Digital Logic Design

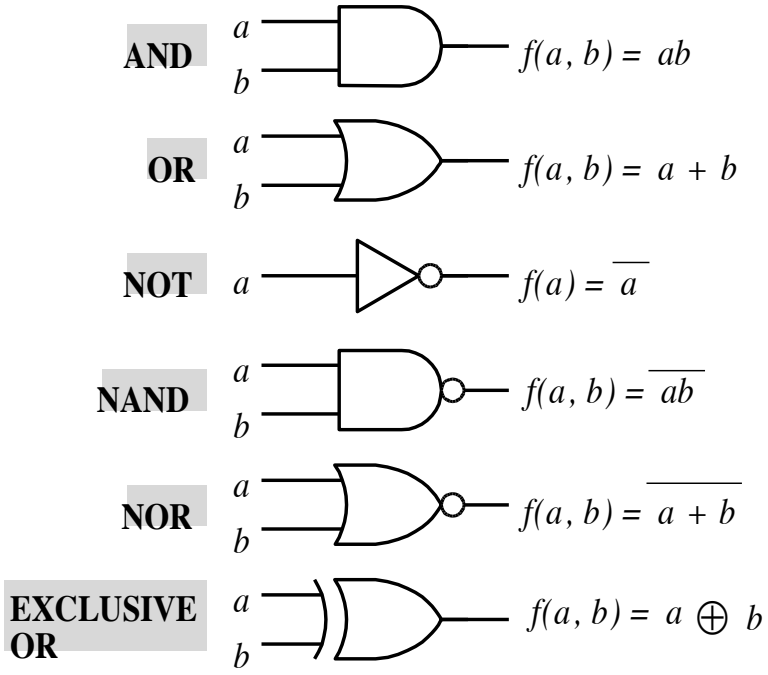
Hajar Falahati

Department of Computer Engineering
IRAN University of Science and Technology

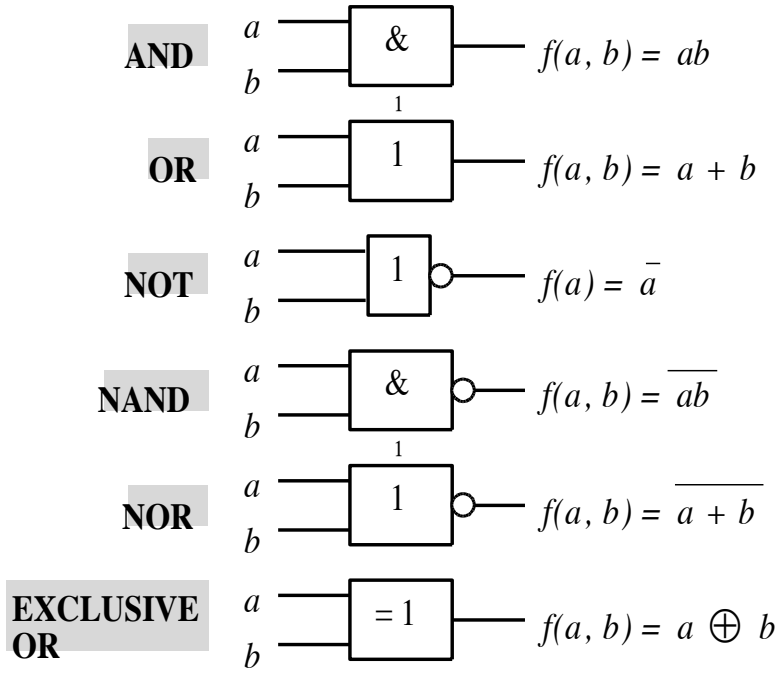
hfalahati@iust.ac.ir

Electronic Logic Gates

- Logic gates



Symbol set 1



Symbol set 2
(ANSI/IEEE Standard 91-1984)

Outline

- Digital Circuit Analysis

Digital circuit Analysis

Digital Circuit Analysis

- Digital Circuit **Design**:
  - Word description of a function

    $\Rightarrow$ A set of switching equations

    $\Rightarrow$ Hardware realization (gates, programmable logic devices, etc.)

- Digital Circuit **Analysis**:
  - Hardware realization

    $\Rightarrow$ Switching expressions, truth tables, timing diagrams, etc.

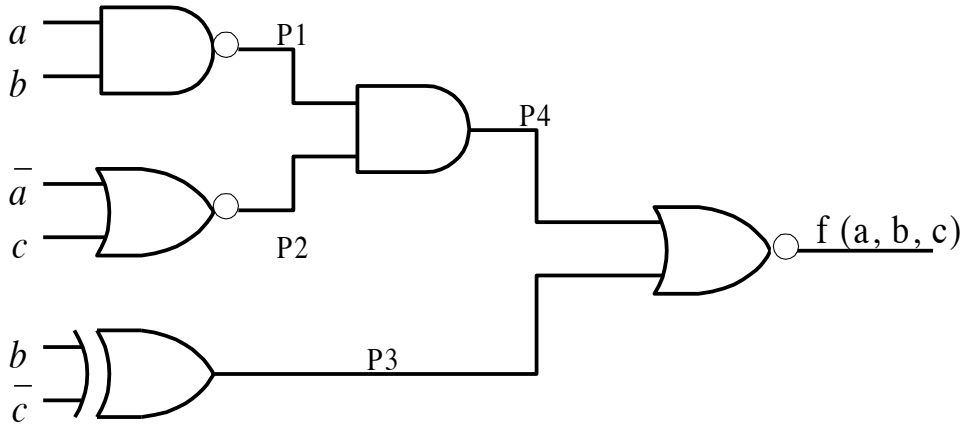
- Analysis is used
  - To determine the behavior of the circuit
  - To verify the correctness of the circuit
  - To assist in converting the circuit to a different form

Digital Circuit Analysis Methods

- Algebraic method
  - Use switching algebra to derive a desired form
- Truth table method
  - Derive the truth table one gate at a time
- Timing diagram
  - Graphical representation of input and output signal relationships over the time dimension.
  - May show intermediate signals and propagation delays

Sample 1:

- Find a simplified switching expression and logic network



(a)

Sample 1: Switching Expression

- Write switching expression for each gate output:

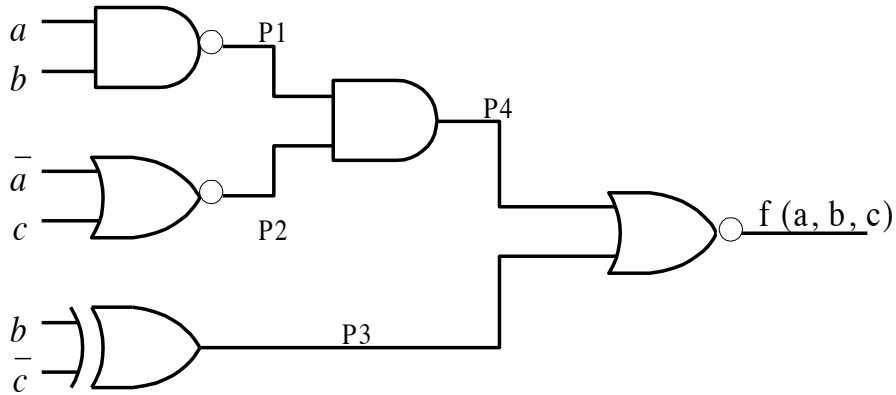
$$P_1 = \overline{ab},$$

$$P_2 = \overline{(\bar{a} + c)}$$

$$P_3 = b \oplus \bar{c},$$

$$P_4 = P_1 \cdot P_2 = \overline{ab} \cdot \overline{(\bar{a} + c)}$$

$$f(a,b,c) = \overline{P_3 + P_4} = \overline{(b \oplus \bar{c}) + \overline{ab} \cdot \overline{(\bar{a} + c)}}$$



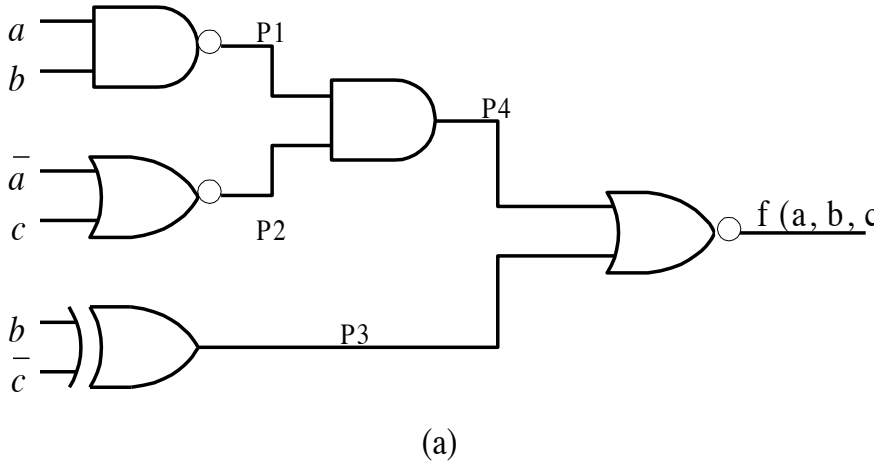
(a)

Sample 1: Simplification

- Simplify the output function using switching algebra:

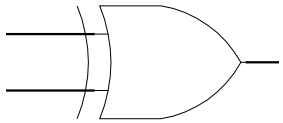
$$f(a,b,c) = \overline{P_3 + P_4} = \overline{(b \oplus \bar{c}) + \overline{ab} \cdot \overline{(\bar{a} + c)}}$$

$$\begin{aligned}\bar{f}(a,b,c) &= (b \oplus \bar{c}) + \overline{ab} \cdot \overline{\bar{a} + c} \\ &= bc + \bar{b}\bar{c} + \overline{ab} \cdot \overline{\bar{a} + c} \\ &= bc + \bar{b}\bar{c} + (\bar{a} + \bar{b})a\bar{c} \\ &= bc + \bar{b}\bar{c} + a\bar{b}\bar{c} \\ &= bc + \bar{b}\bar{c}\end{aligned}$$



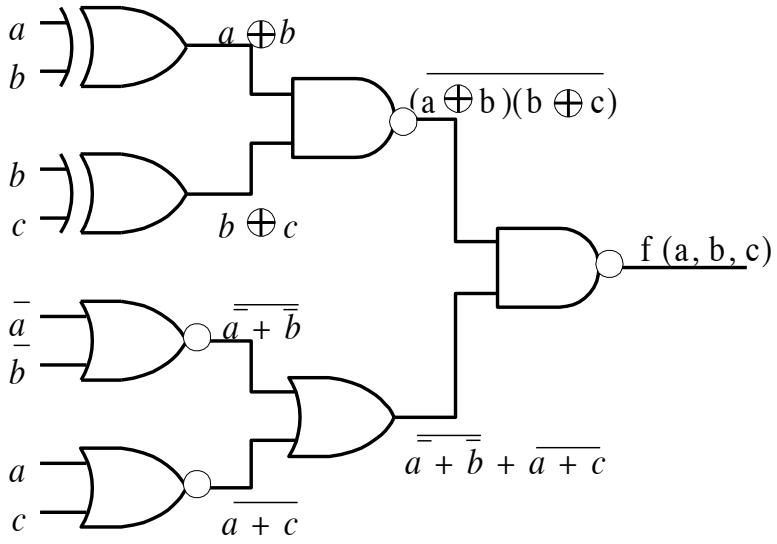
(a)

$$f(a,b,c) = \overline{(b \odot c)} = b \oplus c$$



Sample 2

- Find a simplified switching expressions and logic network.



Sample 2: Algebraic

- Write switching expression for each gate output:

$$\overline{\overline{(a \oplus b)(b \oplus c)} \cdot (\overline{\bar{a} + \bar{\bar{b}} + \overline{a + c}})}$$

$$\overline{(a \oplus b)(b \oplus c)} + \overline{\bar{a} + \bar{\bar{b}} + \overline{a + c}}$$

$$(a \oplus b)(b \oplus c) + (\bar{a} + \bar{b})(a + c)$$

$$(a\bar{b} + \bar{a}b)(b\bar{c} + \bar{b}c) + (\bar{a} + \bar{b})(a + c)$$

$$a\bar{b}b\bar{c} + a\bar{b}\bar{b}c + \bar{a}bb\bar{c} + \bar{a}b\bar{b}c + \bar{a}a + \bar{a}c + a\bar{b} + \bar{b}c$$

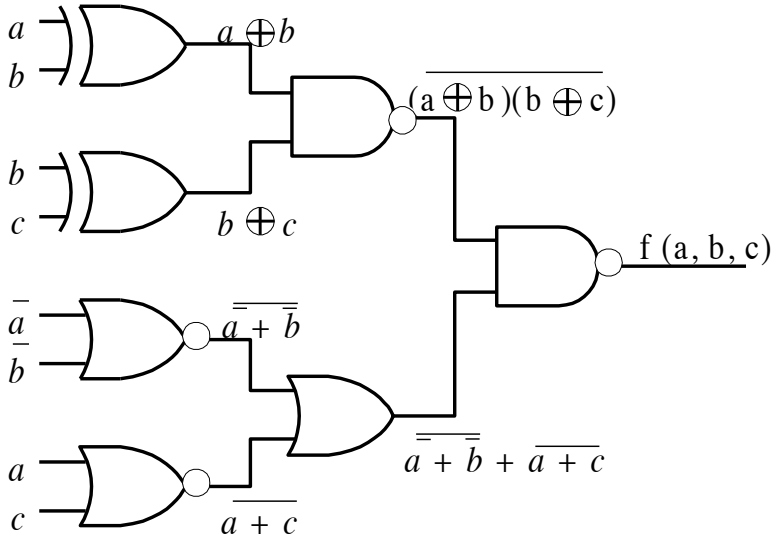
$$a\bar{b}c + \bar{a}b\bar{c} + \bar{a}c + a\bar{b} + \bar{b}c$$

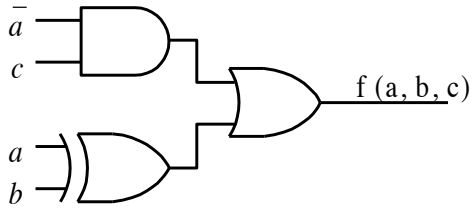
$$\bar{a}b\bar{c} + \bar{a}c + a\bar{b} + \bar{b}c$$

$$\bar{a}b\bar{c} + \bar{a}c + a\bar{b}$$

$$\bar{a}b + \bar{a}c + a\bar{b}$$

$$\bar{a}c + a \oplus b$$



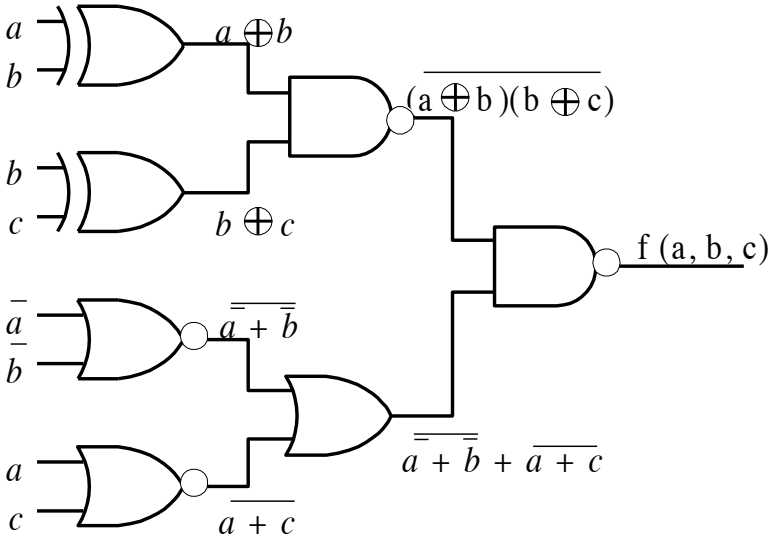


Simplified circuit

Sample 2: Truth Table

- Find a simplified switching expressions and logic network.

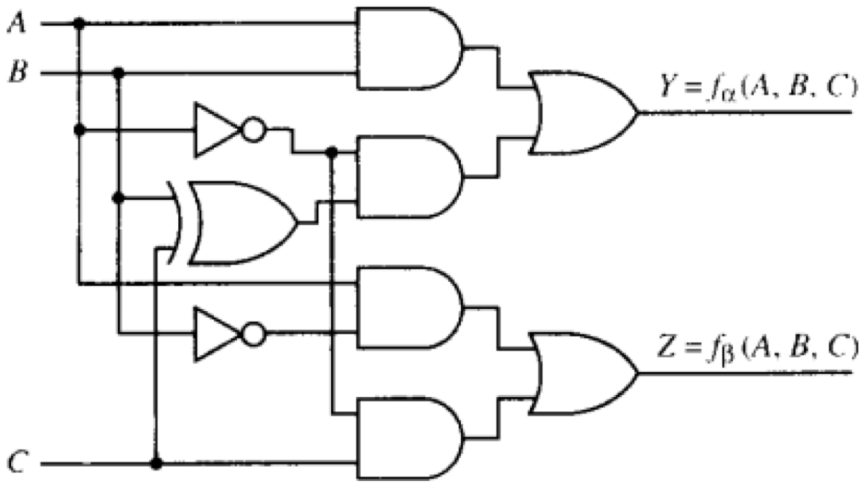
$$\bar{a}c + a \oplus b$$



abc	$\bar{a}c$	$a \oplus b$	$f(a,b,c)$
000	0	0	0
001	1	0	1
010	0	1	1
011	1	1	1
100	0	1	1
101	0	1	1
110	0	0	0
111	0	0	0

Sample 3

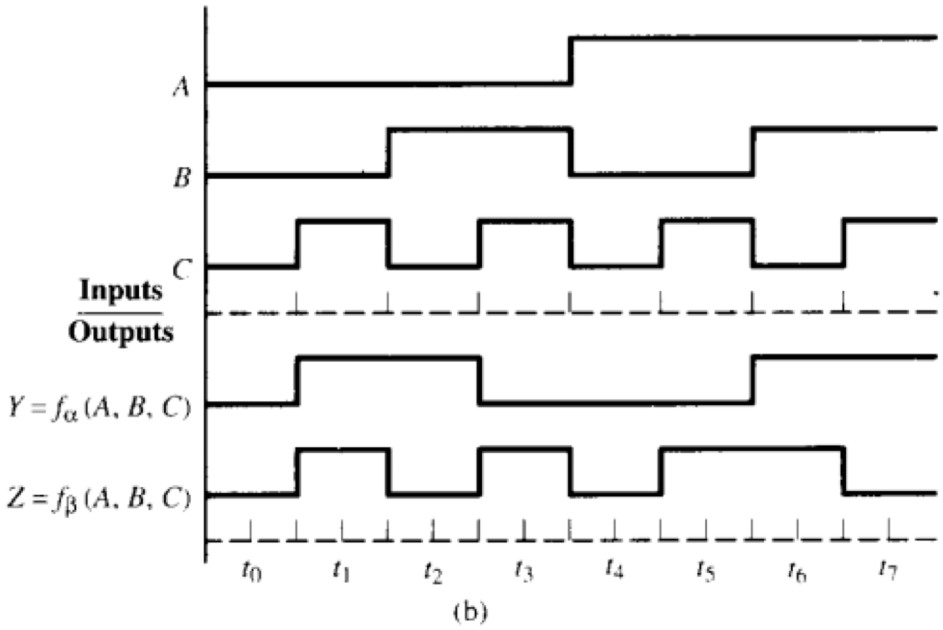
- Derivation of truth table from a timing diagram



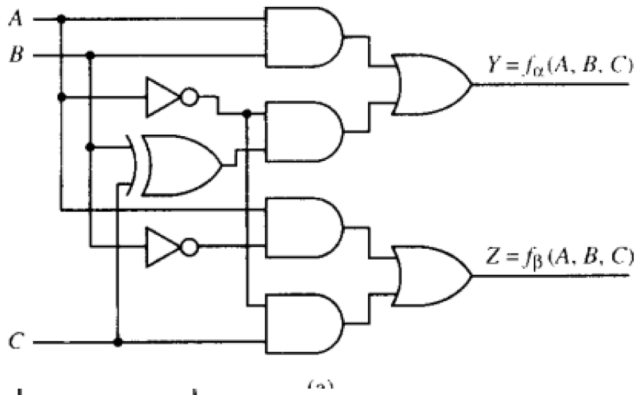
(a)

Sample 3: Timing Diagram

- (b) timing diagram,
- (c) truth table



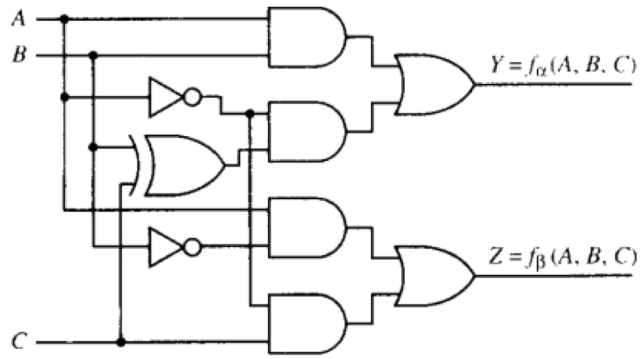
(b)



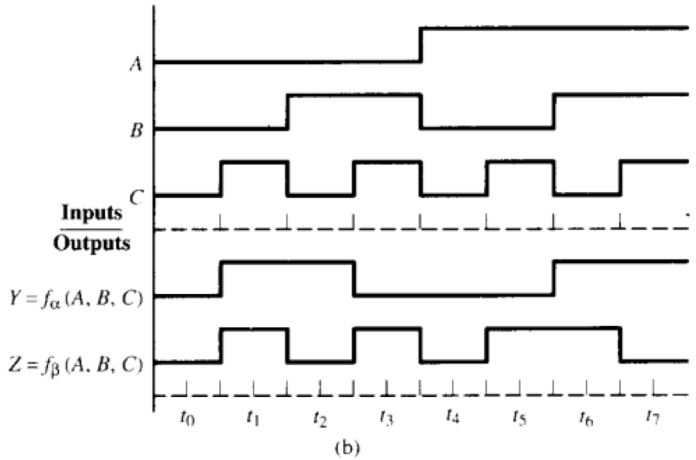
Time	Inputs ABC	Outputs $f_\alpha(A, B, C)$	Outputs $f_\beta(A, B, C)$
t_0	0 0 0	0	0
t_1	0 0 1	1	1
t_2	0 1 0	1	0
t_3	0 1 1	0	1
t_4	1 0 0	0	0
t_5	1 0 1	0	1
t_6	1 1 0	1	1
t_7	1 1 1	1	0

(c)

Sample 3: SOP, POS



(a)



(b)

Time	Inputs ABC	Outputs $f_\alpha(A, B, C)$	$f_\beta(A, B, C)$
t_0	0 0 0	0	0
t_1	0 0 1	1	1
t_2	0 1 0	1	0
t_3	0 1 1	0	1
t_4	1 0 0	0	0
t_5	1 0 1	0	1
t_6	1 1 0	1	1
t_7	1 1 1	1	0

(c)

$$f_\alpha(A, B, C) = \sum m(1, 2, 6, 7)$$
$$= \bar{A}\bar{B}C + \bar{A}B\bar{C} + AB\bar{C} + ABC$$
$$= \bar{A}\bar{B}C + B\bar{C} + AB$$

$$f_\beta(A, B, C) = \sum m(1, 3, 5, 6)$$
$$= \bar{A}\bar{B}C + \bar{A}BC + A\bar{B}C + AB\bar{C}$$
$$= \bar{A}C + A\bar{B}C + AB\bar{C}$$

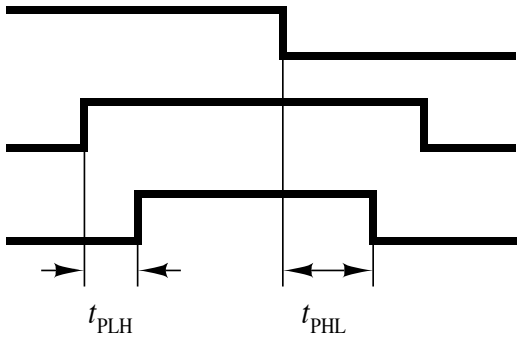
Important Characteristics

- Logic function is important
- How about other characteristics?
  - Physical characteristics of a logic circuit to be considered:
    - Propagation delays
    - Gate fan-in and fan-out restrictions
    - Power consumption
    - Size and weight

Propagation Delay

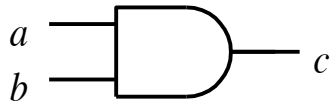
- Delay between the time of an input change and the corresponding output change.
- Typical two propagation delay parameters:
  - t_{PLH} = propagation delay time, low-to-high-level output
  - t_{PHL} = propagation delay time, high-to-low-level output
- Approximation:

$$t_{PD} = \frac{t_{PLH} + t_{PHL}}{2}$$

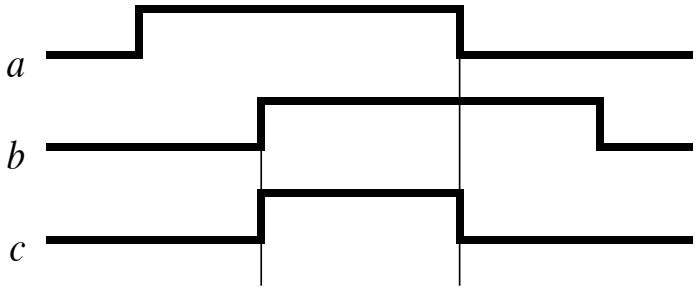


Sample 4: Propagation Delay

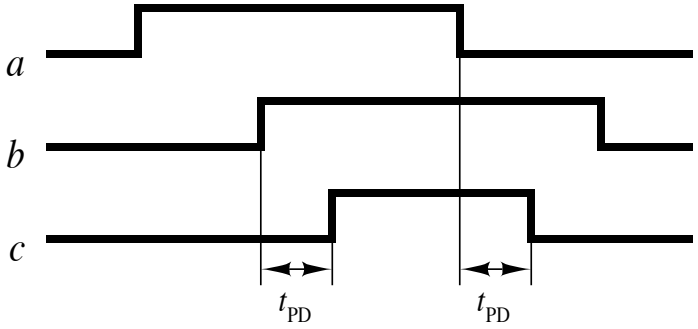
- Propagation delay through a logic gate



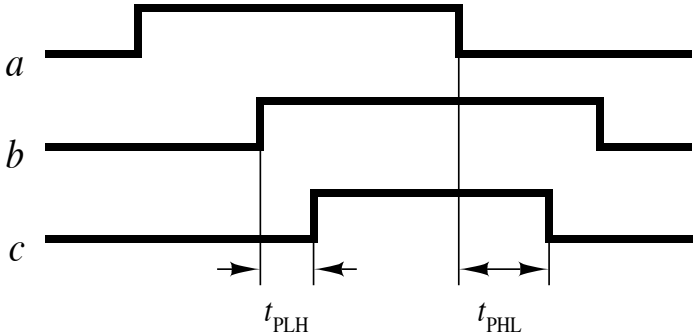
(a) Two-input AND gate



(b) Ideal (zero) delay



(c) $t_{PD} = t_{PLH} = t_{PHL}$



(d) $t_{PLH} < t_{PHL}$

Power

- Power dissipation and propagation delays for several logic families

Logic Family	Propagation Delay t_{PD}(ns)	Power Dissipation Per Gate (mW)	Technology
7400	10	10	Standard TTL
74H00	6	22	High-speed TTL
74L00	33	1	Low-power TTL
74LS00	9.5	2	Low-power Schottky TTL
74S00	3	19	Schottky TTL
74ALS00	3.5	1.3	Advanced low-power Schottky TTL
74AS00	3	8	Advanced Schottky TTL
74HC00	8	0.17	High-speed CMOS

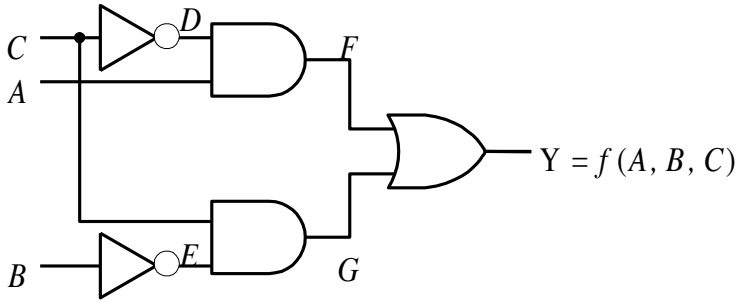
Propagation Delay of 74LS

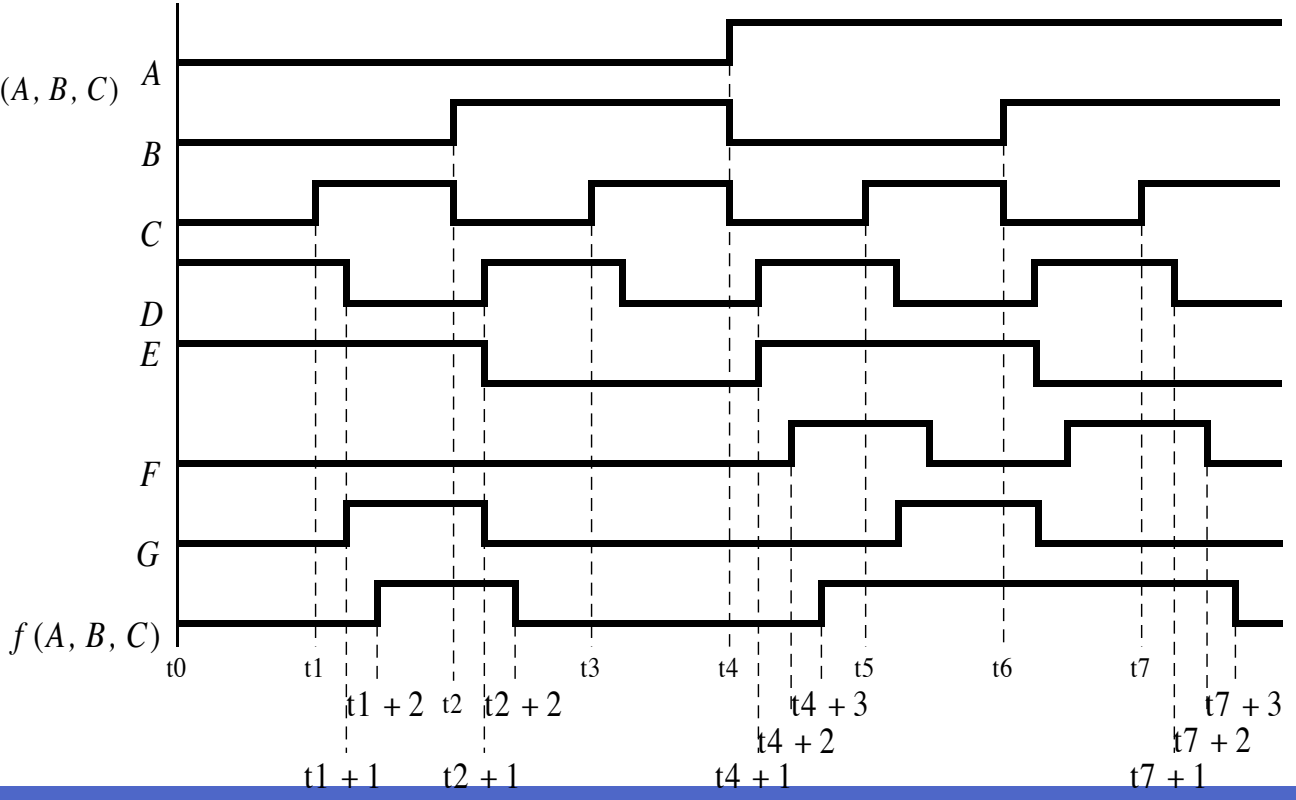
- Propagation delays of primitive 74LS series gates (Table 2.8)

Chip	Function	t_{PLH}		t_{PHL}	
		Typical	Maximum	Typical	Maximum
74LS04	NOT	9	15	10	15
74LS00	NAND	9	15	10	15
74LS02	NOR	10	15	10	15
74LS08	AND	8	15	10	20
74LS32	OR	14	22	14	22

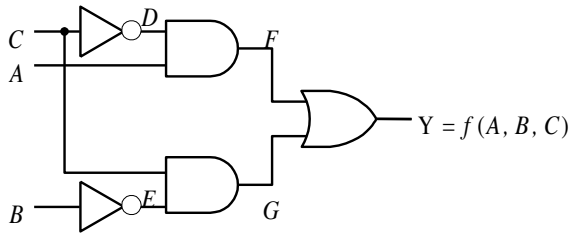
Sample 5

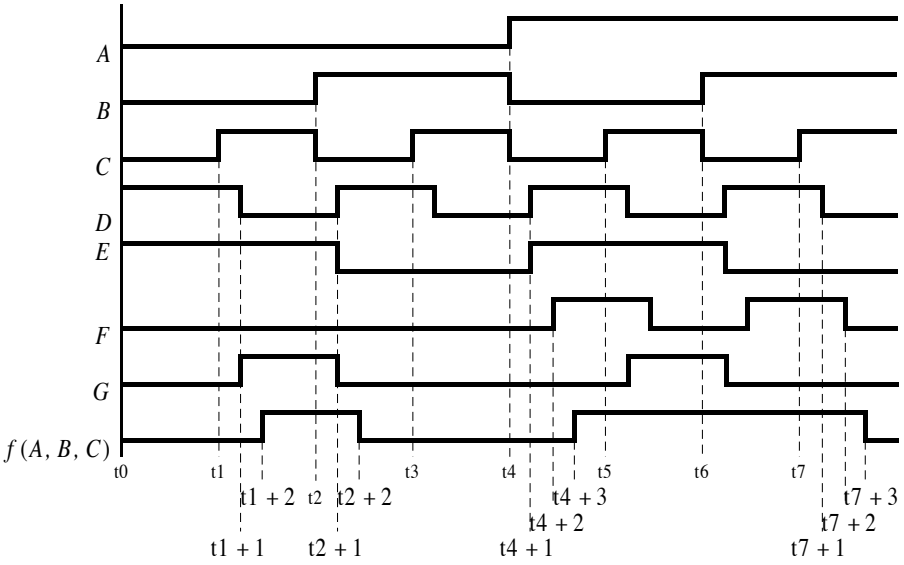
- Find the truth table and minimum switching expression.





Sample 5: Switching Expression





ABC	$f(A,B,C)$
0 0 0	0
0 0 1	1
0 1 0	0
0 1 1	0
1 0 0	1
1 0 1	1
1 1 0	1
1 1 1	0

$$
\begin{aligned}
f(A,B,C) &= \sum m(1,4,5,6) \\
&= \overline{A}\,\overline{B}C + A\overline{B}\,\overline{C} + A\overline{B}C + AB\overline{C} \\
&= A\overline{C} + \overline{B}C
\end{aligned}
$$

Digital Circuit Synthesis

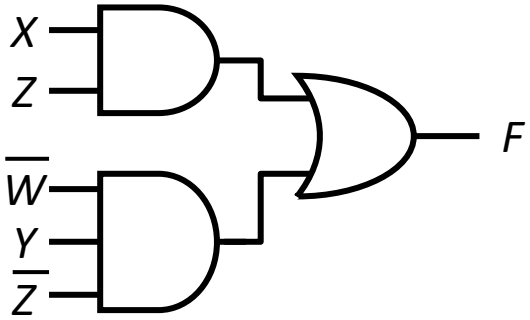
Synthesis

- Realize a logic function using logic gates
- Techniques
  - AND-OR networks
  - NAND networks
  - OR-AND networks
  - NOR networks
  - Two-level circuits
  - AND-OR-inverter networks
  - Factoring

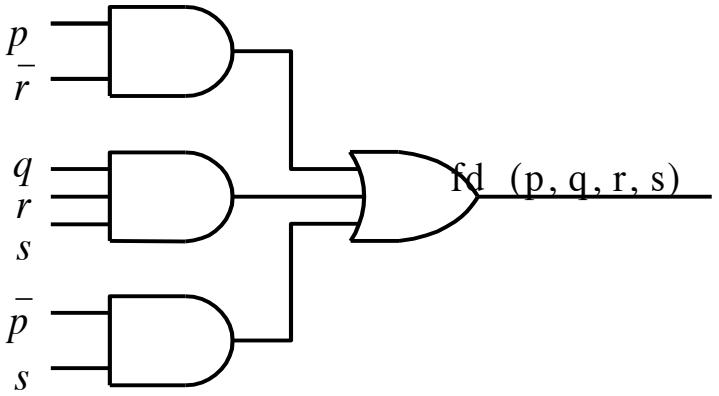
AND-OR Network

- AND gates to form product terms
- An OR gate to form the sum of products
- Switching expression **must** be in SOP form.

$$F = XZ + \overline{W}Y\overline{Z}$$

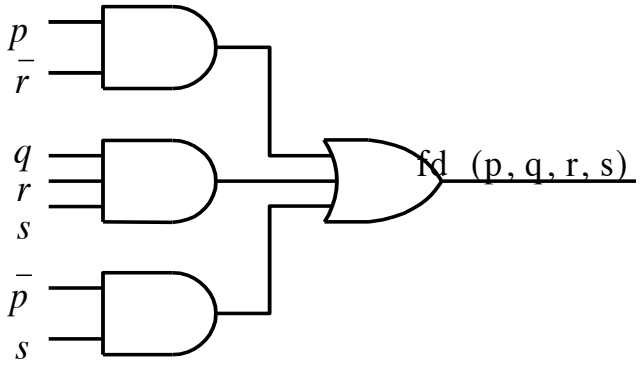


$$f_\delta(p,q,r,s) = p\bar{r} + qrs + \bar{p}s$$

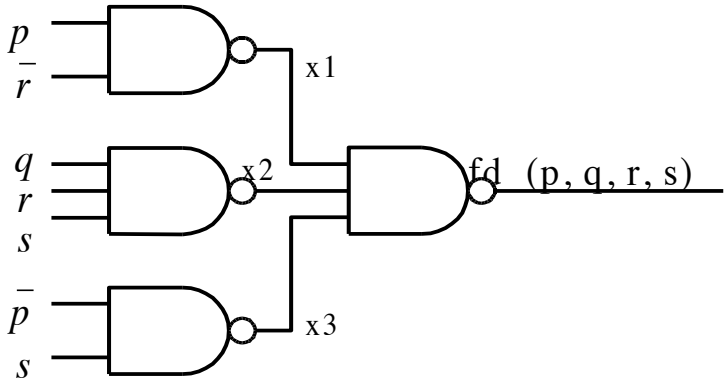


Sample 6

$$f_\delta(p,q,r,s) = p\bar{r} + qrs + \bar{p}s$$



$$f_\delta(p,q,r,s) = \overline{\overline{\overline{p\bar{r} + qrs + \bar{p}s}}}$$
$$= \overline{\overline{p\bar{r}} \cdot \overline{qrs} \cdot \overline{\bar{p}s}}$$
$$= \overline{x_1 \cdot x_2 \cdot x_3}$$
$$x_1 = \overline{p\bar{r}},$$
$$x_2 = \overline{qrs},$$
$$x_3 = \overline{\bar{p}s}$$



Easily conversion from AND-OR Network to NADN network

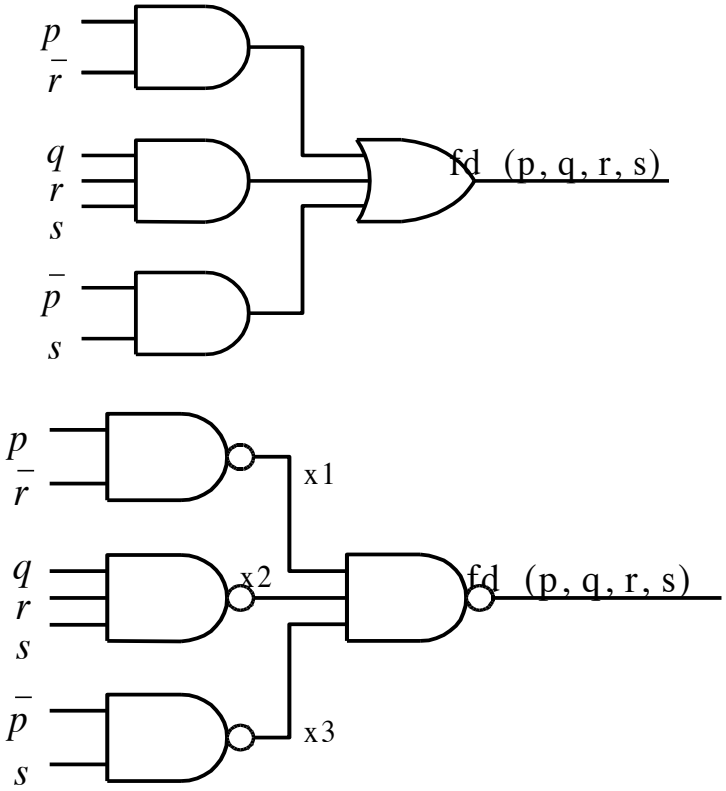
Sample 6 ☺

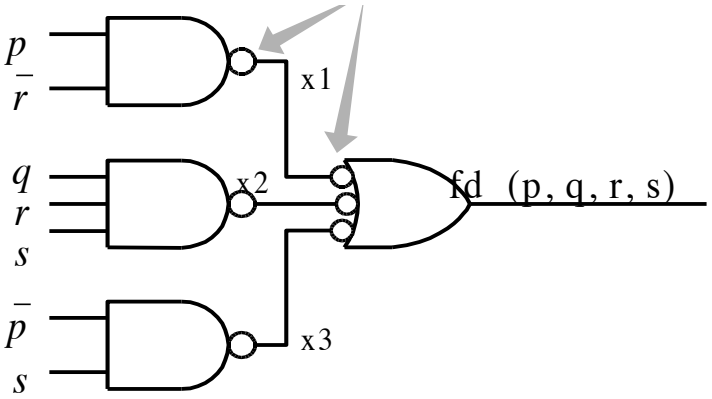
$f_\delta(p,q,r,s) = p\bar{r} + qrs + \bar{p}s$

$f_\delta(p,q,r,s) = \overline{x_1 \cdot x_2 \cdot x_3} \quad x_1 = \overline{p\bar{r}},\ x_2 = \overline{qrs},\ x_3 = \overline{\bar{p}s}$

$= \overline{x_1} + \overline{x_2} + \overline{x_3}$

$= \overline{\overline{p\bar{r}}} + \overline{\overline{qrs}} + \overline{\overline{\bar{p}s}}$

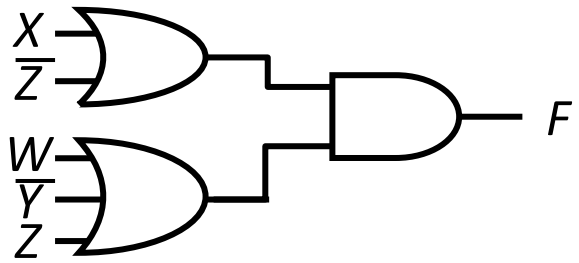




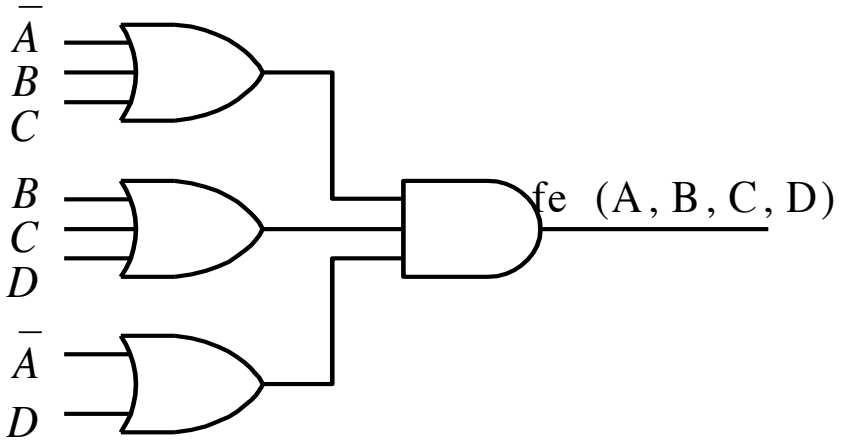
OR-AND Network

- OR gates to form sum terms
- An AND gate to form the product of sums
- Switching expression must be in POS form.

$$F = (X + \overline{Z})(W + \overline{Y} + Z)$$

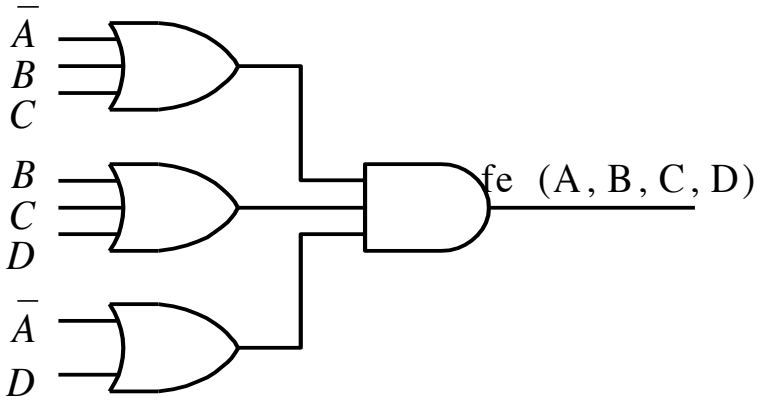


$$f_{\varepsilon}(A,B,C,D) = (\overline{A} + B + C)(B + C + D)(\overline{A} + D)$$

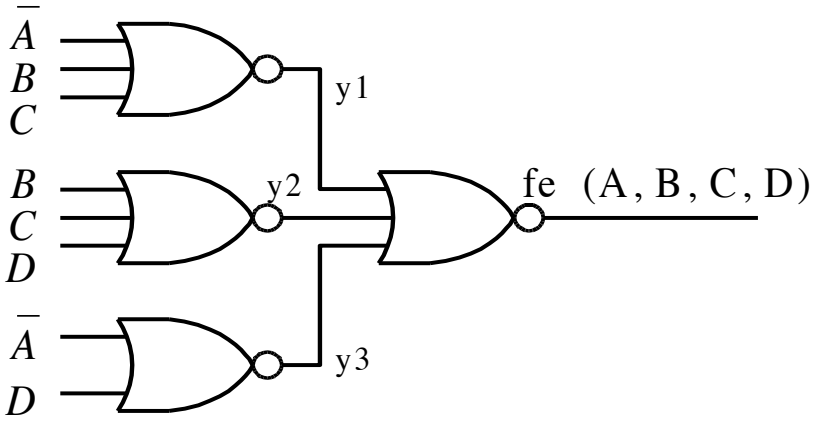


Sample 7

$$f_\varepsilon(A,B,C,D) = (\overline{A}+B+C)(B+C+D)(\overline{A}+D)$$



$$f_\varepsilon(A,B,C,D) = \overline{(\overline{A}+B+C)(B+C+D)(\overline{A}+D)}$$
$$= \overline{\overline{A}+B+C} + \overline{B+C+D} + \overline{\overline{A}+D}$$
$$= \overline{y_1+y_2+y_3}$$
$$y_1 = \overline{\overline{A}+B+C},$$
$$y_2 = \overline{B+C+D},$$
$$y_3 = \overline{\overline{A}+D}$$



Easily conversion from OR-AND Network to NOR network

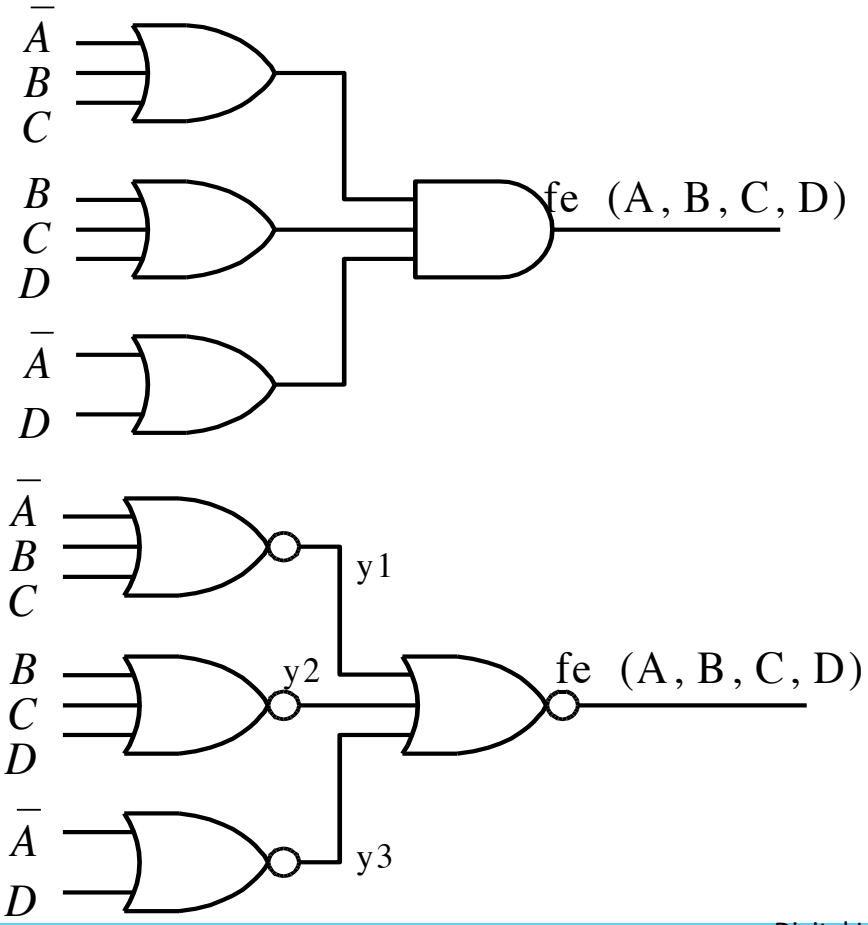
Sample 7 ☺

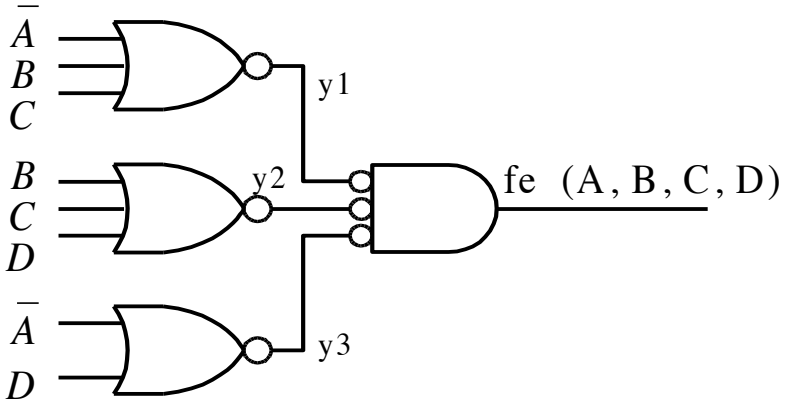
$$f_{\varepsilon}(A,B,C,D) = (\bar{A}+B+C)(B+C+D)(\bar{A}+D)$$

$$= \overline{y_1 + y_2 + y_3}$$

$$= \overline{y_1}.\ \overline{y_2}\ .\ \overline{y_3}$$

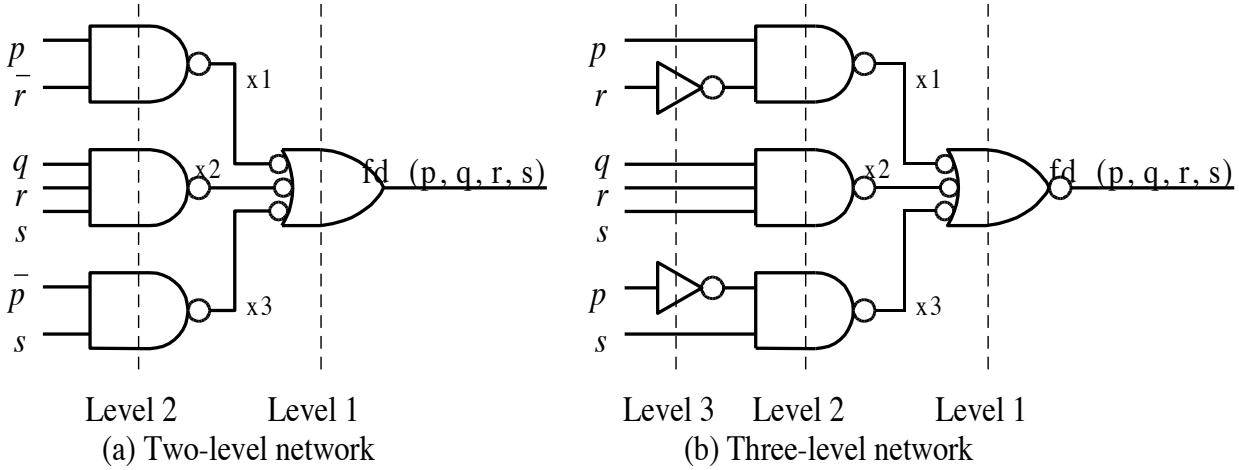
$$= \overline{\bar{A}+B+C}.\ \overline{\overline{B+C+D}}\ .\ \overline{\bar{A}+D}$$





Two-level Circuits

- Input signals pass through two levels of gates before reaching the output.

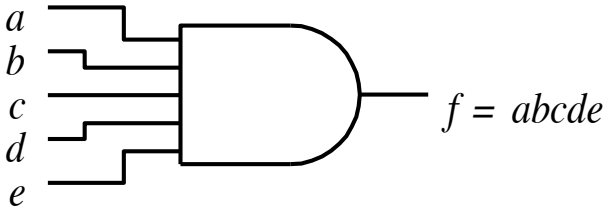


Two-Level: Implementation Procedure

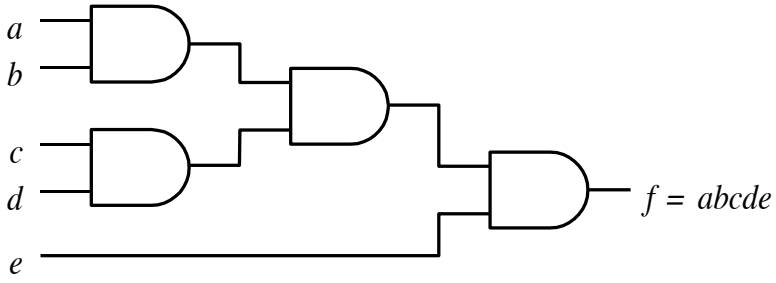
- Step 1. Express the function in minterm (maxterm) list form.
- Step 2. Write out the minterms (maxterms) in algebraic form.
- Step 3. Simplify the function in SOP (POS) form.
- Step 4. Transform the expression into the NAND (NOR) form.
- Step 5. Draw the NAND (NOR) logic diagram.

Are Two-level Circuits Enough?

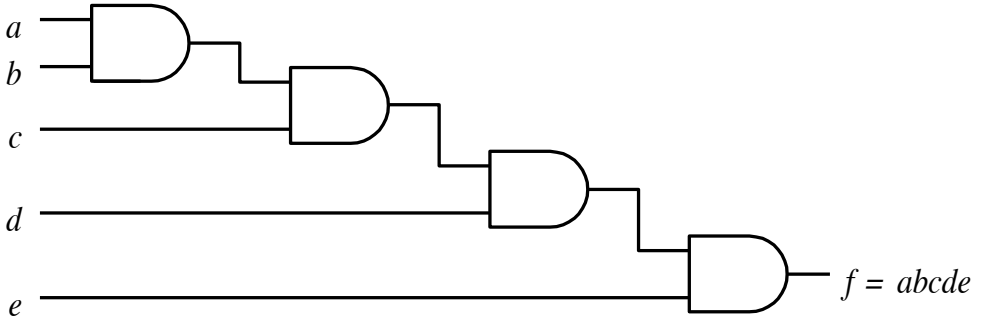
- Circuits with more than two levels are often needed
  - Fan-in constraints



(a) A single five-input AND gate



(b) Three-level network of two-input gates



(c) Four-level network of two-input gates.

Sample 8

- NAND implementation of $f_{\phi}(X,Y,Z) = \Sigma m(0,3,4,5,7)$

Sample 8: NAND

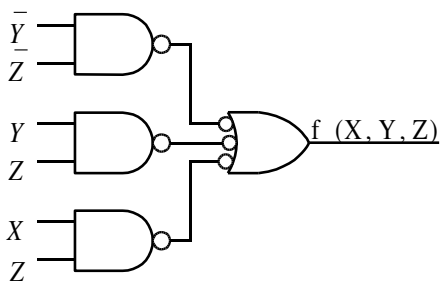
- NAND implementation of $f_\phi(X,Y,Z) = \Sigma m(0,3,4,5,7)$
  1. $f_\phi(X,Y,Z) = \Sigma m(0,3,4,5,7)$
  2. $f_\phi(X,Y,Z) = m_0 + m_3 + m_4 + m_5 + m_7$
  3. Simplification
  4. NAND
  5. Draw diagram

$$\bar{X}\bar{Y}\bar{Z} + \bar{X}YZ + X\bar{Y}\bar{Z} + X\bar{Y}Z + XYZ$$

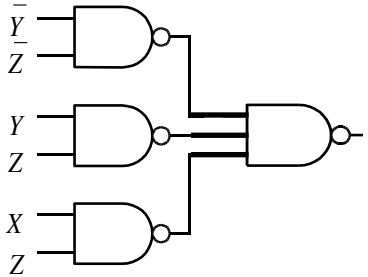
$$f_\phi(X,Y,Z) = \bar{Y}\bar{Z} + YZ + X\,Z$$

$$f_\phi(X,Y,Z) = \overline{\overline{\bar{Y}\bar{Z}} + \overline{YZ} + \overline{X\,Z}}$$

$$f_\phi(X,Y,Z) = \overline{\overline{\bar{Y}\bar{Z}} \cdot \overline{YZ} \cdot \overline{X\,Z}}$$



(a) NAND implementation



(a) NAND implementation

Sample 9

- NOR implementation of $f_{\phi}(X,Y,Z) = \Sigma m(0,3,4,5,7)$

Sample 9: NOR

- NOR implementation of $f_\phi(X,Y,Z) = \Sigma m(0,3,4,5,7)$
  1. $f_\phi(X,Y,Z) = \prod M(1,2,6)$
  2. $f_\phi(X,Y,Z) = M_1 \,.\, M_2 \,.\, M_6$
  3. Simplification
  4. NAND
  5. Draw diagram

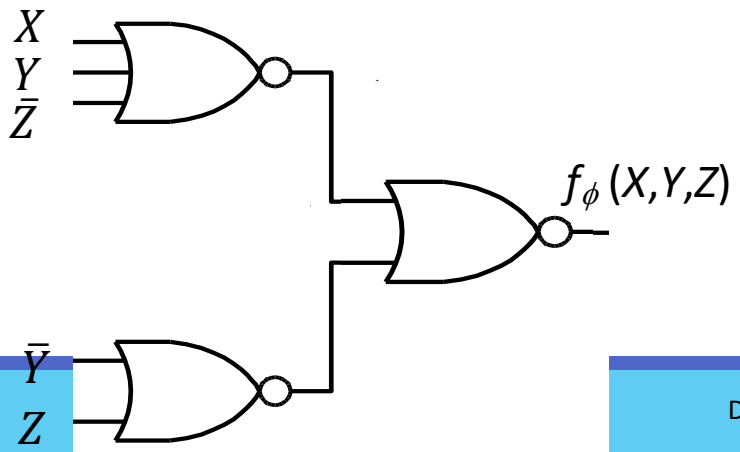
$$= (X + Y + \bar{Z})(X + \bar{Y} + Z)(\bar{X} + \bar{Y} + Z)$$

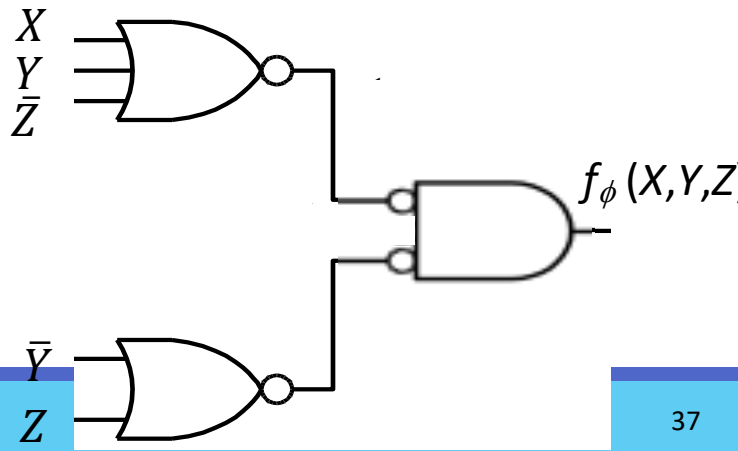
$$= (X + Y + \bar{Z})(\bar{Y} + Z)$$

$$= \overline{\overline{(X + Y + \bar{Z})}.\ \overline{(\bar{Y} + Z)}}$$

$$= \overline{\overline{\big((X + Y + \bar{Z})(\bar{Y} + Z)\big)}}$$

$$= \overline{\overline{(X + Y + \bar{Z})} + \overline{(\bar{Y} + Z)}}$$

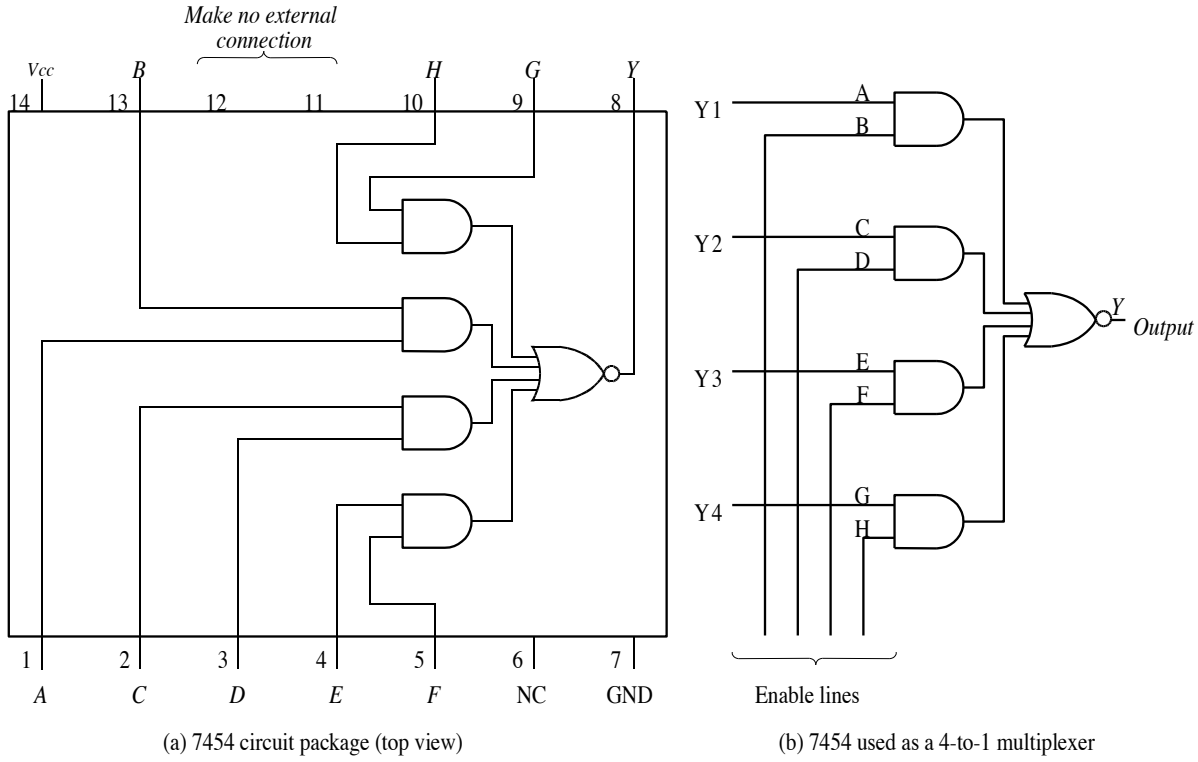




AND-OR-Inverter Circuits

- A set of AND gates followed by a NOR gate.
- Used to readily realize two-level SOP circuits.
- 7454 circuit:

$$F = \overline{AB + CD + EF + GH}$$



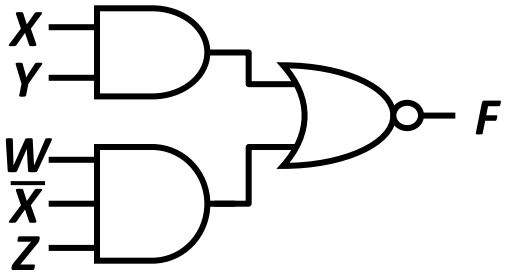
(a) 7454 circuit package (top view)

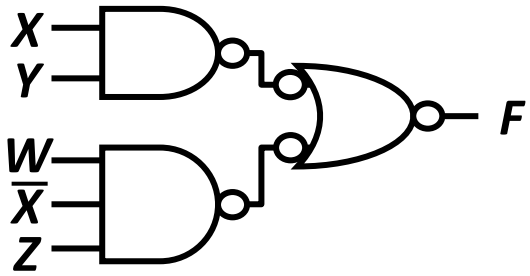
(b) 7454 used as a 4-to-1 multiplexer

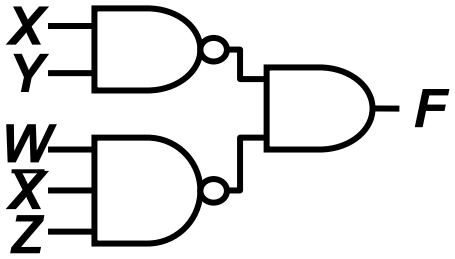
Sample 10

- A set of OR gates followed by a NAND gate.

$$F = \overline{XY + W\overline{X}Z}$$



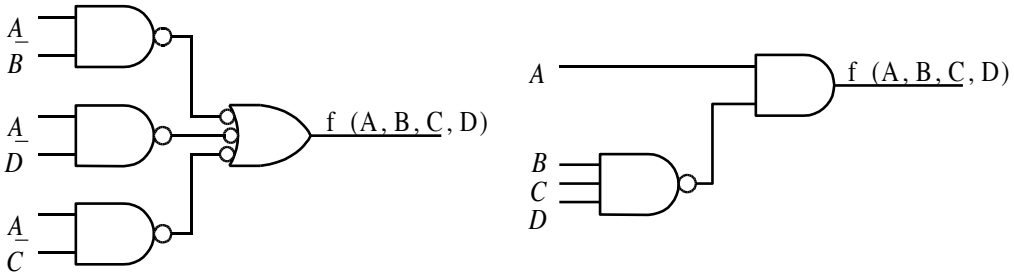




Factoring

- A technique to obtain higher-level forms of switching functions.
- Advantages
  - May need less hardware
  - May be used when there are fan-in constraints
- Disadvantage
  - More difficult to design
  - Slower

$$f(A,B,C,D) = A\overline{B} + A\overline{D} + A\overline{C} = A(\overline{B} + \overline{D} + \overline{C}) = A(\overline{BCD})$$



(a) Original form (b) After factoring

Sample 11

- $f(a,b,c,d) = \Sigma m(8,13)$ with only two-input AND and OR gates.

Sample 11: AND-OR

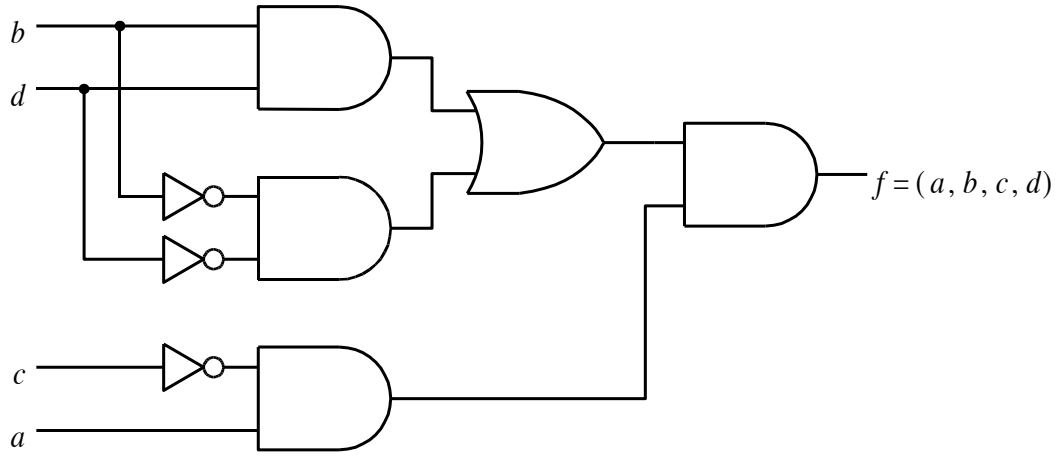
f (*a*,*b*,*c*,*d*) = Σ*m*(8,13) with only two-input AND and OR gates.

- Write the canonical SOP form:

  f (*a*,*b*,*c*,*d*) = Σ*m*(8,13) = $a\bar{b}\bar{c}\bar{d} + ab\bar{c}d$

- Two four-input AND gates and one two-input OR gate are needed.
- Apply factoring:

$$f(a,b,c,d) = a\bar{b}\bar{c}\bar{d} + ab\bar{c}d = (a\bar{c})(bd + \bar{b}\bar{d})$$



Sample 12

- A burglar alarm with four control switches

Sample 12: Burglar Alarm

- A burglar alarm with four control switches
- Each control switch produces logic 1 when:
  - Switch *A*: Secret switch is closed
  - Switch *B*: Safe is in its normal position in the closet
  - Switch *C*: Clock is between 1000 and 1400 hours
  - Switch *D*: Closet door is closed.
- Write the equations of the control logic that produces logic 1 when
  - Safe is moved AND the secret switch is closed, OR
  - Closet is opened after banking hours, OR
  - Closet is opened with the secret switch open.

$$f(A,B,C,D) = A\overline{B} + \overline{C}\,\overline{D} + \overline{A}\,\overline{D}$$

Sample 13

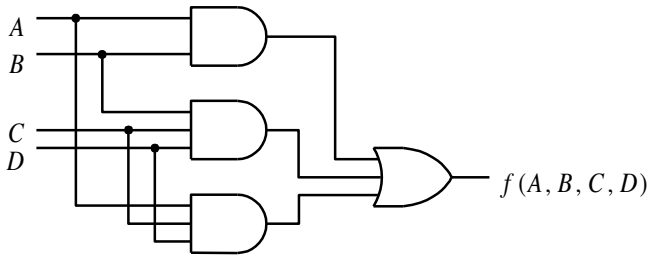
- Write a logic function for Doe family voter:
  - Input: John (Dad): *A*, Jane (Mom):*B*, Joe: *C*, Sue: *D*.
  - Output: Vote for either *hamburgers* (0) or *chicken* (1).
  - Condition:
    - Majority win.
    - Mom and Dad agree, they win.

Sample 13: Doe Family Voter

A (John)	B (Jane)	C (Joe)	D (Sue)	F 1 = chicken
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

$$A,B,C,D) = \overline{A}BCD + A\overline{B}CD + AB\overline{C}\,\overline{D} + AB\overline{C}D + ABC\overline{D} + ABCD$$

$$= \overline{A}BCD + A\overline{B}CD + AB$$
$$= AB + ACD + \overline{A}BCD$$
$$= AB + ACD + BCD$$



Sample 14

- Logic equations for a circuit that adds two binary numbers

Sample 14: Specification

- Logic equations for a circuit that adds two **2-bit** binary numbers
- Input: $(A_1A_0)_2$ and $(B_1B_0)_2$
- Output: produces sum bits $(S_1S_0)_2$ and carry bit C_1;

$$\begin{array}{r} A_1\,A_0 \\ +\ B_1\,B_0 \\ \hline C_1\,S_1\,S_0 \end{array}$$

2-bit Adder: Truth Table

A1	A0	B1	B0	C1	S1	S0
0	0	0	0	0	0	0
0	0	0	1	0	0	1
0	0	1	0	0	1	0
0	0	1	1	0	1	1
0	1	0	0	0	0	1
0	1	0	1	0	1	0
0	1	1	0	0	1	1
0	1	1	1	1	0	0
1	0	0	0	0	1	0
1	0	0	1	0	1	1
1	0	1	0	1	0	0
1	0	1	1	1	0	1
1	1	0	0	0	1	1
1	1	0	1	1	0	0
1	1	1	0	1	0	1
1	1	1	1	1	1	0

2-bit Adder: S0

A1	A0	B1	B0	C1	S1	S0
0	0	0	0	0	0	0
0	0	0	1	0	0	1
0	0	1	0	0	1	0
0	0	1	1	0	1	1
0	1	0	0	0	0	1
0	1	0	1	0	1	0
0	1	1	0	0	1	1
0	1	1	1	1	0	0
1	0	0	0	0	1	0
1	0	0	1	0	1	1
1	0	1	0	1	0	0
1	0	1	1	1	0	1
1	1	0	0	0	1	1
1	1	0	1	1	0	0
1	1	1	0	1	0	1
1	1	1	1	1	1	0

$$S_0 = \overline{A}_1\overline{A}_0\overline{B}_1B_0 + \overline{A}_1\overline{A}_0B_1B_0 + \overline{A}_1A_0\overline{B}_1\overline{B}_0 + \overline{A}_1A_0B_1\overline{B}_0 + A_1\overline{A}_0\overline{B}_1B_0 + A_1\overline{A}_0B_1B_0 + A_1A_0\overline{B}_1\overline{B}_0 + A_1A_0B_1\overline{B}_0$$

$$= A_0\overline{B_0} + \overline{A_0}\,B_0$$

2-bit Adder: S1

A1	A0	B1	B0	C1	S1	S0
0	0	0	0	0	0	0
0	0	0	1	0	0	1
0	0	1	0	0	1	0
0	0	1	1	0	1	1
0	1	0	0	0	0	1
0	1	0	1	0	1	0
0	1	1	0	0	1	1
0	1	1	1	1	0	0
1	0	0	0	0	1	0
1	0	0	1	0	1	1
1	0	1	0	1	0	0
1	0	1	1	1	0	1
1	1	0	0	0	1	1
1	1	0	1	1	0	0
1	1	1	0	1	0	1
1	1	1	1	1	1	0

$$S_1 = \bar{A}_1\bar{A}_0B_1\bar{B}_0 + A_1\bar{A}_0B_1B_0 + \bar{A}_1A_0\bar{B}_1B_0 + A_1A_0B_1\bar{B}_0 + A_1\bar{A}_0\bar{B}_1\bar{B}_0 + \bar{A}_1\bar{A}_0\bar{B}_1B_0 + A_1A_0\bar{B}_1\bar{B}_0 + A_1A_0B_1B_0$$

$$\bar{A}_1\bar{A}_0B_1 + \bar{A}_1B_1\bar{B}_0 + \bar{A}_1A_0\bar{B}_1B_0 + A_1A_0B_1B_0 + A_1\bar{B}_1\bar{B}_0 + A_1\bar{A}_0\bar{B}_1$$

2-bit Adder: C1

A1	A0	B1	B0	C1	S1	S0
0	0	0	0	0	0	0
0	0	0	1	0	0	1
0	0	1	0	0	1	0
0	0	1	1	0	1	1
0	1	0	0	0	0	1
0	1	0	1	0	1	0
0	1	1	0	0	1	1
0	1	1	1	1	0	0
1	0	0	0	0	1	0
1	0	0	1	0	1	1
1	0	1	0	1	0	0
1	0	1	1	1	0	1
1	1	0	0	0	1	1
1	1	0	1	1	0	0
1	1	1	0	1	0	1
1	1	1	1	1	1	0

$$C_1 = \overline{A}_1 A_0 B_1 B_0 + A_1 \overline{A}_0 B_1 \overline{B}_0 + A_1 \overline{A}_0 B_1 B_0 + A_1 A_0 \overline{B}_1 B_0 + A_1 A_0 B_1 \overline{B}_0 + A_1 A_0 B_1 B_0$$

$$= \overline{A_1} A_0 B_1 B_0 + A_1 A_0 \overline{B_1}\, B_0 + A_0 B_0\, A_1 B_1 + A_1 B_1$$

$$= A_0 B_0\, (\overline{A_1} B_1 + A_1 \overline{B_1} + A_1 B_1) + A_1 B_1$$

$$= A_0 B_0\, (B_1 + A_1 \overline{B_1}) + A_1 B_1$$

$$= A_0 B_0\, (B_1 + A_1) + A_1 B_1$$

$$= A_0 B_0\, B_1 + A_0 B_0 A_1 + A_1 B_1$$

Sample 15

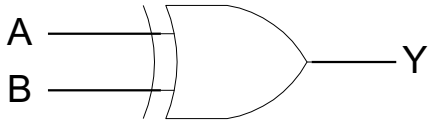
- Implement a XOR gate with 3-state buffers

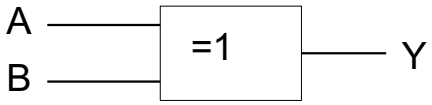
XOR & Tristate: Truth Table

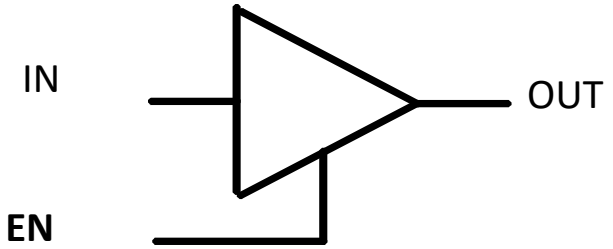
- Implement a XOR gate with 3-state buffers

$a\ b$	$f_{XOR}(a, b) = a \oplus b$	$A\ B$	Y
0 0	0	L L	L
0 1	1	L H	H
1 0	1	H L	H
1 1	0	H H	L

EN	IN	OUT
0	X	Hi-Z
1	0	0
1	1	1



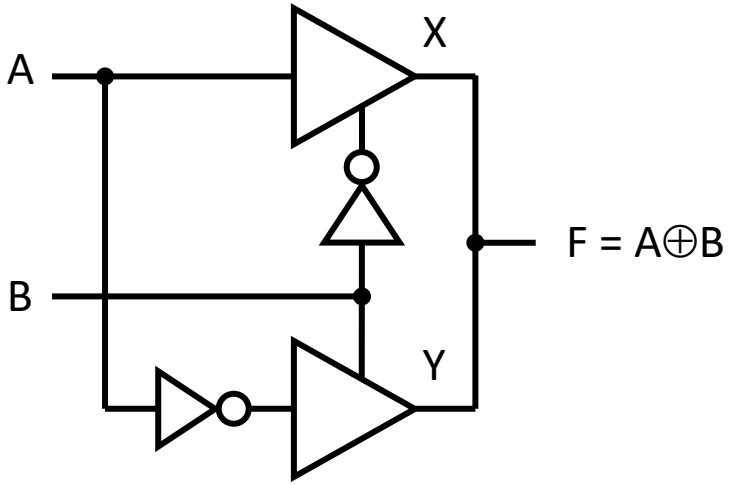




XOR & Tristate: Realization

- Implement a XOR gate with 3-state buffers

A	B	X	Y	F
0	0	0	Hi-Z	0
0	1	Hi-Z	1	1
1	0	1	Hi-Z	1
1	1	Hi-Z	0	0



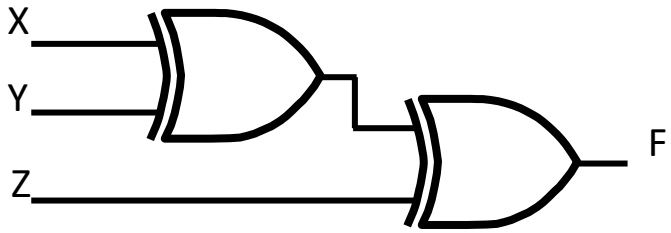
Sample 16

- Implement an odd function with 2-input XOR
  - Input: X, Y, Z
  - Odd functions: Odd number of 1

Odd Function: Realization

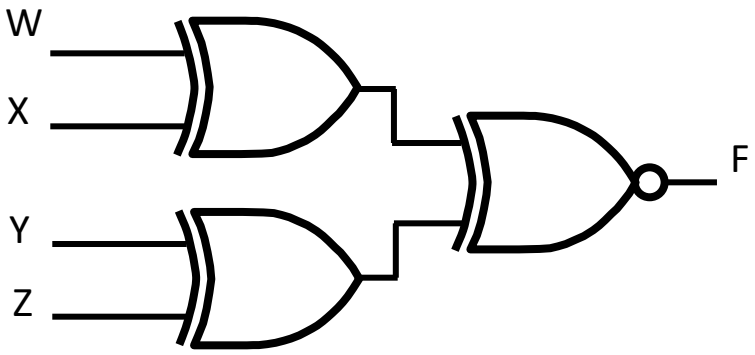
- Implement an odd function with 2-input XOR
  - Odd function: A XOR gate with more than 3 inputs

$$X \oplus Y \oplus Z = \overline{X}\,\overline{Y}\,Z + \overline{X}\,Y\,\overline{Z} + X\,\overline{Y}\,\overline{Z} + X\,Y\,Z$$



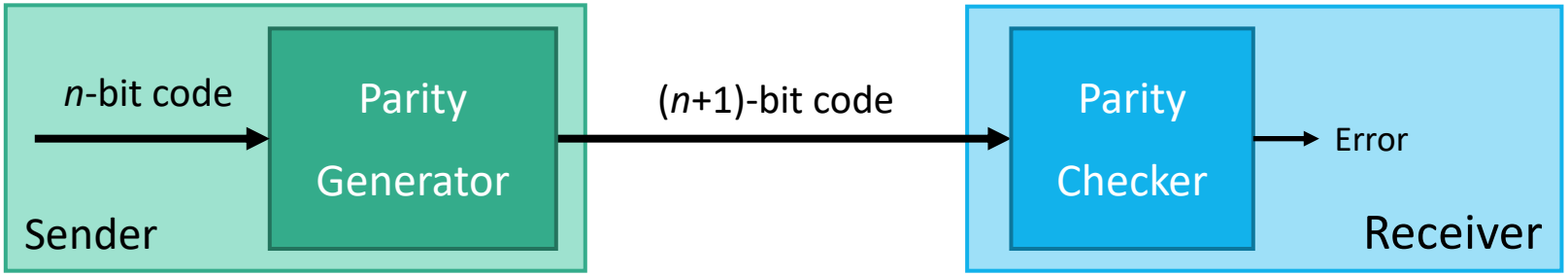
Sample 17

- Implement an even function with 2-input XOR
  - Input: X, Y, Z, W
  - Even functions: even number of 1



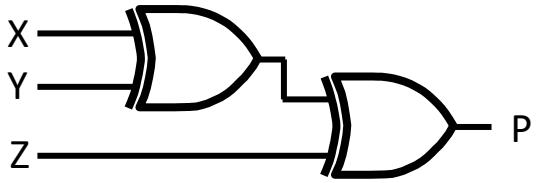
Sample 18

- Design an even parity generator
- Design a parity checker
  - Input: X, Y, Z
  - Output: even parity, P



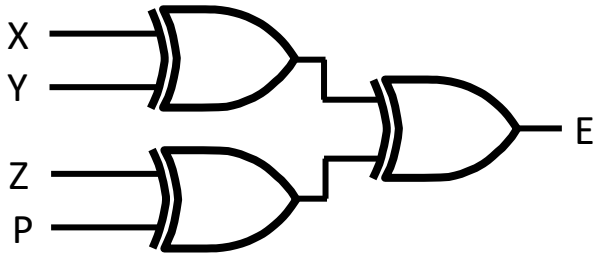
Parity Generator

- Design an even parity generator
  - 4-bit final code must have even number of 1
  - 3-bit input code must have odd number of 1
  - $P = (X \oplus Y) \oplus Z$



Parity Checker

- Design a parity checker
  - 4-bit input must have even number of 1
  - E = (X ⊕ Y) ⊕ (Z ⊕ P)



Thank You