L2: Combinational Logic Design

(Construction and Boolean Algebra)

Acknowledgements:

Review: Noise Margin

IN ▷○ OUT

Truth Table

IN	OUT
0	1
1	0

"1"
V_{OH}
V_{IH}

*Undefined
Region*

"0"
V_{IL}
V_{OL}

V(y)

V_{OH}

Slope = -1

V_{OL}

Slope = -1

V(x)

V_{OL} V_{IL} V_{IH} V_{OH}
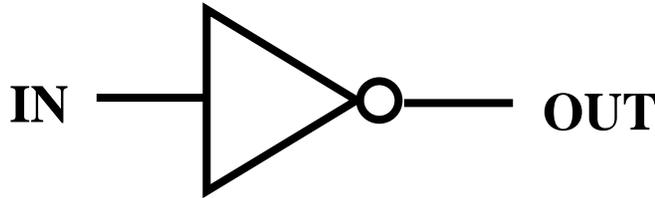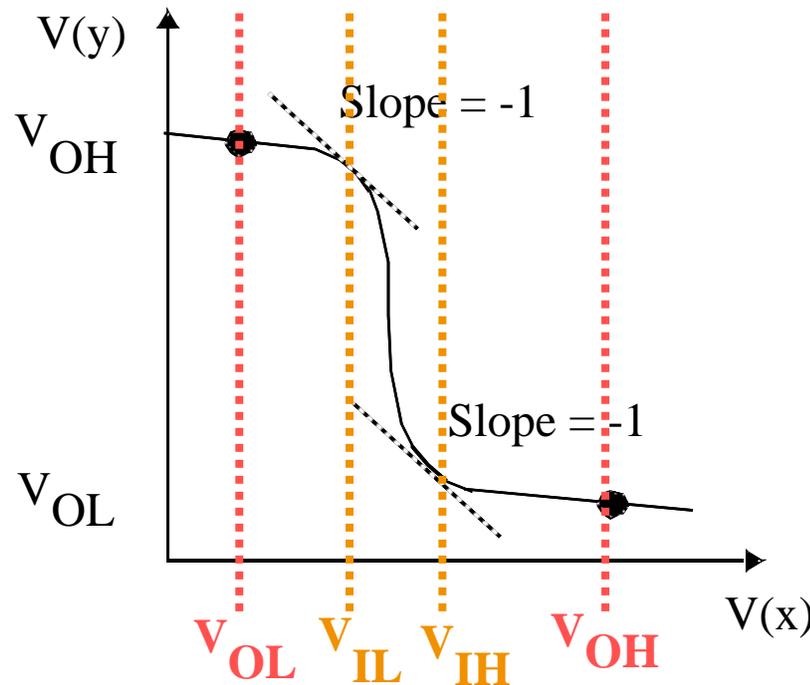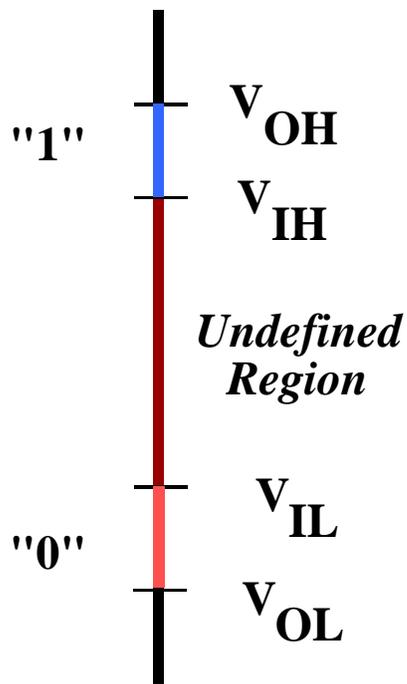
$$NM_L = V_{IL} - V_{OL}$$
$$NM_H = V_{OH} - V_{IH}$$

- ▪ **Large noise margins** **protect against various noise sources**

74LS04 (courtesy Texas Instruments)
Image removed due to copyright restrictions

# MOS Technology: The NMOS Switch

gate

source

drain

D

N+

N+

P-substrate

V_s

G

$V_T = 0.5V$

S

Switch Model

OFF

R_{NMOS}

$V_{GS} < V_T$

ON

R_{NMOS}

$V_{GS} > V_T$

NMOS ON when Switch Input is High

NMOS Device Characteristics
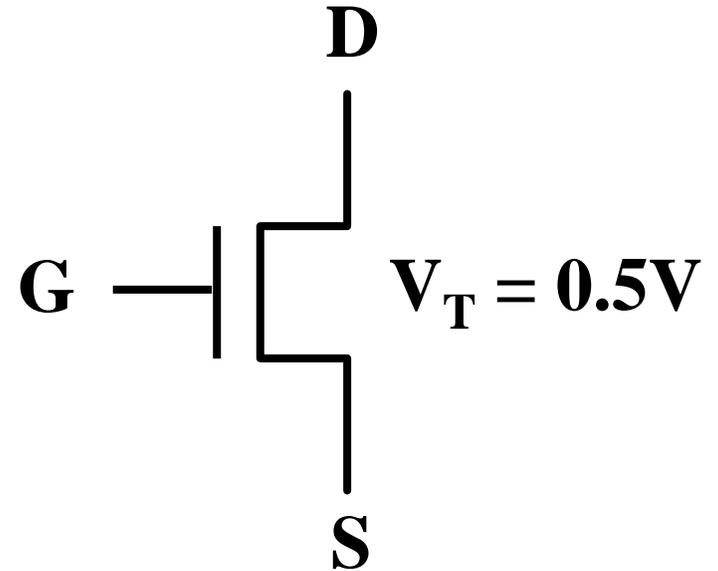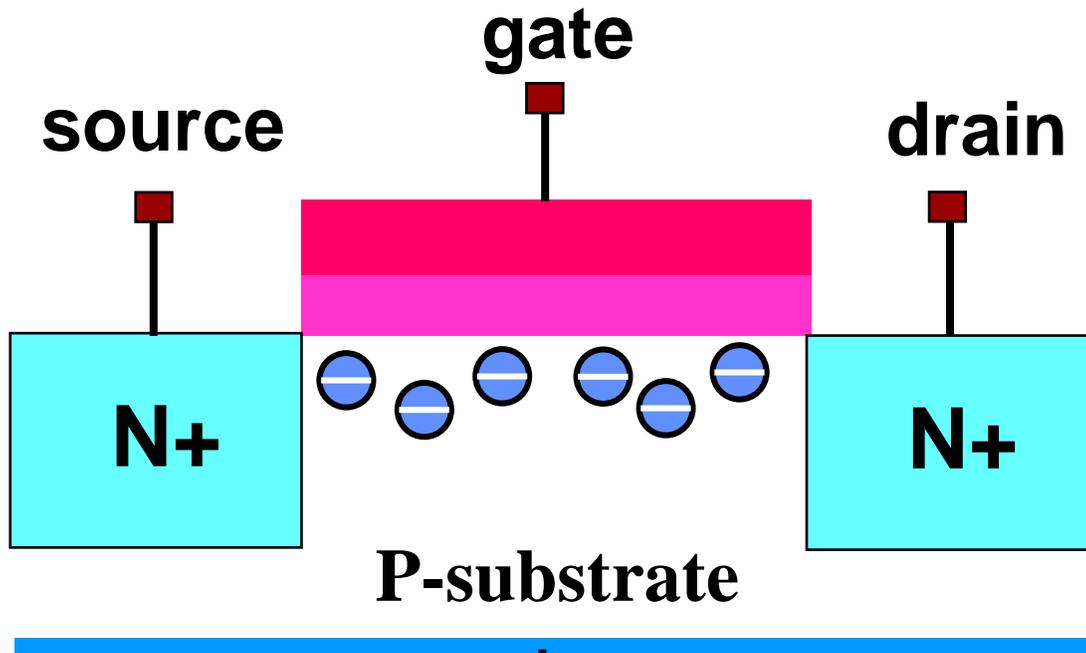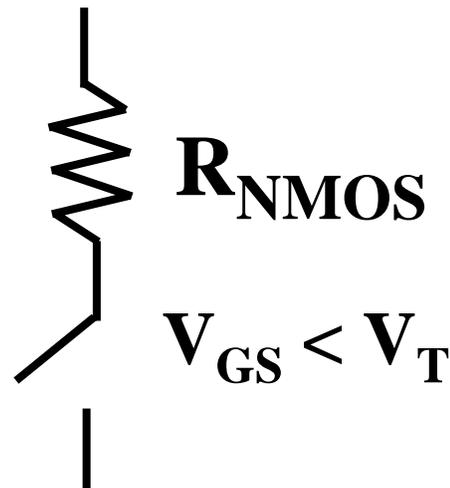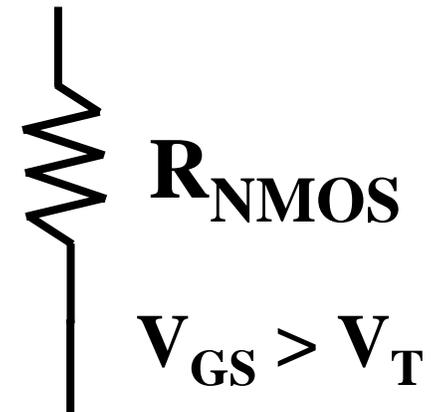
Picture of an NMOS Device
Picture removed due to copyright restrictions

D

I_D

G

+

V_{GS}

-

S

$V_T = 0.5V$
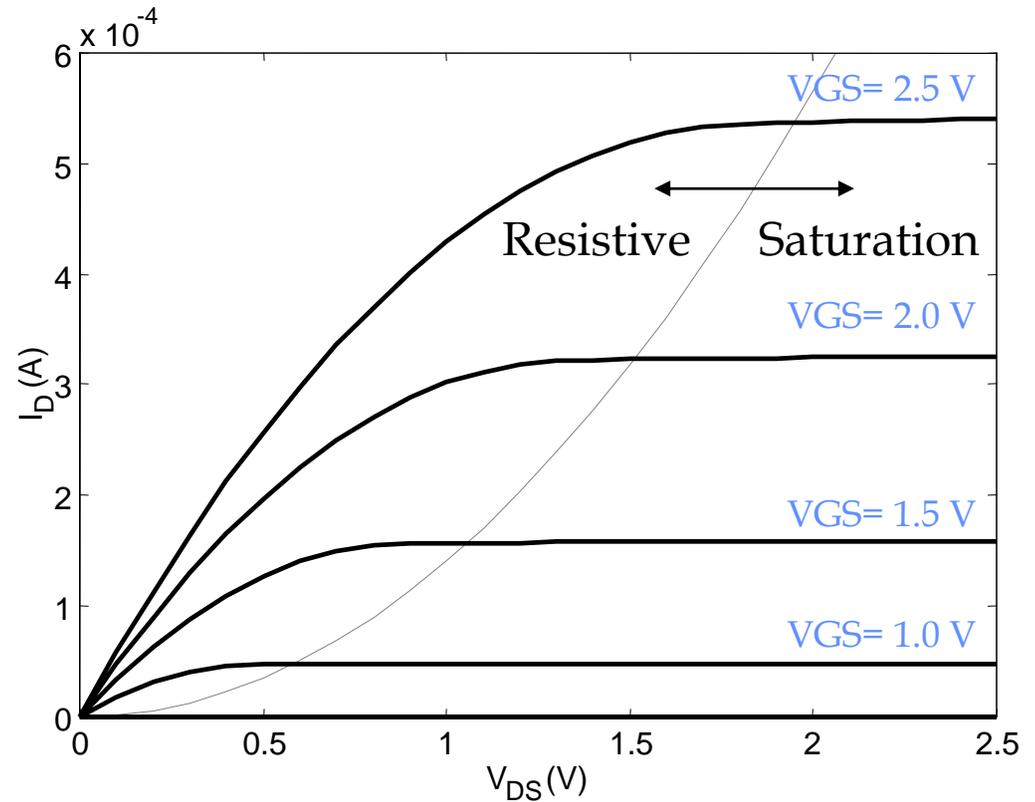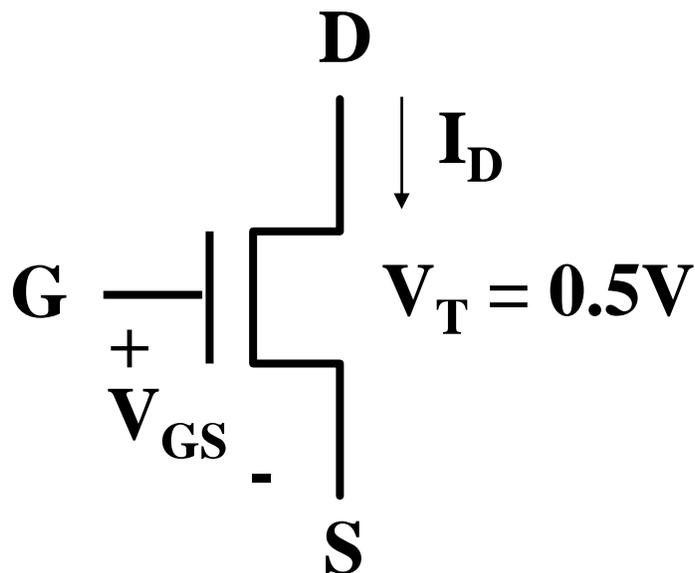
➢ **MOS is a very non-linear.**
➢ **Switch-resistor model sufficient for first order analysis.**

PMOS: The Complementary Switch

gate

source

drain

P+

P+

N-substrate

V_{DD}

S

G

$V_T = -0.5V$

D

Switch Model

OFF

R_{PMOS}

$V_{GS} > V_T$

ON

R_{PMOS}

$V_{GS} < V_T$
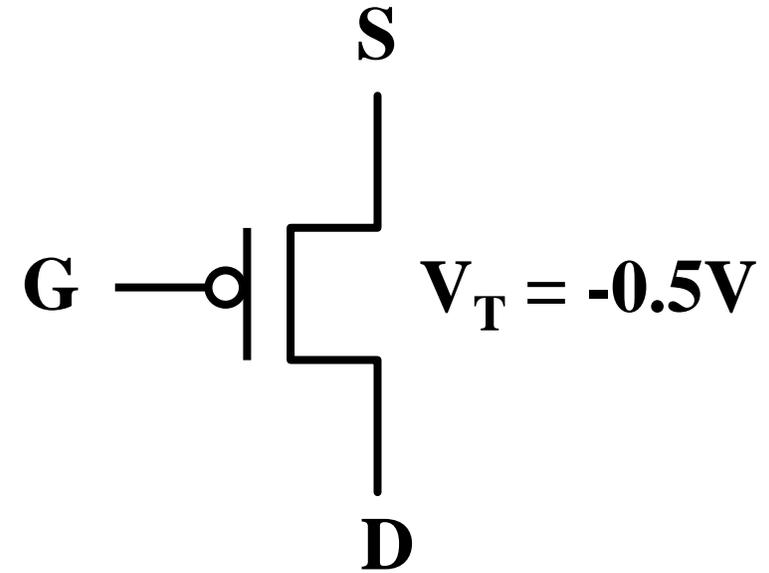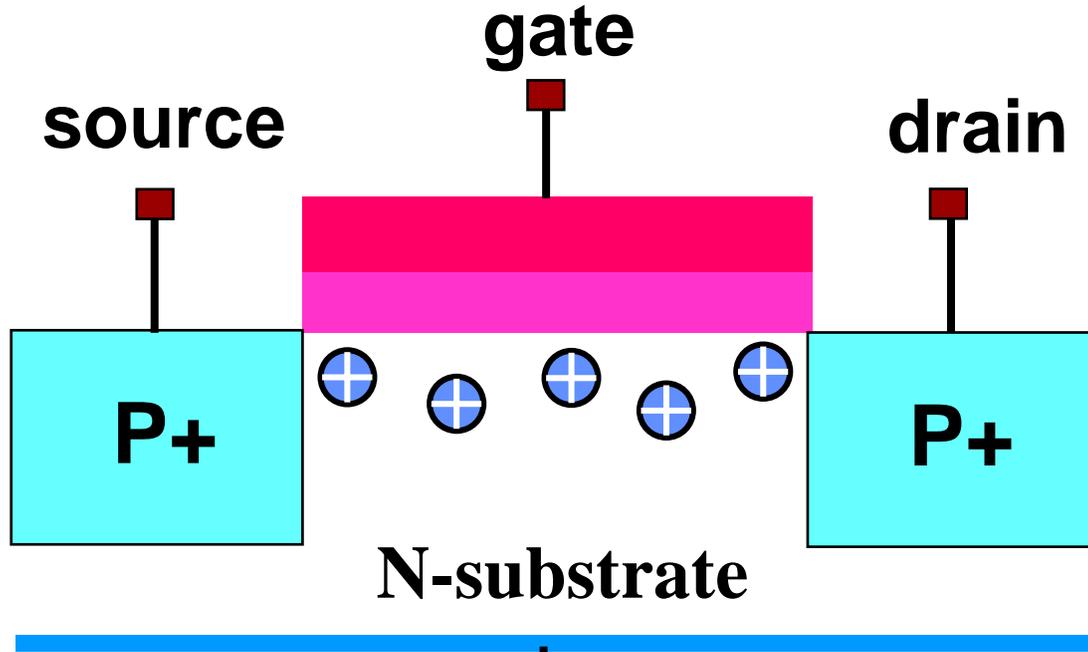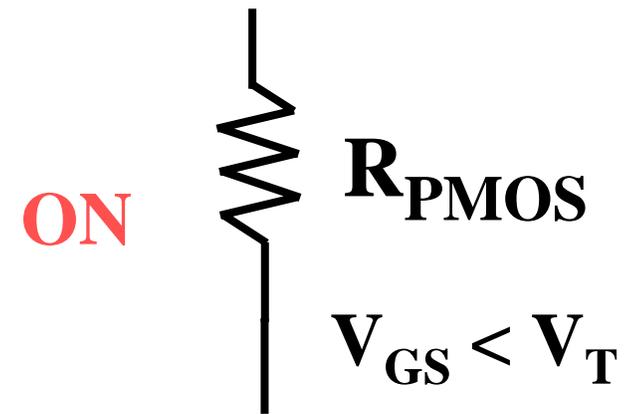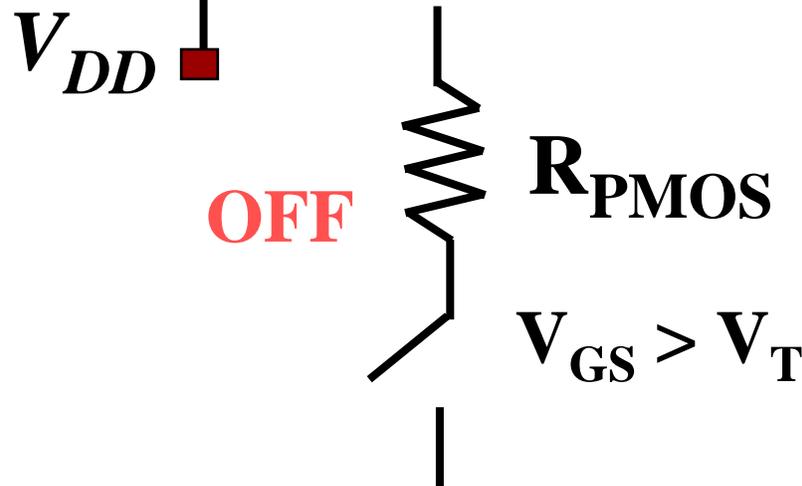
PMOS ON when Switch Input is Low

Switch Model

V_{DD}

S

G

IN

D

OUT

D

G

S

V_{DD}

R_{PMOS}

$\overline{IN}$

OUT

R_{NMOS}

IN
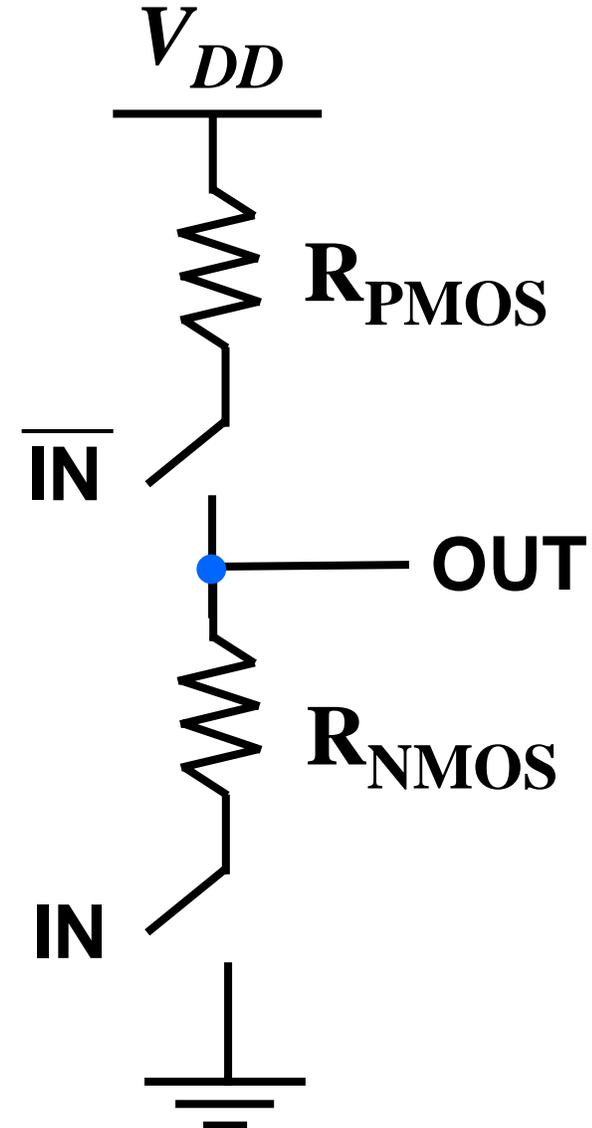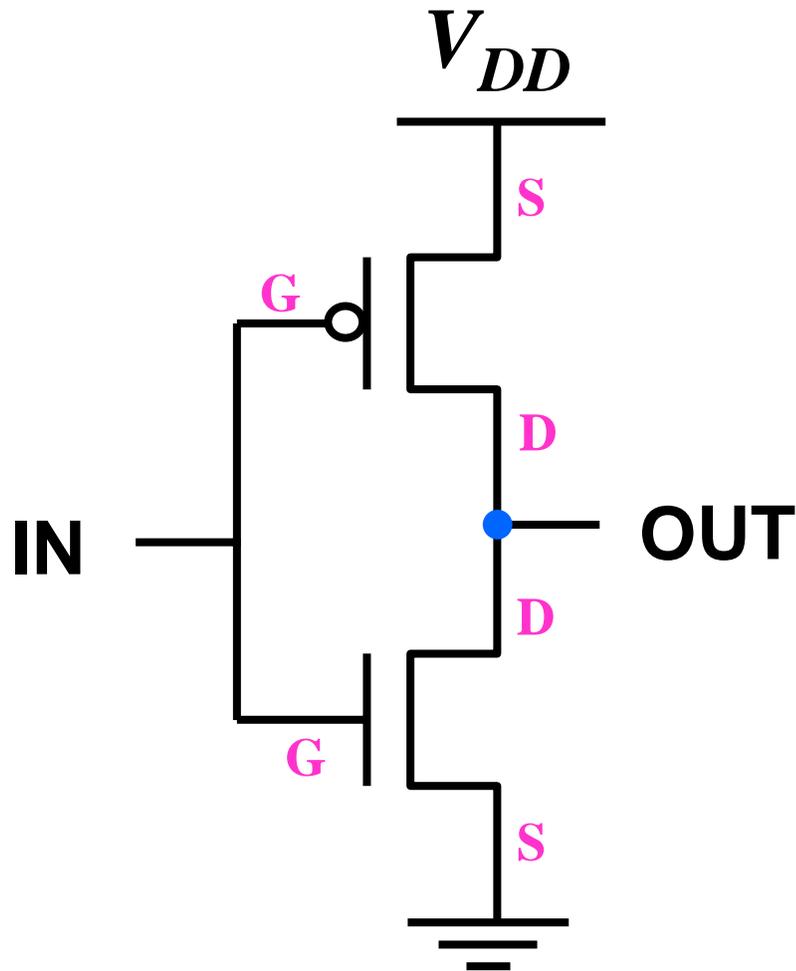
Rail-to-rail Swing in CMOS
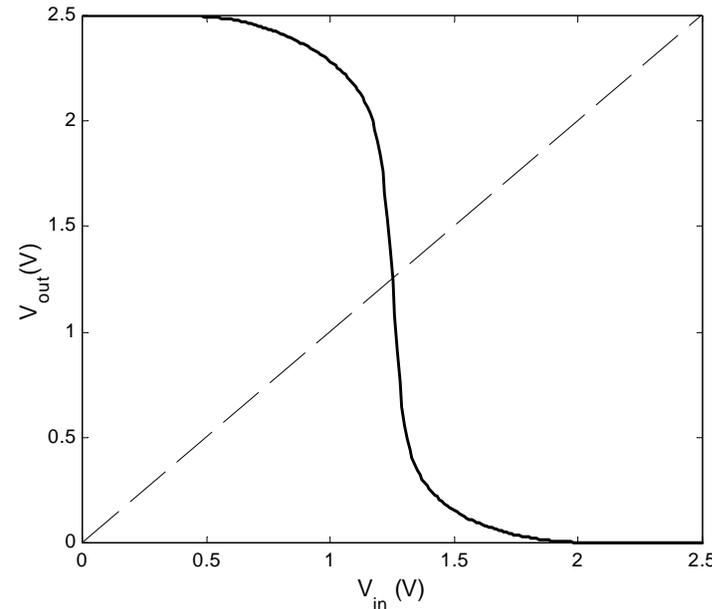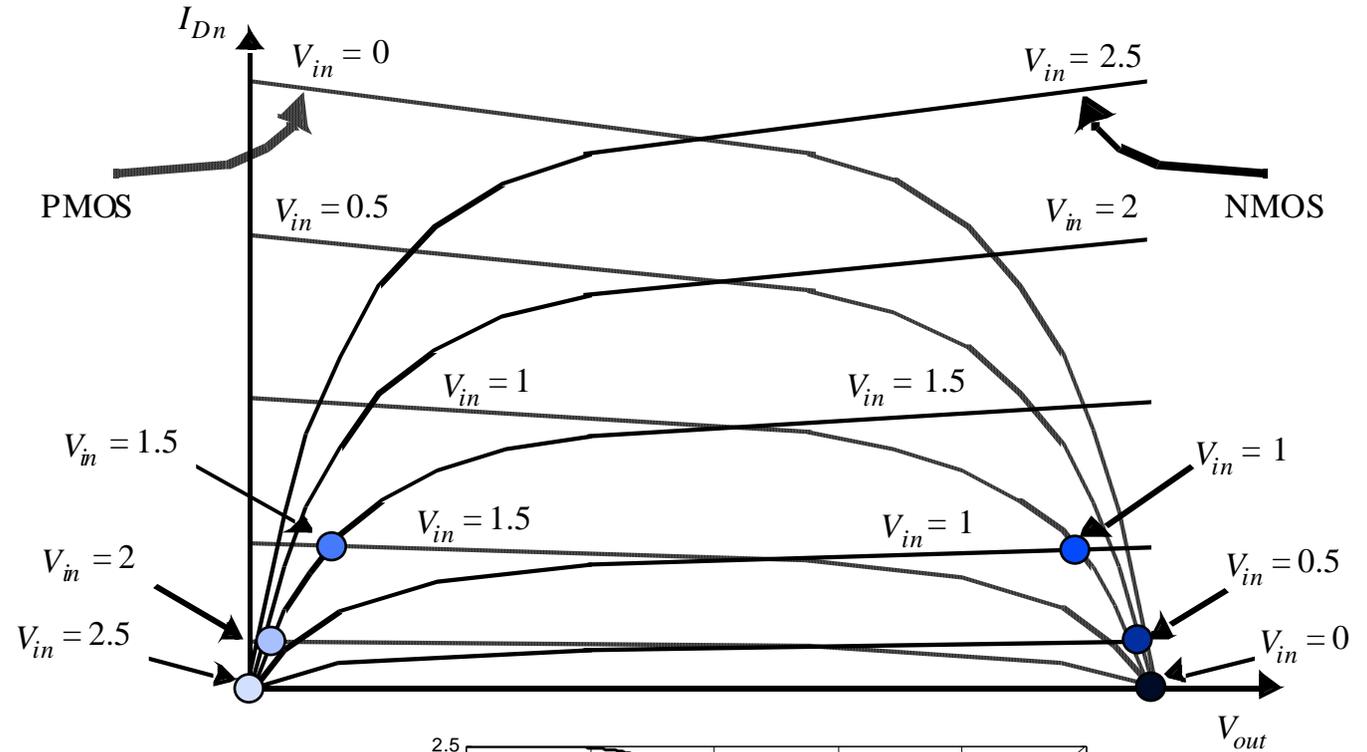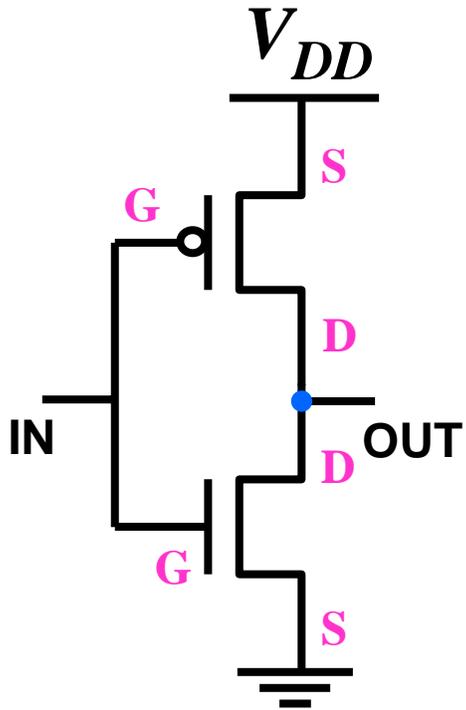
CMOS gates have:

- **Rail-to-rail swing (0V to V_{DD})**
- **Large noise margins**
- **"zero" static power dissipation**

There are 16 possible functions of 2 input variables:

X	Y																
						16 possible functions (F_0–F_{15})											
0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

0   X   Y   X XOR Y   X = Y   NOT Y   NOT X   1

X AND Y   X OR Y   X NOR Y   X NAND Y

NOT (X OR Y)   NOT (X AND Y)

In general, there are $2^{(2^n)}$ functions of n inputs

Common Logic Gates

Gate	Symbol	Truth-Table	Expression

NAND
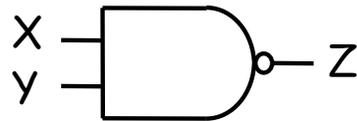
X	Y	Z
0	0	1
0	1	1
1	0	1
1	1	0

$$Z = \overline{X \cdot Y}$$

AND
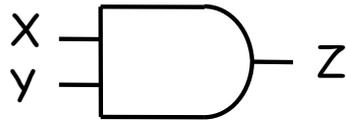
X	Y	Z
0	0	0
0	1	0
1	0	0
1	1	1

$$Z = X \cdot Y$$

NOR
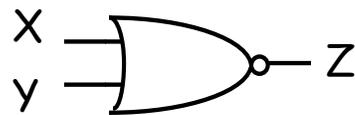
X	Y	Z
0	0	1
0	1	0
1	0	0
1	1	0

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

OR
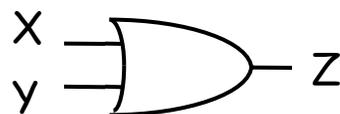
X	Y	Z
0	0	0
0	1	1
1	0	1
1	1	1

$$Z = X + Y$$

XOR

$(X \oplus Y)$

X, Y → Z
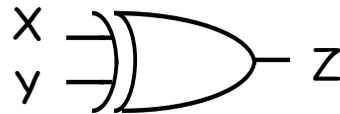
X	Y	Z
0	0	0
0	1	1
1	0	1
1	1	0

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

X or Y but not both
("inequality", "difference")

XNOR

$\overline{(X \oplus Y)}$

X, Y → Z
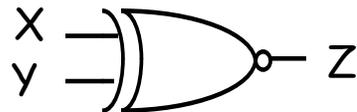
| X	Y	Z
0	0	1
0	1	0
1	0	0
1	1	1

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

X and Y the same
("equality")

Widely used in arithmetic structures such as adders and multipliers

V_{dd}

pullup: make this connection when we want $F(A_1,…,A_n) = 1$

$F(A_1,…,A_n)$

A_1
...
A_n

pulldown: make this connection when we want $F(A_1,…,A_n) = 0$
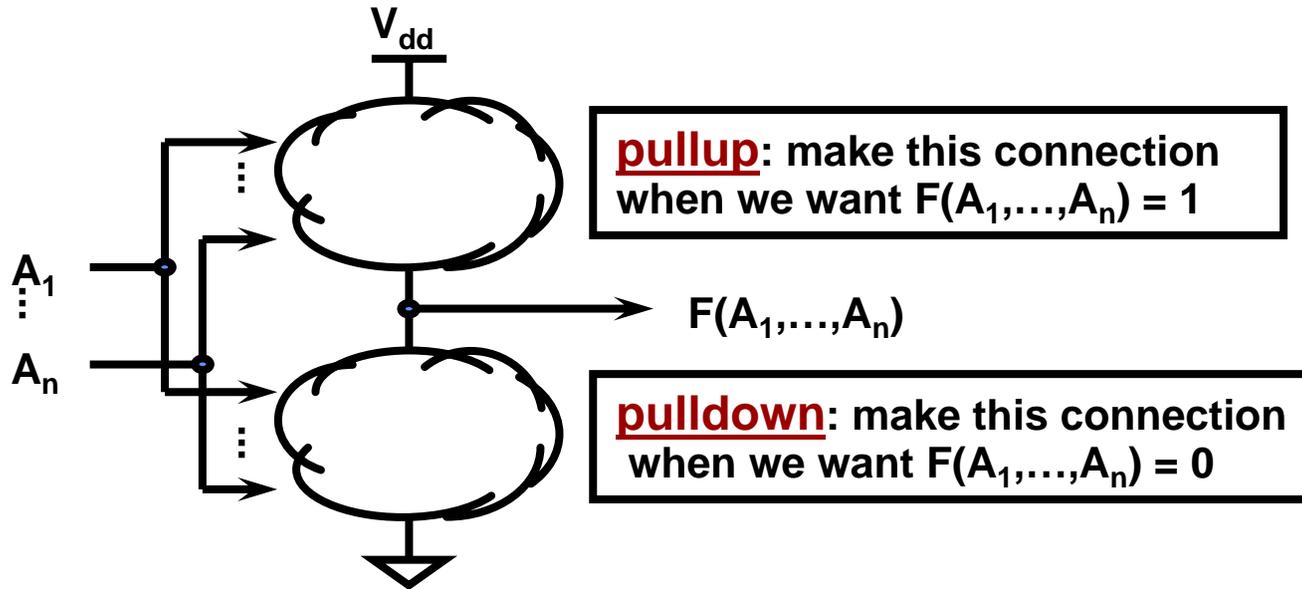
Note: **CMOS gates result in inverting functions!** (easier to build NAND vs. AND)

PUN

A   B   O

PDN

C_L
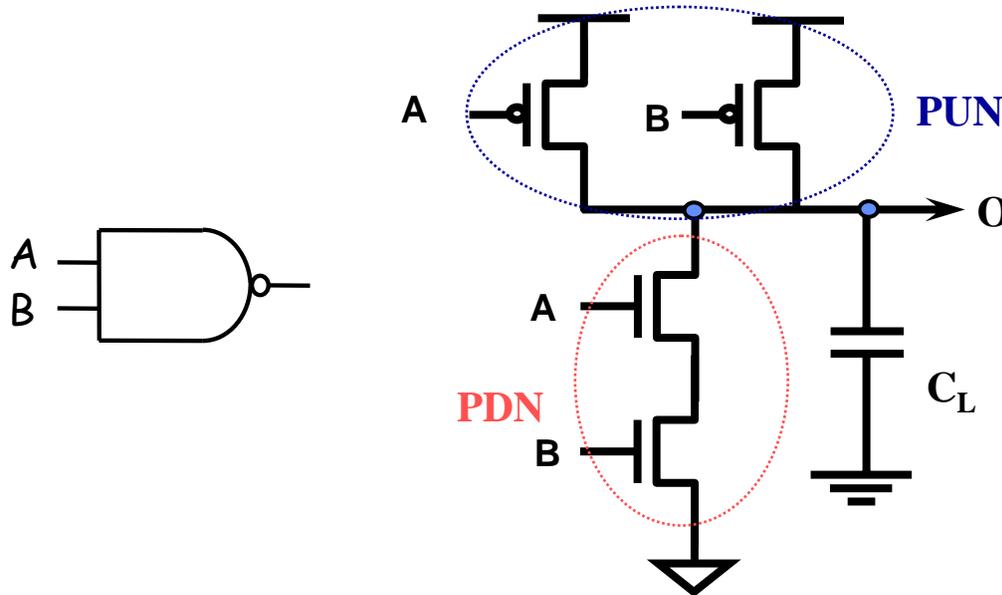
A	B	PDN	PUN	O
0	0	0ff	0n	1
0	1	0ff	0n	1
1	0	0ff	0n	1
1	1	0n	0ff	0

How do you build a 2-input NOR Gate?

Theorems of Boolean Algebra (I)

- **Elementary**

1. $X + 0 = X$	1D. $X \cdot 1 = X$
2. $X + 1 = 1$	2D. $X \cdot 0 = 0$
3. $X + X = X$	3D. $X \cdot X = X$
4. $(\overline{\overline{X}}) = X$	
5. $X + \overline{X} = 1$	5D. $X \cdot \overline{X} = 0$

- **Commutativity:**

6. $X + Y = Y + X$	6D. $X \cdot Y = Y \cdot X$

- **Associativity:**

7. $(X + Y) + Z = X + (Y + Z)$	7D. $(X \cdot Y) \cdot Z = X \cdot (Y \cdot Z)$

- **Distributivity:**

8. $X \cdot (Y + Z) = (X \cdot Y) + (X \cdot Z)$	8D. $X + (Y \cdot Z) = (X + Y) \cdot (X + Z)$

- **Uniting:**

9. $X \cdot Y + X \cdot \overline{Y} = X$	9D. $(X + Y) \cdot (X + \overline{Y}) = X$

- **Absorption:**

10. $X + X \cdot Y = X$	10D. $X \cdot (X + Y) = X$
11. $(X + \overline{Y}) \cdot Y = X \cdot Y$	11D. $(X \cdot \overline{Y}) + Y = X + Y$

Theorems of Boolean Algebra (II)

- **Factoring:**

12. $(X \bullet Y) + (X \bullet Z) =$ $X \bullet (Y + Z)$	12D. $(X + Y) \bullet (X + Z) =$ $X + (Y \bullet Z)$

- **Consensus:**

13. $(X \bullet Y) + (Y \bullet Z) + (\overline{X} \bullet Z) =$ $X \bullet Y + \overline{X} \bullet Z$	13D. $(X + Y) \bullet (Y + Z) \bullet (\overline{X} + Z) =$ $(X + Y) \bullet (\overline{X} + Z)$

- **De Morgan's:**

14. $\overline{(X + Y + \ldots)} = \overline{X} \bullet \overline{Y} \bullet \ldots$	14D. $\overline{(X \bullet Y \bullet \ldots)} = \overline{X} + \overline{Y} + \ldots$

- **Generalized De Morgan's:**

  15. $\overline{f(X1, X2, \ldots, Xn, 0, 1, +, \bullet)} = f(\overline{X1}, \overline{X2}, \ldots, \overline{Xn}, 1, 0, \bullet, +)$

- **Duality**
  - Dual of a Boolean expression is derived by replacing $\bullet$ by $+$, $+$ by $\bullet$, 0 by 1, and 1 by 0, and leaving variables unchanged
  - $f(X1, X2, \ldots, Xn, 0, 1, +, \bullet) \Leftrightarrow f(X1, X2, \ldots, Xn, 1, 0, \bullet, +)$

- **1-bit binary adder**
  - inputs: A, B, Carry-in
  - outputs: Sum, Carry-out

A → [+] → S
B →
Cin → → Cout
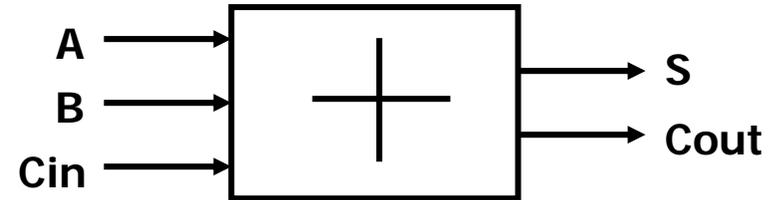
A	B	Cin	S	Cout
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Sum-of-Products Canonical Form

$$S = \overline{A}\ \overline{B}\ Cin + \overline{A}\ B\ \overline{Cin} + A\ \overline{B}\ \overline{Cin} + A\ B\ Cin$$

$$Cout = \overline{A}\ B\ Cin + A\ \overline{B}\ Cin + A\ B\ \overline{Cin} + A\ B\ Cin$$

- **Product term (or minterm)**
  - ANDed product of literals – input combination for which output is true
  - Each variable appears exactly once, in true or inverted form (but not both)

Simplify Boolean Expressions

$$\text{Cout} = \bar{A}\,B\,Cin + A\,\bar{B}\,Cin + A\,B\,\bar{Cin} + A\,B\,Cin$$

$$= \bar{A}\,B\,Cin + A\,B\,Cin + A\,\bar{B}\,Cin + A\,B\,Cin + A\,B\,\bar{Cin} + A\,B\,Cin$$

$$= (\bar{A} + A)\,B\,Cin + A\,(\bar{B} + B)\,Cin + A\,B\,(\bar{Cin} + Cin)$$

$$= B\,Cin + A\,Cin + A\,B$$

$$= (B + A)\,Cin + A\,B$$

$$S = \bar{A}\,\bar{B}\,Cin + \bar{A}\,B\,\bar{Cin} + A\,\bar{B}\,\bar{Cin} + A\,B\,Cin$$

$$= (\bar{A}\,\bar{B} + A\,B)\,Cin + (A\,\bar{B} + \bar{A}\,B)\,\bar{Cin}$$

$$= \overline{(A \oplus B)}\,Cin + (A \oplus B)\,\bar{Cin}$$

$$= A \oplus B \oplus Cin$$

Sum-of-Products & Product-of-Sum

- **Product term** (or minterm): ANDed product of literals – input combination for which output is true

A	B	C	minterms	
0	0	0	$\overline{A}\ \overline{B}\ \overline{C}$	m0
0	0	1	$\overline{A}\ \overline{B}\ C$	m1
0	1	0	$\overline{A}\ B\ \overline{C}$	m2
0	1	1	$\overline{A}\ B\ C$	m3
1	0	0	$A\ \overline{B}\ \overline{C}$	m4
1	0	1	$A\ \overline{B}\ C$	m5
1	1	0	$A\ B\ \overline{C}$	m6
1	1	1	$A\ B\ C$	m7

short-hand notation form in terms of 3 variables

F in canonical form:

$$F(A, B, C) = \Sigma m(1,3,5,6,7)$$
$$= m1 + m3 + m5 + m6 + m7$$
$$F = \overline{A}\ \overline{B}\ C + \overline{A}\ B\ C + A\ \overline{B}\ C + A\ B\ \overline{C} + ABC$$

canonical form $\neq$ minimal form

$$F(A, B, C) = \overline{A}\ \overline{B}\ C + \overline{A}\ B\ C + A\overline{B}\ C + ABC + AB\overline{C}$$
$$= (\overline{A}\ \overline{B} + \overline{A}\ B + A\overline{B} + AB)C + AB\overline{C}$$
$$= ((\overline{A} + A)(\overline{B} + B))C + AB\overline{C}$$
$$= C + AB\overline{C} = AB\overline{C} + C = AB + C$$

- **Sum term** (or maxterm) - ORed sum of literals – input combination for which output is false

A	B	C	maxterms	
0	0	0	$A + B + C$	M0
0	0	1	$A + B + \overline{C}$	M1
0	1	0	$A + \overline{B} + C$	M2
0	1	1	$A + \overline{B} + \overline{C}$	M3
1	0	0	$\overline{A} + B + C$	M4
1	0	1	$\overline{A} + B + \overline{C}$	M5
1	1	0	$\overline{A} + \overline{B} + C$	M6
1	1	1	$\overline{A} + \overline{B} + \overline{C}$	M7

short-hand notation for maxterms of 3 variables

F in canonical form:

$$F(A, B, C) = \Pi M(0,2,4)$$
$$= M0 \cdot M2 \cdot M4$$
$$= (A + B + C)(A + \overline{B} + C)(\overline{A} + B + C)$$

canonical form $\neq$ minimal form

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

Mapping Between Forms

1. **Minterm to Maxterm conversion:**
   rewrite minterm shorthand using maxterm shorthand
   replace minterm indices with the indices not already used

   E.g., $F(A,B,C) = \Sigma m(3,4,5,6,7) = \Pi M(0,1,2)$

2. **Maxterm to Minterm conversion:**
   rewrite maxterm shorthand using minterm shorthand
   replace maxterm indices with the indices not already used

   E.g., $F(A,B,C) = \Pi M(0,1,2) = \Sigma m(3,4,5,6,7)$

3. **Minterm expansion of F to Minterm expansion of F':**
   in minterm shorthand form, list the indices not already used in F

   E.g., $F(A,B,C) = \Sigma m(3,4,5,6,7)$ $\longrightarrow$ $F'(A,B,C) = \Sigma m(0,1,2)$
   $\qquad\qquad = \Pi M(0,1,2)$ $\longrightarrow$ $\qquad\qquad = \Pi M(3,4,5,6,7)$

4. **Minterm expansion of F to Maxterm expansion of F':**
   rewrite in Maxterm form, using the same indices as F

   E.g., $F(A,B,C) = \Sigma m(3,4,5,6,7)$ $\longrightarrow$ $F'(A,B,C) = \Pi M(3,4,5,6,7)$
   $\qquad\qquad = \Pi M(0,1,2)$ $\longrightarrow$ $\qquad\qquad = \Sigma m(0,1,2)$

- **Key tool to simplification: $A (\overline{B} + B) = A$**

- **Essence of simplification of two-level logic**
  - Find two element subsets of the ON-set where only one variable changes its value – this single varying variable can be eliminated and a single product term used to represent both elements

$$F = \overline{A}\,\overline{B} + A\overline{B} = (\overline{A} + A)\overline{B} = \overline{B}$$

A	B	F
0	0	1
0	1	0
1	0	1
1	1	0

B has the same value in both on-set rows
– B remains

A has a different value in the two rows
– A is eliminated

Boolean Cubes
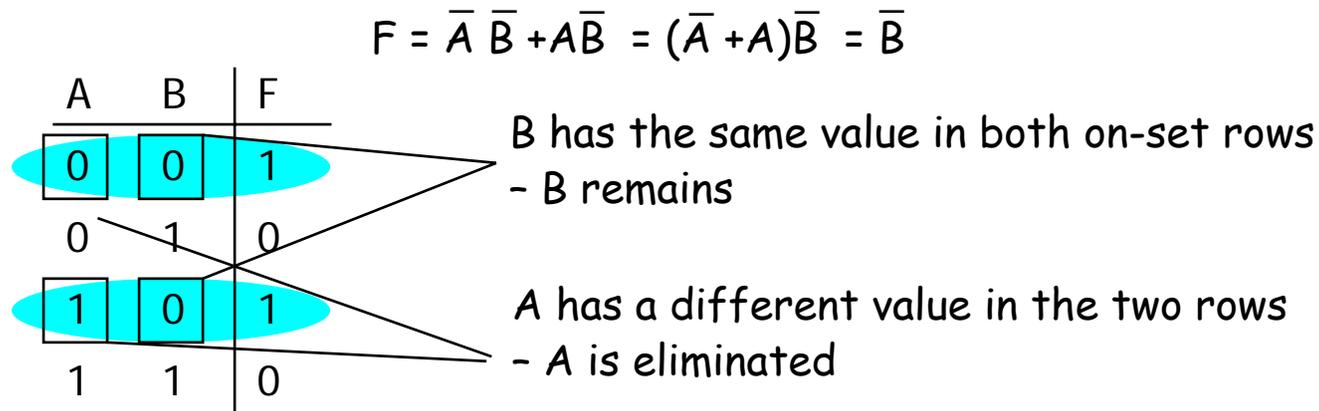
- **Just another way to represent truth table**

- **Visual technique for identifying when the uniting theorem can be applied**

- **n input variables = n-dimensional "cube"**

1-cube

2-cube

3-cube

4-cube

Mapping Truth Tables onto Boolean Cubes
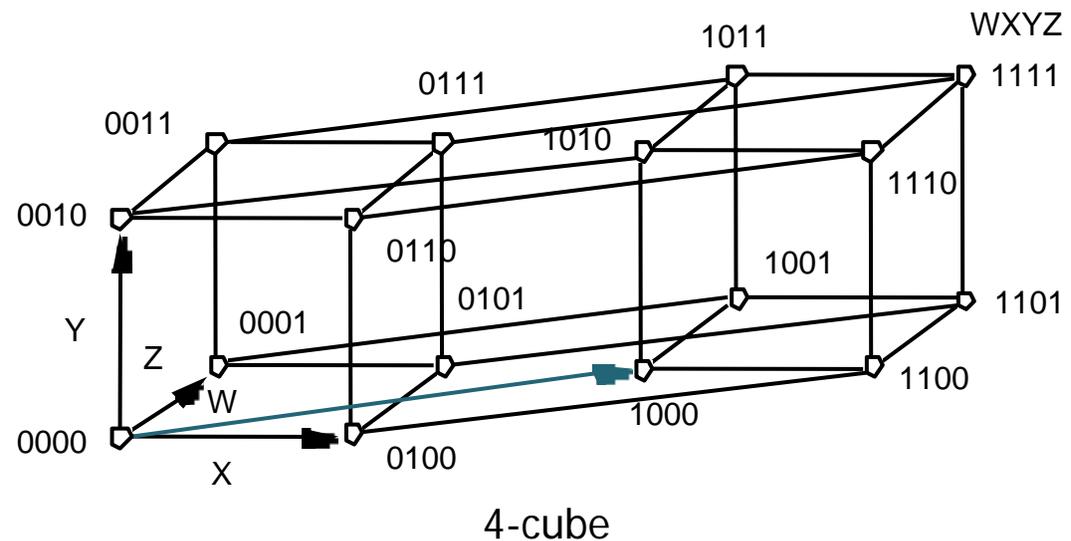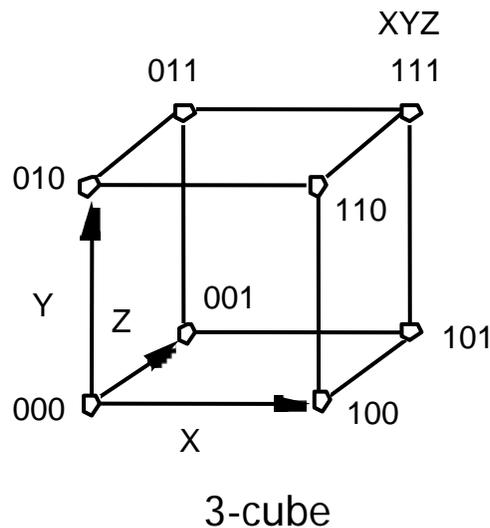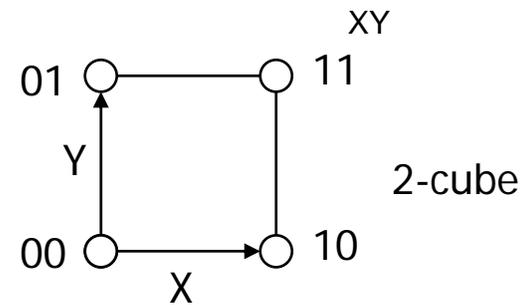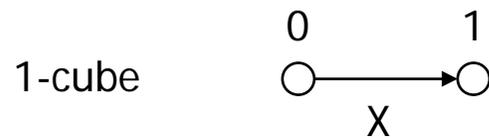
- **Uniting theorem**

A	B	F
0	0	1
0	1	0
1	0	1
1	1	0
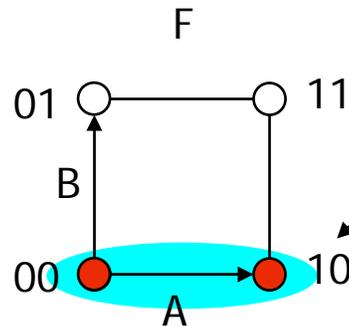
Circled group of the on-set is called the *adjacency* plane. Each adjacency plane corresponds to a product term.

ON-set = solid nodes
OFF-set = empty nodes

A varies within face, B does not this face represents the literal $\overline{B}$

- **Three variable example: Binary full-adder carry-out logic**

A	B	Cin	Cout
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

$(\overline{A}+A)BCin$

$AB(\overline{Cin}+Cin)$
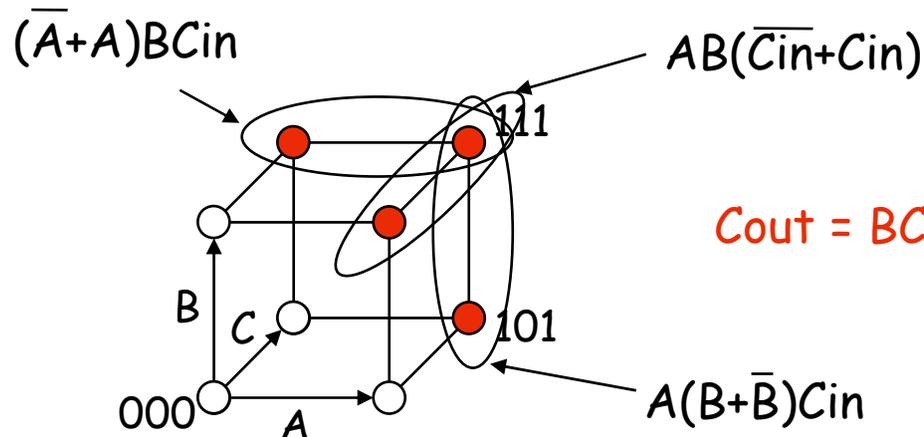
Cout = BCin+AB+ACin

$A(B+\overline{B})Cin$

The on-set is completely covered by the combination (OR) of the subcubes of lower dimensionality - note that "111" is covered three times

Higher Dimension Cubes
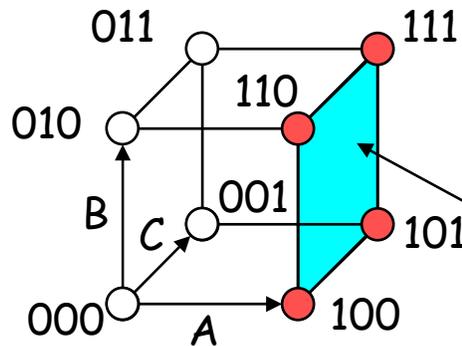
$F(A,B,C) = \Sigma m(4,5,6,7)$

on-set forms a square
i.e., a cube of dimension 2 (2-D adjacency plane)

represents an expression in one variable
i.e., 3 dimensions – 2 dimensions

A is asserted (true) and unchanged
B and C vary

This subcube represents the literal A

- **In a 3-cube (three variables):**
  - □ **0-cube, i.e., a single node, yields a term in 3 literals**
  - □ **1-cube, i.e., a line of two nodes, yields a term in 2 literals**
  - □ **2-cube, i.e., a plane of four nodes, yields a term in 1 literal**
  - □ **3-cube, i.e., a cube of eight nodes, yields a constant term "1"**

- **In general,**
  - □ **m-subcube within an n-cube (m < n) yields a term with n – m literals**

Karnaugh Maps

- **Alternative to truth-tables to help visualize adjacencies**
    - Guide to applying the uniting theorem - On-set elements with only one variable changing value are adjacent unlike in a linear truth-table


A	B	F
0	0	1
0	1	0
1	0	1
1	1	0

- **Numbering scheme based on Gray–code**
   - e.g., 00, 01, 11, 10 (only a single bit changes in code for adjacent map cells)

2-variable K-map

**3-variable K-map**

4-variable K-map

K-Map Examples

A B

	A			
Cin	00	01	11	10
0	0	0	1	0
1	0	1	1	1

B

Cout =

AB

	A			
C	00	01	11	10
0	0	0	1	1
1	0	0	1	1

B

F(A,B,C) =

AB

	A			
C	00	01	11	10
0	1	0	0	1
1	0	0	1	1

B

F(A,B,C) = $\Sigma m(0,4,5,7)$

F =

AB

	A			
C	00	01	11	10
0	0	1	1	0
1	1	1	0	0

B
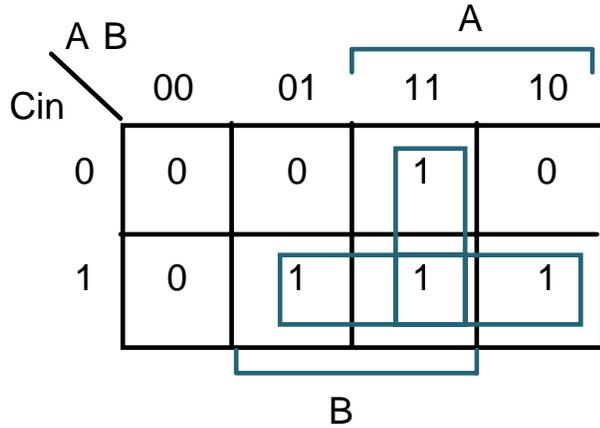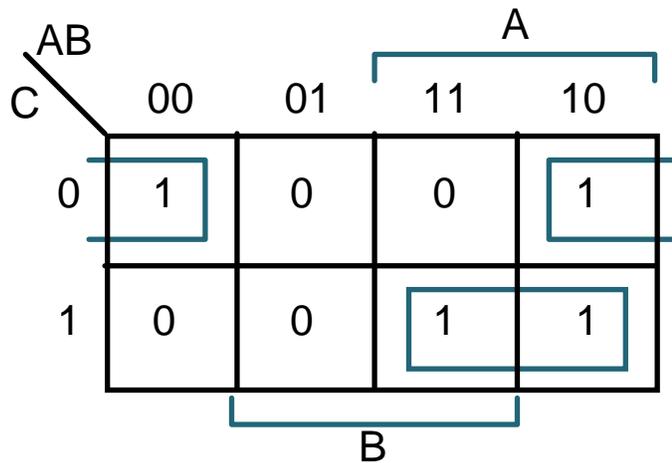
F' simply replace 1's with 0's and vice versa

F'(A,B,C) = $\Sigma m(1,2,3,6)$

F' =

Four Variable Karnaugh Map
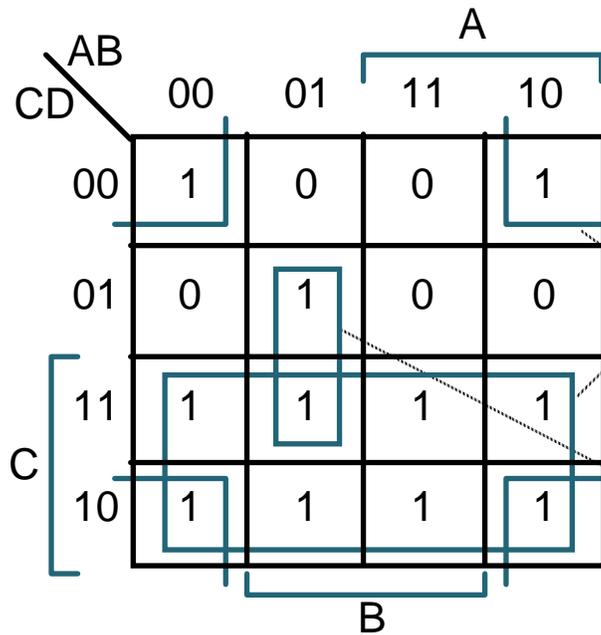
$$F(A,B,C,D) = \Sigma m(0,2,3,5,6,7,8,10,11,14,15)$$

$$F = C + \overline{A}\,B\,D + \overline{B}\,\overline{D}$$

Find the smallest number of the largest possible subcubes that cover the ON-set
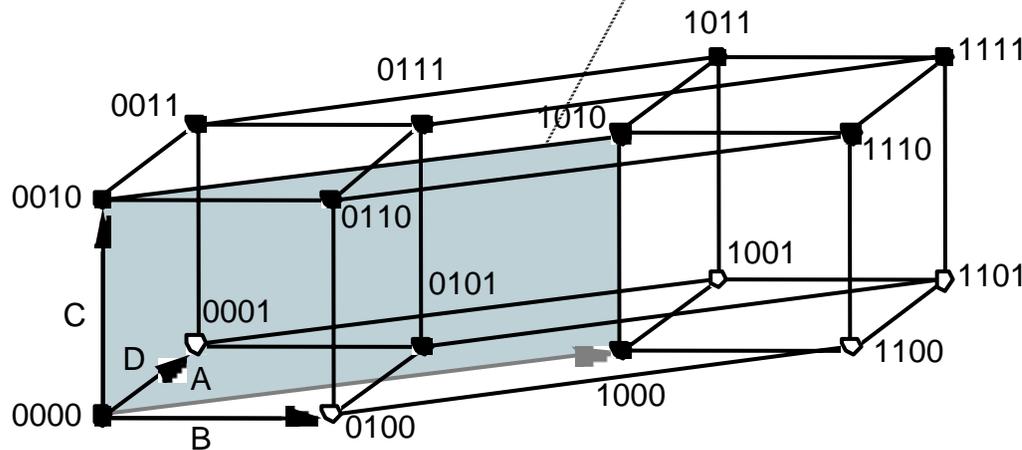
K-map Corner Adjacency Illustrated in the 4-Cube

Don't Cares can be treated as 1's or 0's if it is advantageous to do so
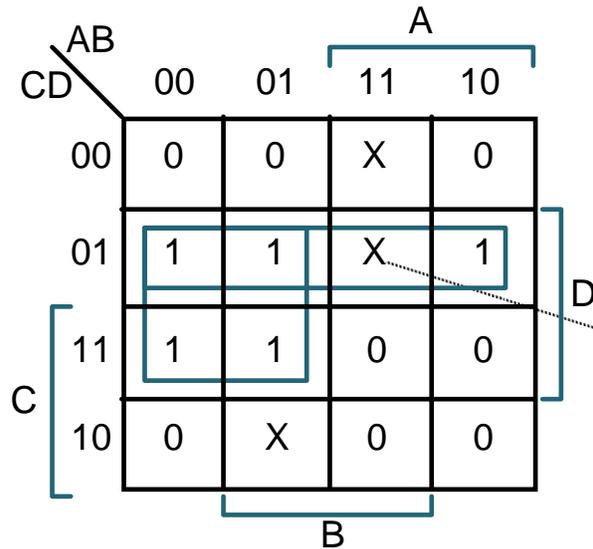
$$F(A,B,C,D) = \Sigma m(1,3,5,7,9) + \Sigma d(6,12,13)$$

$$F = \overline{A}\,D + \overline{B}\,\overline{C}\,D \quad \text{w/o don't cares}$$

$$F = \overline{C}\,D + \overline{A}\,D \quad \text{w/ don't cares}$$

**By treating this DC as a "1", a 2-cube can be formed rather than one 0-cube**

In PoS form: $F = D\,(\overline{A} + \overline{C})$

Equivalent answer as above, but fewer literals
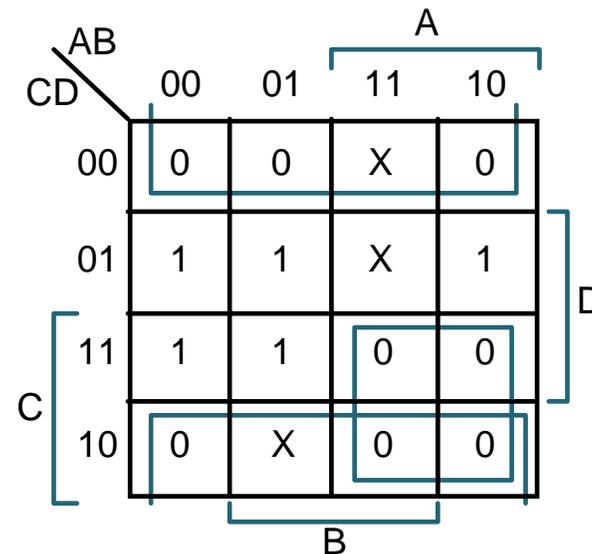
Static Hazards: Consider this function:

$$F = A * \overline{C} + B * C$$

AB

C	00	01	11	10
0	0	0	1	1
1	0	1	1	0

A = B = 1

Glitch

Gate Delay
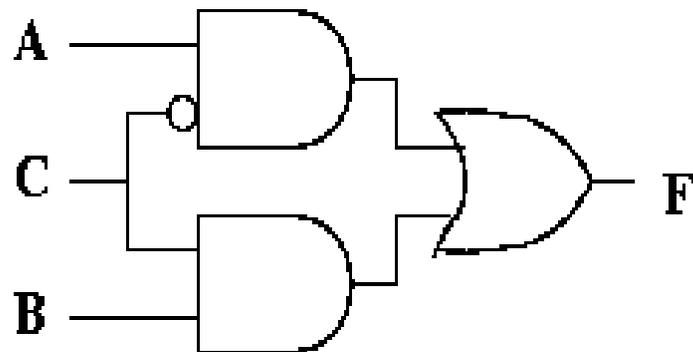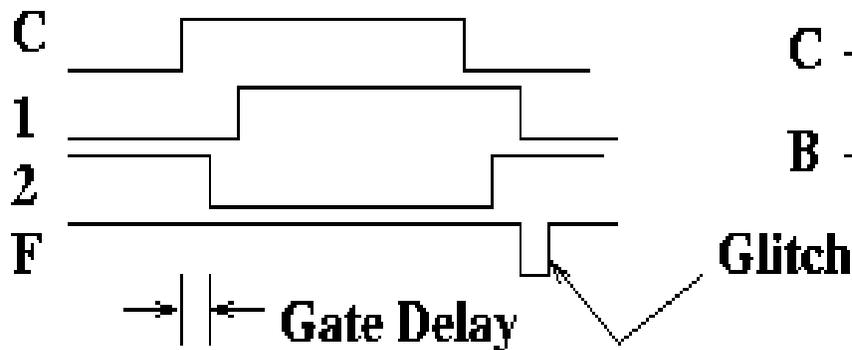
Implemented with MSI gates:
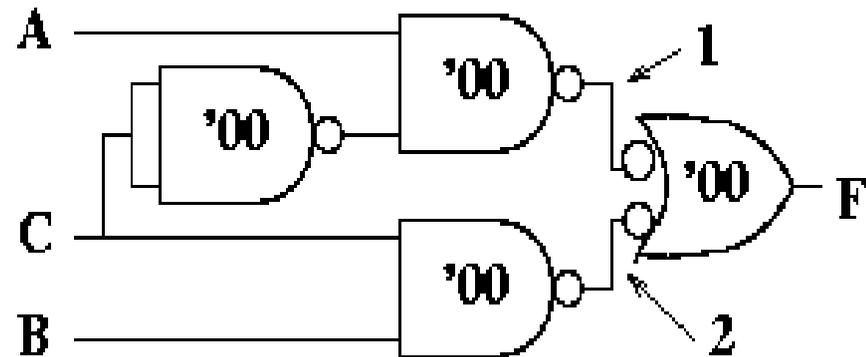
The glitch is the result of timing differences in parallel data paths. It is associated with the function jumping between groupings or product terms on the K-map. To fix it, cover it up with another grouping or product term!
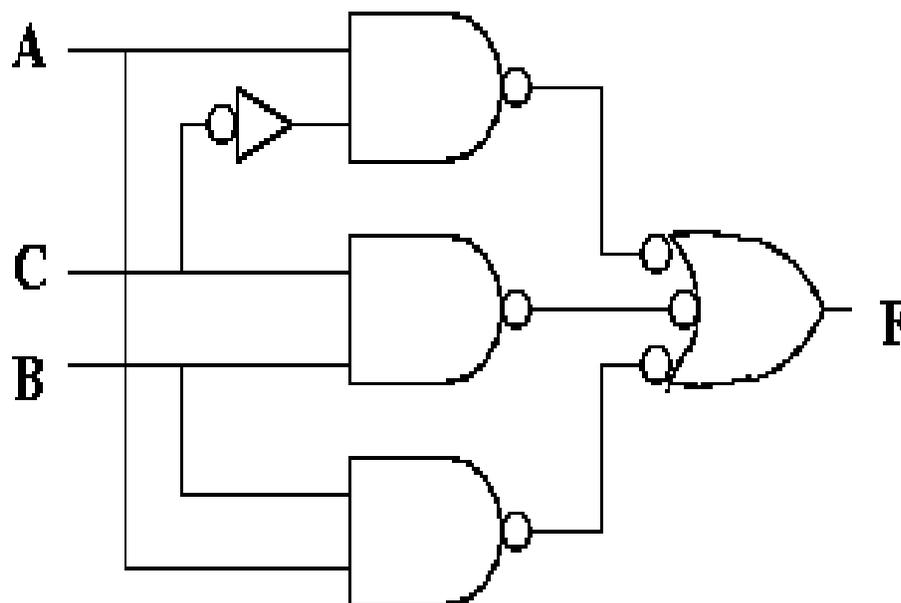
	00	01	11	10
0	0	0	1	1
1	0	1	1	0

$$F = A * \overline{C} + B * C + A * B$$

- **In general, it is difficult to avoid hazards – need a robust design methodology to deal with hazards.**