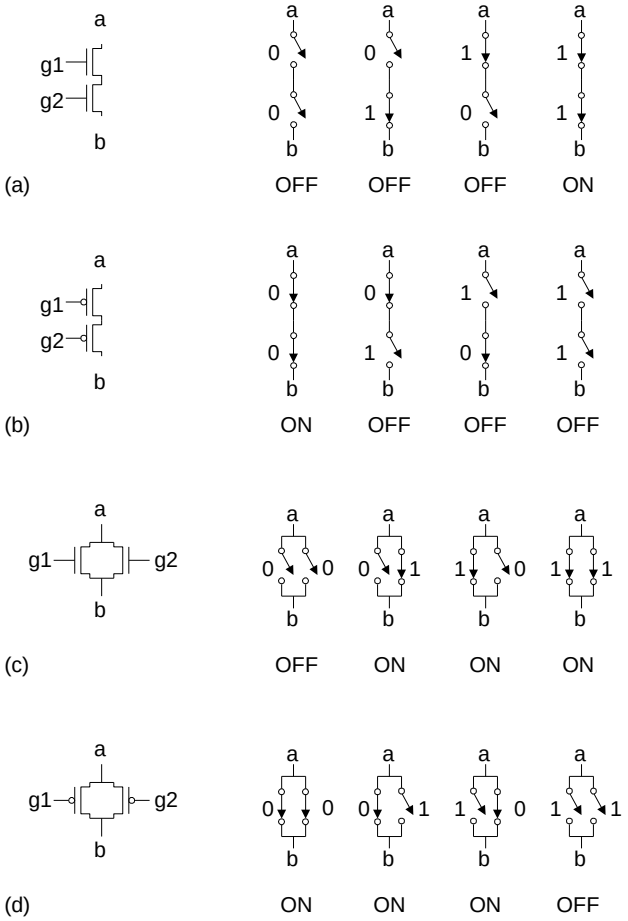
CMOS VLSI Design

Digital Design

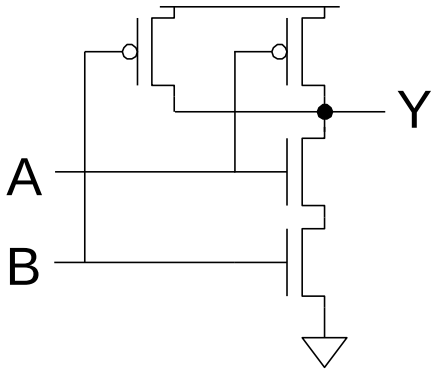
Series and Parallel

- nMOS: 1 = ON
- pMOS: 0 = ON
- *Series*: both must be ON
- *Parallel*: either can be ON



Conduction Complement

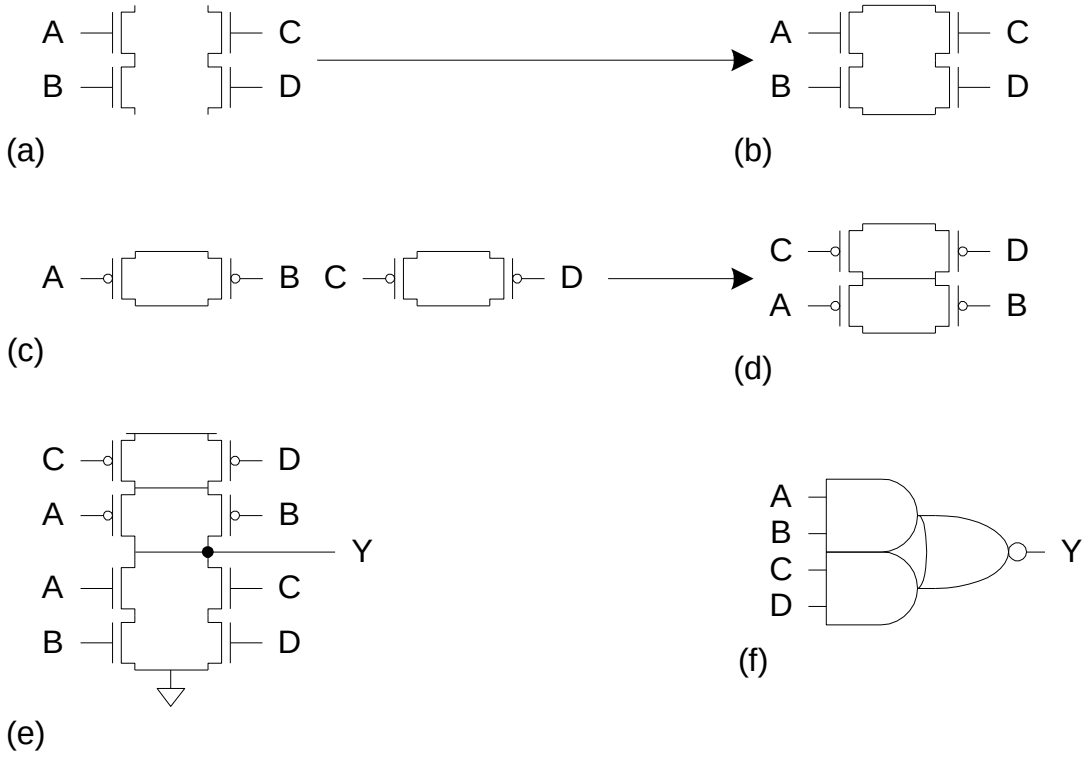
- Complementary CMOS gates always produce 0 or 1
- Ex: NAND gate
  - Series nMOS: Y=0 when both inputs are 1
  - Thus Y=1 when either input is 0
  - Requires parallel pMOS



- Rule of *Conduction Complements*
  - Pull-up network is complement of pull-down
  - Parallel -> series, series -> parallel

Compound Gates

- *Compound gates* can do any inverting function
- Ex: Y = (A.B + C.D)'

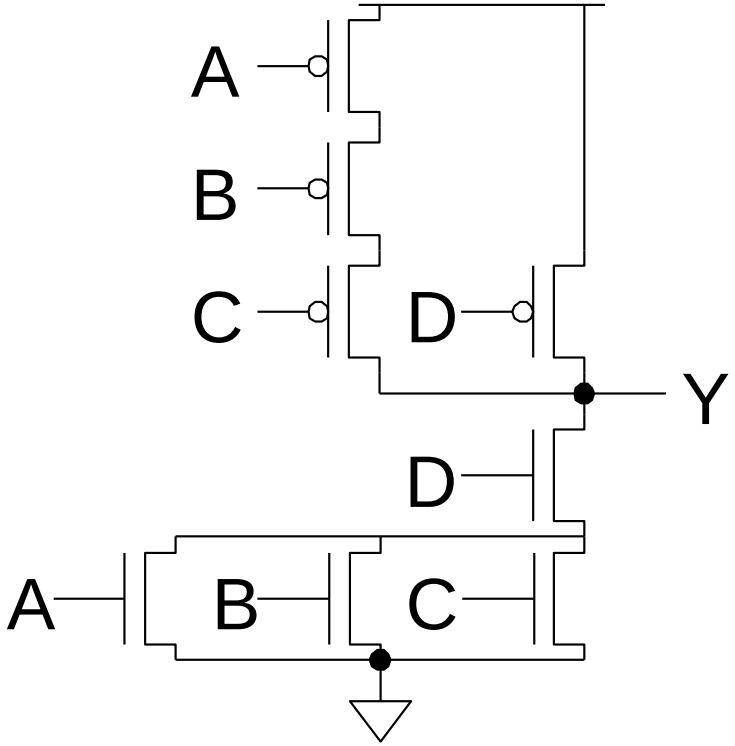


Example:

- Y = ((A+B+C).D)’

Example:

- Y = ((A+B+C).D)'

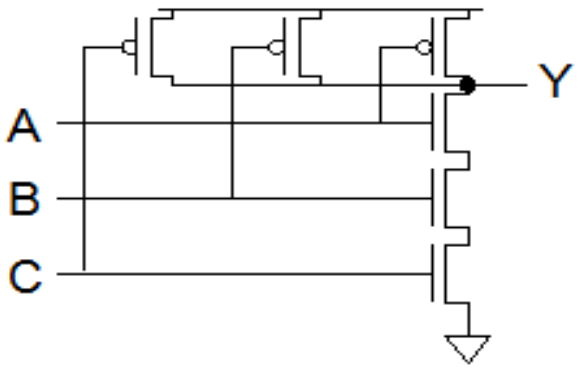


CMOS Gate Design

- Activity:
  - Sketch a 3-input CMOS NAND gate

3-input NAND Gate

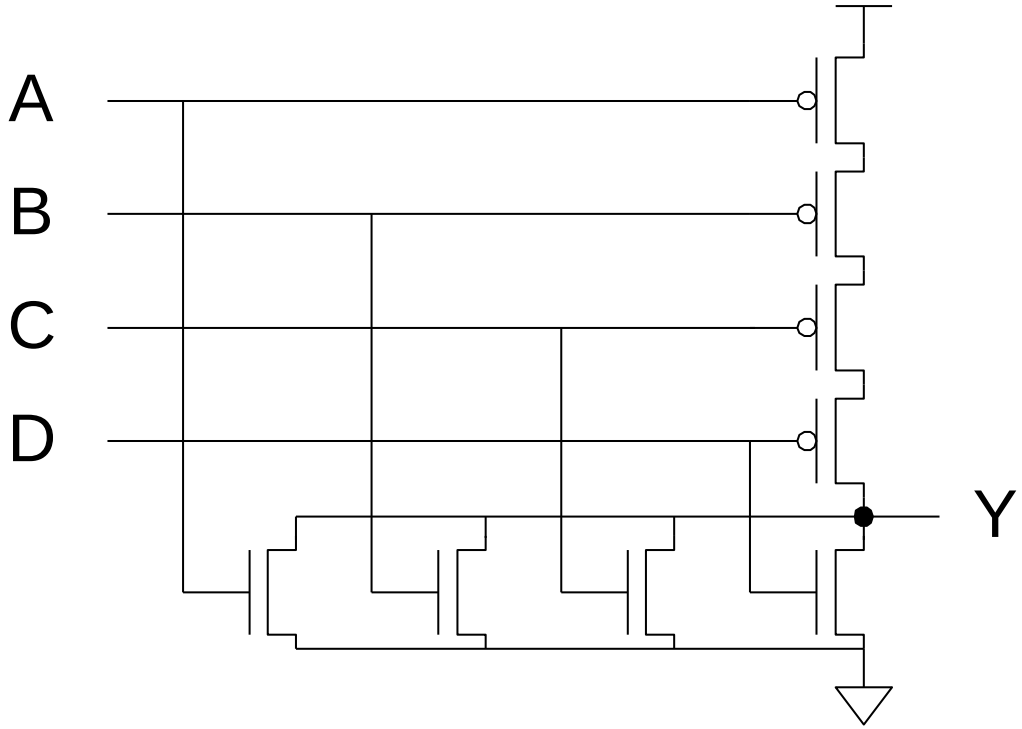
- Y pulls low if ALL inputs are 1
- Y pulls high if ANY input is 0



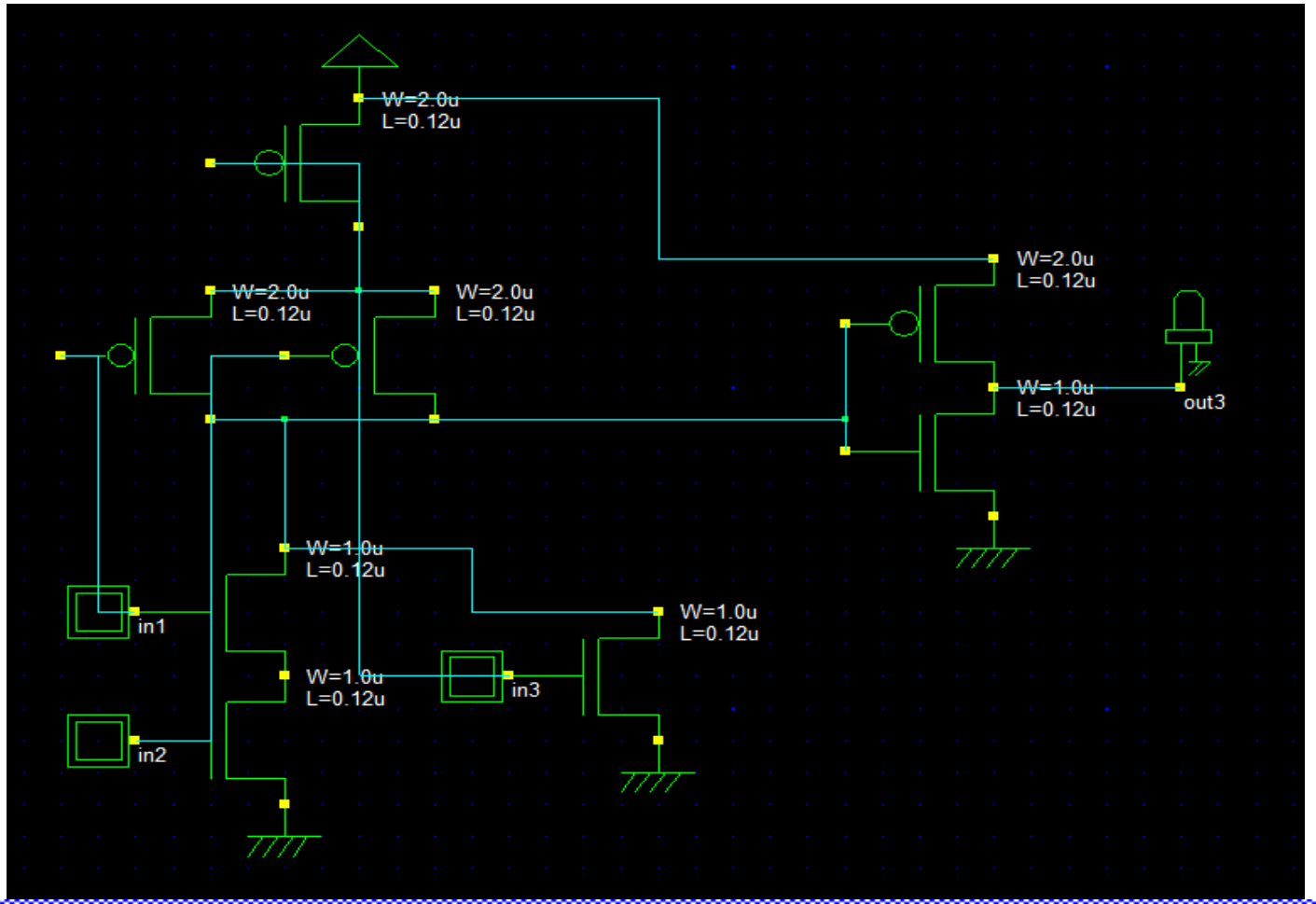
0: Introduction CMOS VLSI Design Slide 22

CMOS Gate Design

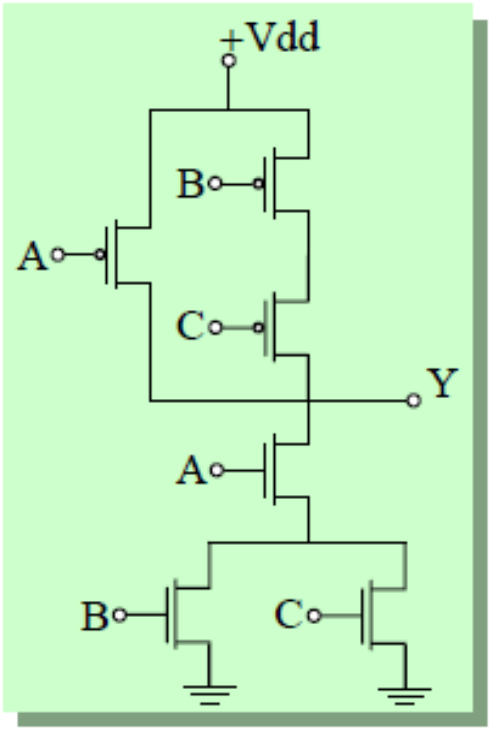
- Activity:
  - Sketch a 4-input CMOS NOR gate



Example: A+(BC)


W=2.0u
L=0.12u
W=2.0u
L=0.12u
W=2.0u
L=0.12u
W=2.0u
L=0.12u
W=1.0u
L=0.12u
out3
W=1.0u
L=0.12u
W=1.0u
L=0.12u
in1
W=1.0u
L=0.12u
in3
in2

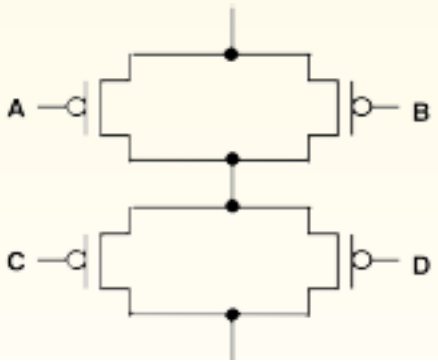
- Realization of More Complicated Gate Circuits
  - a) $Y = \overline{A(B+C)}$
    - Transistor Level
      - It need only 6 transistors

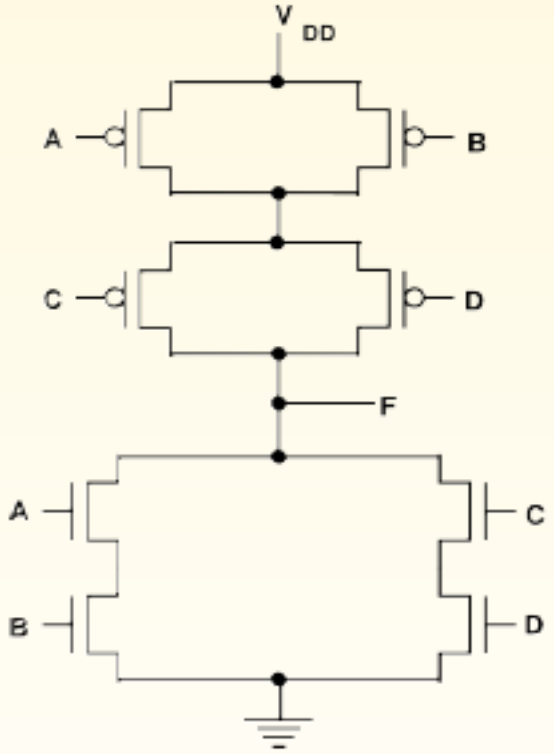


Constructing Complex Gates

Example: $F = \overline{(A \cdot B) + (C \cdot D)}$

4. Construct P-network by taking complement of N-expression $(AB + CD)$, which gives the expression, $(\overline{A} + \overline{B}) \cdot (\overline{C} + \overline{D})$
5. Combine P and N circuits





Example of Compound Gate

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

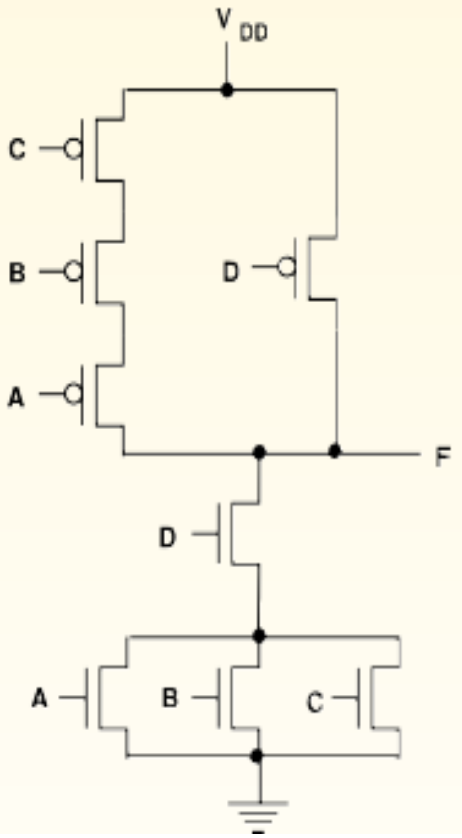
Note:

N- and P- graphs are duals of each other

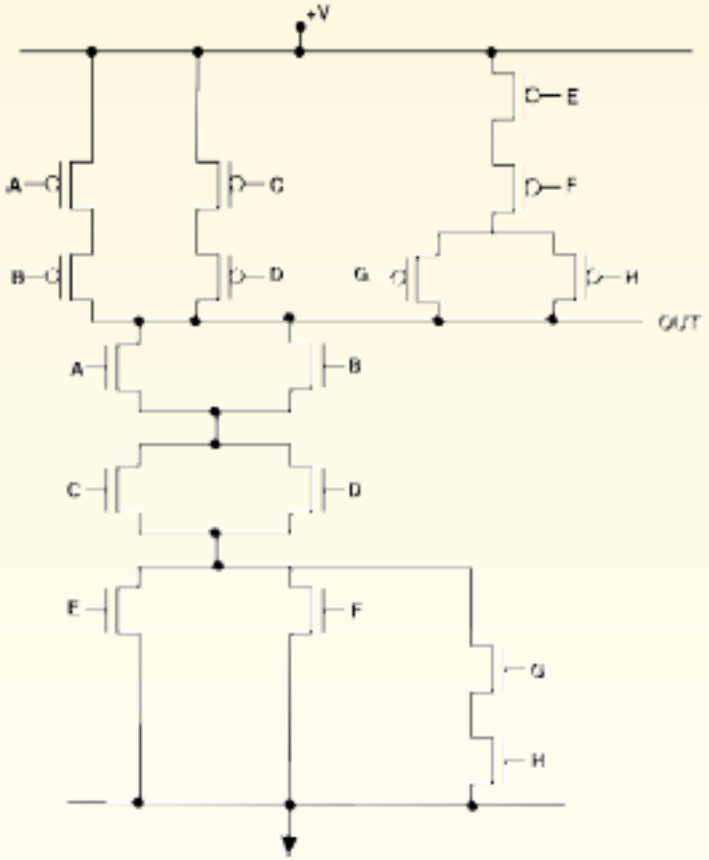
In this case, the function is the complement of the switching function between F and GND

Question: Does it make any difference to the function if the transistor with input D is connected between the parallel A, B, C, transistors and GND?

What about the electrical behavior?

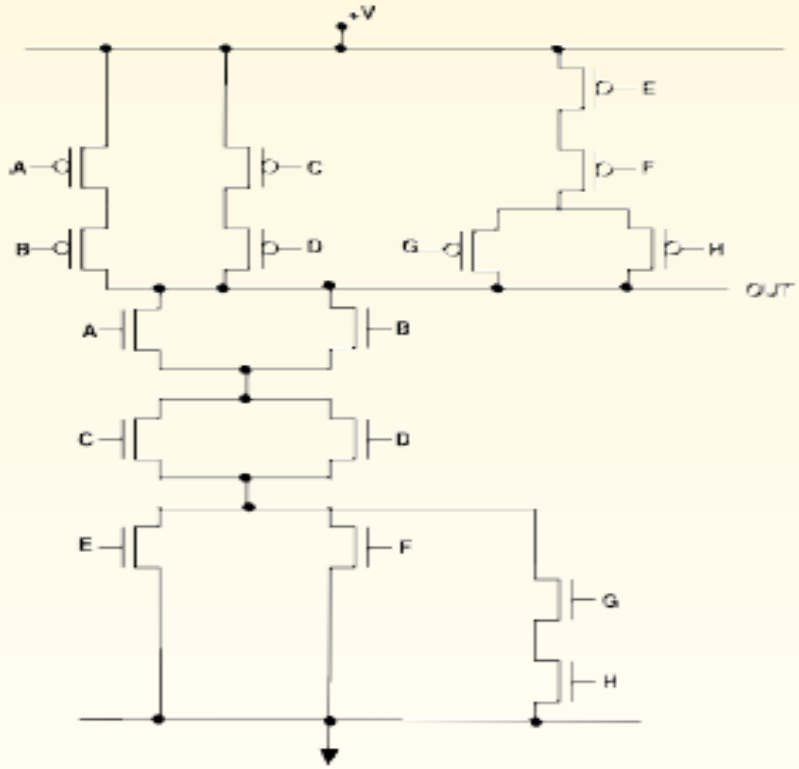


Example of More Complex Gate



$$OUT = \overline{(A+B)\cdot(C+D)\cdot(E+F+(G\cdot H))}$$

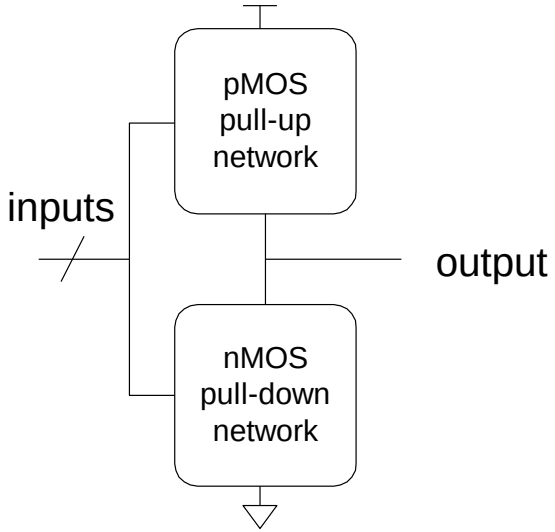
Example of More Complex Gate



$$OUT = \overline{(A+B)\cdot(C+D)\cdot(E+F+(G\cdot H))}$$

Complementary CMOS

- Complementary CMOS logic gates
  - nMOS *pull-down network*
  - pMOS *pull-up network*
  - a.k.a. static CMOS



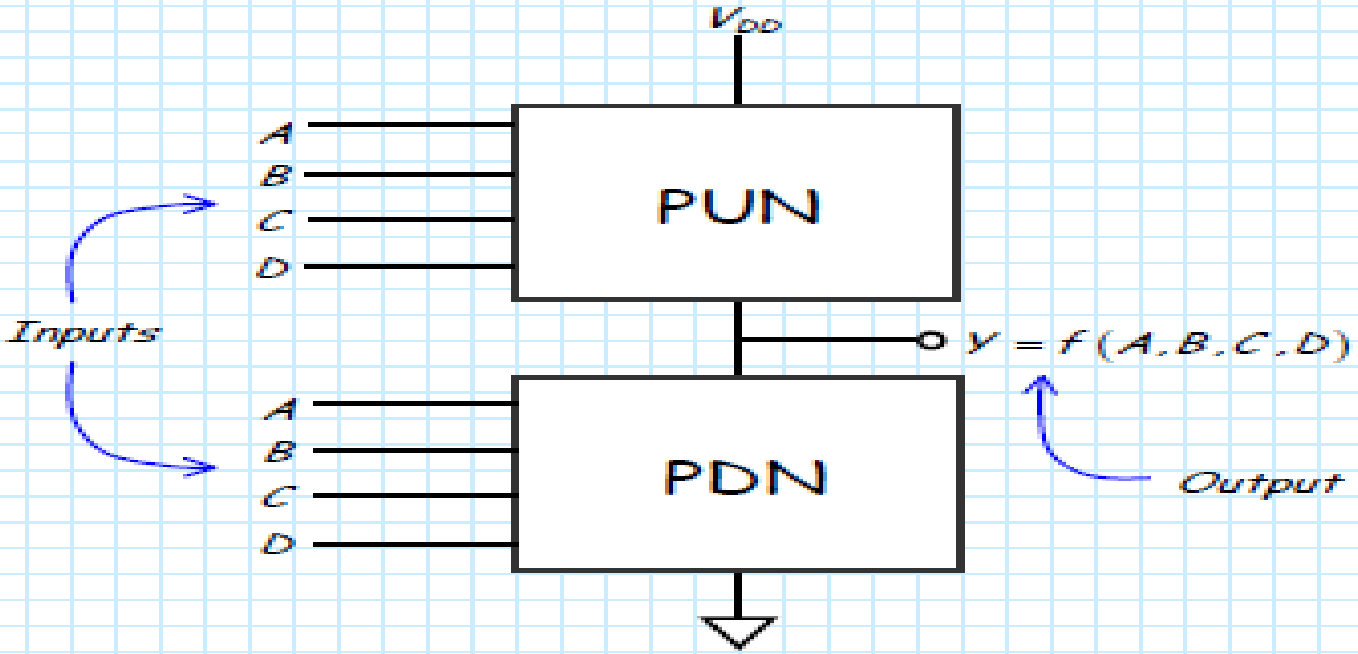
CMOS Device Structure

- For every CMOS device, there are essentially two separate circuits:

  1) *The Pull-Up Network*

  2) *The Pull-Down Network*

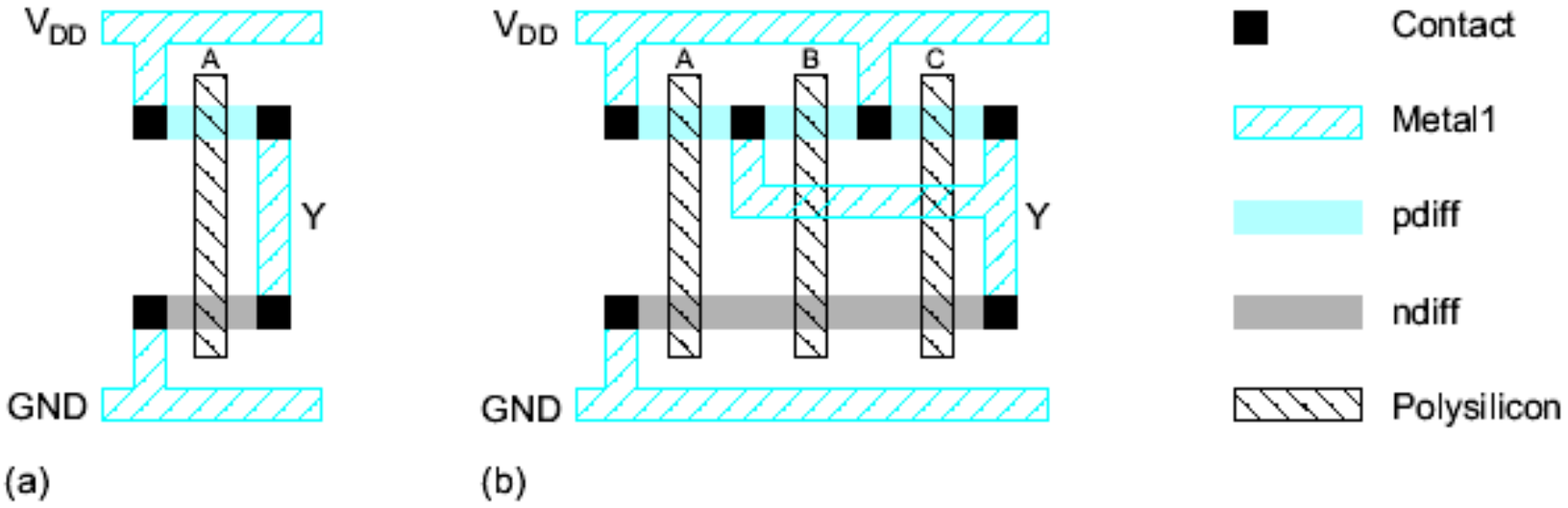
The basic **CMOS structure** is:



- A CMOS logic gate **must** be in one of **two** states!

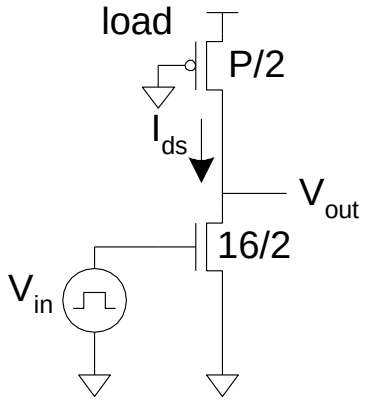
Stick Diagrams

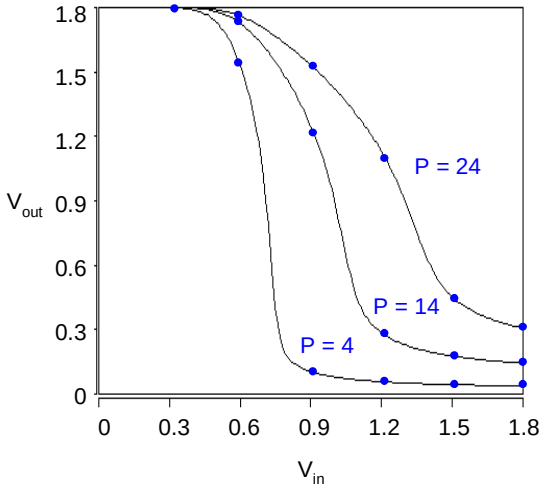
- *Stick diagrams* help plan layout quickly
  - Need not be to scale
  - Draw with color pencils or dry-erase markers



Pseudo-nMOS

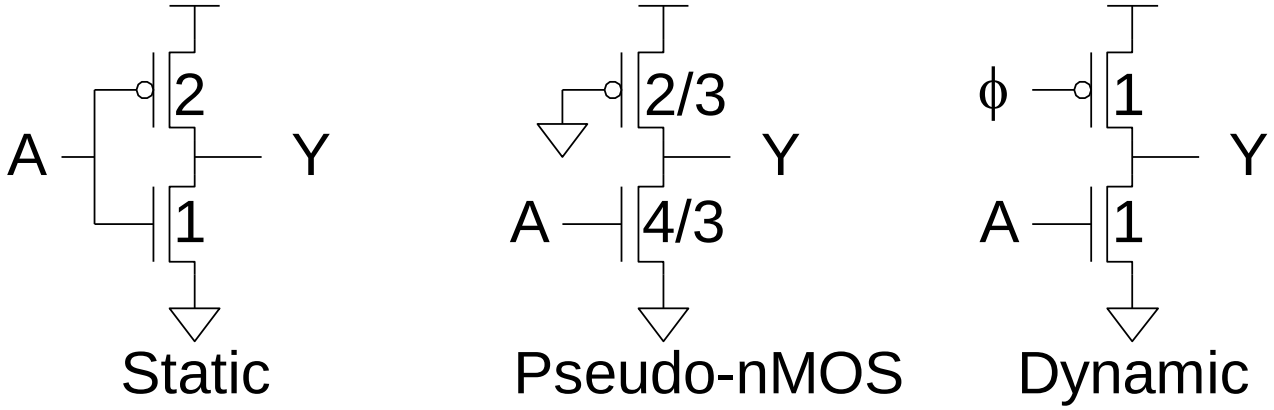
- In the old days, nMOS processes had no pMOS
  - Instead, use pull-up transistor that is always ON
- In CMOS, use a pMOS that is always ON
  - *Ratio* issue
  - Make pMOS about ¼ effective strength of pulldown network

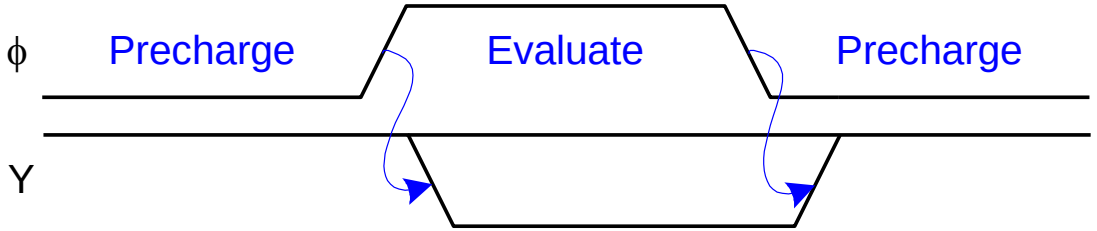




Dynamic Logic

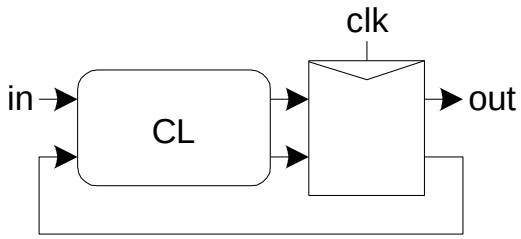
- *Dynamic* gates uses a clocked pMOS pullup
- Two modes: *precharge* and *evaluate*



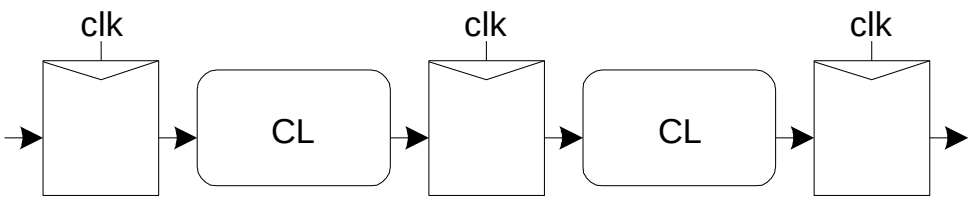


Sequencing

- *Combinational logic*
  - output depends on current inputs
- *Sequential logic*
  - output depends on current and previous inputs
  - Requires separating previous, current, future
  - Called *state* or *tokens*
  - Ex: FSM, pipeline



Finite State Machine

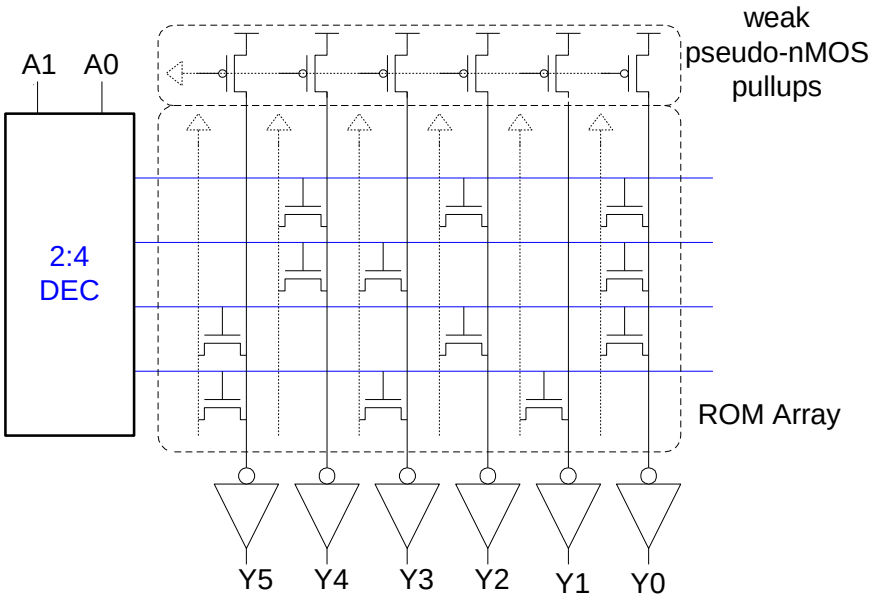


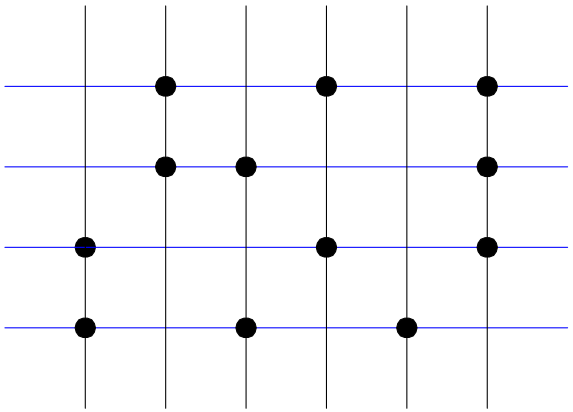
Pipeline

ROM Example

- 4-word x 6-bit ROM
  - Represented with dot diagram
  - Dots indicate 1's in ROM

```
Word 0: 010101
Word 1: 011001
Word 2: 100101
Word 3: 101010
```

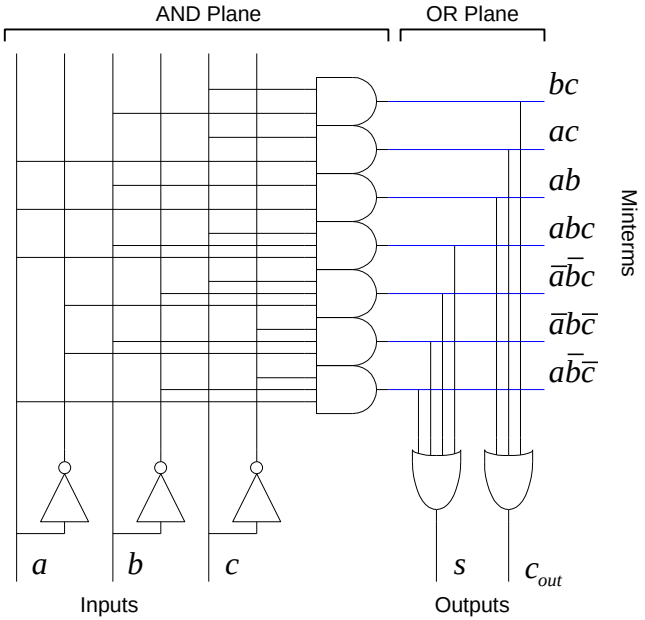




Looks like 6 4-input pseudo-nMOS NORs

PLAs

- A *Programmable Logic Array* performs any function in sum-of-products form.
- *Literals*: inputs & complements
- *Products / Minterms*: AND of literals
- *Outputs*: OR of Minterms



- Example: Full Adder

$$s = a\bar{b}\bar{c} + \bar{a}b\bar{c} + \bar{a}\bar{b}c + abc$$

$$c_{\text{out}} = ab + bc + ac$$

PLA Schematic & Layout