CS30 Fall 2009 Midterm

Instructions
• Please turn off all pagers, cell phones and beepers. Remove all hats & headphones. Place your backpacks, laptops and jackets at the front.
• Sit in every other seat. Nothing may be placed in the “no fly zone” spare seat/desk between students.
• You have 75 minutes to complete this exam.
• This exam is closed book, closed notes, no computers, PDAs or calculators.
• WRITE YOUR NAME on EACH PAGE OF THIS TEST. You will be deducted 2 points otherwise.

<table>
<thead>
<tr>
<th>n</th>
<th>$2^n$</th>
<th>n</th>
<th>$2^n$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>17</td>
<td>131072</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>18</td>
<td>262144</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>19</td>
<td>524288</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>20</td>
<td>1048576</td>
</tr>
<tr>
<td>5</td>
<td>32</td>
<td>21</td>
<td>2097152</td>
</tr>
<tr>
<td>6</td>
<td>64</td>
<td>22</td>
<td>4194304</td>
</tr>
<tr>
<td>7</td>
<td>128</td>
<td>23</td>
<td>8388608</td>
</tr>
<tr>
<td>8</td>
<td>256</td>
<td>24</td>
<td>16777216</td>
</tr>
<tr>
<td>9</td>
<td>512</td>
<td>25</td>
<td>33554432</td>
</tr>
<tr>
<td>10</td>
<td>1024</td>
<td>26</td>
<td>67108864</td>
</tr>
<tr>
<td>11</td>
<td>2048</td>
<td>27</td>
<td>134217728</td>
</tr>
<tr>
<td>12</td>
<td>4096</td>
<td>28</td>
<td>268435456</td>
</tr>
<tr>
<td>13</td>
<td>8192</td>
<td>29</td>
<td>536870912</td>
</tr>
<tr>
<td>14</td>
<td>16384</td>
<td>30</td>
<td>1073741824</td>
</tr>
<tr>
<td>15</td>
<td>32768</td>
<td>31</td>
<td>2147483648</td>
</tr>
<tr>
<td>16</td>
<td>65536</td>
<td>32</td>
<td>4294967296</td>
</tr>
</tbody>
</table>
Important – please note: The MIPS instructions shown in this table are the ones that you must use on the entire exam. Do not use any instructions that are not in this table. If you use any instructions not listed below, you will lose points.

<table>
<thead>
<tr>
<th>Name</th>
<th>Syntax</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>add</td>
<td><code>add rd, rs, rt</code></td>
<td><code>rd=rs+rt</code></td>
</tr>
<tr>
<td>sub</td>
<td><code>sub rd, rs, rt</code></td>
<td><code>rd=rs-rt</code></td>
</tr>
<tr>
<td>and</td>
<td><code>and rd, rs, rt</code></td>
<td><code>rd=rs AND rt</code></td>
</tr>
<tr>
<td>or</td>
<td><code>or rd, rs, rt</code></td>
<td><code>rd=rs OR rt</code></td>
</tr>
<tr>
<td>sll</td>
<td><code>sll rd, rt, shamt</code></td>
<td><code>rd=logic shift rt left shamt bits</code></td>
</tr>
<tr>
<td>srl</td>
<td><code>srl rd, rt, shamt</code></td>
<td><code>rd=logic shift rt right shamt bits</code></td>
</tr>
<tr>
<td>sra</td>
<td><code>sra rd, rt, shamt</code></td>
<td><code>rd=arithmetic shift rt right shamt bits</code></td>
</tr>
<tr>
<td>slt</td>
<td><code>slt rd, rs, rt</code></td>
<td>If rs&lt;rt, rd=1, else rd=0</td>
</tr>
<tr>
<td>slti</td>
<td><code>slti rd, rs, imm</code></td>
<td>If rs&lt;imm, rd=1, else rd=0</td>
</tr>
<tr>
<td>jr</td>
<td><code>jr rs</code></td>
<td>jump to the instruction held by the memory location indicated by rs</td>
</tr>
<tr>
<td>addi</td>
<td><code>addi rt, rs, imm</code></td>
<td><code>rt=rs+imm</code></td>
</tr>
<tr>
<td>andi</td>
<td><code>andi rt, rs, imm</code></td>
<td><code>rt=rs AND imm</code></td>
</tr>
<tr>
<td>ori</td>
<td><code>ori rt, rs, imm</code></td>
<td><code>rt=rs OR imm</code></td>
</tr>
<tr>
<td>lw</td>
<td><code>lw rt, imm(rd)</code></td>
<td><code>rt=MEMORY[rd+imm]</code></td>
</tr>
<tr>
<td>sw</td>
<td><code>sw rt, imm(rd)</code></td>
<td><code>MEMORY[rd+imm]=rt</code></td>
</tr>
<tr>
<td>beq</td>
<td><code>beq rs, rt, label</code></td>
<td>Branch if rs==rt</td>
</tr>
<tr>
<td>bne</td>
<td><code>bne rs, rt, label</code></td>
<td>Branch if rs != rt</td>
</tr>
<tr>
<td>j</td>
<td><code>j label</code></td>
<td>Jump to label</td>
</tr>
<tr>
<td>jal</td>
<td><code>jal label</code></td>
<td>Jump to label and link</td>
</tr>
</tbody>
</table>
Part I: General concept of MIPS [5pt, 5 minutes]:

1. MIPS was invented in 1985 with 32 integer registers. According to Moore’s Law, named after Intel founder Gordon Moore, the number of transistors per microprocessor doubles every 1.5 years. Thus, microprocessors could have 1000 times the number of transistors in 2000 as they could in 1985. It would seem that we could easily build microprocessors with, say, 256 registers.

Select all the reasons why MIPS has not increased the number of integer registers from 32 to 256.

A) 32 registers are much smaller than 256 registers, and since smaller is faster, the 32 registers makes it easier to build fast microprocessors than if there were 256 registers.

B) There is no need for more than 32 registers, as compilers have difficulty using the 32 registers in the MIPS architecture now.

C) Moore’s Law applies to Intel microprocessors, not MIPS microprocessors, hence the hypothesis is false. The MIPS chip would be too expensive if it had 256 registers.

D) None of the above

Your Answer _________________________________________________

A
Part II: Number Representation
2. (10pt, 5 min) Assume a 16-bit system

Consider the following Hex pattern 0xFFFF
What is the (decimal) value of this bit pattern assuming that it is in one's complement format
a) -1
b) -32767
c) 32767
d) 0
e) 1
f) None of the above

Your Answer: d

What is the (decimal) value of this bit pattern assuming that it is in two's complement format
a) -1
b) -32767
c) 32767
d) 0
e) 1
f) None of the above

Your Answer: a

3. (5pt, 3min) Assume a 16-bit system, explain why one's complement format and sign and magnitude format have a “dirty zero.”

Both have 2 zeros;
we can list the zeros in each format:
one's complement: 0x0000, 0xFFFF
sign and magnitude: 0x0000, 0x8000
4. (10pt, 5min) Assume a 16-bit system
What is the largest positive number that can be represented by unsigned integer format? Write it in both decimal and hexadecimal forms.

\[ 2^{16} - 1 \]
\[ 0xFFFF \]

What are the largest positive number and smallest negative number that can be represented by two's complement format? Write them in both decimal and hexadecimal forms.

\[ 2^{15} - 1 \ (0xFFFF) \]
\[ -2^{15} \ (0x8000) \]

What are the largest positive number and smallest negative number that can be represented by sign and magnitude format? Write them in both decimal and hexadecimal forms.

\[ 2^{15} - 1 \ (0xFFFF) \]
\[ -2^{15} + 1 \ (0xFFFF) \]
Part III. Assembly Language

5. (10pt, 10min) Assume a 32-bit system with two's complement format
Register $a0 contains 0xFFFF0FFF
Register $a1 contains 0x10011111
Register $a2 contains 0x00FFFFFF
at the beginning of each of the following instructions. For each instruction, give the contents of the destination register in hexadecimal format

\[
\begin{align*}
&\text{add }$s0, $a0, $a1 & 0x10002110 \\
&\text{sub }$s0, $a0, $a1 & 0xEFFDFEEE \\
&\text{andi }$s0, $a0, 0xFF00 & 0x00000F00 \text{ note: andi uses zero extension} \\
&\text{or }$s0, $a0, $a2 & 0xFFFFFFFF \text{ note: addi uses sign extension} \\
&\text{addi }$s0, $a0, 0xF0 & 0xFFF0FEF \\
\end{align*}
\]

6. (5pt, 2min) Memory access
Which of the following addresses are valid memory addresses? Select all that apply:

a) 0x01000201
b) 0x00014430
c) 0x000B073C
d) 0x0E0D8846

Your Answer b, c
7. (10pt, 10min) Each part below specifies an instruction that could be defined to be a pseudo-instruction for a particular MIPS simulator. The effect of each instruction is described so there will be no confusion. You need to:

I. For each, specify whether it can be translated into a single MIPS native instruction by circling YES or NO. The native instructions are listed in the 2nd page of this exam.

II. Also for each write out the shortest sequence of native instructions that could be used to simulate the given instruction. Aside from the registered declared in the instruction, you can only use $at and $0.

A) copy $reg1, $reg2
Effect: The value in 'reg2' is copied into 'reg1'.

Yes  No

B) sbg $reg1,$reg2,label
Effect: 'reg1' gets the value (reg2-reg1*0.5). If this value is positive, branch to the given 'label'.

Yes  No

```assembly
add $reg1, $reg2, $0
```
8. (15pt, 15min) Implement the following function in MIPS, filling the blanks in Assembly (no need to add comments)

C program  

```c
int sum( int n ) {
    if ( n < 1 ) return 1;
    else return (n + sum(n - 1));
}
```

Corresponding MIPS program: note that the input n is assigned to $a0

```
sum:    sub    $sp, $sp, 4
        # no a native instruction
sw     $ra, 4($sp)   # save the return address on the stack
sw     $a0, 0($sp)   # save the argument on the stack
slt    $t0, $a0, 1
beq    $t0, $zero, ___
add    $v0, $zero, 1
        j return
else:    addi   $a0, $a0, -1
        jal    sum
lw     $a0, 0($sp)
lw     $ra, 4($sp)    # restore the return address
add    $v0, $a0, $v0
return: addi   $sp, $sp, 4     # release the save area on the stack
        jr    $ra     # return to main
```
Congratulations!

You have finished the Midterm. To ensure that you get credit for your hard work, WRITE YOUR NAME on EACH PAGE OF THIS TEST. You will be deducted -2 points you can not finish this task.