Problem Set 3
Due: 29 September, 2003 in class

Instructions:
The homework you turn in should be your own work. You may discuss problems
with your colleagues. But though you may share ideas and possible approaches to
solving problems, you may not share actual solutions.

Do not forget to write your TA's name and your discussion session time on
the homework you turn in. You may use the coversheet provided on the Problem
Sets page.

Please staple your homework.

Note: Numbers in parentheses are problem numbers from text.

1. Consider a shift register which can hold 3-bits at a time. New bits can be
   pushed into the register. Every time a bit is pushed in, the existing bits are
   shifted one position to the right and the rightmost bit is discarded. So if the
   register contained 011 and we push in a 1, the register will now contain 101.
   This register has 8 distinct states. Assuming that the register has 010 to begin
   with, draw a state transition diagram that describes the behavior of the
   register as we start shifting in bits.

2. (3.24)
   a. The figure below shows a logic circuit that appears in many of today's
      processors. Each of the boxes is a full-adder circuit. What does the value
      on the wire X do? That is, what is the difference in the output of this circuit
      if X=0 versus if X=1?
   b. Construct a logic diagram that implements an adder/subtractor. That is,
      the logic circuit will compute A+B or A-B depending on the value of X.
      Hint: Use the given logic diagram as a building block.
3. (3.26)
Recall that the adder was built with individual "slices" that produced a sum bit and carryout bit based on the two operands bits A and B and the carry in bit. We called such an element the full adder. Suppose we have a 3-to-8 decoder and two six-input OR gates, as shown in the figure below. Can we connect them so that we have a full adder? If so, please do. (Hint: If an input to an OR gate is not needed, we can simply put an input 0 on it and it will have no effect on anything. For example, refer to the figure.

4. (3.28)
Design a 2-bit by 2-bit unsigned binary multiplier. The multiplier takes two 2-bit inputs $A[1:0]$ and $B[1:0]$ and produces an output $Y$ which is the product of $A[1:0]$ and $B[1:0]$. The standard notation for this is:

a. What is the maximum value that can be represented in 2 bits for A (A[1:0])?
b. What is the maximum value that can be represented in 2 bits for B (B[1:0])?
c. What is the maximum possible value of Y?
d. What is the number of required bits to represent the maximum value of Y?
e. Write a truth table for the multiplier described above. You will have a four-input truth table with the inputs being A[1], A[0], B[1] and B[0].
f. Implement the third bit of the output, Y[2] from the truth table using only AND, OR and NOT gates.

5. (3.30) Refer to the following two figures for this problem.
A comparator circuit has two 1-bit inputs A and B and three 1-bit outputs G (greater), E (Equal), and L (less than).

a. Draw the truth table for a 1-bit comparator.

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>G</th>
<th>E</th>
<th>L</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

b. Implement G, E, and L using AND, OR, and NOT gates.
c. Using the 1-bit comparator as a building block, construct a four-bit equality checker, such that output EQUAL is 1 if A[3:0] = B[3:0], 0 otherwise.
6. \((3.34)\)
For the memory shown in the figure:

a. What is the address space?

b. What is the addressability?

c. What is the data at address 2?

7.
G1 and G2 are gated D latches. G1 is level sensitive while G2 is edge triggered. Let the outputs of G1 and G2 be Q1 and Q2 respectively. The waveforms shown in the figure below are applied to the data input and write enable lines of both latches. Draw the waveforms for the outputs Q1 and Q2.

<table>
<thead>
<tr>
<th>D</th>
<th>WE</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
8. An R-S latch can be implemented using two 2-input NAND gates as discussed in section 3.4.1. Complete the state transition diagram given below for an R-S latch. You may assume that S and R will not be set to 0 simultaneously.

![State Transition Diagram]

9. (3.44) Prove that the NAND gate, by itself, is logically complete (see Section 3.3.5) by constructing a logic circuit that performs the AND function, a logic circuit that performs the NOT function, and a logic circuit that performs the OR function. Use only NAND gates in these three logic circuits.