UCR CS 122A
Prof. Frank Vahid
Fall 2004
Quiz 1

1. (C-level question, 3 points, id: cs122a_04fal_q1p1)
   What is combinational logic, and if you want a process to represent combinational logic, why must all read signals appear in the sensitivity list?

   Combinational logic is logic whose output is solely dependent on current inputs, without any notion of state.

   All read signals must appear in the sensitivity list of a process describing combinational logic because omitting a read signal will cause that signal to be latched, violating the definition of combinational logic.

2. (B-level question, 4 points, id: cs122a_04falq1p2)
   Write a VHDL entity and single-process architecture to represent combinational logic that outputs in binary the number of 1's appearing on 4 inputs I3, I2, I1, I0.

```
entity countOnes is
    port(
        I3, I2, I1, I0: in STD_LOGIC;
        output:         out STD_LOGIC_VECTOR(2 downto 0)
    );
end countOnes;

architecture bhv of countOnes is
begin
    process (I3, I2, I1, I0)
        variable count: STD_LOGIC_VECTOR(2 downto 0);
    begin
        count := "00";
        count = count + I3;
        count = count + I2;
        count = count + I1;
        count = count + I0;
        output <= count;
    end process;
end bhv;
```

3. (A-level question, 3 points: id: cs122a_04fal_q1p3) You already know that a combinational process must have all its read inputs in the process sensitivity list; must have loops that can be fully unrolled if loops exist; and must not have any wait statements.
Suppose the process writes to two outputs P and Q. If we want this process to represent combinational logic, then EVERY time that the process is activated, MUST **BOTH** P AND Q BE ASSIGNED A VALUE, or could only **ONE** of them be assigned? For example, would the following be an acceptable complete body of a combinational process? Provide an explanation as well as an example in your answer.

```
if (input0 = '0') then
   P <= '0'; Q <= '0';
end if;
if (input1 = '1') then
   P <= '1';
else
   Q <= '1';
end if;
```

The above code is not a valid way to represent combinational logic. There are sequences of input that can cause the output to rely on previous values. For example, if input0 and input1 are initially '0' and '1', respectively, P is assigned '1' and Q is assigned '0'. If input0 changes to '1' while input1 holds at '1', P is assigned '1' and Q retains its previous value of '0' (notice that no new assignment of Q occurs). Both P and Q must be assigned a value.

(Note: within a process, only the last signal assignment is "seen" outside of the process.)

Questions vary in difficulty
Question 1 tests whether one remembers some of our discussions in class.
Question 2 tests whether one has developed basic skill in VHDL coding.
Question 3 tests a deeper understanding of the relationship of VHDL processes to combinational logic. While the question was not directly covered in class, it follows directly from the concepts discussed in class.