University of Virginia, Department of Computer Science
CS/ECE 3330: Computer Architecture, Fall 2010

Exam 1
September 27, 2010

Name: ____________________

Problem	Score	Possible
1		20
2		20
3		8
4		20
5		20
6		12
Bonus		5
Total		105

COMP 015 001 **Student Name:** ____________________

Student Username: ____________________

a b c d e f g h i j k l m n o p q r s t u v w x y z 0 1 2 3 4 5 6 7 8 9
a b c d e f g h i j k l m n o p q r s t u v w x y z 0 1 2 3 4 5 6 7 8 9
a b c d e f g h i j k l m n o p q r s t u v w x y z 0 1 2 3 4 5 6 7 8 9
a b c d e f g h i j k l m n o p q r s t u v w x y z 0 1 2 3 4 5 6 7 8 9
a b c d e f g h i j k l m n o p q r s t u v w x y z 0 1 2 3 4 5 6 7 8 9
a b c d e f g h i j k l m n o p q r s t u v w x y z 0 1 2 3 4 5 6 7 8 9
a b c d e f g h i j k l m n o p q r s t u v w x y z 0 1 2 3 4 5 6 7 8 9
a b c d e f g h i j k l m n o p q r s t u v w x y z 0 1 2 3 4 5 6 7 8 9

1. True or False (20 points)

T F Multicore architectures consist of multiple processor cores on a single chip

T F Google data centers consume roughly as much electricity as a typical household

T F Power is measured in joules; energy is measured in watts

T F MIPS is an example of an ISA

T F Compilers can convert high-level C programs into assembly programs

T F There are more registers than memory locations

T F A load operation takes data from memory and places it in a register

T F `sll $t1,$s3,6` is functionally equivalent to `mul $t1,$s3,64`

T F The encoded jump target is relative, the branch target is absolute

T F Leaf procedures do not call any other procedures

T F The loader operates every time a program is executed

T F The first argument to a callee is placed in the `$t0` register

T F The callee uses the "jump register" (`jr`) instruction to return to the caller

T F The program stack grows and shrinks while that program is being compiled

T F It is impossible to have overflow when adding a positive and negative number

T F The value 0.100×10^6 is in normalized form

T F The LO register stores the most significant 32-bits of the result of a multiply

T F When performing integer division, the remainder is initialized to the dividend

T F To produce a 32-bit quotient, you must iterate through the algorithm exactly 33 times

T F For single precision floating point, the exponent bias is always 1023.

2. Performance Analysis (20 pts)

a. (4 points) Consider the following values: 2, 5, 100.

(1) What is the geometric mean?

(2) What is the arithmetic mean?

(3) If the values 2, 5, 100 represent IPCs, which is the proper mean to use?

b. (8 points) Computer A has an overall CPI of 1.3 and can be run at a clock rate of 600MHz. Computer B has a CPI of 2.5 and can be run at a clock rate of 750 MHz. We have a particular program we wish to run. When compiled for computer A, this program has exactly 100,000 instructions. How many instructions would the program need to have when compiled for Computer B, in order for the two computers to have exactly the same execution time for this program?

c. (8 points) Suppose that we can improve the floating point instruction performance of machine by a factor of 15 (the same floating point instructions run 15 times faster on this new machine). What percent of the instructions must be floating point to achieve a Speedup of at least 4?

3. ISA: MIPS to/from C (8 pts)

a. (4 points) We began converting the following C code to MIPS assembly. Fill in the blanks. Assume a, b, c are in $s1, $s2, $s3.

```
if (a < b) c++
```

```
      slt _____,_____,_____
      beq _____,_____ ,L
      addi _____,_____,_____
L:
```

b. (4 points) We began converting the following C code to MIPS assembly. Fill in the blanks. Assume h is in $s2, and the base address of A is in $s3.

```
A[12] = h + A[8]
```

```
lw $t0,_____ (_____ )
add $t0,_____ , $t0
sw $t0,_____ (_____)
```

4. ISA: MIPS to/from Binary (20 pts)

a. (9 points) Consider the hex value 0xAE0BFFFC.

(1) What binary number does it represent?

(2) What MIPS instruction does it represent? (A copy of the tear-out card is shown on the last page of the exam.) Show the entire instruction, with all fields.

(3) What type (R-type, I-type, J-type) does it represent?

b. (5 points) Encode the following instruction. Show your answer as a word in hex.

```
and $t1, $t2, $t3
```

c. (6 points) Encode the branch/jump instructions below by filling in the TWO missing fields.

<table>
<tr><td><code>Loop: sll $t1, $s3, 2</code></td><td>40000</td><td>0</td><td>0</td><td>19</td><td>9</td><td>4</td><td>0</td></tr>
<tr><td><code>   add $t1, $t1, $s6</code></td><td>40004</td><td>0</td><td>9</td><td>22</td><td>9</td><td>0</td><td>32</td></tr>
<tr><td><code>   bne $t0, $s5, Exit</code></td><td>40008</td><td>5</td><td>8</td><td>21</td><td colspan="3"></td></tr>
<tr><td><code>   lw $t0, 0($t1)</code></td><td>40012</td><td>35</td><td>9</td><td>8</td><td colspan="3">0</td></tr>
<tr><td><code>   addi $s3, $s3, 1</code></td><td>40016</td><td>8</td><td>19</td><td>19</td><td colspan="3">1</td></tr>
<tr><td><code>   j   Loop</code></td><td>40020</td><td>2</td><td colspan="5"></td></tr>
<tr><td><code>Exit: …</code></td><td>40024</td><td colspan="6">0</td></tr>
</table>

5. Integer Arithmetic (20 pts)

a. **(8 points)** Perform the following operations by converting the operands to 2's complement binary numbers and then doing the addition or subtraction shown. Please show all work in binary, operating on 8-bit numbers.

(1) 3 + 13

Overflow?
Circle one: Y N

(2) 13 – 3

Overflow?
Circle one: Y N

b. **(12 points)** Perform the following operations using sign magnitude form. Please show all work in binary, operating on 8-bit numbers.

(1) (-10) + (-3)

Overflow?
Circle one: Y N

(2) 7 – 24

Overflow?
Circle one: Y N

(3) (-24) – (-3)

Overflow?
Circle one: Y N

6. Floating-Point (12 pts)

a. **(6 points)** Assuming single-precision IEEE 754 format, what decimal is represented by this word? (Hint: remember to use the biased form of the exponent)

`1 01111101 00100000000000000000000`

b. **(6 points)** Perform rounding on the following fractional binary numbers, use the bit positions in italics to determine rounding (use the rightmost 3 bits). Your result should have six bits to the right of the decimal point.

(1) Round toward positive infinity: `+0.100101`*`110`* $_{\text{binary}}$

(2) Round toward negative infinity: `-0.001111`*`001`* $_{\text{binary}}$

(3) Round toward zero: `+0.100101`*`000`* $_{\text{binary}}$

7. Bonus Points for Coming to Class (5 pts)

a. (1 points) How many machine instructions executed on Prof. Hazelwood's system when she ran a standard Hello World program in Linux?

b. (1 points) What happened during the YouTube video that we watched in class?

c. (2 points) During two different lectures, annoying noises disrupted class. Name the two distinct noises.

d. (1 points) Where was Prof. Hazelwood last week?

8. Optional Feedback (no credit)

Do you believe your performance on this exam will fairly and adequately reflect your understanding of the course material? If not, explain why.