The University of Alabama in Huntsville
Electrical & Computer Engineering Department
CPE 431 01
Test 2 Solution
November 15, 2007

1. (1 point) A __data hazard___ is an occurrence in which a planned instruction cannot execute in the proper clock cycle because data that is needed to execute the instruction is not yet available.

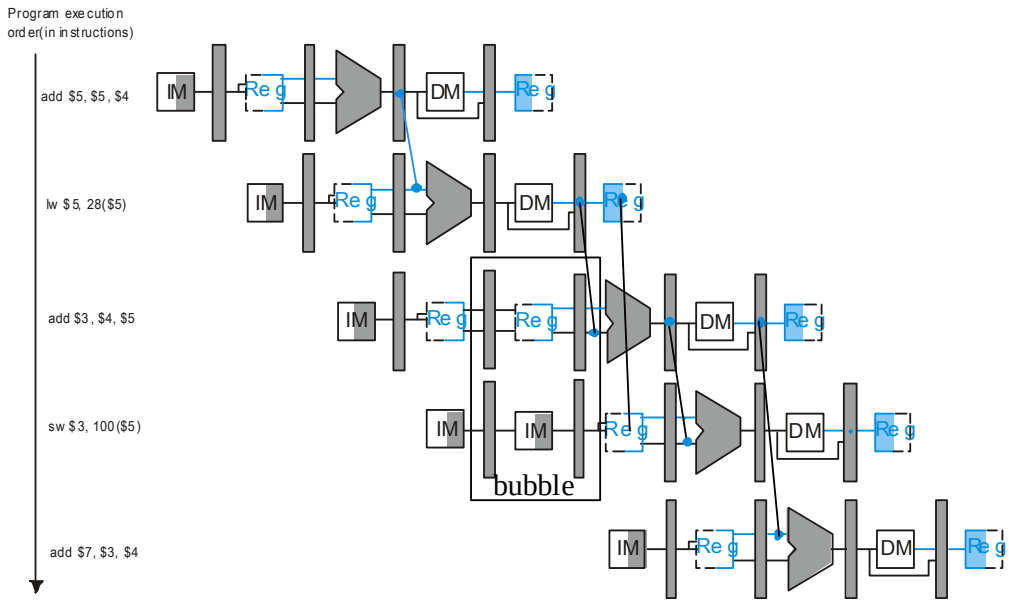
2. (3 points) The three Cs of cache misses are _compulsory _, _capacity_, and _conflict_.

3. (1 point) A _page fault_ occurs when an accessed page is not present in main memory.

4. (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   $5, $5, $4  (a)
lw    $5, 28($5)  (b)
add   $3, $4, $5  (c)
sw    $3, 100($5) (d)
add   $7, $3, $4  (e)
```

```
Dependencies   Hazard?
     a-b         yes     forward from EX/MEM pipeline register
     b-c         yes     stall, forward from MEM/WB pipeline register
     b-d          no     data is available in register file after stall
     c-d         yes     forward from EX/MEM pipeline register
     c-e         yes     forward from MEM/WB pipeline register
```



5. (5 points) Given the pipelined processor of Chapter 6 with forwarding, determine which instruction is being executed in each stage of the pipeline in cycle 12 of the following instruction sequence if it begins executing in cycle 1

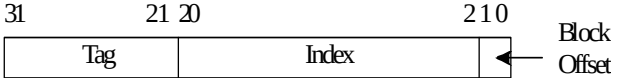
```
1.    addi  $23, $29, 12          8.    lw    $16, 4($2)
2.    sw    $2, 0($29)            9.    sw    $17, 0($2)
3.    sw    $15, 4($29)           10.   sw    $18, 4($2)
4.    sw    $16, 8($29)           11.   lw    $2, 0($29)
5.    muli  $8, $5, 4             12.   lw    $15, 4($29)
6.    add   $7, $4, $2            13.   lw    $16, 8($29)
7.    lw    $15, 0($2)            14.   addi  $29, $29, 12
```

IF __lw $15__ ID __lw $2__ EX __sw $18__ MEM ___sw $17__ WB __lw $16_

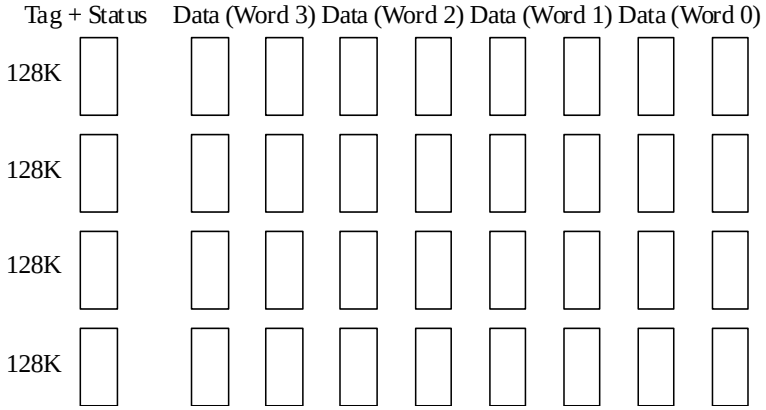
6. (15 points) You have been given 50 128K x 16-bit SRAMS to build an instruction cache for a processor with a 32-bit address. You do not have a byte offset. You do need 4 bits of storage per block for valid, dirty and other status bits. What is the largest size (i.e., the largest size of the data storage area in bytes) direct-mapped instruction cache that you can build with four-word blocks? Show the breakdown on the address into its cache access components and describe how the various SRAM chips are used.

Byte offset is 0 bits, Block offset is 2 bits, Index n bits, Tag 32 – n – 2. It takes 8 chips to get 4 words, and we have increments of 512K words because we have 4 word blocks. 512K words comes from 128K sets, for which 17 bits of index are required, leaving 13 bits of tag. 13 bits for each tag plus 4 status bits requires 2 chips. The data requires 8 chips (2 chips/word, 4 words/block, 1 block per set). Tag plus status plus data for 512K words requires 10 total chips. We can quadruple this and still have extra chips. So, the total data storage is 512K words * 4 *4 bytes/word for a total data storage of 8 MB and the address is broken down as

Byte offset – 0 bits, Block offset – 2 bits, Index – 19 bits, Tag – 11 bits



Now, tag plus storage (11 + 4) fits in one chip, so we have 4 chips for tag plus status and 32 chips for data.



7. (15 points) Here is a series of address references given as word addresses: 1, 4, 8, 5, 20, 17, 19, 9, 56, 11, 4, 43, 5, 6. Assuming a two-way set associative cache with two-word blocks and a total size of 16 words that is initially empty and used LRU, (a) label each reference in the list as a hit or a miss and (b) show the final contents of the cache.

$$16 words \times \frac{1 block}{2 words} \times \frac{1 set}{2 block} = 4 sets$$

	Tag	Index	Block Offset	Hit/Miss
1	0000	00	1	Miss
4	0000	10	0	Miss
8	0001	00	0	Miss
5	0000	10	1	Hit
20	0010	10	0	Miss
17	0010	00	1	Miss
19	0010	01	1	Miss
9	0001	00	1	Hit
56	0111	00	0	Miss
11	0001	01	1	Miss
4	0000	10	0	Hit
43	0101	01	1	Miss
5	0000	10	1	Hit
6	0000	11	0	Miss

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

8. (10 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 considere

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

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

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

9. (20 points) Unroll the following code so that three iterations of the loop are done at once. Assume the loop index is a multiple of 3 (i.e., $10 is a multiple of twelve):

```
Loop: lw    $2, 0($10)            Loop is executed 120 times
      sub   $4, $2, $3
      sw    $4, 0($10)
      addi  $10, $10, 4
      bne   $10, $30, Loop
```

Schedule this code for fast execution on the standard MIPS pipeline with forwarding (assume that it supports addi instruction). Assume initially $10 is 0 and $30 is 480 and that branches are resolved in the ID stage. How does the unrolled, scheduled code compare against the original code in terms of total execution time?

Original

```
Loop:   lw    $2, 0($10)
        sub   $4, $2, $3
        sw    $4, 0($10)
        addi  $10, $10, 4
        bne   $10, $30, Loop
```

C	IF	ID	EX	MEM	WB
1	lw $2				
2	sub $4	lw $2			
3	sw $4	sub $4	lw $2		
4	sw $4	sub $4	bubble	lw $2	
5	addi	sw $4	sub $4	bubble	lw $2
6	bne	addi	sw $4	sub $4	bubble
7	after1	bne	addi	sw $4	sub $4
8	lw $2	bubble	bne	addi	sw $4

119 iterations * 7 cycles per iteration + 1 iteration * 6 cycles per iteration = 839 cycles

Unrolled, Scheduled

```
Loop:   lw    $2, 0($10)
        lw    $5, 4($10)
        lw    $6, 8($10)
        sub   $4, $2, $3
        sw    $4, 0($10)
        sub   $7, $5, $3
        sw    $7, 4($10)
        sub   $8, $6, $3
        sw    $8, 8($10)
        addi  $10, $10, 12
        bne   $10, $30, Loop
```

C	IF	ID	EX	MEM	WB
1	lw $2				
2	lw $5	lw $2			
3	lw $6	lw $5	lw $2		
4	sub $4	lw $6	lw $5	lw $2	
5	sw $4	sub $4	lw $6	lw $5	lw $2
6	sub $7	sw $4	sub $4	lw $6	lw $5
7	sw $7	sub $7	sw $4	sub $4	lw $6
8	sub $8	sw $7	sub $7	sw $4	sub $4
9	sw $8	sub $8	sw $7	sub $7	sw $4
10	addi	sw $8	sub $8	sw $7	sub $7
11	bne	addi	sw $8	sub $8	sw $7
12	after1	bne	addi	sw $8	sub $8
13	lw $2	bubble	bne	addi	sw $8

39 iterations * 12 cycles per iteration + 1 iteration * 11 cycles per iteration = 479 cycles

10. (15 points) Consider three processors with different cache configurations:

Cache 1: Direct-mapped with one-word blocks
Cache 2: Direct-mapped with four-word blocks
Cache 3: Two-way set associative with four-word blocks

The following miss rate measurements have been made:

Cache 1: Instruction miss rate is 4%; data miss rate is 6%.
Cache 2: Instruction miss rate is 2%; data miss rate is 4%.
Cache 3: Instruction miss rate is 2%; data miss rate is 3%.

For these processors, one-half of the instructions contain a data reference. Assume that the cache miss penalty is 6 + Block size in words. The CPI for this workload was measured on a processor with cache 1 and was found to be 2.0. The cycle times for the processors are 420 ps for the first and second processors and 310 ps for the third processor. Determine which processor is the fastest and which is the slowest.

Miss cycles = (IC*instruction miss rate + IC*0.5*data miss rate)*miss penalty

Cache1: Miss cycles = (IC*0.04 + IC*0.5*0.06)*7 cycles = 7*(0.04+0.03)*IC = 0.49 IC
Cache 2: Miss cycles = (IC*0.02 + IC*0.5*0.04)*10 cycles = 10*(0.02+0.02)*IC = 0.4 IC
Cache 3: Miss cycles = (IC*0.02 + IC*0.5*0.03)*10 cycles = 10*(0.02+0.015)*IC = 0.35 IC

$CPI_{measured} = CPI_{base} + CPI_{miss}$
Cache 1: $2.0 = CPI_{base} + 0.49$, $CPI_{base} = 1.51$
Cache 2: $CPI_{measured} = 1.51 + 0.4 = 1.91$
Cache 3: $CPI_{measured} = 1.51 + 0.35 = 1.86$

$ET = IC \times CPI \times CT$

$ET_{Cache1} = IC \times 2.0 \times 420 \text{ ps} = 840 \text{ IC}$
$ET_{Cache2} = IC \times 1.91 \times 420 \text{ ps} = 802.2 \text{ IC}$
$ET_{Cache3} = IC \times 1.86 \times 310 \text{ ps} = 576.6 \text{ IC}$

The processor with cache 3 is the fastest and the processor with cache 2 is the slowest.