Practice Questions for Exam 1 in CSCI 320

Chapter 1 Questions
(Introduction)

1. We looked at 4 generations of computers, namely the first, second, third and fourth. What technology distinguished each generation from the previous?

2. The von Neumann architecture, which is the basis for most digital computers today, suffers from the von Neumann bottleneck. Explain.

3. Explain why modern machines consist of multiple levels of virtual machines. Why not just have two levels, the digital logic level and the high-language programming level?

4. (True or False) To allow current flow through a nMOS transistor, the gate should be connected to ground?

5. What does semiconductor mean?

6. What is the meaning of Instruction Set Architecture (ISA)?

Chapter 2 Questions
(Data Representation)

1. Below is a table containing 5 Hex values. Complete the table by filling in the binary and base10 representations of each value. Assume that a 2’s complement representation is being used. After completing the table, answer the questions below.

<table>
<thead>
<tr>
<th>Base10</th>
<th>Hex</th>
<th>Binary</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>FF</td>
<td></td>
</tr>
<tr>
<td></td>
<td>07</td>
<td></td>
</tr>
<tr>
<td></td>
<td>F8</td>
<td></td>
</tr>
<tr>
<td></td>
<td>87</td>
<td></td>
</tr>
<tr>
<td></td>
<td>80</td>
<td></td>
</tr>
</tbody>
</table>

a) Which of the above 5 numbers is the largest 8-bit unsigned binary number?

b) Which of the above 5 numbers is equivalent to the same base 10 number when it is interpreted by a computer using signed-magnitude representation as opposed to a computer using 2’s complement representation?
c) Suppose the Hex value F8 is interpreted as a 1's complement number, would its base 10 and binary representations change? If so, what would they be?

d) Which of the above 5 numbers is the largest 8-bit 2's complement number?

e) Which of the above 5 numbers is the smallest 8-bit 2's complement number?

2. Add the following pairs of six-bit two's complement numbers and indicate which additions result in an overflow.

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>011101</td>
<td>001010</td>
</tr>
<tr>
<td>+ 101010</td>
<td>+ 011100</td>
</tr>
<tr>
<td>111111</td>
<td>001101</td>
</tr>
<tr>
<td>+ 011111</td>
<td>+ 001011</td>
</tr>
</tbody>
</table>

3. Subtract the following pairs of six-bit two's complement numbers and indicate which subtractions result in an overflow.

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>011101</td>
<td>001010</td>
</tr>
<tr>
<td>- 101010</td>
<td>- 011100</td>
</tr>
<tr>
<td>111111</td>
<td>001101</td>
</tr>
<tr>
<td>- 011111</td>
<td>- 001011</td>
</tr>
</tbody>
</table>

4. Assume we are using a simple model for floating-point representation that similar to the one suggested in your textbook: the representation uses a 14-bit format, 5 bits for the exponent with a bias of 16, a normalized mantissa of 8 bits with an implied 1 to the left of the radix pt, and a single sign bit for the number. Show how the computer would represent the numbers 96.5 and -0.75 using this floating-point format.

5. Given that the ASCII code for the character "A" is 1000001, the ASCII code for "H" would be:
   a. 1010101  b. 1110101  c. 1000100  d. 1001000  e. none of these
6. What is the minimum number of base-3 digits required to obtain at least as many combinations as can be done with 5 binary digits?

   a. eleven       b. three       c. four
   d. five         e. this can't be done

7. (True or False) The largest value that a 60-bit unsigned binary integer can represent is \(2^{60} - 1\).

**Chapter 3 Questions**  
(Digital Logic)

1. Prove that the 2-variable form of DeMorgan's Laws are valid.

2. For each component below, (1) Identify whether it is based on combinational logic (C) or sequential logic (S), and (2) briefly describe its functionality.
   a. SR latch  
   b. register  
   c. decoder  
   d. clocked D latch  
   e. ALU  
   f. memory  
   g. half-adder  
   h. a digital circuit to implement \(F(x,y,z) = x'y + xy'\)

3. A 16-input multiplexer can select between any one of 16 inputs and requires 16/2=8 "select" or control lines.

4. (True or False) It is possible to have two different truth tables that describe the same Boolean function.

5. Tyrone Shoelaces has invested a huge amount of money into the stock market and doesn't trust just anyone to give him buying and selling information. Before he will buy a certain stock, he must get input from three sources. His first source is Pain Webster, a famous stock broker. His second source is Meg A. Cash, a self-made millionaire in the stock market, and his third source is Madame LaZora, world famous psychic. After several months of receiving advice from all three, he has come to the following conclusions:
   a. Buy if Pain and Meg both say yes and the psychic says no.
   b. Buy if the psychic says yes.
   c. Don't buy otherwise.
Represent Tyrone’s algorithm in a (1) truth table, as a (2) Boolean algebra expression, and as a (3) schematic.

6. Give the truth table for the schematic shown below. The inputs are on the left, the output is on the right.

![Schematic Diagram]

7. Construct a FSM parity checker, first implement the parity checker as a Moore machine and then as a Mealy machine. The FSM receives a bit-stream (1 bit at a time) as input, and outputs “1” for odd parity and “0” for even parity. Each implementation should include the following:

- the state transition diagram
- the state transition table
- the state transition and output table(s) rewritten with binary encodings
- the Boolean algebra equations for the next state and output logic
- the circuit schematic

**Verilog**

Draw a schematic of the circuit defined in each Verilog module below.

1. module mystery1(input  a, b, c,
                      output    y);
   assign y = ~a & ~b | ~a & ~c | ~b & ~c;
endmodule

2. module mystery2(input  [3:0] a,
                     output [3:0] y);
assign y = ~a;
endmodule

3. module mystery3(input [3:0] d0, d1,
    input s,
    output [3:0] y);
    assign y = s ? d1 : d0;
endmodule

4. module mystery4(
    output co, s,
    input ci, a, b);
    wire a_xor_b;
    wire a_and_b;
    wire ci_and_a_xor_b;
    xor u1( a_xor_b, a, b );
    and u2( a_and_b, a, b );
    and u3( ci_and_a_xor_b, ci, a_xor_b );
    or  u4( co, a_and_b, ci_and_a_xor_b );
    xor u5( s, ci, a_xor_b );
endmodule

5. module mystery5(input clk,
    input reset,
    input [3:0] d,
    output reg [3:0] q);
    always @ (posedge clk, posedge reset)
      if (reset) q <= 4'b0;
      else q <= d;
endmodule