Homework #5

due: Thursday, March 10, 2011

- Write your name at the top of each page of your solutions.
- Clearly indicate the start of the solution for each problem.
- Properly order and staple all pages of your solution.
- Show all of your work!
- A solution that requires physical units is incorrect without them.
- Clearly identify your result (e.g. box or circle the result).
- Always write neatly! If it cannot be read, it will not be graded!

- Always read the associated sections of the textbook before attempting the homework problems!

1. Given the following logic function:

\[ F(A,B,C,D) = \Sigma m(4, 6, 7, 8, 9, 10) \]

and using AND and OR gates,

(a) Design a minimum two-level logic circuit to realize this logic function.

(b) Design a minimum three-level logic circuit to realize this logic function.
   (12 gate inputs minimum).

*Hint: For part (a) you must consider both minimum SOP and minimum POS. A similar consideration must be made for part (b).*

2. Given the following logic function:

\[ F(A,B,C,D) = \Pi M(0, 1, 2, 3, 4, 6, 7, 8, 9, 14, 15) \]

Design a minimum three-level NAND-gate (only) logic circuit to realize this logic function. (only four gates are required).
3. Given the following logic function:

\[ F(A,B,C,D) = \Sigma m(0, 1, 2, 3, 11, 13, 15) \]

Design a minimum three-level NOR-gate (only) logic circuit to realize this logic function. (only five gates are required).

4. Given the following logic functions:

\[ F(A,B,C,D) = \Sigma m(2, 3, 8, 9, 14, 15) \]
\[ G(A,B,C,D) = \Sigma m(0, 1, 5, 8, 9, 14, 15) \]

Design a minimum two-level OR-AND logic circuit to realize both logic functions. (a minimum solution has eight gates).

5. A combinational logic circuit has four inputs (A, B, C, D) and one output (F). The output is a logic 0 if and only if three or four of the inputs are 0; the output is a logic 1 otherwise.

(a) Determine the maxterm expansion for \( F(A,B,C,D) \).

(b) Using AND and OR gates, design a minimum three-level logic circuit to realize \( F \). (a minimum solution requires five gates and 12 gate inputs).

6. Using the Repeated-Division and Repeated-Multiplication processes, convert the following decimal number to binary:

\[ 347.3125_{10} \]  
(Show your work!)

7. Convert the following unsigned binary numbers to decimal, octal, and hexadecimal:

(a) \( 1011010011_2 \)

(b) \( 101011010_2 \)

(c) \( 1000010_2 \)
8. Convert the following decimal numbers to 8-bit Sign and Magnitude binary:
   (a) $112_{10}$
   (b) $-75_{10}$
   (c) $-106_{10}$
   (d) $139_{10}$

9. Convert the following decimal numbers to 8-bit 1's Complement binary:
   (a) $113_{10}$
   (b) $-75_{10}$
   (c) $-121_{10}$
   (d) $142_{10}$

10. Convert the following decimal numbers to 8-bit 2's Complement binary:
    (a) $119_{10}$
    (b) $-47_{10}$
    (c) $-106_{10}$
    (d) $154_{10}$
1. \( F(A, B, C, D) = \sum m(4, 6, 7, 8, 9, 10) \)

(a) \[
\begin{align*}
F = A'B'D' + A'BC + \\
AB'C' + AB'D' &. \\
\text{cost} = 5 \text{ gates} + 16 \text{ inputs}.
\end{align*}
\]

(b) \[
\begin{align*}
F = A'B(C+D') + \\
AB'(C'+D') &. \\
\text{cost} = 2 \times 2\text{-input OR} \\
& 2 \times 3\text{-input AND} \\
& 1 \times 2\text{-input OR} \\
& = 5 \text{ gates} + 12 \text{ inputs}.
\end{align*}
\]

The POS expression is the minimum 2-level logic circuit.

\[
\begin{align*}
F = (A+B)(A'+B') \\
(D'+B+C)(A'+C') &. \\
= (A+B)(A'+B') \\
(D'+A'B+B'C+A'C) &. \\
\text{cost} = 3 \times 2\text{-input AND} \\
& 2 \times 2\text{-input OR} \\
& 1 \times 4\text{-input OR} \\
& 1 \times 3\text{-input AND} \\
& = 7 \text{ gates} + 17 \text{ inputs}.
\end{align*}
\]
4.2. \( F(A,B,C,D) = \Pi M(0,1,2,3,4,6,7,8,9,14,15) \)

\[
\begin{array}{c|c|c|c|c|c|c|c|c|c|c}
\text{CD} & 00 & 01 & 10 & 11 & 00 & 01 & 10 & 11 & 00 & 01 & 10 & 11 \\
\hline
\text{AB} & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\
\text{ABC} & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 \\
\text{BC'D} & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 \\
\text{AB'C} & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 \\
\end{array}
\]

\[ F = ABC' + BC'D + AB'C \]

* Note: To design a NAND-only circuit, with 3 levels, we must start with an expression that describes an OR-AND-OR logic circuit (because an OR-AND-OR circuit can be easily converted to a NAND-only circuit). The output gate (at level 1) must be an OR gate.

\[ F = BC'(A+D) + AB'C \]
#3. \( F(A, B, C, D) = \sum m(0, 1, 2, 3, 11, 13, 15) \)

\[
\begin{array}{cccc}
& A & B & (A+B') \\
\hline
C & D & \text{CD} & \\
0 & 0 & 0 & 1 \quad 0 \quad 0 \quad 1 \\
0 & 1 & 1 & 0 \quad 0 \quad 0 \quad 1 \\
1 & 1 & 0 & 1 \quad 0 \quad 1 \quad 0 \\
1 & 0 & 1 & 1 \quad 0 \quad 1 \quad 0
\end{array}
\]

\((A'+B+C)\)  
\((A'+D)\)

*Note*: to design a 3-level **NOR**-only circuit, derive an expression that leads to an **AND-OR-AND** circuit first; this **AND-OR-AND** circuit can then be easily converted to a **NOR**-only circuit.

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

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

\[
= (A+B')(A'+BD+CD)
\]
4. \( F(A, B, C, D) = \Sigma_m(2, 3, 8, 9, 14, 15) \)
\( G(A, B, C, D) = \Sigma_m(0, 1, 5, 8, 9, 14, 15) \)

*note: a POS expression leads to a 2-level or-and circuit.*

*note: when simplifying functions \( F \) and \( G \), do not simplify the independently; rather, simplify them as a pair (even if \( F \) and \( G \) are not completely simplified).*

\[
F = (A+C)(A+B')(A'+B'+C)(A'+B+C')
\]
\[
G = (A+C')(B'+C+D)(A'+B'+C)(A'+B+C')
\]
#5. \( F(A, B, C, D) = 0 \) iff 3 or 4 of the inputs are 0;

\[ F = 1, \text{ otherwise.} \]

<table>
<thead>
<tr>
<th>#</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>9</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>10</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>11</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>12</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>13</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>14</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>15</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

**Truth Table**

\[ F(A, B, C, D) = \prod M(0, 1, 2, 4, 8) \]

Maxterm Expansion
(b) \[ F = \overline{A+C+D}(B+C+D)(A+B+D)(A+B+C) \]

**POS; OR-AND ckt; 5 gates, 16 inputs.**

\[ F = (AB + (C+D))(A+B + CD) \]

**3-level; AND-OR-AND ckt; 5 gates, 12 inputs.**
6. Convert 347.3125\textsubscript{10} to binary (base 2).

\[
\begin{align*}
347/2 & = 173, \quad \text{Rem} = 1 \quad (a_0) \\
173/2 & = 86 \quad \text{Rem} = 1 \\
86/2 & = 43 \quad \text{Rem} = 0 \\
43/2 & = 21 \quad \text{Rem} = 1 \\
21/2 & = 10 \quad \text{Rem} = 1 \\
10/2 & = 5 \quad \text{Rem} = 0 \\
5/2 & = 2 \quad \text{Rem} = 1 \\
2/2 & = 1 \quad \text{Rem} = 0 \\
1/2 & = 0 \quad \text{Rem} = 1 \\
\end{align*}
\]

\[
\begin{align*}
0.3125 \times 2 & = 0.625, \quad a_{-1} = 0 \\
0.625 \times 2 & = 1.250, \quad a_{-2} = 1 \\
0.250 \times 2 & = 0.500, \quad a_{-3} = 0 \\
0.500 \times 2 & = 1.000, \quad a_{-4} = 1 \\
\end{align*}
\]

\[
\therefore \quad 347.3125\textsubscript{10} = 101011011.0101\textsubscript{2}
\]
#7. (a) \(1011010011_2\)

- decimal: \(2^8 + 2^7 + 2^5 + 2^4 + 2^3 + 2^0 = 723_{10}\)
- octal: \(0010110010011 = 1323_8\)
- hexadecimal: \(0010110010011 = 2D3_{16}\)

(b) \(101011010_2\)

- decimal: \(2^8 + 2^6 + 2^4 + 2^3 + 2^1 = 346_{10}\)
- octal: \(101011010 = 532_8\)
- hexadecimal: \(000101011010 = 15A_{16}\)

(c) \(1000010_2\)

- decimal: \(2^5 + 2^1 = 64 + 2 = 66_{10}\)
- octal: \(001000010 = 102_8\)
- hexadecimal: \(01000010 = 42_{16}\)
#8. 8-bit Sign and Magnitude binary:

(a) \( 112_{10} = \overline{01110000}_S \text{ MAGNITUDE} \)

(b) \(-75_{10} \Rightarrow +75_{10} = 01001011 \)

\[ \therefore -75_{10} = \overline{11001011}_S \text{ MAGNITUDE} \]

(c) \(-106_{10} \Rightarrow +106_{10} = 01101010 \)

\[ \therefore -106_{10} = \overline{11101010} \]

(d) \(139_{10} = \text{ OUT OF RANGE}; \text{ Cannot be represented} \)

in 8-bit Sign and Magnitude binary.

Range: \(-127 \leq N \leq +127\).
9. 8-bit 1's Complement binary:

(a) \( 113_{10} = 0111 \, 0001 \)
    \[ \text{positive #} \]

(b) \(-75_{10} \Rightarrow +75_{10} = 0100 \, 1011 \) (from 8(b)).
    \[ \therefore -75_{10} = 1011 \, 0100 \]
    \[ \text{negative #} \]

(c) \(-121_{10} \Rightarrow +121_{10} = 0111 \, 1011 \)
    \[ \therefore -121_{10} = 1000 \, 0100 \]
    \[ \text{negative #} \]

(d) \(142_{10} = \text{OUT OF RANGE}; \) cannot be represented
    in 8-bit 1's Complement binary.

Range: \(-127 \leq N \leq +127\).
10. 8-bit 2's Complement binary:

(a) \(119_{10} = 0111\ 0111\ \uparrow\ \text{positive #.}\\

(b) \(-47_{10} \Rightarrow +47_{10} = 0010\ 1111\\
\begin{array}{c}
\hline
-4_{10} = 1101\ 0000 \\
+1 \\
\hline
1101\ 0001 \\
\uparrow\ \text{negative #.}\\
\end{array}

(c) \(-106_{10} \Rightarrow +106 = 0110\ 1010 \ (\text{from 8(c)}).\\
\begin{array}{c}
\hline
-106_{10} = 1001\ 0101 \\
+1 \\
\hline
1001\ 0110
\end{array}

(d) \ 154_{10} = \text{out of range; cannot be represented} \\
in 8-bit 2's Complement binary

range: \(-128 \leq N \leq +127\).