Name: _____

CIS 351 Practice Test 3

Updated March 16, 2022

1. What is the role of registers on the CPU?
2. What is a computer's "word size"?
3. How does a computer's word size affect its design?
4. What are the values stored as `here` and `there` in the code below?

```
0x00400000 | addi $s0, $s1, 2
0x00400004 | here: beq $s0, $s2, there
0x00400008 | addi $s0, $s0, 1
0x0040000C | j here
0x00400010 | there: sub $t0, $t0, $t1
```

5. Review both questions from Homework 4. Make sure you are comfortable assembling or disassembling instructions by hand, not just by using an online tool.
6. Give an example of a change you could make to a microarchitecture that would not affect the implemented architecture (it does not need to be a particularly major or useful change).
7. Give an example of an instruction for which `RegDest` is `X` (i.e., "Don't care").
8. Modify the example MIPS datapath we have been working with to include the jump instruction. You may add additional control wires if needed.
9. Give the result of `0x0123 AND 0x0F04`.
10. Give the result of `0x0123 XOR 0x0F04`.
11. Convert the following line of Java code to assembly: `t0 = t1 + t2 + t3 - t4 + t5`
12. Convert the following line of Java code to assembly: `t0 = (t1 ^ t2) & (t3 | !t4)`
13. Convert the following Java code to assembly. Your answer *must* use `slt` and `beq` or `bne`. Using branching pseudoinstructions, such as `blt`, will result in partial credit.

```
if (t1 - 6 < t2) {
  t0 = t1;
} else {
   t0 = t2 + 4;
}
t1 = t1 + 7
```

14. Describe in common English what the following function does. Hint: It takes three parameters, all integers.

Name: _____

```
mysteryFunction1:
slt $t0, $a0, $a1
slt $t1, $a1, $a2
and $v0, $t0, $t1
jr $ra
```

15. Describe in common English what the following function does. Hint: It takes two integer parameters. `sra` stands for "shift right arithmetic". It moves all the bits in the register to the right the specified amount.

```
mysteryFunction2:
add $v0, $a0, $a1
sra $v0, $v0, 1
jr $ra
```

16. How many distinct opcodes are possible for MIPS instructions? What would change about MIPS insructions if we wanted to allow 100 distinct opcodes? What problems might this cause?