Introduction to CMOS VLSI Design

Circuits Lecture B

Peter Kogge
University of Notre Dame
Fall 2015,2018

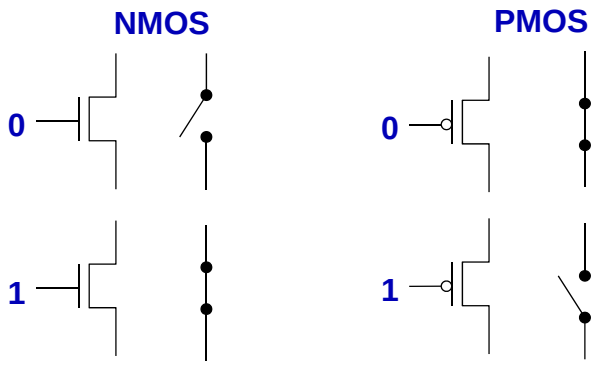
Based on material from
Prof. Jay Brockman, Joseph Nahas: University of Notre Dame
Prof. David Harris, Harvey Mudd College
http://www.cmosvlsi.com/coursematerials.html

Outline: Circuits

- Lecture A
  - Physics, EE 101
  - Semiconductors
  - CMOS Transistors
- ***Lecture B***
  - ***NMOS Logic***
  - ***CMOS Inverter and NAND Gate Operation***
  - ***CMOS Gate Design***
  - ***Adders***
  - ***Multipliers***
- Lecture C
  - Pass Transistors
  - Tri-states
  - Multiplexors
  - Latches
  - Barrel Shifters

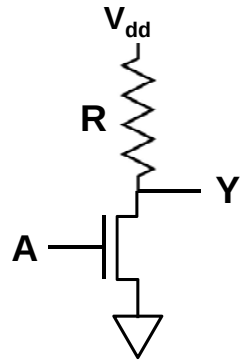
MOS Transistors as Switches

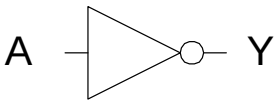
- View MOS transistors as electrically controlled switches
- Voltage at gate controls path from source to drain



NMOS Inverter

A	Y
0	
1	

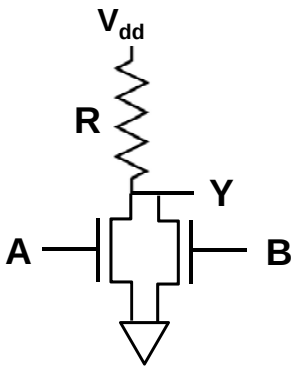




Questions:
- How to make R?
- What is current when A=1?
- What is power when A = 1?

An NMOS Gate

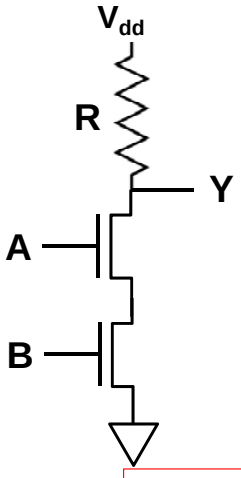
A	B	Y
0	0	
0	1	
1	0	
1	1	



What is logic function?

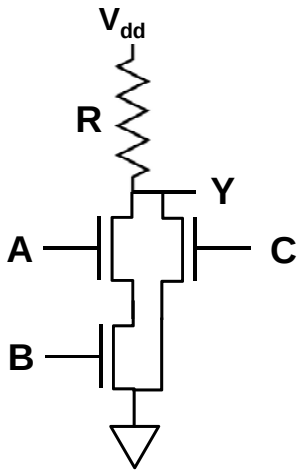
Another NMOS Gate

A	B	Y
0	0	
0	1	
1	0	
1	1	



What is logic function?

What About Now?



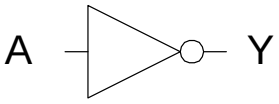
What is logic function?

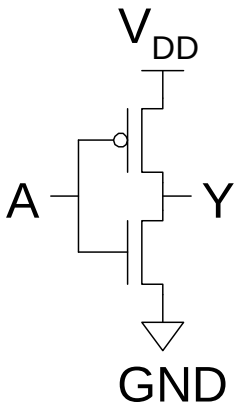
NMOS Take-Aways

- Input voltages turn N-types either on or off
- Series transistors produce “AND”
- Parallel transistors produce “OR”
- Need to implement resistors separately
- Significant static power

CMOS Inverter

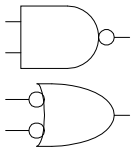
A	Y
0	
1	

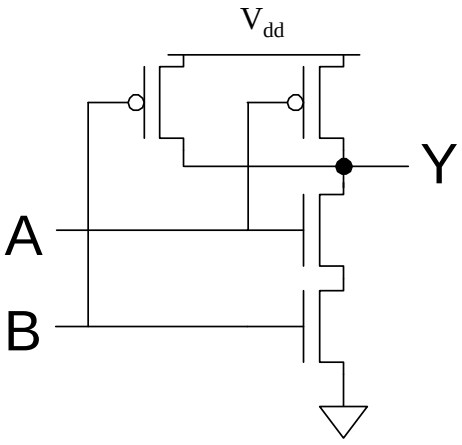




CMOS NAND Gate

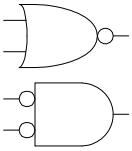
A	B	Y
0	0	
0	1	
1	0	
1	1	

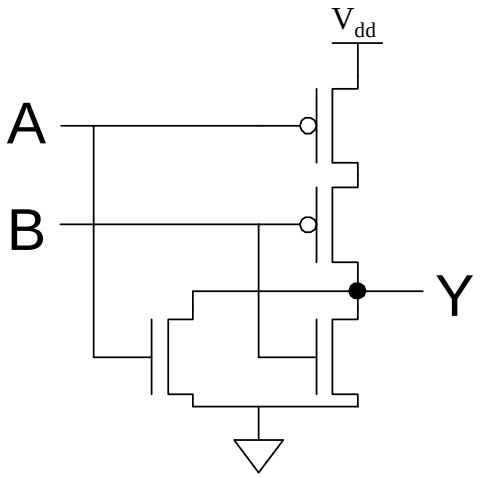




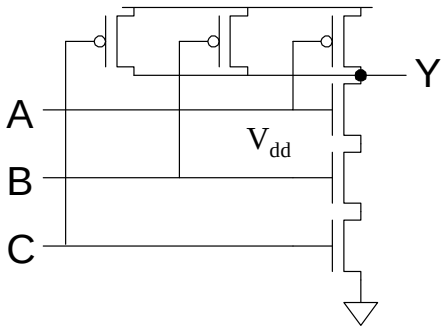
CMOS NOR Gate

A	B	Y
0	0	1
0	1	0
1	0	0
1	1	0





3-input NAND Gate

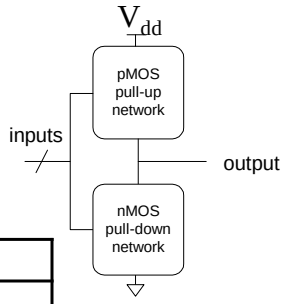


- **Y pulls low if ALL inputs are 1**
- **Y pulls high if ANY input is 0**

General CMOS Gates

Complementary CMOS

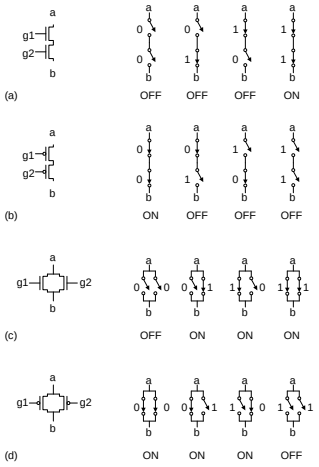
- **Complementary CMOS logic gates**
  - **nMOS *pull-down network***
  - **pMOS *pull-up network***
  - **a.k.a. static CMOS**



	Pull-up OFF	Pull-up ON
Pull-down OFF	Z (float)	1
Pull-down ON	0	X (crowbar)

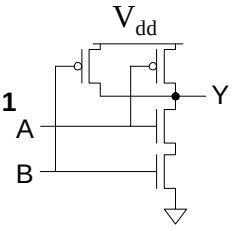
Series and Parallel

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



Conduction Complement

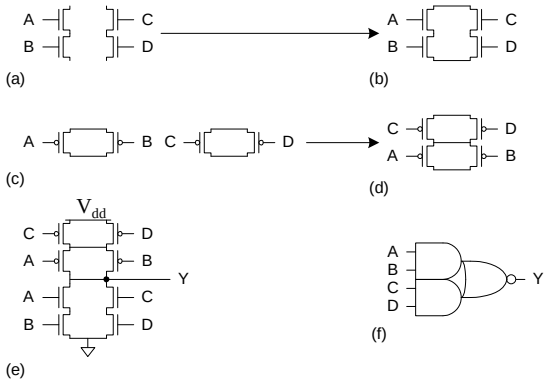
- Complementary CMOS gates ***always*** produce 0 or 1
- Ex: NAND gate
  - Series nMOS: Y=0 when both inputs are 1
  - Thus Y=1 when either input is 0
  - Requires parallel pMOS



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

Compound Gates

- ***Compound gates*** **can do any inverted function**
- **Ex:** $Y = \overline{AB + CD}$ AND-OR-INVERT, AOI22

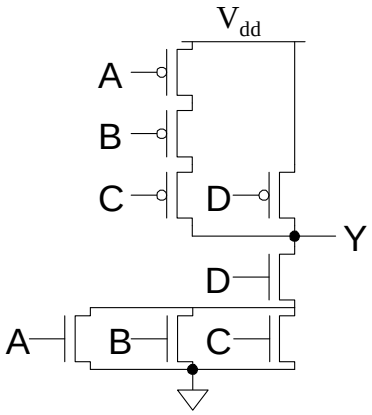


Example: O3AI

- $Y = \overline{(A + B + C)D}$

Example: O3AI

- $Y = \overline{(A + B + C)D}$



Signal Strength

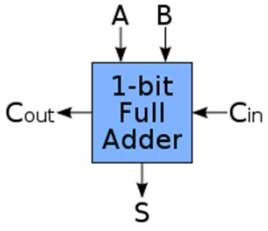
- ***Strength*** **of signal**
  - **How close it approximates ideal voltage source**
- **V_{DD} and GND rails are strongest 1 and 0**
- **nMOS pass strong 0**
  - **But degraded or weak 1**
- **pMOS pass strong 1**
  - **But degraded or weak 0**
- **Thus**
  - **pMOS are best for pull-up network**
  - **nMOS are best for pull-down network**

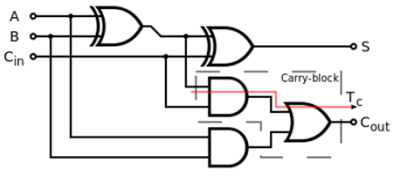
Adders

1-Bit Adder

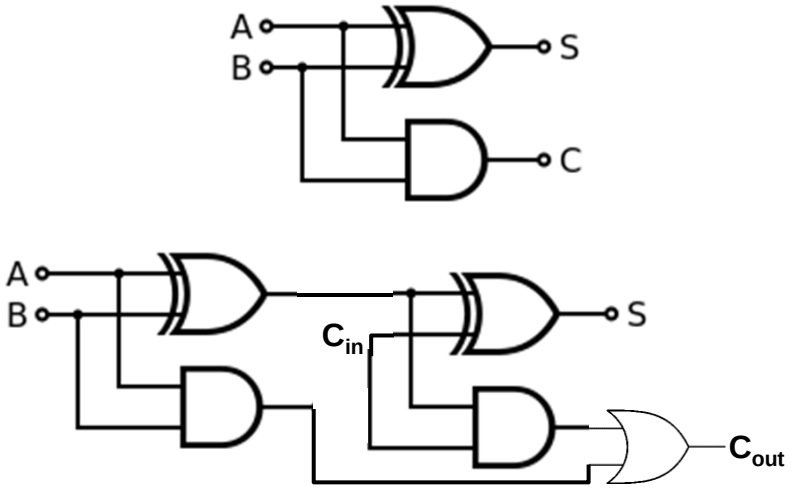
- **Inputs:**
  - **2 data inputs A and B**
  - **1 Carry Input C**
- **Outputs**
  - **1 bit S (sum)**
  - **1 bit Carry Out**

A	B	C_{IN}	C_{out}	Sum
0	0	0	0	0
0	1	0	0	1
1	0	0	0	1
1	1	0	1	0
0	0	1	0	1
0	1	1	0	1
1	0	1	1	0
1	1	1	1	1

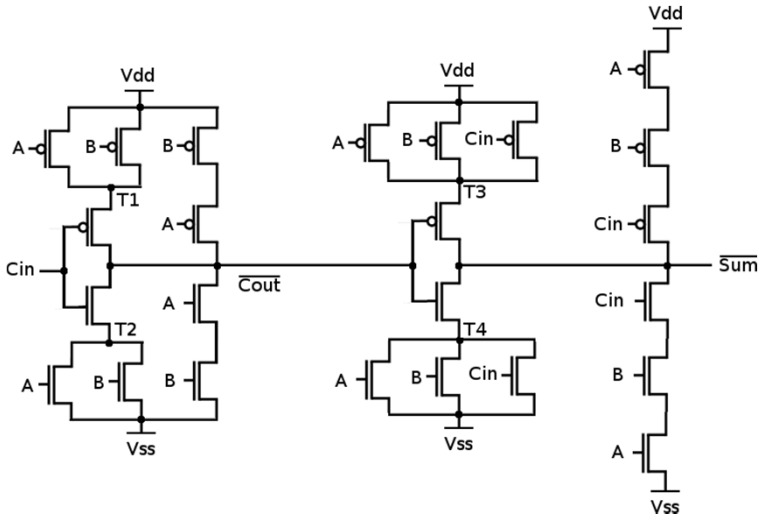




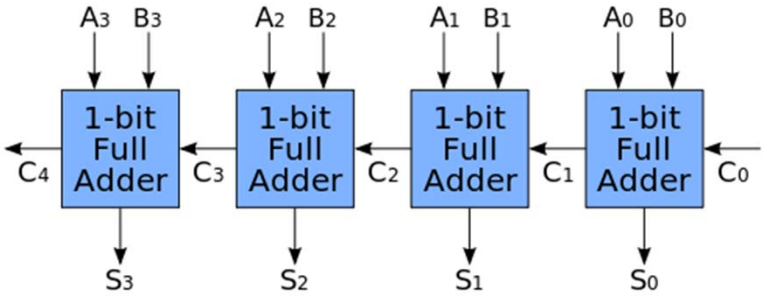
Half Adders and Full Adders



CMOS Full Adder



Multi-Bit Ripple Adder



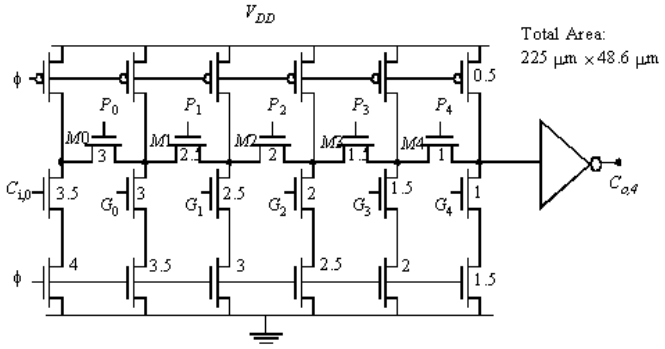
For N bits, time ~N Full Adder Delays

Carry LookAhead Adders

- **Instead of pure carry signals, generate 2 signals**
  - **P: Carry Propagate: if C_{in} = 1 then C_{out} should be 1**
  - **G: Carry Generate: C_{out} should be 1 regardless**
- **Example: 1 bit**
  - **$G_i = A_iB_i$**
  - **$P_i = A_i + B_i$**
  - **Then $C_{i+1} = G_i + P_iC_i$**
- **Compute Ps and Gs for each bit**
- **Then compute C_{i+4} from just C_i**

$$C_1 = G_0 + P_0 \cdot C_0$$
$$C_2 = G_1 + G_0 \cdot P_1 + C_0 \cdot P_0 \cdot P_1$$
$$C_3 = G_2 + G_1 \cdot P_2 + G_0 \cdot P_1 \cdot P_2 + C_0 \cdot P_0 \cdot P_1 \cdot P_2$$
$$C_4 = G_3 + G_2 \cdot P_3 + G_1 \cdot P_2 \cdot P_3 + G_0 \cdot P_1 \cdot P_2 \cdot P_3 + C_0 \cdot P_0 \cdot P_1 \cdot P_2 \cdot P_3$$

Manchester Carry Chain Adder



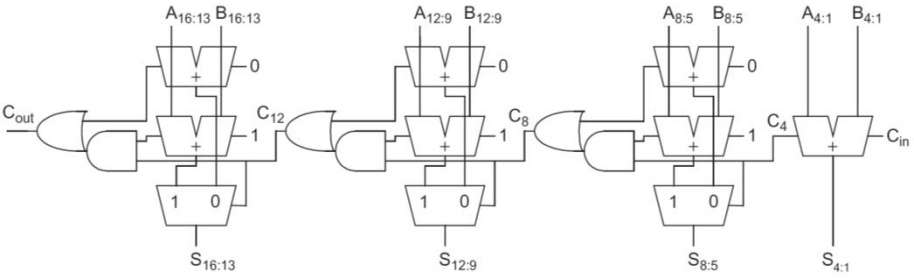
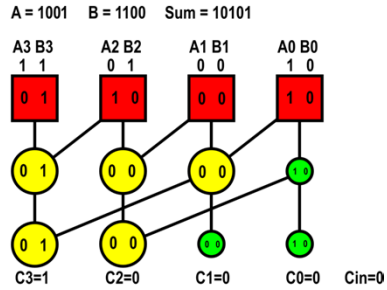
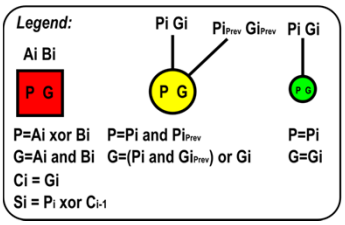


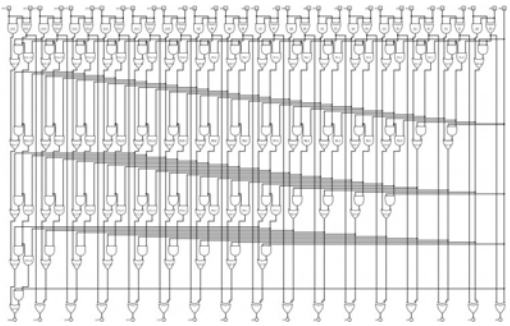
FIGURE 11.24 Carry-select adder

Kogge-Stone Adder



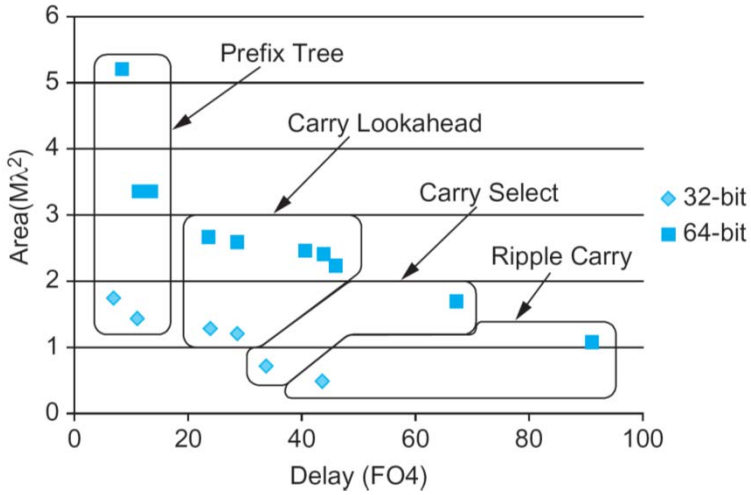


4-bit



16-bit

FIGURE 11.40 Area vs. delay of synthesized adders



Multipliers

```
      011001  :  25(10)      multiplicand
    × 100111  :  39(10)      multiplier
      011001
     011001
    011001                   partial
   000000                    products
  000000
+011001
001111001111  :  975(10)      product
```

FIGURE 11.71 Multiplication example

						y_5	y_4	y_3	y_2	y_1	y_0	Multiplicand
						x_5	x_4	x_3	x_2	x_1	x_0	Multiplier
						x_0y_5	x_0y_4	x_0y_3	x_0y_2	x_0y_1	x_0y_0	
					x_1y_5	x_1y_4	x_1y_3	x_1y_2	x_1y_1	x_1y_0		
				x_2y_5	x_2y_4	x_2y_3	x_2y_2	x_2y_1	x_2y_0			Partial Products
			x_3y_5	x_3y_4	x_3y_3	x_3y_2	x_3y_1	x_3y_0				
		x_4y_5	x_4y_4	x_4y_3	x_4y_2	x_4y_1	x_4y_0					
	x_5y_5	x_5y_4	x_5y_3	x_5y_2	x_5y_1	x_5y_0						
p_{11}	p_{10}	p_9	p_8	p_7	p_6	p_5	p_4	p_3	p_2	p_1	p_0	Product

FIGURE 11.72 Partial products

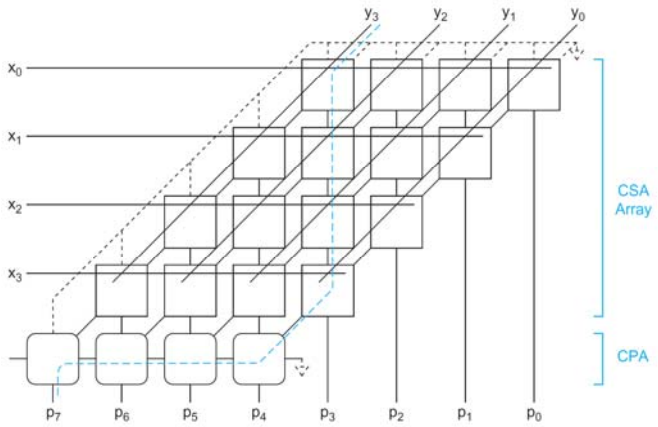
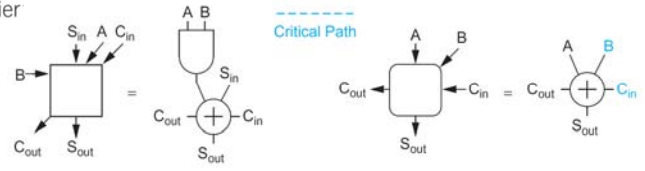


FIGURE 11.74 Array multiplier



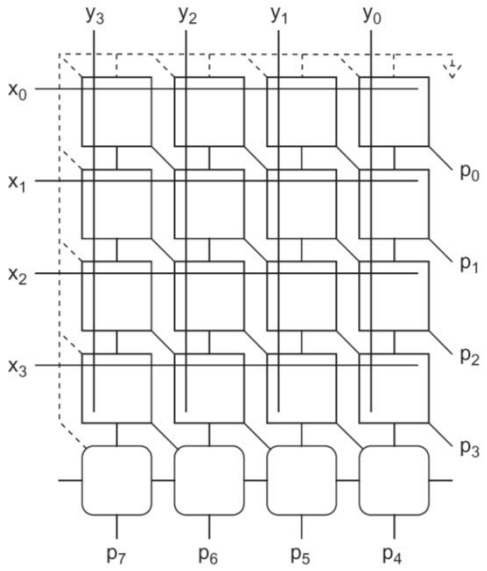


FIGURE 11.75 Rectangular array multiplier