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 must be clearly legible and combined into a single file—simply sending a picture of each scanned page is not an acceptable form of submission.

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 **all** 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    $t4, $t0, -3
sub     $t4, $t4, $t1
add     $t5, $t4, $t0
```

b.
```
or      $t5, $t0, $t1
ori     $t6, $t5, 0x1110
sra     $t7, $t6, 3
```

c.
```
lw      $t4, 0($t2)
addi    $t5, $t4, 9
sw      $t5, 4($t2)
```

d.
```
lui     $s0, 0x0027
ori     $s0, $s0, 0x3000
lh      $s2, 2($s0)
lh      $s3, 4($s0)
and     $s4, $s2, $s3
sw      $s3, 4($s0)
```

e.
```
slt     $s0, $t1, $t0
beq     $s0, $zero, L
add     $t0, $t0, $t1
L: add  $t3, $t0, $t0
```

Note: Indicate if the branch is taken.

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 $t1 holds the value of `var` and $t2 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;
}
```