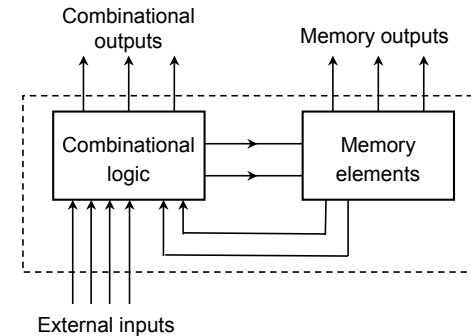


Topic #3 – Combinational Circuit Analysis

- **Combinational logic circuits:**
 - Outputs depend only on its current inputs.
 - No feedback loops -
 - A connection from the output of one gate to propagate back into the input of the same gate
 - Described using Boolean expressions and/or truth tables.
- **Sequential logic circuits:**
 - Outputs depend not only on the current inputs but also on the past sequence of inputs.
 - Contain combinational logic as well as memory elements formed with feedback loops.
 - Described using state transition tables and diagrams.

S. Jay Yang, CE, RIT

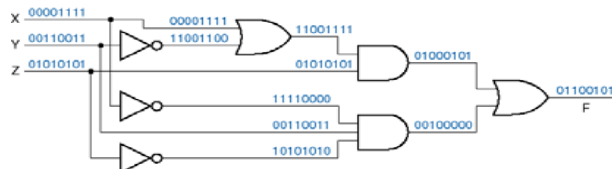
Combinational vs. Sequential Logic



S. Jay Yang, CE, RIT

What is combinational circuit analysis?

- Recall the representations of digital logic ...
 - Truth table
 - Digital circuit diagram
 - Boolean function ...
- Given a circuit diagram, determine the corresponding Boolean expression, and simplify it if possible.
- Example: find the logic expression of F ...

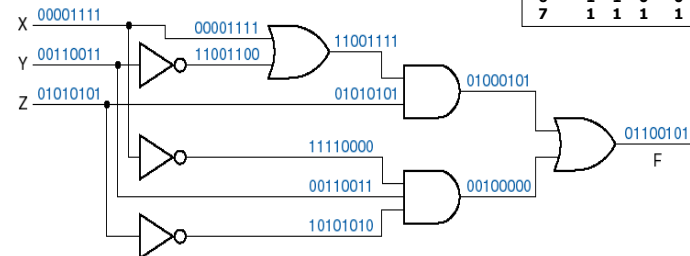


J, CE, RIT

Approaches

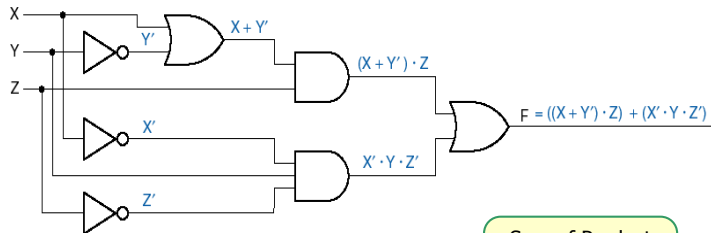
- Determine the outputs of each gate one by one.
- Or ... generate the truth table and find the canonical form, hence the SoP or PoS expression.

Truth table					
Row	X	Y	Z	F	
0	0	0	0	0	
1	0	0	1	1	
2	0	1	0	1	
3	0	1	1	0	
4	1	0	0	0	
5	1	0	1	1	
6	1	1	0	0	
7	1	1	1	1	



S. Jay Yang, CE, RIT

Let's get the outputs one by one



- Multiply out:

$$F = ((X + Y') \cdot Z) + (X' \cdot Y \cdot Z')$$

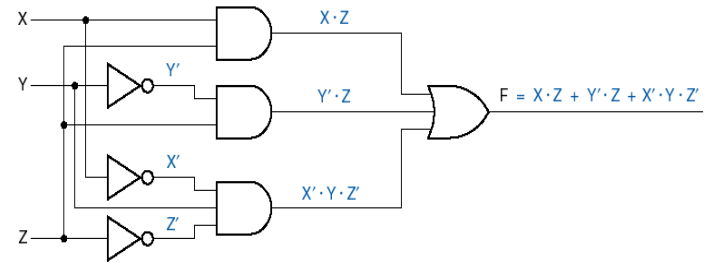
$$= (X \cdot Z) + (Y' \cdot Z) + (X' \cdot Y \cdot Z')$$

- Same function, different circuit?

Sum of Product
⇒ Canonical Sum
representation

S. Jay Yang, CE, RIT

New circuit, same function



- And we can even get another one ...
 - How about the trick we use to change the SoP to PoS?

S. Jay Yang, CE, RIT

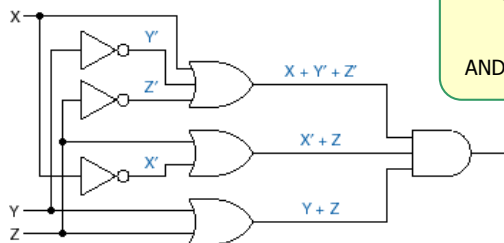
"Add out" logic function

$$F = ((X+Y') \cdot Z) + (X' \cdot Y \cdot Z')$$

$$= (X+Y'+X') \cdot (X+Y'+Y) \cdot (X+Y'+Z) \cdot (Z+X') \cdot (Z+Y) \cdot (Z+Z')$$

$$= 1 \cdot 1 \cdot (X+Y'+Z) \cdot (X'+Z) \cdot (Y+Z) \cdot 1$$

$$= (X+Y'+Z) \cdot (X'+Z) \cdot (Y+Z)$$

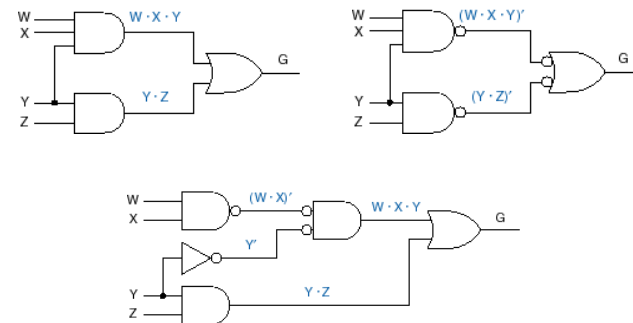


SoP ⇔ PoS
⇕ ⇕
AND-OR ⇔ OR-AND

S. Jay Yang, CE, RIT

Another example

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



S. Jay Yang, CE, RIT

Which circuit is better? SoP, PoS, or ...?

- Recall at the transistor level (e.g., TTL), NAND/NOR have fewer transistor than AND/OR.
- Recall the DeMorgan's theorem
 - Convert SoP to NAND-NAND

$$F = A \cdot B + C \cdot D$$

$$= ((A \cdot B)') + ((C \cdot D)')$$
 (T4)

$$= ((A \cdot B)' \cdot (C \cdot D)')$$
 (DeMorgan's theorem T13)
 - Convert PoS to NOR-NOR

$$F = (A + B) \cdot (C + D)$$

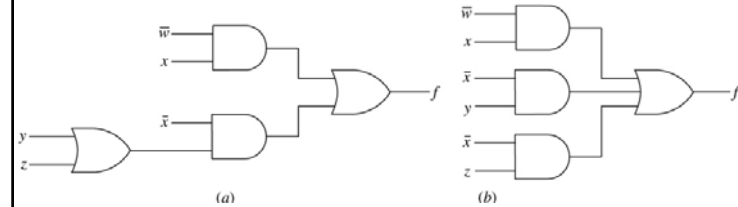
$$= ((A + B)')' \cdot ((C + D)')$$
 (T4)

$$= ((A + B)' + (C + D)')$$
 (DeMorgan's theorem T13')

S. Jay Yang, CE, RIT

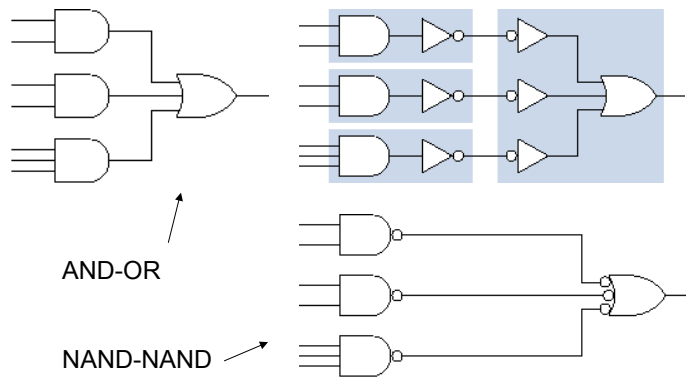
2-level vs. 3-level circuit to realize same function

(a) $f(w,x,y,z) = \bar{w}x + \bar{x}(y + z)$. (b) $f(w,x,y,z) = \bar{w}x + \bar{x}y + \bar{x}z$.



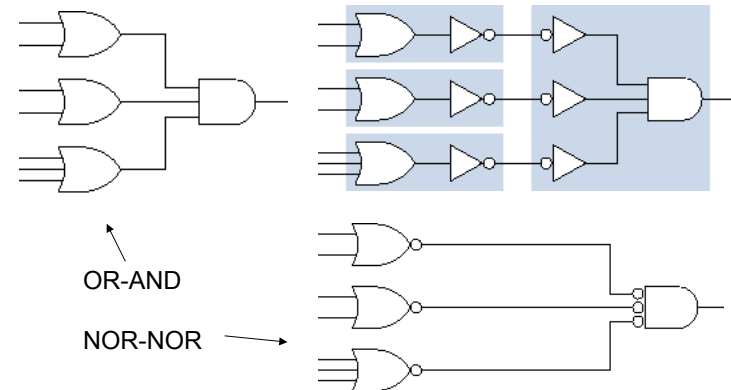
S. Jay Yang, CE, RIT

Realizing SoP using NAND-NAND



S. Jay Yang, CE, RIT

Realizing PoS using NOR-NOR



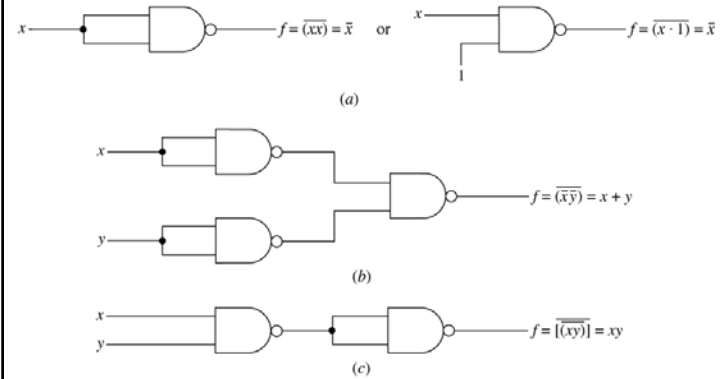
S. Jay Yang, CE, RIT

What we have learned?

- Deriving Boolean functions from digital circuits
- Simplify Boolean functions using the switching algebra theorems
 - (T10,T10'), (T13,T13') DeMorgan's theorem, ...
- Construct alternative circuits based on simplified Boolean functions
 - AND-OR, OR-AND, and others
- Convert AND-OR to NAND-NAND, & OR-AND to NOR-NOR

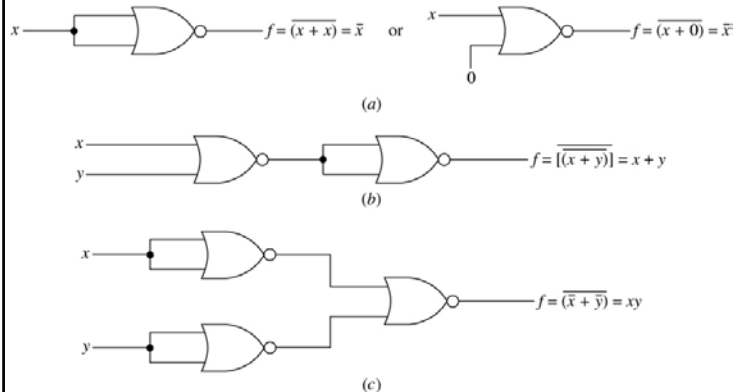
S. Jay Yang, CE, RIT

Self-Sufficient NAND: NOT, OR, AND



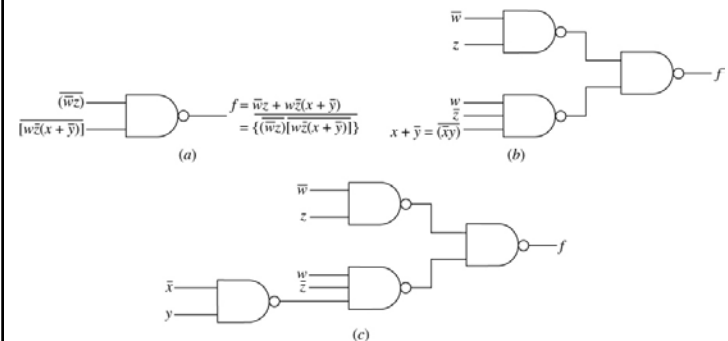
S. Jay Yang, CE, RIT

Self-Sufficient NOR: NOT, OR, AND



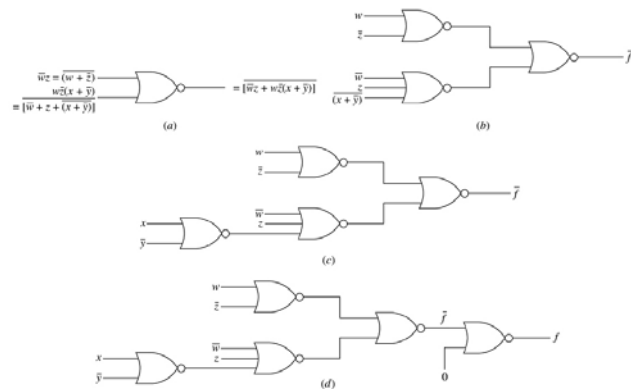
S. Jay Yang, CE, RIT

Realize logic function using NAND



S. Jay Yang, CE, RIT

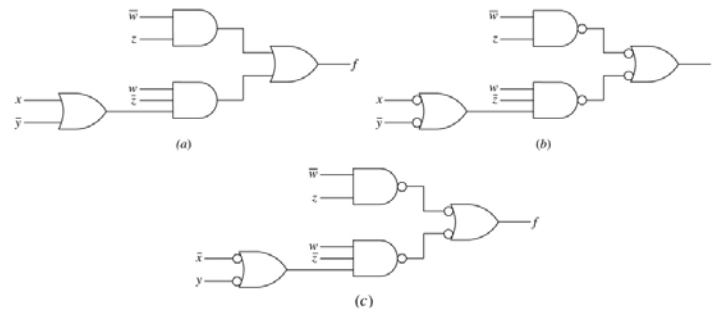
Realize logic function using NOR



S. Jay Yang, CE, RIT

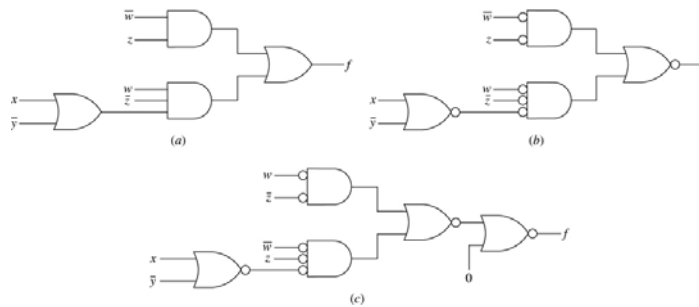
From OR-AND-OR to NAND-NAND-NAND

$$f(w,x,y,z) = w' \cdot z + w \cdot z' \cdot (x + y')$$



S. Jay Yang, CE, RIT

From OR-AND-OR to NOR-NOR-NOR



S. Jay Yang, CE, RIT