Homework #3 (150 pts)
Submit hard copy to me
You must show all work for full credit.
For #7, submit Logisim circuit (.circ) to Blackboard

1) CPU instruction cycle (20 pts)
   a. What are the five stages of the CPU instruction cycle? (5 pts)

   b. For each stage, write 1-2 sentences describing what happens. (15 pts)

2) What does RISC stand for? (5 pts)

3) What does CISC stand for? (5 pts)

4) Describe the difference between big-endian and little-endian. (5 pts)
5) Store the 4-byte hexadecimal number 0xABCDEF12: (10 pts)

a. using the big-endian convention. (5 pts)

<table>
<thead>
<tr>
<th>INDEX</th>
<th>BYTE</th>
</tr>
</thead>
<tbody>
<tr>
<td>2000</td>
<td></td>
</tr>
<tr>
<td>2001</td>
<td></td>
</tr>
<tr>
<td>2002</td>
<td></td>
</tr>
<tr>
<td>2003</td>
<td></td>
</tr>
</tbody>
</table>

b. using the little-endian convention. (5 pts)

<table>
<thead>
<tr>
<th>INDEX</th>
<th>BYTE</th>
</tr>
</thead>
<tbody>
<tr>
<td>2000</td>
<td></td>
</tr>
<tr>
<td>2001</td>
<td></td>
</tr>
<tr>
<td>2002</td>
<td></td>
</tr>
<tr>
<td>2003</td>
<td></td>
</tr>
</tbody>
</table>

6) Suppose a divide instruction takes 57 cycles to complete. What is the total execution time (in seconds) required to complete 43 divide instructions using a: (15 pts)

a. 1.6 kHz CPU? (5 pts)

b. 3.2 MHz CPU? (5 pts)

c. 4.8 GHz CPU? (5 pts)
7) (50 pts). Using *Logisim*, create a circuit that takes *any number between 0 and 3* as keyboard input and displays the corresponding decimal number (i.e., 0, 1, 2, or 3) on a 7-segment display. A few additional requirements:

a. When there is no input from the keyboard, the 7-segment display must be blank.
b. You **CANNOT** use a ready-made decoder (e.g., BDC) for the 7-segment display.
c. You **CANNOT** use the hex digit display.
d. Logic gates **CANNOT** have more than 2 inputs.
e. You are **NOT REQUIRED** to handle keyboard input other than 0, 1, 2, and 3.
   i. For example, it’s **OK** for the 7-segment display in your circuit to display ‘1’ for keyboard input ‘A’.

![Diagram of keyboard and 7-segment display]

**Hints:**

1) Experiment with the 7-segment display first.
   a. E.g., How do you control the “bottom left” segment?
2) Work incrementally.
3) Consider the bits representing the numbers, not the numbers themselves.
   a. What does Logisim’s keyboard output?
4) Build a truth table and write Boolean expressions for each number and segment.
5) Can a transistor be used as a switch?

8) (40 pts). Write C++ code that will take any unsigned integer between 0 and 65535 on standard input and output the 16-bit binary equivalent to standard output. If the input is less than 0 or greater than 65535, your code must output an error message.

a. Your code **must compile and run on the EECS Linux servers**
   i. (You’ll need to ssh into the servers).
b. Please **DO NOT COPY / PASTE** your code into Blackboard
   i. Get WinSCP to copy your code from EECS to your machine
c. **ONLY** submit the .cpp file, not the executable.
   i. Name your file <username>_hw3.cpp (e.g., mine is rubinma_hw3.cpp)
d. Use a command line editor to write your code (e.g., vim, nano)
e. To compile your code, run this command: “**g++ code.cpp**” (minus quotes)
f. To run your code, type “**./a.out**” (minus quotes)
g. Post questions about C++ coding, compiling, running, etc. to Piazza!
h. (This is a warm up for assembly programming in Linux).