Combinational Circuit Analysis

- **Combinational circuit**: Output depends only on the current input values (called an input combination)
  - *Sequential circuit*’s output depends not only on its current input but also on the past sequence of inputs that have been applied to it.
    - I.e., a sequential circuit has memory of past events

- **Combinational circuit analysis**: we are given a logic diagram and need to find its formal description (truth table, logic expression)
Kinds of Combinational Analysis

- Exhaustive (truth table)
- Algebraic (expressions)
- Simulation / test bench (in the laboratory)

Exhaustive — Truth Table

Given:

Find truth table by all input combinations:

\[
\begin{array}{cccc}
X & Y & Z & F \\
000 & 011 & 111 & 011 \\
011 & 001 & 111 & 011 \\
111 & 101 & 001 & 011 \\
\end{array}
\]
Exhaustive — Truth Table

<table>
<thead>
<tr>
<th>X</th>
<th>Y</th>
<th>Z</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Find truth table by all input combinations:

Algebraic — Signal Expressions

- Use theorems to transform F into another form
- E.g., “multiplying out”:

\[ F = ((X+Y')Z) + (X'YZ') \]

\[ = (XZ) + (Y'Z) + (X'YZ') \]
Algebraic — Signal Expressions

...and obtain a new circuit but the same function:

\[ F = ((X+Y') \cdot Z) + (X' \cdot Y' \cdot Z') \]

Two-level AND-OR circuit

"Add out" Logic Function

"Add out" logic function is OR-AND circuit:

\[ F = ((X + Y') \cdot Z) + (X' \cdot Y \cdot Z') \]

\[ = (X + Y' + X) \cdot (X + Y + Y') \cdot (X + Y' + Z') \cdot (Z + X') \cdot (Z + Y) \cdot (Z + Z') \]

\[ = 1 \cdot 1 \cdot (X + Y' + Z) \cdot (X' + Z) \cdot (Y + Z) \]

\[ \leftarrow \text{two-level OR-AND circuit} \]

- Two-level OR-AND circuit:

\[ F = (X + Y' + Z') \cdot (X' + Z) \cdot (Y + Z) \]
Another Example

\[ G(W, X, Y, Z) = W \cdot X \cdot Y + Y \cdot Z \]

\[
\begin{align*}
\text{two-level AND-OR} & \quad \text{two-level NAND-NAND} \\
\end{align*}
\]

with 2-input gates only

Yet Another Example (1)

using NAND and NOR gates:

\[
F = \left[ (W \cdot X)' \cdot Y' + (W' + X + Y)' + (W + Z)' \right]
\]
[RECALL from Lecture #4] DeMorgan Symbols

\[ X + Y \quad \text{OR} \quad (X' \cdot Y')' \]

\[ (X + Y)' \quad \text{NOR} \quad X' \cdot Y' \]

\[ X \cdot Y \quad \text{AND} \quad (X' + Y')' \]

\[ (X \cdot Y)' \quad \text{NAND} \quad X' + Y' \]

\[ X \quad \text{BUFFER} \quad (X')' \]

\[ X \quad \text{INVERTER} \quad X' \]

---

Yet Another Example (1)

using NAND and NOR gates:

\[ F = \left[ (W \cdot X)' \cdot Y \right] \cdot (W + Z) \]

after substitution of some NAND and NOR gates:

\[ F = (W' \cdot X) \cdot (W' \cdot X + Y)' \cdot (W + Z) \]

...same function, according to the generalized DeMorgan's theorem
Yet Another Example (2)

different circuit but the same function:

\[
W \cdot X + Y = \left( (W' \cdot X) \cdot Y \right) \cdot \left( W' + X + Y' \right) \cdot (W + Z)
\]

here, majority are AND and OR gates