CSE 370 Homework 3 Solutions

1. Static Hazards

a) \[ F = AB + A'C' \]

Kmap of this function shown here; add logic for blue box to eliminate static 1 hazard

![Kmap of AB + A'C']

Static 1 Hazards:

\((A,B,C) = (1,1,0) \rightarrow (0,1,0)\)

New circuit is: \( F = AB + A'C' + BC' \)

![New Circuit](image)

b) \[ F = A'C + B'C' + ABD \]

Kmap for this function is shown below; red lines are already implemented in function, blue terms must be added to eliminate the 3 static 1 hazards

![Kmap of A'C + B'C' + ABD](image)
Static 1 Hazards:

(A,B,C,D) = (0,0,1,1) -> (0,0,0,1)
(A,B,C,D) = (0,0,1,0) -> (0,0,0,0)
(A,B,C,D) = (1,1,1,1) -> (0,1,1,1)
(A,B,C,D) = (1,1,0,1) -> (1,0,0,1)

New circuit is: \( F = A'C + B'C' + ABD + A'B' + A'BD + AC'D \)

Note: OR gates should be 1 6-input OR gate, but Visio only has 5-input OR gates
c) \[ F = (W+X+Y)(X'+Z') \]

Kmap for this function is shown below; red lines are already implemented in function, blue term must be added to eliminate the static 0 hazard

Static 0 Hazards:

\[(W,X,Y,Z) = (0,0,0,1) \rightarrow (0,1,0,1)\]

New circuit is:

2. **Timing Diagrams**

The non-oscillating steady state for this circuit can easily be found by assuming the node to the right of S is a 1 and tracing the resulting path. From S, you can find that B is a 1, and C is therefore a 0. D is then known to be a 1, and at the other end of the loop, A is of course the same value as the node to the right of S, a 1.
Note: for this timing diagram, assume that the steady state is at 0ps, and the switch is changed to the down position at t=10ns. Also, assume $T_{pd}=10\text{ns}$.

3. **2 bit adder**

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>X</th>
<th>Y</th>
<th>Z</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

$X = AC + ABD + BCD$

$Y = A'B'C + A'CD' + A'BC'D + AB'C' + AC'D' + ABCD$

$Z = BD' + B'D = B \oplus D$

To implement this circuit, you can use an 8 to 1 multiplexer for each output. Since some outputs (Y and Z) depend on more than 3 inputs (the number of select bits for an 8 to 1 mux), you can use the other input for the 2 bit adder as an input to the multiplexer. Many implementations are possible, one is shown here:
4. Full Adder using multiplexers

a) using 2 8-to-1 multiplexers

b) using 2 4-to-1 multiplexers

c) see solution for d)
d) using 5 2-to-1 MUXes

5. Decoders

a) \( f(P, Q, R) = (PQ + R) \) using 3:8 decoder
b) \[ f(P, Q, R, S, T) = (P+Q)S + (R+T)S' \] using 2 2:4 decoders

![2 to 4 Decoder Diagram]

Note: Only 1 6-input OR gate is necessary, but visio only has up to 5 inputs, so the OR gate is drawn this way

6. Implementation Methods

Note: Function can be reduced to \( F = AC + AD \)

a) 8:1 mux

![8 to 1 MUX Diagram]

b) 4:16 decoder
Note: Once again, visio does not have 6-input OR gates, so a tree of OR gates is used.

c) PLA-like structure
7. Rotate – assume this is supposed to implement a rotate right function (left is also acceptable), inputs are A[7:0] and R[2:0], outputs are B[7:0]