EECS 322 Test 4 solution Monday April 30, 2001

Name: _____ email: _____

For the following instruction sequence fill in the direct-mapped writeback data cache. **The word size is 32 bits.** Memory[0]=0x05370899; Memory[12]=0x10871625; Memory[60]=0x16581727;

Problem 1a. (25%) Fill in the miss cache column.

tag	index	byte offset	instruction	Cache Miss?
00	00	01	lbu $6, 1($0)	YES
11	11	10	lw $1, 62($0)	YES
00	00	11	sb $0, 3($0)	
11	11	01	lbu $2, 61($0)	
00	11	00	lw $4, 12($0)	YES
11	11	00	sw $0, 60($0)	YES

Problem 1b. (25%) Show **all states** and **underline the final state** of the direct mapped data cache:

Index	Valid	Dirty	Tag	32 bit Data
00	N→Y→Y	X→N→Y	X → 00 → 00	0x05370899 → 0x05370800
01	N	X		
10	N	X		
11	N→Y→Y	X → N → N → Y	X → 11 → 00 → 11	0x16581727 → 0x10871625 → 0x00000000

P2a. (25%) Assume only 1024 bytes of **real memory** (0-511)(512-1023); LRU, a page size of 512 bytes and no pages loaded in memory. Fill in the columns. (Blank space implies No or none)

Instruction virtual address	Page fault?	Flush which real page?	Write flushed page to disk?	Load what new *virtual* page	Load into what *real* page
lw $1, 742($0)	YES			1 (512..1023)	0 (0..511)
sw $2, 1412($0)	YES			2 (1024..1535)	1 (512..1023)
lbu $3, 1769($0)	YES	0 (0..511)		3 (1536..2047)	0 (0..511)
sw $4, 814($0)	YES	1 (512..1023)	YES	1 (512..1023)	1 (512..1023)
lw $5, 1431($0)	YES	0 (0..511)		2 (1024..1535)	0 (0..511)
lbu $6, 1821($0)	YES	1 (512..1023)	YES	3 (1536..2047)	1 (512..1023)

P2b. (10%) After execution of part **P2a**: what **virtual** page is contained in the **real** page 0 = **2 or (1024..1535)** and **real** page 1 = **3 or (1536..2047)**

P2c. (15%) After execution of part **P2a**: fill out the TLB (Hint: think of a fully associative cache)

Valid	Dirty	Virtual Page Tag	Physical Page Number
N→Y→N→Y→N	X→N→N→Y→N	1 or (512..1023)	X → 0 → X → 1 → X
N→Y→N→Y	X→Y→N→N	2 or (1024..1535)	X → 1 → X → 0
N→Y→N→Y	X→N→N→N	3 or (1536..2047)	X → 0 → X → 1

Note: page 0 = 0..511, page 1 = 512..1023, page 2 = 1024..1535, page 3 = 1536..2047, …

Robertson Stephens, *typecast as the boutique bank specializing in Silicon Valley IPOs*, underwrites IPOs, specializing in such high-tech issues as Sun Microsystems, Pixar Animation Studios, and E*TRADE. (Revenue per employee: $1,191,895.11) www.robertsonstephens.com would like you to have access to the following *extra credit which can be used for this and previous exams.*

RS1 (10%) Using problem **P2a,** rewrite the instruction sequence which **minimizes** page faults.
Multiple solutions: 3 page faults or misses down from 6 page faults in P2a.

```
lw    $1, 742($0)
sw    $4, 814($0)
lbu   $3, 1769($0)
lbu   $6, 1821($0)
lw    $2, 1412($0)
lw    $5, 1431($0)
```

```
lw    $1, 742($0)
sw    $4, 814($0)
lbu   $3, 1769($0)
lw    $2, 1412($0)
lbu   $6, 1821($0)
lw    $5, 1431($0)
```

```
lw    $1, 742($0)
lw    $2, 1412($0)
sw    $4, 814($0)
lw    $5, 1431($0)
lbu   $3, 1769($0)
lbu   $6, 1821($0)
```

RS2 (10%). Assemble the following machine instructions into **binary**

Field 1 =sw	$rs=$fp $rt=$s3 offset=4	instruction
101011	11110 10011 0000 0000 0000 0100 Note: No divide by 4 on the immediate field. Only the branch & jumps require that.	`sw $s3, 4($fp)`

RS3 (10%) Draw lines showing all the data dependencies in "Time" column. Using forwarding, show the 5-stage MIPS pipeline sequence (IF, ID, EX, M, WB) and draw lines showing the forwarding.
(Hint: sw $rt,offset($rs) when does a store word stall? $rt or $rs?)

Time	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
sub $2, $1, $4	IF	ID	EX	M	W											
sw $2, 4($1)		IF	ID	EX	M	W										
add $5, $2, $1			IF	ID	EX	M	W									
lw $3, 8($5)				IF	ID	EX	M	W								

See Page 481, Figure 6.37.
Page 489, sec 6.5: …one case where forwarding cannot save the day …a load instruction…

RS4. (20%) Translate the following C code into MIPS. Please comment your code. Assume **w** is $a1 and points to integers; **cp** is $a2 and points to unsigned char; **int x** is $t3; **int y** is $t4;.

Note: `sw $rt,offset($rs)` equals `*($rs+offset)=$rt` or `$rs[offset]=$rs`

(a) cp[y] = x;

```
add  $t0, $a2, $t4      # $t0 = (char *cp)$a2 + (int y)$t4
sb   $t3, 0($t0)        # *(cp + y) = (int x)$t3
```

(b) w[y] = x;

```
sll  $t0, $t4, 2        # $t0 = y << 2;
add  $t0, $a1, $t0      # $t0 = (int *w)$a1  +  y<<2;
sw   $t3, 0($t0)        # *(w + y) = (int x)$t3
```

```
#alternate solution
add  $t0, $t4, $t4      # $t0 = y*2;
add  $t0, $t0, $t0      # $t0 = (y*2)*2
add  $t0, $a1, $t0      # $t0 = (int *w)$a1  +  y<<2;
sw   $t3, 0($t0)        # *(w + y) = (int x)$t3
```