The University of Alabama in Huntsville
Electrical & Computer Engineering Department
CPE 431 01
Test 2 Solution
Fall 2003

1. (15 points) Consider the case of a four-deep pipeline where the branch is resolved at the end of the second stage for unconditional branches and at the end of the third 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%
Conditional branches	60% 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?

j			
after 1	j		
target	bubble	j	

beq			
after1	beq		
after2	after1	beq	
target	bubble	bubble	beq

The penalty for an unconditional branch being taken is 1 cycle, the penalty for a conditional branch being taken is 2 cycles. An unconditional branch being taken occurs 5% of the time. A conditional branch being taken occurs 0.2*0.6 or 12% of the time.

$CPI_{perfect}$=1.0, $CC_{perfect} = CC_{branchhazards}$, $IC_{perfect} = IC_{branchhazards}$

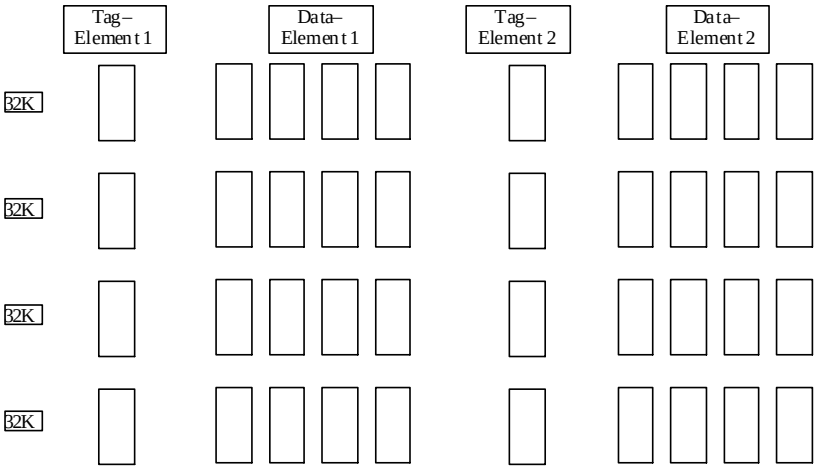
$CPI_{branch\ hazards} = CPI_{perfect} + CPI_{branch\ penalties} = 1.0 + 0.05*1 + (0.2*0.6*2) = 1.0 + 0.05 + 0.24 = 1.29$

$$\frac{P_{perfect}}{P_{branchhazards}} = \frac{ET_{branchhazards}}{ET_{perfect}} = \frac{IC_{perfect} \times CC_{perfect} \times CPI_{perfect}}{IC_{branchhazards} \times CC_{branchhazards} \times CPI_{branchhazards}} = \frac{CPI_{perfect}}{CPI_{branchhazards}} = \frac{1.29}{1} = 1.29$$

2. (15 points) You have been given 50 32K x 16-bit SRAMS to build an instruction cache for a processor with a 32-bit address. You do not have a byte offset. What is the largest size (i.e., the largest size of the data storage area in bytes) two-way set associative instruction cache that you can build with two-word blocks? Show the breakdown on the address into its cache access components and describe how the various SRAM chips will be used.

Byte offset is 0 bits, Block offset is 1 bits, Index n bits, Tag 32 – n – 1. It takes 4 chips to get two words, and we have increments of 128 Kwords because we have two-way set associativity and two word blocks. 128 Kwords comes from 32 Ksets, for which 15 bits of index are required, leaving 16 bits of tag. 16 bits for each of two tags (for each element in the set) requires 2 chips. The data requires 8 chips (2 chips/word, 2 words/block, 2 blocks per set). Tag plus data for 128 Kwords requires 10 total chips. We can quadruple this and still have extra chips. So, the total data storage is 128Kwords * 4 *4 bytes/word for a total data storage of 2 MB and the address is broken down as

Byte offset – 0 bits
Block offset – 1 bit
Index – 17 bits
Tag – 14 bits



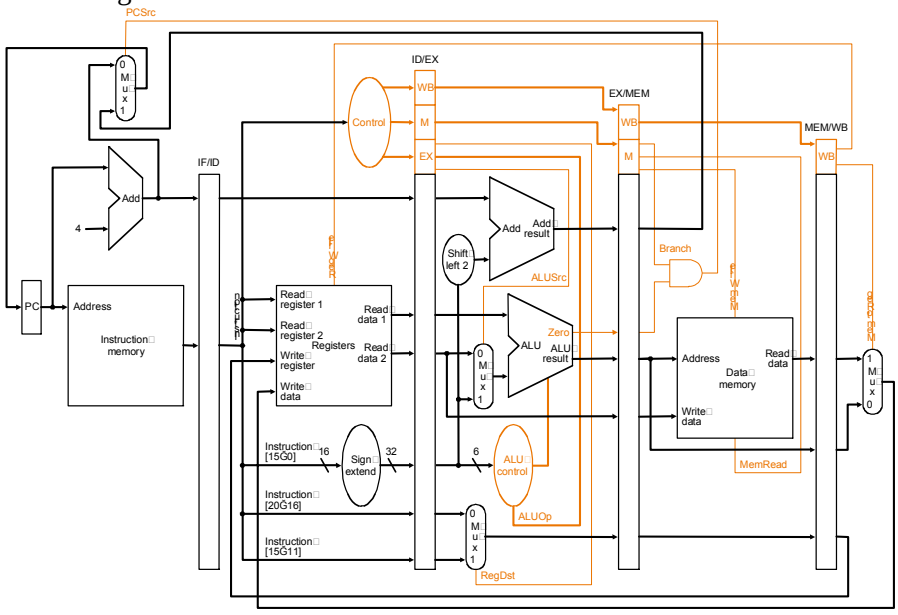
3. (1 point) A _____structural hazard___ means that the hardware cannot support the combination of instructions that we want to execute in the same clock cycle.

4. (1 point) The one case where forwarding cannot save the day is the case where a ___lw___ instruction is followed by __use of the register loaded by lw_____.

5. (1 point) A ___superscalar__ computer can launch multiple instructions in a single cycle.

6. (1 point) __Spatial locality___ states that if an item is referenced, items whose addresses are close by will tend to be referenced soon.

7. (20 points) Consider executing the following code on a pipelined datapath like the one shown, except that it has 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)
         add $s2, $a0, $zero
         add $s3, $a1, $zero
         add $s0, $zero, $zero
for1tst: slt $t0, $s0, $s3
         beq $t0, $zero, exit1
         addi$s1, $s0, -1
for2tst: slt $t0, $s1, $zero
         bne $t0, $zero, exit2
         add $t1, $s1, $s1
         add $t1, $t1, $t1
         add $t2, $s2, $t1
         lw  $t3, 0($t2)
         lw    $t4, 4($t2)
         slt   $t0, $t4, $t3
         beq   $t0, $zero, exit2
         add   $a0, $s2, $zero
         add   $a1, $s1, $zero
         jal   swap
         addi  $s1, $s1, -1
         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 `sw $s3` instruction two instructions after the `sort` label begins executing in cycle 1 and the `beq $t0, $zero, exit1` is taken, what are the values stored in the following fields of the ID/EX pipeline register in the 8th cycle? Assume that before the instructions are executed, the state of the machine was as follows:

The PC has the value 500_{10}, the address of the `addi` instruction (with the label sort).
Every register has the initial value 20_{10} plus the register number.
Every memory word accessed as data has the initial value 10000_{10} plus the byte address of the word.

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

ID/EX.WB = 10_2
ID/EX.MEM = 000
ID/EX.EX = 1100_2
ID/EX.PCInc = 532_{10} if start at `addi`, 524_{10} if start at `sw $s3`
ID/EX.Read1 = $a1 = 25
ID/EX.Read2 = $zero = 0
ID/EX.SignEx = SignExtend($1001\ 1000\ 0010\ 0000_2$) = $FFFF9820_{16}$
ID/EX.Writert = 0_{10}
ID/EX.Writerd = 19_{10}

Cycle	IF	ID	EX	MEM	WB
1	`sw $s3`				
2	`sw $s2`	`sw $s3`			
3	`sw $s1`	`sw $s2`	`sw $s3`		
4	`sw $s0`	`sw $s1`	`sw $s2`	`sw $s3`	
5	`add $s2`	`sw $s0`	`sw $s1`	`sw $s2`	`sw $s3`
6	`add $s3`	`add $s2`	`sw $s0`	`sw $s1`	`sw $s2`
7	`add $s0`	`add $s3`	`add $s2`	`sw $s0`	`sw $s1`
8	`slt $t0`	`add $s0`	`add $s3`	`add $s2`	`sw $s0`

The instruction of interest is add $s3, $a1, $zero

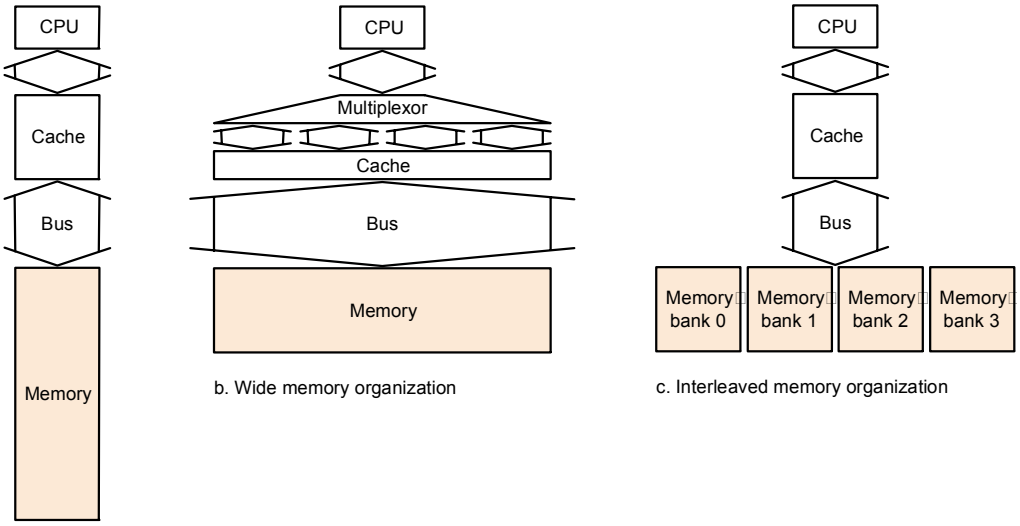
The fields of the instruction are as follows:

op	000000,	rs	00101,	rt	00000
rd	10011,	shamt	00000	funct	100000

8. (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___

9. (1 point) In the case where memory and cache have different values for the same memory location, the cache and memory are said to be ___inconsistent_.

10. (15 points) Consider a memory hierarchy using one of the three organizations for main memory shown below. Assume that the cache block size is 16 words, that the width of organization b of the figure is eight words, and that the number of banks in organization c is four. If the main memory latency for a new access is 30 cycles and the transfer time is 3 cycles, what are the miss penalties for organizations a, b and c?



a. One-word-wide memory organization

b. Wide memory organization

c. Interleaved memory organization

a. 3 send address + 16 * (30 read + 3 transfer) = 3 + 16 * 33 = 3 + 528 = 531 cycles
b. 3 send address + 2 * (30 read + 3 transfer) = 3 + 2 * 33 = 69 cycles
c. 3 send address + 4 * 30 read + 16 * 3 transfer = 3 + 120 + 48 = 171 cycles

11. (15 points) (a) Identify all of the data dependencies in the following code. (b) How is each data dependency either handled or not handled by forwarding? Draw a multiple clock cycle style diagram to support your answer.

```
add   $2, $5, $4
lw    $4, 28($2)
add   $3, $2, $5
sw    $4, 100($2)
add   $3, $3, $4
```

Data dependencies:	**Hazard?**	**Forward?**
1. add $2, $5, $4 – lw $4, 28($2)	Yes	Yes, from EX/MEM
2. add $2, $5, $4 – add $3, $2, $5	Yes	Yes, from MEM/WB
3. add $2, $5, $4 – sw $4, 100($2)	No	not needed
4. lw $4, 28($2) – sw $4, 100($2)	Yes	Yes, from MEM/WB
5. lw $4, 28($2) – add $3, $3, $4	No	not needed
6. add $3, $2, $5– add $3, $3, $4	Yes	Yes, from MEM/WB

12. (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 two-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.

$$16 words * \frac{1 block}{2 words} * \frac{1 set}{1 block} = 8 sets$$

Block Offset - 1 bit, Index - 3 bits

1	00 000 1	m
4	00 010 0	m
8	00 100 0	m
5	00 010 1	h
20	01 010 0	m
17	01 000 1	m
19	01 001 1	m
56	11 100 0	m
9	00 100 1	m
11	00 101 1	m
4	00 010 0	m
43	10 101 1	m
5	00 010 1	h
6	00 011 0	m

0	~~0~~, 16	~~1~~, 17
1	18	19
2	~~4~~, ~~20~~, 4	~~5~~, ~~21~~, 5
3	6	7
4	~~8~~, ~~56~~, 8	~~9~~, ~~57~~, 9
5	~~10~~, 42	~~11~~, 43
6		
7		