~~ M[4]	~~M[5],~~ ~~M[21],~~ M[5]	~~M[6]~~, ~~M[22],~~ M[6]	~~M[7],~~ ~~M[23],~~ M[7]
2	~~M[8],~~ ~~M[56],~~ ~~M[8],~~ M[40]	~~M[9],~~ ~~M[57],~~ ~~M[9],~~ M[41]	~~M[10],~~ ~~M[58],~~ ~~M[10],~~ M[42]	~~M[11],~~ ~~M[59],~~ ~~M[11],~~ M[43]
3				

12. (10 points) Consider the case of a six-deep pipeline where the branch is resolved at the end of the third stage for unconditional branches and at the end of the fifth cycle for conditional branches. The program run on this pipeline has the following branch frequencies (as percentages of all instructions) are as follows:

Conditional branches 20%

Jumps and calls 5%

60 % of the conditional branches are taken. Assuming that the CPI of the program, neglecting branch hazards, is 1.0, how much slower is the real number, when branch hazards are considered?

First, determine the penalties involved for the conditional and unconditional branches

Unconditional Branches

after 2	after 1	branch			
target	after 2	after 1	branch		
	target	after 2	after 1	branch	

Conditional Branches

after 4	after 3	after 2	after 1	branch	
target	after 4	after 3	after 2	after 1	branch
	target	after 4	after 3	after 2	after 1

So, the penalty for unconditional branches is two clock cycles and the penalty for conditional branches is four clock cycles.

$$CPI_{branch\ hazards} = CPI_{perfect} + \text{Conditional branches taken} * \text{penalty} + \text{Unconditional branches taken} * \text{penalty}$$
$$= 1.0 + 0.2*0.6*4 + 0.05*2 = 1 + 0.48 + 0.1 = 1.58$$

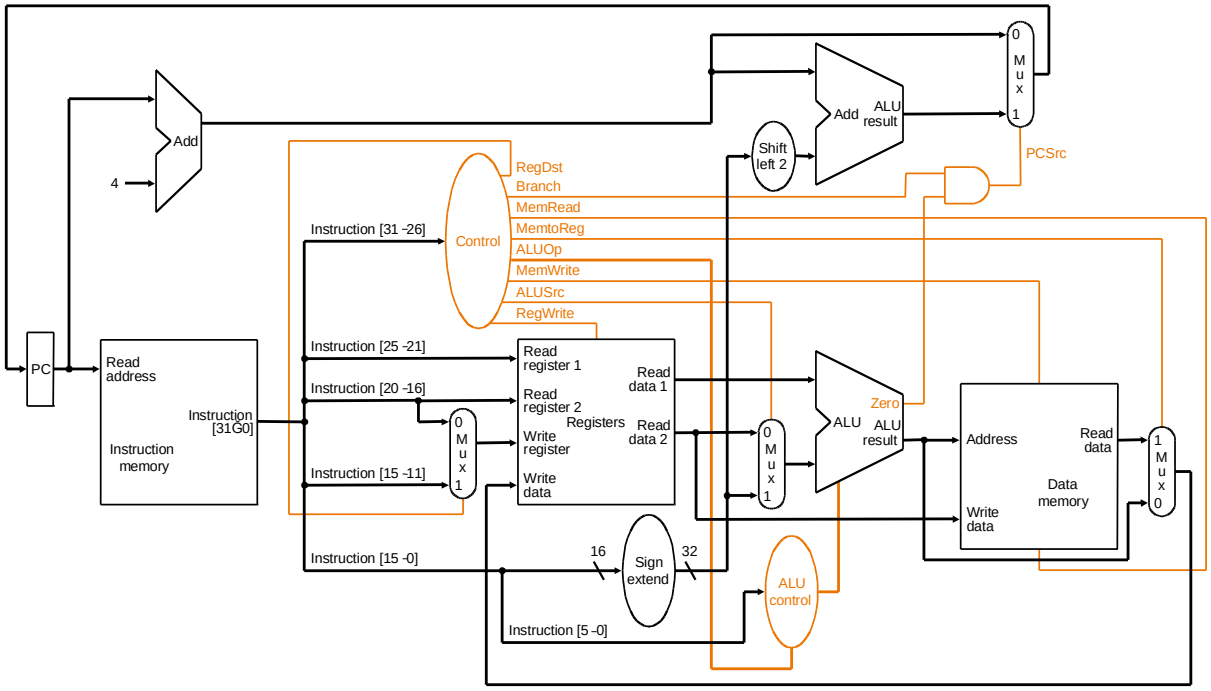
13. (5 points) What single precision floating-point number does the following collection of 32 bits represent?

```
1111 1100 1010 1111 1111 0000 0000 0000
```

Sign = 1, Exponent = 1111 1001, Fraction = 01011111111

Number = $(-1)^{Sign} * (1.Fraction)*2^{(Exponent - Bias)} = (-1)1* (1.01011111111)*2^{(249-127)}$
$= -1 * (1 + 2^{-2} + 2^{-4} + 2^{-5} + 2^{-6} + 2^{-7} + 2^{-8} + 2^{-9} + 2^{-10} + 2^{-11}) * 2^{122} = -7.31 * 10^{36}$

14. (10 points) Describe the effect that a single stuck-at1 fault (i.e., regardless of what it should be, the signal is always 1) would have for the signals shown below, in the single-cycle datapath in Figure 5.17 on page 307. Which instructions, if any, will not work correctly? Explain why. Consider each of the following faults separately: RegDst = 0, ALUSrc =0, MemtoReg = 0, Zero = 0.



- RegDst s-a-0: The rt field is always selected.
- ALUSrc s-a-0: The value of Read Data 2 is always selected.
- MemtoReg s-a-0: The ALU output is always selected
- Zero s-a-0: The status output of the ALU is always zero.

Instruction	RegDst	ALUSrc	Memto-Reg	Reg Write	Mem Read	Mem Write	Branch	ALU Op1	ALU Op0
R-format	1	0	0	1	0	0	0	1	0
`lw`	0	1	1	1	1	0	0	0	0
`sw`	d	1	d	0	0	1	0	0	0
`beq`	d	0	d	0	0	0	1	0	1

d = don't care

	R-format	`lw`	`sw`	`beq`
RegDst s-a-0	doesn't work	works	works	works
ALUSrc s-a-0	works	doesn't work	doesn't work	works
MemtoReg s-a-0	works	doesn't work	works	works
Zero s-a-0	works	works	works	doesn't work

15. (10 points) A secret agency simultaneously monitors 200 cellular phone conversations and multiplexes the data onto a network with a bandwidth of 10 MB/sec and an overhead latency of 250 μs per 5 KB message. Calculate the transmission time per message and determine whether there is sufficient bandwidth to support this application. Assume that the phone conversation data consists of 4 bytes sampled at a rate of 5 KHz. The question is, can the data gathered in one second be transferred in one second?

$$200 conversations \times \frac{5000 samples}{1\text{sec}} \times \frac{4 bytes}{conversation} = 4 \times 10^6 \frac{bytes}{\text{sec}}$$

$$4 \times 10^6 \frac{bytes}{\text{sec}} \times \frac{1 message}{5 \times 10^3 bytes} = 800 \frac{messages}{\text{sec}}$$

$$time = latency + transfer\_time$$

$$= 800 messages * 250 \times 10^{-6} \frac{\text{sec}}{message} + 4 \times 10^6 bytes \times \frac{1\text{sec}}{10 Mbytes}$$

$$= 200 ms + 400 ms = 600 ms$$

Since 600 ms < 1 sec, yes, there is sufficient bandwidth.

16. (10 points) Consider a write-back cache used for a processor with a bus and memory system as described in the book on page 665 (assume that writes require the same amount of time as reads). The time to transfer 8-word blocks is 49 cycles and the time to transfer 16-word blocks is 57 cycles. Additionally, the following performance measurements have been made: The cache miss rate is 0.05 misses per instruction for block sizes of 8 words. The cache miss rate is 0.03 misses per instruction for block sizes of 16 words. For either block size, 40% of the misses require a write-back operation, while the other 60% require only a read.

Assuming that the processor is stalled for the duration of a miss (including the write-back time if a write-back is needed), find the number of cycles per instruction that are spent handling cache misses for each block size. (Hint: First compute the miss penalty.)

$CPI_{\text{cache miss}}$ = Cache miss rate * (writeback * (write + read) + no writeback* read)

8 words

$CPI_{\text{cache miss}} = 0.05(0.4*(49 + 49) + 0.6(49)) = 0.05*1.4*49 = 3.43$

16 words

$CPI_{\text{cache miss}} = 0.03(0.4*(57+ 57) + 0.6(57)) = 0.03*1.4*57 = 2.39$