Homework #3  Computer Organization  
Due: Sept. 21, 2018 (Friday) by 3 PM

1. Recall that the sum-of-products (SOP) Boolean formula for the carry-out \((c_{i+1})\) of a \(n\)-bit adder was:

- 1-bit adder: \(c_{i+1} = x_i y_i + x_i c_i + y_i c_i\)
- 2-bit adder: \(c_{i+1} = x_i y_i + x_i (x_i x_i y_i + x_i x_i c_i) + y_i (x_i y_i + x_i c_i + y_i c_i)\)
- 3-bit adder: \(c_{i+1} = x_i y_i + x_i (7\ \text{product terms of 2-bit adder}) + y_i (7\ \text{product terms of 2-bit adder})\)
- 4-bit adder: \(c_{i+1} = x_i y_i + x_i (15\ \text{product terms of 3-bit adder}) + y_i (15\ \text{product terms of 3-bit adder})\)

a) Complete the following table showing the gate delays for different types of adders **assuming a 9-input limit into any gate.**

<table>
<thead>
<tr>
<th>Type of Adder</th>
<th># of product terms in SOP expression to be OR’ed</th>
<th>Most # of inputs in any of the product terms</th>
<th># gate delays due to product/AND terms</th>
<th># gate delays due to sum/OR terms</th>
<th>Gate Delay per Adder</th>
<th>Gate Delays for 32-bit ripple adder using adders of this type</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-bit</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>32 × 2 = 64</td>
</tr>
<tr>
<td>2-bit</td>
<td>7</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>16 × 2 = 32</td>
</tr>
<tr>
<td>3-bit</td>
<td>15</td>
<td>4</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>10 × 3 + 2 = 32</td>
</tr>
<tr>
<td>4-bit</td>
<td>31</td>
<td>5</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>8 × 3 = 24</td>
</tr>
<tr>
<td>5-bit</td>
<td>63</td>
<td>6</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>6 × 3 + 2 = 20</td>
</tr>
<tr>
<td>6-bit</td>
<td>127</td>
<td>7</td>
<td>1</td>
<td>3</td>
<td>4</td>
<td>5 × 3 + 2 = 16</td>
</tr>
<tr>
<td>7-bit</td>
<td>255</td>
<td>8</td>
<td>1</td>
<td>3</td>
<td>4</td>
<td>4 × 3 + 3 = 19</td>
</tr>
<tr>
<td>8-bit</td>
<td>511</td>
<td>9</td>
<td>1</td>
<td>3</td>
<td>4</td>
<td>4 × 4 + 1 = 16</td>
</tr>
<tr>
<td>9-bit</td>
<td>1,023</td>
<td>10</td>
<td>2</td>
<td>4</td>
<td>6</td>
<td>3 × 6 + 3 = 21</td>
</tr>
<tr>
<td>10-bit</td>
<td>2,047</td>
<td>11</td>
<td>2</td>
<td>4</td>
<td>6</td>
<td>3 × 6 + 7 = 20</td>
</tr>
<tr>
<td>11-bit</td>
<td>4,095</td>
<td>12</td>
<td>2</td>
<td>4</td>
<td>6</td>
<td>2 × 6 + 6 = 18</td>
</tr>
<tr>
<td>16-bit</td>
<td>131,072</td>
<td>17</td>
<td>2</td>
<td>6</td>
<td>8</td>
<td>2 × 8 = 16</td>
</tr>
</tbody>
</table>

b) Consider the SOP Boolean formula for the carry-out \((c_{i+1})\) of a 11-bit adder:

\[
x_i y_i x_{i-1} y_{i-1} \cdots x_{i-8} y_{i-8} x_{i-9} y_{i-9} x_{i-10} y_{i-10} \\
\]

\[
c_{i+1} \leftarrow c_{i-10}
\]

Give an example of each of the following:

i) a product term with only two terms \(x_{i-10} y_{i-10}\)

ii) a product term with 12 terms \(2^{11} \times 3\ \text{combining } i \text{s}\)

\[
x_{i-10} x_{i-10} x_{i-10} x_{i-7} x_{i-3} x_{i-4}
\]

HW #3 -1
3. Suppose we have a register file with the following specifications (see Memory Supplement in Lecture 8 on the eLearning system and in your course packet):

- 16 registers numbered from 0 to 15
- each register has 32-bits
- one write port
- two read ports

a) What how many bits ("wires") would be need for each of the following?
   - data to be written for a write port $32$
   - specifying the register number of a read port $4$
   - specifying the register number of a write port $4$
   - output read from a read port $32$

b) How many write enable wires would be needed for the whole register file?

c) How many decoders would be needed in the implementation of the whole register file? Explain how you arrived at that number and specify the type of decoders (i.e., # of inputs and # of outputs for each decoder)

   $$\text{one decoder that are } 4 \rightarrow 16$$

d) How many MUXs would be needed in the implementation of the whole register file? Explain how you arrived at that number and specify the type of MUXs (i.e., # of inputs and # of outputs for each MUX)

   $$\text{two MUX that are } 3 \rightarrow 16$$

4. Complete the below diagram of a 4-bit register so that it is able to perform the following operations:

- parallel read/output of all bits (just look at the Q values)
- Control the MUXs using the following codes:
  - $00_2$ - parallel write/load/input of all bits
  - $01_2$ - arithmetic shift right (sign-extend the most-significant bit)
  - $10_2$ - circular shift right two bit positions (bit(s) shifted out of least-significant bit is shifted into the most-significant bit)
  - $11_2$ - logical shift left one bit positions (bit(s) shifted out of most-significant bit are lost and a "0" is shifted into the least-significant bit)

   Note: For each D-flip flop, the output of a MUX is used as the D-input as shown below.