Multiple Function Minimization

\[ F_1 = \Sigma(11,12,13,14,15) = AB + ACD \]

\[ F_2 = \Sigma(3,7,11,12,13,15) = ABC' + CD \]

\[ F_3 = \Sigma(3,7,12,13,14,15) = A'CD + AB \]

Minimize separately

Implementation

\[ F_1 = \Sigma(11,12,13,14,15) = AB + ACD \]

\[ F_2 = \Sigma(3,7,11,12,13,15) = ABC' + CD \]

\[ F_3 = \Sigma(3,7,12,13,14,15) = A'CD + AB \]

Look for Shared Terms

\[ F_1 = \Sigma(11,12,13,14,15) = AB + ACD \]

\[ F_2 = \Sigma(3,7,11,12,13,15) = A'CD + A'CD + ABC' + ACD \]

Minimize separately
Implementation with Shared Terms

\[ F_1 = AB + ACD \]

\[ F_2 = A'CD + ABC' + ACD \]

\[ F_3 = A'CD + AB \]

Lab 6: SSN Decoder

- Create a four output combinational block that will recognize fields in your SSN
  - SSN is three fields: XXX - YY - ZZZZ
  - \( F_1 = 1 \) if input is equal to one of the numbers in the first group (XXX)
  - \( F_2 = 1 \) if input is equal to one of the numbers in the 2nd group (YY)
  - \( F_3 = 1 \) if input is equal to one of the numbers in the 3rd group (ZZZZ)
  - \( F_4 = 1 \) if input is equal to numbers in any group

Constraints

- Can only use 1 each of the following devices
  - 7400 (4 two-input NANDs)
  - 7402 (4 two-input NORs)
  - 7404 (6 Inverters)
  - 7408 (4 two-input AND)
  - 7410 (3 three-input NAND)
  - 7432 (4 two-input OR)
  - 7451 (AND-OR-INVERT function)
  - 7486 (4 two-input XOR gates)
What good is a 7451?

If a SOP equation has two product terms, with only two terms for each product term, then can implement with a 7451.

If minimize ZEROS on K-Map, then get an SOP form for $F'$. The inverter on the output of the 7451 will convert it to $F$!!!

OR, can minimize '1's, then put an external inverter on the output of the 7451 to get the high true version.

Example for SSN= 458 70 2198

<table>
<thead>
<tr>
<th>Row</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>F1</th>
<th>F2</th>
<th>F3</th>
<th>F4</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</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>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<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>7</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</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>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>9</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>10</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>11</td>
<td>1</td>
<td>0</td>
<td>1</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>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>13</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>14</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>15</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
First Try: Minimize each map, ignore shared terms

<table>
<thead>
<tr>
<th></th>
<th>00</th>
<th>01</th>
<th>11</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>C</td>
<td>0</td>
<td>X</td>
<td>0</td>
<td>X</td>
</tr>
<tr>
<td>D</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

\[ F_1 = B'C' + AD' \]

<table>
<thead>
<tr>
<th></th>
<th>00</th>
<th>01</th>
<th>11</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>B</td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>C</td>
<td></td>
<td>0</td>
<td>0</td>
<td>X</td>
</tr>
<tr>
<td>D</td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

\[ F_2 = BCD + A'B'C'D' \]

---

First Try: Minimize each map, ignore shared terms

<table>
<thead>
<tr>
<th></th>
<th>00</th>
<th>01</th>
<th>11</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>C</td>
<td>0</td>
<td>X</td>
<td>0</td>
<td>X</td>
</tr>
<tr>
<td>D</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

\[ F_3 = A + B'CD' + B'CD' \]

\[ F_4 = A + C' + BD + B'D' \]

But F4 is simply F1 + F2 + F3

---

An Implementation

Will try to implement F1, F2, F3 directly, then implement F4 = F1 + F2 + F3

Need four inverters for A', B', C', D'

\[ F_1 = B'C' + AD' \] (use 7451 + inverter, this the 5th inverter)

What about F2?

\[ F_2 = A'B'C'D' + BCD \]

Do NOT have a 4 input NAND gate????
F2 Implementation

Create a 4 input NAND gate from 2-input ANDs, inverter

\[ F_2 = A'B'C'D' + BCD \]

3 gates from '08, 1 from '10, 1 from '00

F3 Implementation

\[ F_3 = A + B'C'D + B'CD' \]

Cannot do this! Do not have enough '10 gates! Already used one!!!

F3 Implementation (cont)

\[ F_3 = A + B'C'D + B'CD' = A + B'(C'D + CD') = A + B'(C \oplus D) \]

Was able to use the XOR gate ('86)
F4 Implementation

\[ F_3 = F_1 + F_2 + F_3 \]

I used Nand-Nand implementation for SOP forms so that I could save the OR-gates in the '32 for this.

---

Final Gate Count

- All gates from the '04 (6 inverters)
- Part of the 7451
- Three gates from the '08
- One gate from the '10
- Three gates from the '10
- One gate from the '86
- Two gates from the '32

There are MOST certainly other solutions ….. Did not consider shared terms, did not minimize zeros. Many other avenues to try if these failed.

---

Debugging

- Wire up ONE function at a time
  - Start with whatever your simplest function is, then work towards hardest function.
  - No need to proceed to next function until current one works
- If F1, F2, F3 all work, then getting F4 to work will be trivial.
- The TAs do NOT know what the correct solution is for your SSN! They will only be able to offer general debugging help.