16.482 / 16.561: Computer Architecture and Design

Fall 2013

Homework #2 Due **Monday**, **9/23/13**

Notes:

- While typed submissions are preferred, handwritten submissions are acceptable.
- Any handwritten solutions that are scanned and submitted electronically <u>must</u> be clearly legible and combined into a single file—<u>simply sending a picture of each scanned page is not an acceptable form of submission.</u>
- 1. (50 points) For each part of the following question, assume the following initial state. Note that your answers for each sequence should start with the values below—your answer to part (a), for example, should not affect your answer to part (b), but the first instruction in part (a) affects the second instruction in part (a).

```
• $t0 = 0x00000009, $t1 = 0x00000004, $t2 = 0x00273000
```

• mem[0x273000] = 0x02732011, mem[0x273004] = 0x8899AABB

For each sequence of instructions below, list <u>all</u> changed registers or memory locations and their new values. When listing memory values, list the entire word—for example, if a byte is written to 0x00273000, show the values at addresses 0x00273000-0x00273003.

a. addi		\$t0,	-3	d.	lui	lui \$s0,		0×0027	
ub	\$t4,	\$t4,	\$t1		ori	\$s0,	\$s0,	0x3000	
dd	\$t5,	\$t4,	\$t0		lh	\$s2,	2(\$s0)		
					lh	\$s3,	4(\$s))	
r	\$t5,	\$t0,	\$t1		and	\$s4,	\$s2,	\$s3	
ri	\$t6,	\$t5,	0x1110		SW	\$s3,	4(\$s))	
ra	\$t7,	\$t6,	3						
				e.	slt	\$s0,	\$t1,	\$t0	
c. lw \$t4		4, 0(\$t2)			beq	\$s0,	\$zero), L	
ddi	\$t5,	\$t4,	9		add	\$t0,	\$t0,	\$t1	
sw		\$t5, 4(\$t2)		r:	add	\$t3,	\$t0,	\$t0	
				No	Note: Indicate if the branch is taken.				
ו ו	ub dd r ri ra w ddi	ab \$t4, dd \$t5, r \$t5, ri \$t6, ra \$t7, w \$t4, ddi \$t5,	ab \$t4, \$t4, \$t4, \$td, \$t5, \$t4, \$t5, \$t0, \$t6, \$t5, \$t6, \$t7, \$t6, \$t4, 0(\$t2, \$t5, \$t4, \$t5, \$t4, \$t5, \$t4, \$t5, \$t4,	ab \$t4, \$t4, \$t1 dd \$t5, \$t4, \$t0 r \$t5, \$t0, \$t1 ri \$t6, \$t5, 0x1110 ra \$t7, \$t6, 3 w \$t4, 0(\$t2) ddi \$t5, \$t4, 9	ab \$t4, \$t4, \$t1 dd \$t5, \$t4, \$t0 r \$t5, \$t0, \$t1 ri \$t6, \$t5, 0x1110 ra \$t7, \$t6, 3 e. w \$t4, 0(\$t2) ddi \$t5, \$t4, 9 w \$t5, 4(\$t2) L:	ab \$t4, \$t4, \$t1 ori dd \$t5, \$t4, \$t0 lh lh lh r \$t5, \$t0, \$t1 and ri \$t6, \$t5, 0x1110 sw ra \$t7, \$t6, 3 e. slt w \$t4, 0(\$t2) beq ddi \$t5, \$t4, 9 add w \$t5, 4(\$t2) L: add	ab \$t4, \$t4, \$t1 ori \$s0, dd \$t5, \$t4, \$t0 lh \$s2, lh \$s3, r \$t5, \$t0, \$t1 and \$s4, ri \$t6, \$t5, 0x1110 sw \$s3, ra \$t7, \$t6, 3 e. slt \$s0, w \$t4, 0(\$t2) beq \$s0, ddi \$t5, \$t4, 9 add \$t0, w \$t5, 4(\$t2) L: add \$t3,	ab \$t4, \$t4, \$t1 ori \$s0, \$s0, \$dd dd \$t5, \$t4, \$t0 lh \$s2, 2(\$s0, \$dd lh \$s3, 4(\$s0, \$dd \$s4, \$s2, \$dd ri \$t6, \$t5, 0x1110 \$sw \$s3, 4(\$s0, \$dd ra \$t7, \$t6, 3 \$s0, \$t1, \$dd \$s0, \$t1, \$dd w \$t4, 0(\$t2) \$beq \$s0, \$zero \$dd \$t0, \$t0, \$dd ddi \$t5, \$t4, 9 add \$t0, \$t0, \$dd \$t0, \$t0, \$dd \$t3, \$t0, \$dd	

2. (25 points) Translate the high-level code below into MIPS assembly. You can assume that \$s0 holds the value of x and \$s1 holds the value of y.

```
int x, y;
...
x = 0;
while (x < 30) {
   y = x >> 4;
   x = x + 2;
}
```

3. (25 points) Translate the high-level code below into MIPS assembly. Assume that \$11 holds the value of var and \$12 holds the value of i. Assume that you must store the value of i to address 0x2000A000 at the end of this code.

```
char i;
int var;
. . .
switch (var) {
  case 0:
     i++;
     break;
  case 1:
     i--;
     break;
  case 2:
     i = 0x12345678;
     break;
  default:
     i = 0;
}
```