CSE 140 Midterm 1 Solution
Tajana Simunic Rosing
Spring 2010

- Do not start the exam until you are told to.
- Turn off any cell phones or pagers.
- Write your name and PID at the top of every page. Do not separate the pages.
- This is a closed-book, closed-notes, no-calculator exam. You may only refer to one page of your own notes.
- Do not look at anyone else’s exam. Do not talk to anyone but an exam proctor during the exam.
- If you have a question, raise your hand and an exam proctor will come to you.
- You have 80 minutes to finish the exam. When the time is finished, you must stop writing.
- Write your answers in the space provided.
- To get the most partial credit, clearly and neatly show all steps of your work.

<table>
<thead>
<tr>
<th></th>
<th>1. 15 points</th>
<th>2. 15 points</th>
<th>3. 20 points</th>
<th>4. 15 points</th>
<th>5. 15 points</th>
<th>6. 20 points</th>
<th>Total (100 points)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Problem 1
(15 points)
1) A single bit full subtractor has three inputs, $x$, $y$, $b_{in}$ and two outputs $d$, $b_{out}$. Its truth table is shown below.

$$
\begin{array}{cccc}
 x & y & b_{in} & d & b_{out} \\
 0 & 0 & 0 & 0 & 0 \\
 0 & 0 & 1 & 1 & 1 \\
 0 & 1 & 0 & 1 & 1 \\
 0 & 1 & 1 & 0 & 1 \\
 1 & 0 & 0 & 1 & 0 \\
 1 & 0 & 1 & 0 & 0 \\
 1 & 1 & 0 & 0 & 0 \\
 1 & 1 & 1 & 1 & 1 \\
\end{array}
$$

An incomplete gate-level design of a full subtractor is shown below. Please complete this design by filling in each of the dotted box with exactly one gate.
2) A 2-bit subtractor is constructed using the full subtractor you designed in Part (1). Its block-level diagram is shown below. Please add a minimum amount of logic to convert this 2-bit subtractor to a circuit that computes the **absolute value** of a 2-bit two’s complement number $A_1A_0$. 

![Block-level diagram of a 2-bit subtractor](image)
Problem 2
(15 points)

A PLA with only three rows is provided below. Is it possible to implement the following two functions in this PLA? If you answer is YES, show your implementation by marking the appropriate connections. If you think more rows are needed, add as many rows as necessary and draw your implementation. Minimize the number of rows in your implementation for full credit.

\[
F = a'c + cd' \\
G = acd' + bcd'
\]

Conver F = a’c + cd’ to F = a’c + acd’, share bcd’ with G
Problem 3
(15 points)

A 4-variable Kmap is shown below

1) Please list all the prime implicates of this Kmap

B’+C  A+C+D’  A+B+D’  A+B+C’  B+C’+D

2) Please list all the Essential prime implicates of this Kmap

B’+C  B+C’+D

3) What is the minimum product-of-sums implementation of this function?

(B’+C)(A+B+D’)(B+C’+D)

4) Is the minimum product-of-sums implementation of this function prone to static hazard? If Yes, eliminate the hazard by designing the minimum hazard-free product-of-sums implementation. If No, give a reasoning.

Yes, (B’+C)(A+B+D’)(B+C’+D)(A+C+D’)(A+B+C’)
Problem 4
(15 points)

1) Design multiplexer implementations for the following functions using the Karnaugh map method. Do using C as the data variable and A,B as the select variables.

\[ Z = f(A,B,C) = A'B + B'C + BC + AB'C' \]

Using C as the data variable and A,B as the select variables.
2) Implement the following using 2:4 decoder and any size OR gate.

\[ F(A, B, C) = A'C' + BC' \]

\[
F(A, B, C) = m_0 + m_1 + m_3
\]

\[ = C'(A' + B) = C' \]

no decoder needed.

\[ C \rightarrow \bigtriangleup \rightarrow F \]
Problem 5  
(15 points)

Draw the timing diagram for the following circuit. The D latch is level-sensitive (enabled when clock=1). The D flip-flop is negative-edge triggered. Assume both the latch and the flip-flop have zero delay and an initial state of 0.
Problem 6
(15 points)

There is 4-bit ALU with two inputs (A, B), input carry (C_in), select signals (S_1, S_0), output (O) and carry out (C_out). The ALU performs following arithmetic operations:

<table>
<thead>
<tr>
<th>S_1S_0</th>
<th>C_in = 0</th>
<th>C_in = 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>F = A + B (add)</td>
<td>F = A + B + 1</td>
</tr>
<tr>
<td>01</td>
<td>F = A (transfer)</td>
<td>F = A + 1 (increment)</td>
</tr>
<tr>
<td>10</td>
<td>F = B' (complement)</td>
<td>F = B' + 1 (negate)</td>
</tr>
<tr>
<td>11</td>
<td>F = A + B'</td>
<td>F = A + B' + 1 (subtract)</td>
</tr>
</tbody>
</table>

Implement the ALU using only multiplexers, one 4-bit adder and a minimum number of other gates if necessary.
| First Name: | Last Name: | PID: |