

# EE 330

## Lecture 37

High Frequency Operation of Amplifiers  
Digital Circuit Design

- Basic Logic Gates
- Properties of Logic Families

# Exam Schedule

Final

Wed May 11 7:30 a.m.

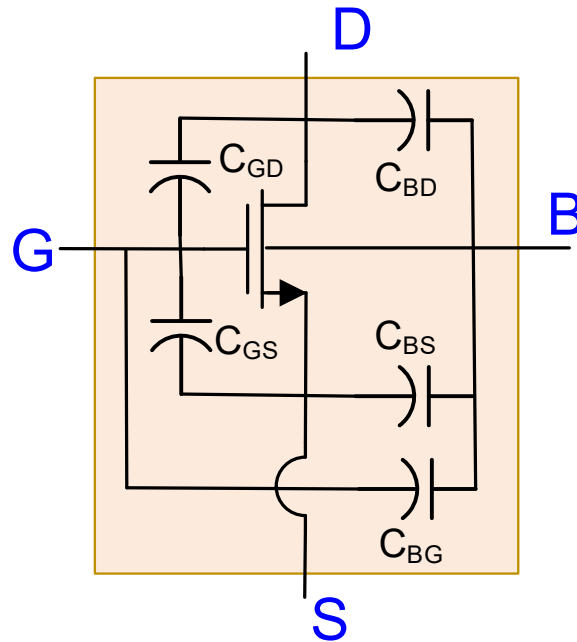
Photo courtesy of the director of the National Institute of Health ( NIH)



As a courtesy to fellow classmates, TAs, and the instructor

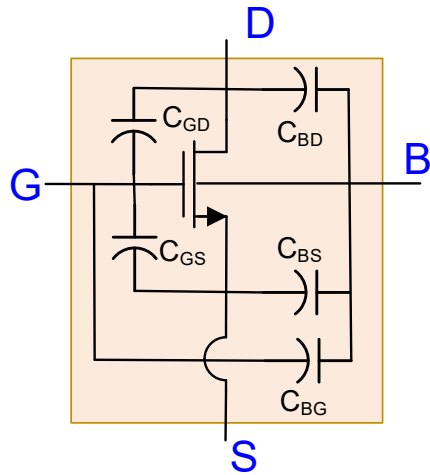
**Wearing of masks during lectures and in the laboratories for this course would be appreciated irrespective of vaccination status**

# Parasitic Capacitance Summary

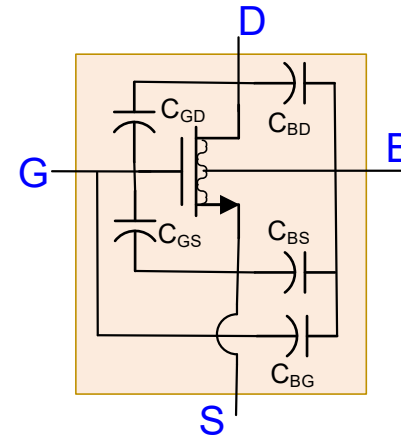


	<b>Cutoff</b>	<b>Ohmic</b>	<b>Saturation</b>
<b><math>C_{GS}</math></b>	$C_{ox}WL_D$	$0.5C_{ox}WL$	$C_{ox}WL_D + (2/3)C_{ox}WL$
<b><math>C_{GD}</math></b>	$C_{ox}WL_D$	$0.5C_{ox}WL$	$C_{ox}WL_D$
<b><math>C_{BG}</math></b>	$C_{ox}WL$ (or less)	0	0
<b><math>C_{BS}</math></b>	$C_{BOT}A_S + C_{SW}P_S$	$C_{BOT}A_S + C_{SW}P_S + 0.5WLC_{BOTCH}$	$C_{BOT}A_S + C_{SW}P_S + (2/3)WLC_{BOTCH}$
<b><math>C_{BD}</math></b>	$C_{BOT}A_D + C_{SW}P_D$	$C_{BOT}A_D + C_{SW}P_D + 0.5WLC_{BOTCH}$	$C_{BOT}A_D + C_{SW}P_D$

# Parasitic Capacitance Summary

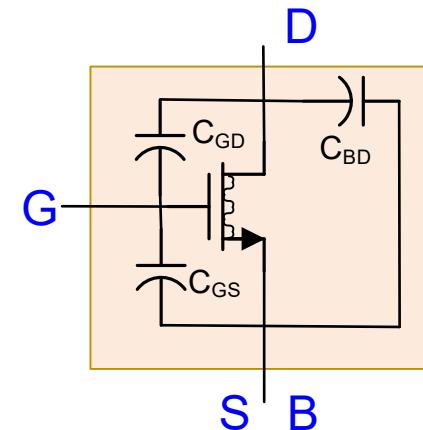
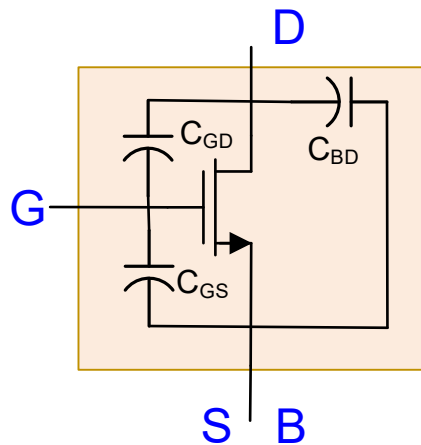


High Frequency Large Signal Model

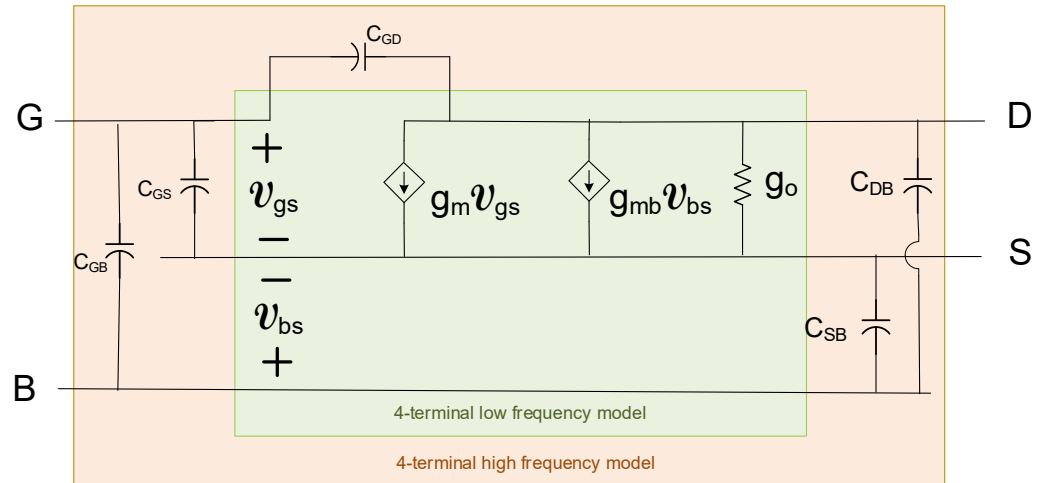
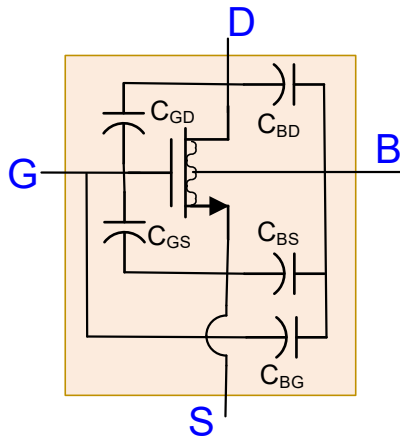


High Frequency Small Signal Model

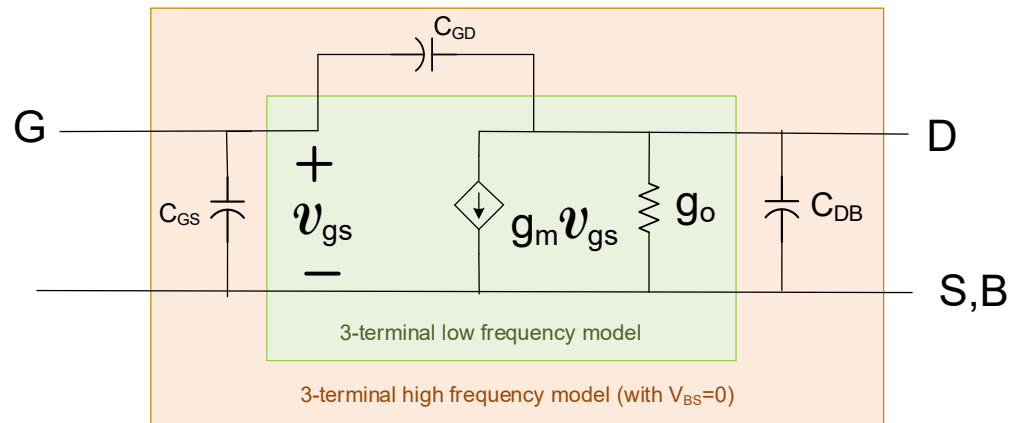
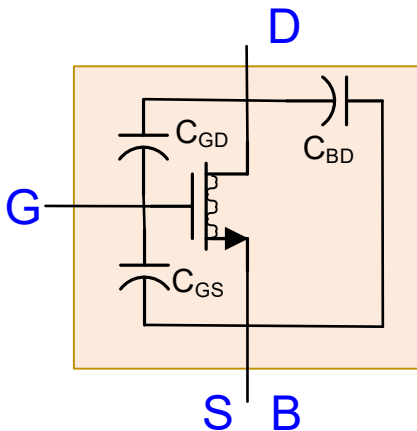
Often  $V_{BS}=0$  and  $C_{BG}=0$ , so simplifies to



# High Frequency Small-Signal Model



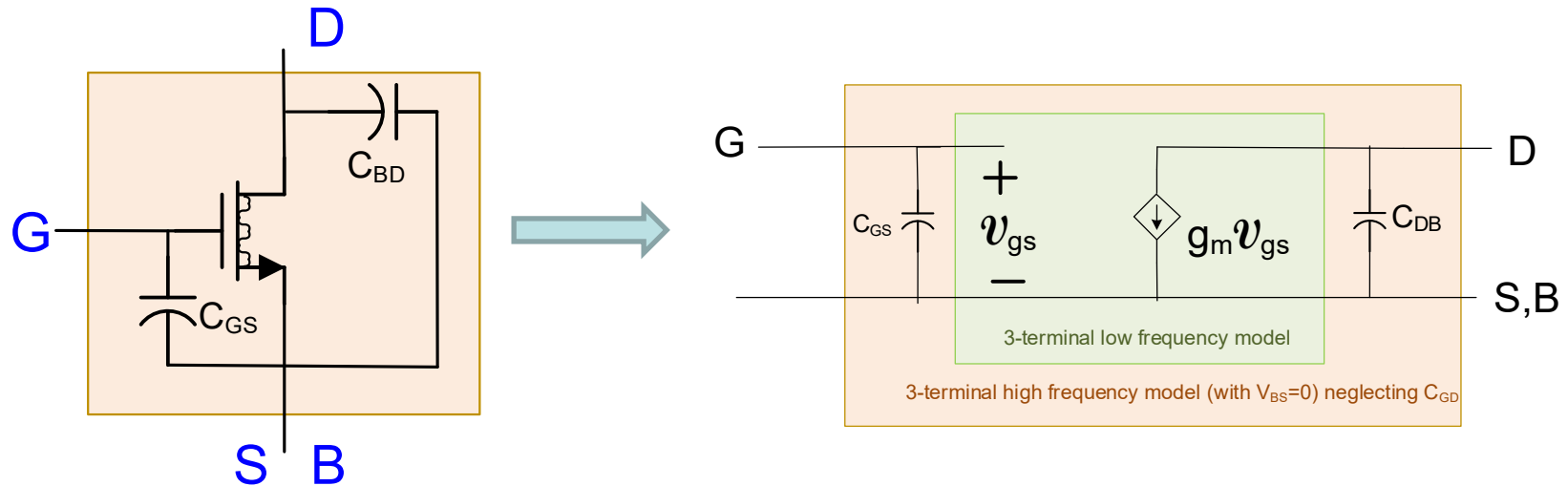
Often  $V_{BS}=0$  and  $C_{BG}=0$ , so simplifies to



Review from Last Lecture

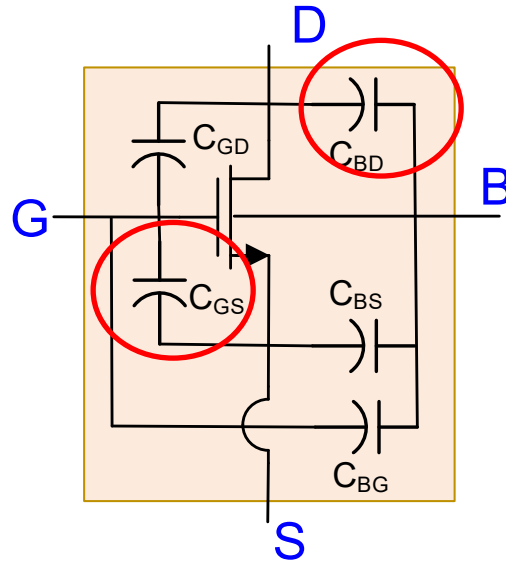
# High Frequency Small-Signal Model

Often  $V_{BS}=0$  and  $C_{BG}=0$  and  $C_{GD}$  and  $g_0$  can be neglected so simplifies farther to



Neglecting  $C_{GD}$  which is high frequency feedback from output to input often simplifies analysis considerably

# Parasitic Capacitance Implications



The parasitic capacitances inherently introduce an upper limit on how fast either digital circuits or analog circuits can operate in a given process

Two parameters,  $f_{MAX}$  and  $f_T$ , (not defined here) are two metrics that are used to specify the fundamental speed limit in a semiconductor process

The dominant parasitic capacitances for most circuits are  $C_{GS}$  and  $C_{BD}$



# $f_T$ and $f_{MAX}$ for a semiconductor process

$f_T$  is defined to be the frequency where the short-circuit current gain of a transistor drops to unity

$f_{MAX}$  is defined to be the frequency where the power gain of the transistor drops to unity (related to the maximum frequency of oscillation in a process)

$$f_T \approx \frac{3}{4\pi} \frac{\mu V_{EB}}{L_{min}^2} = \frac{3}{16\pi} \frac{\mu |V_{DD} - V_{TH}|}{(\lambda - LD)^2}$$

$f_T$  strongly dependent on  $V_{EB}$

for the ON 0.5u process

$$\mu_n C_{OX} = 100 \mu A/V^2$$

$$C_{OX} = 2.4 \text{ fF}/\mu^2$$

$$\lambda = 0.2 \mu$$

$$LD = .05 \mu$$

$$V_{THn} = 0.8 \text{ V}$$

$$\mu_n = 4E10 \text{ A}\mu^2\text{F}^{-1}\text{V}^{-2}$$

$$\mu_n = 400 \text{ cm}^2\text{AF}^{-1}\text{V}^{-2}$$

At  $V_{EB} = 1 \text{ V}$ ,  $f_T = 25 \text{ GHz}$

Note: As feature sizes shrink with process nodes,  $V_{EB-MAX}$  will typically drop linearly but  $L_{min}$  will drop quadratically thus  $f_T$  gets much larger in small feature processes

# $f_T$ and $f_{MAX}$ for a semiconductor process

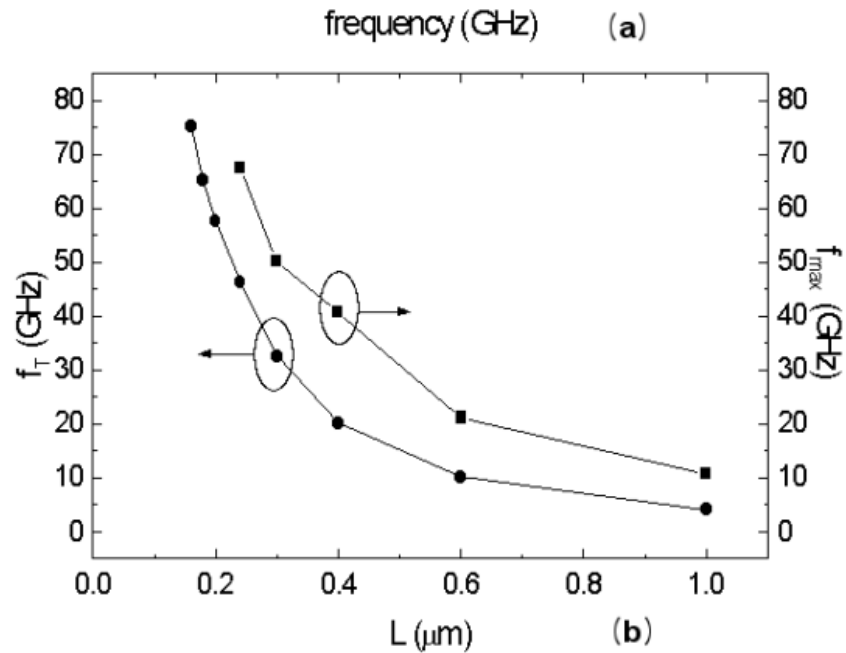
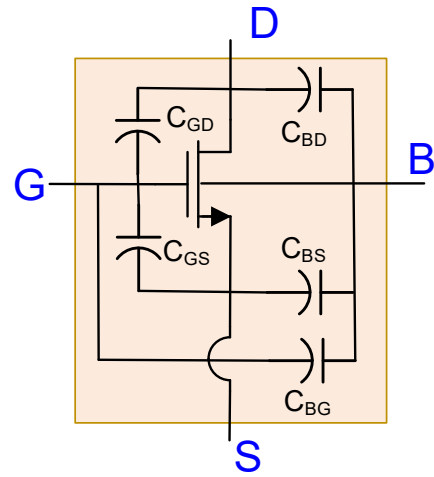


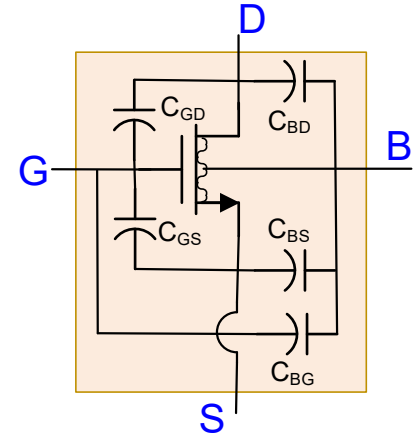
Fig. 7. (a) Maximum stable gain (MSG) and maximum available gain (MAG) for different channel lengths and (b) the cutoff frequency ( $f_T$ ) and maximum oscillation frequency ( $f_{max}$ ) as functions of the channel length.

For 0.18 $\mu$  process,  $V_D=2V$ ,  $V_G=1.2V$

# Parasitic Capacitance Summary

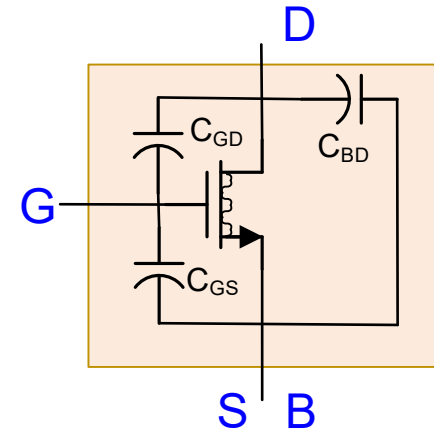
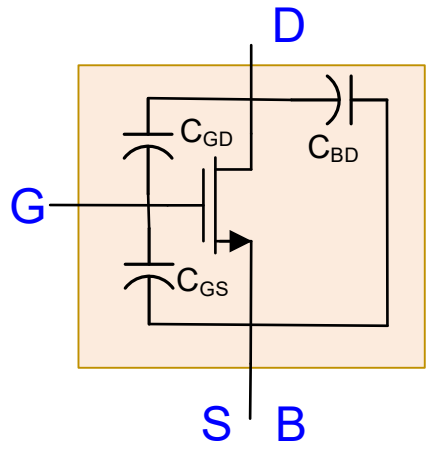


High Frequency Large Signal Model



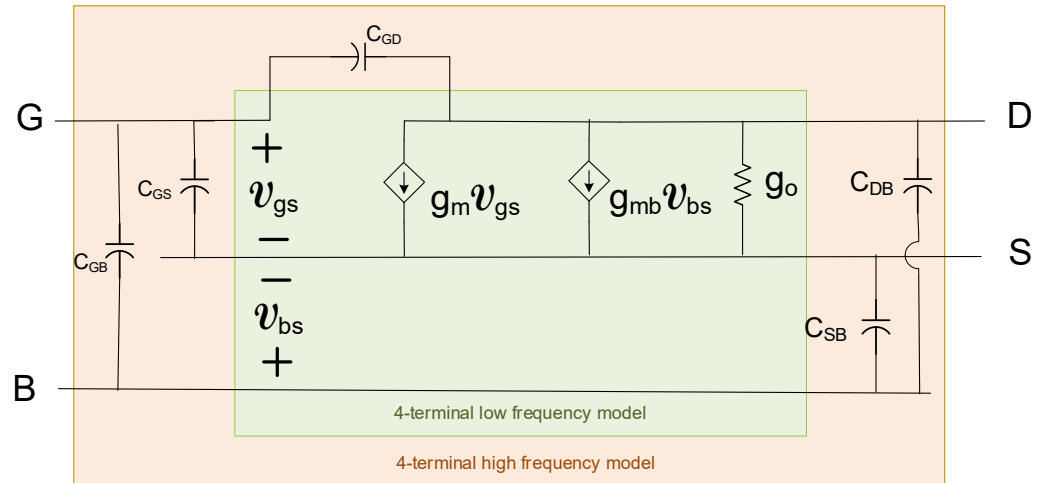
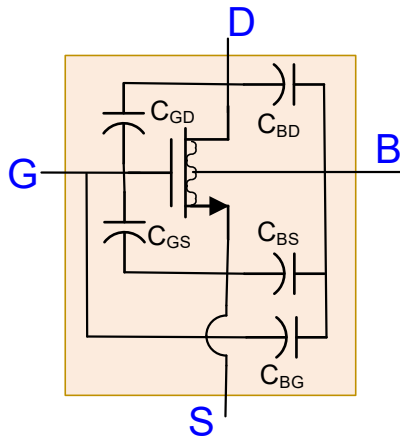
High Frequency Small Signal Model (saturation region)

Often  $V_{BS}=0$  and  $C_{BG}=0$  in saturation, so simplifies to

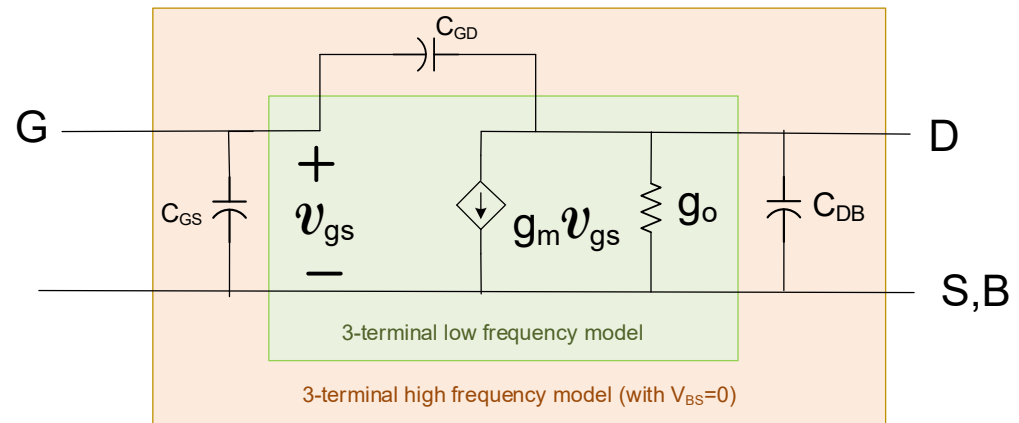
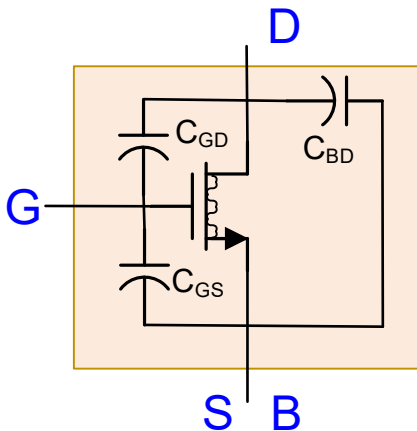


# High Frequency Small-Signal Model

(Saturation Region)

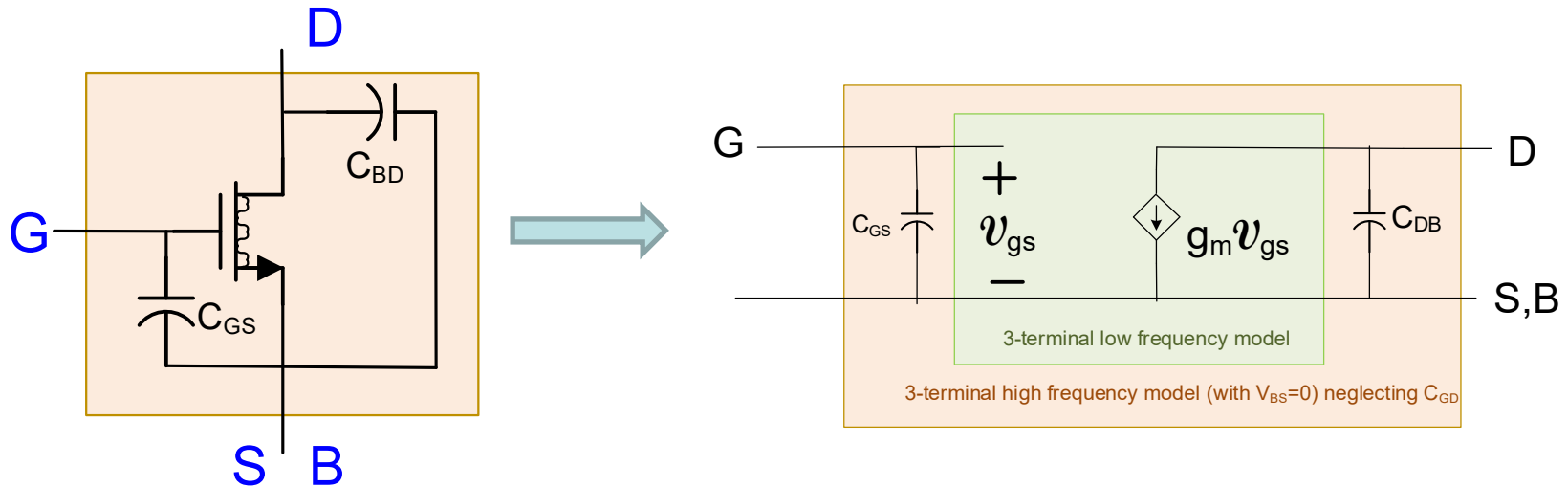


Often  $V_{BS}=0$  and  $C_{BG}=0$ , so simplifies to



# High Frequency Small-Signal Model

Often  $V_{BS}=0$  and  $C_{BG}=0$  and  $C_{GD}$  and  $g_0$  can be neglected so simplifies farther to



Neglecting  $C_{GD}$  which is high frequency feedback from output to input often simplifies analysis considerably

# Recall:



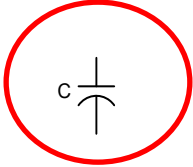



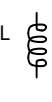
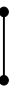









## Small-signal and simplified dc equivalent elements

	Element	ss equivalent	Simplified dc equivalent
Capacitors	C Large		
	C Small		
Inductors	L Large		
	L Small		
Diodes			 Simplified
MOS transistors (MOSFET (enhancement or depletion), JFET)			 Simplified
			 Simplified

Have not yet considered situations where the small capacitor is relevant in small-signal analysis

# Recall:

## Small-signal and simplified dc equivalent elements

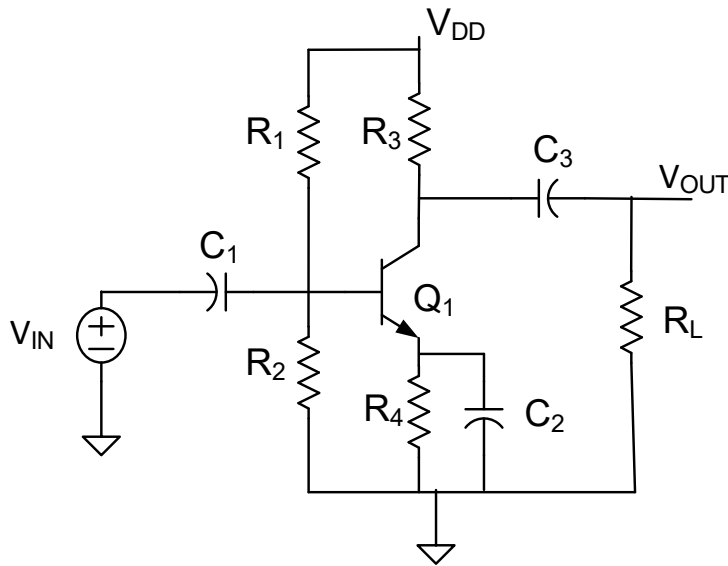
	Element	ss equivalent	Simplified dc equivalent
Capacitors	C Large		
	C Small		
Inductors	L Large		
	L Small		
Diodes			 Simplified
MOS transistors (MOSFET (enhancement or depletion), JFET)			 Simplified
			 Simplified

Have not yet considered situations where the small capacitor is relevant in small-signal analysis

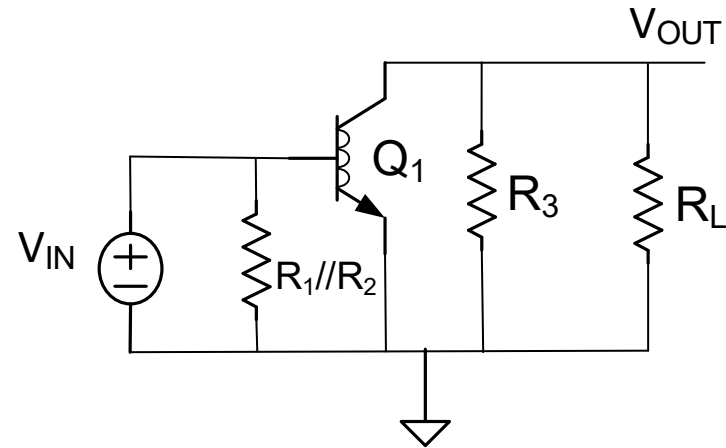
# Amplifiers with Small Capacitors

Consider a bipolar amplifier first where  $C_3$  is a small capacitor but not a parasitic capacitor

Recall:



If capacitors are large

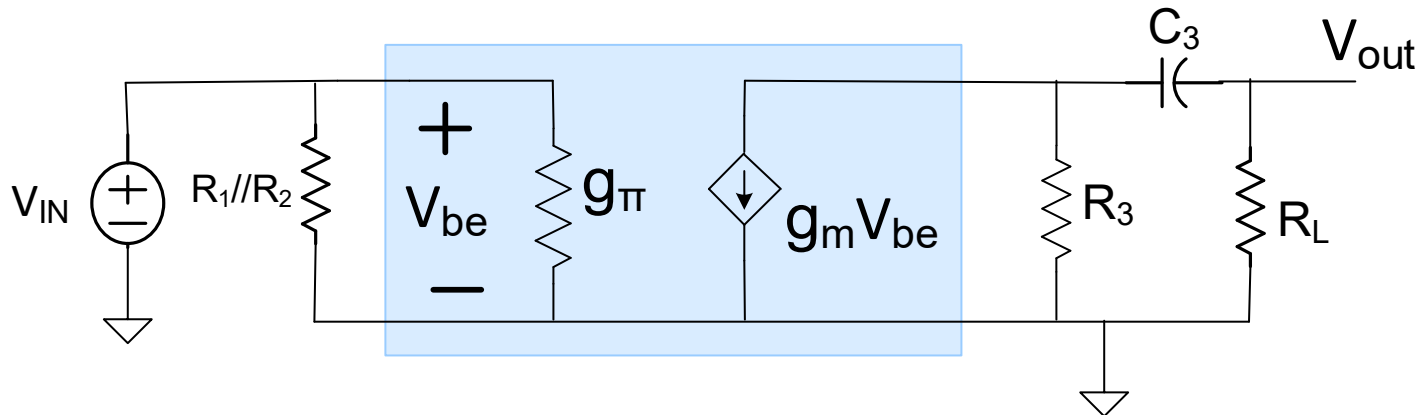
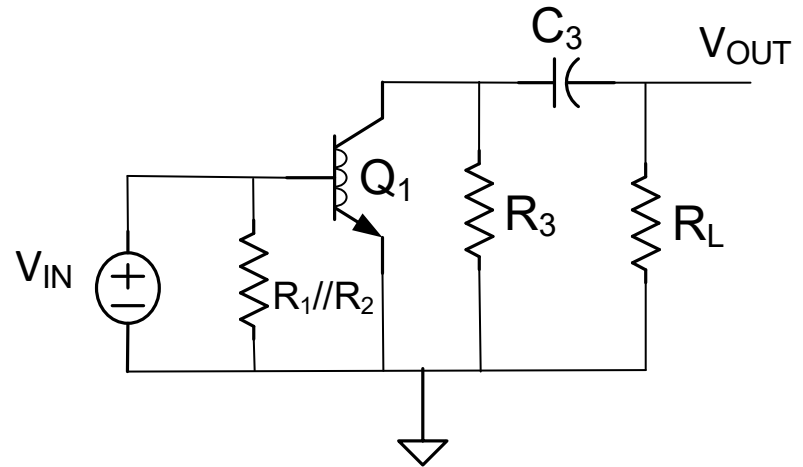
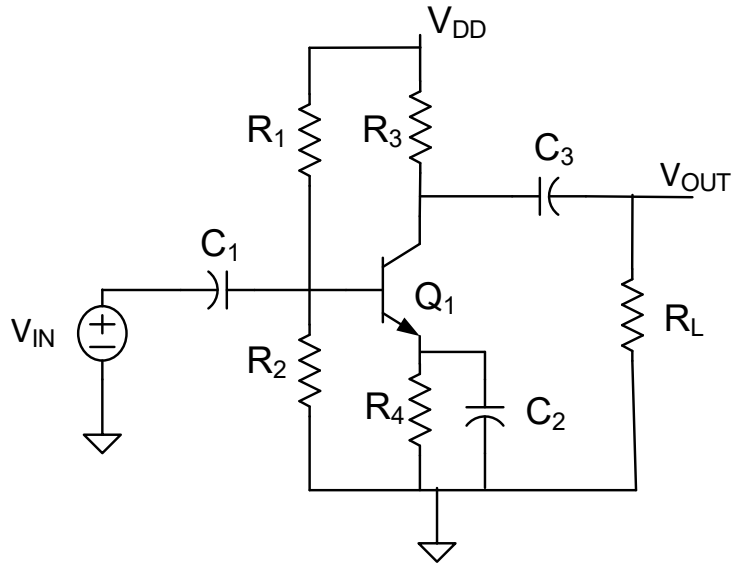


$$A_V = -g_{m1} \bullet R_3 // R_L$$



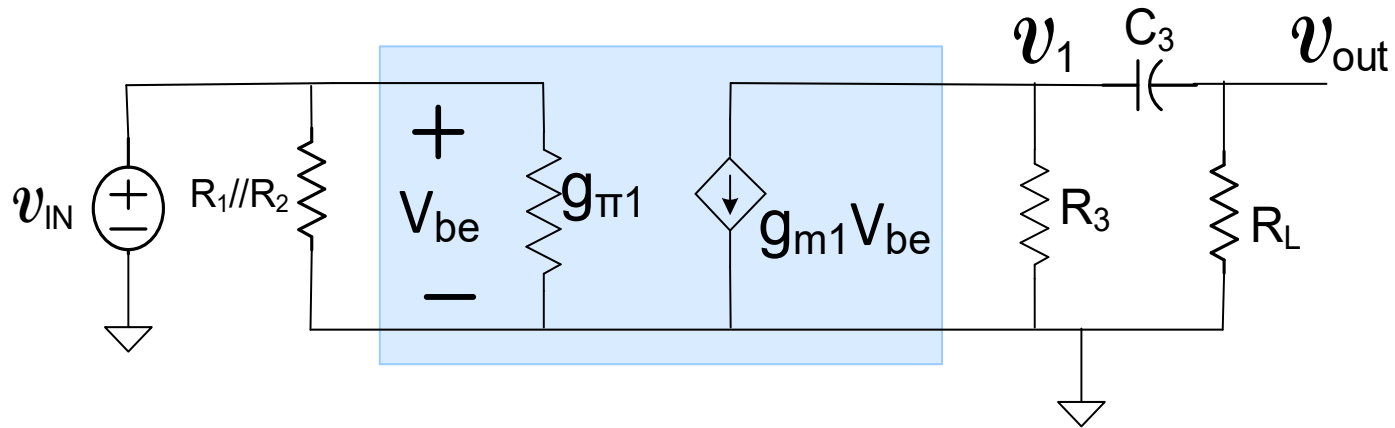
# Amplifiers with Small Capacitors

What if  $C_1$  and  $C_2$  large but  $C_3$  is not large?:



# Amplifiers with Small Capacitors

What if  $C_1$  and  $C_2$  large but  $C_3$  not large?:



From KCL:

$$\left. \begin{aligned} v_{OUT} (sC_3 + G_L) &= v_1 sC_3 \\ v_1 (sC_3 + G_3) + g_{m1} v_{IN} &= v_{OUT} sC_3 \end{aligned} \right\}$$

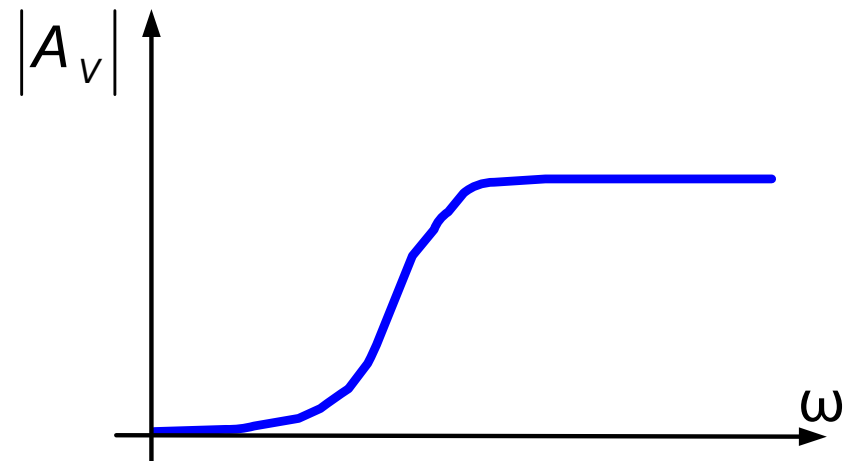
Solving:

$$\frac{v_{OUT}}{v_{IN}} = -\frac{-sC_3 g_{m1}}{sC_3 (G_L + G_3) + G_3 G_L}$$

Equivalently:

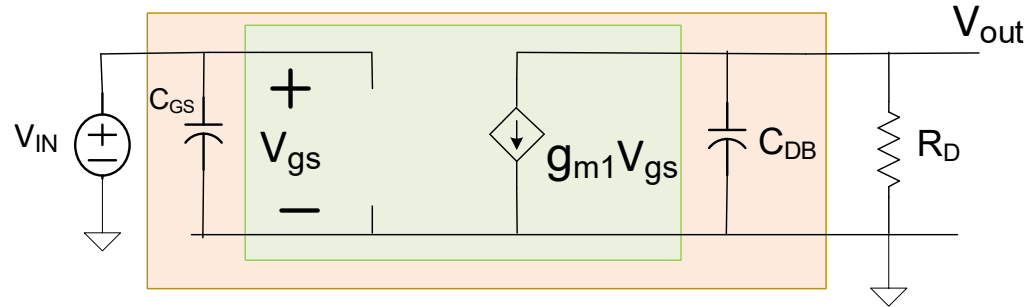
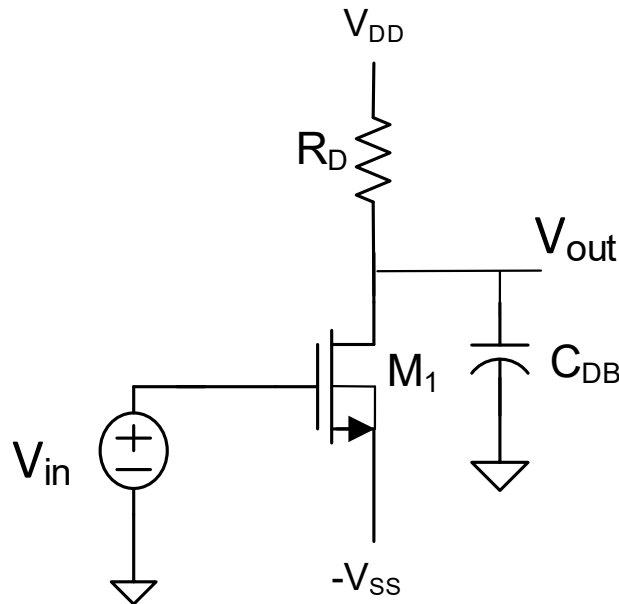
$$\frac{v_{OUT}}{v_{IN}} = -\frac{g_{m1} sC_3 R_3 R_L}{sC_3 (R_L + R_3) + 1}$$

Serves as a first-order high-pass filter



# Amplifiers with Small Capacitors

Consider parasitic  $C_{GS}$  and  $C_{DB}$



By KCL:

$$v_{OUT} (sC_{DB} + G_D) = -g_{m1} v_{IN}$$

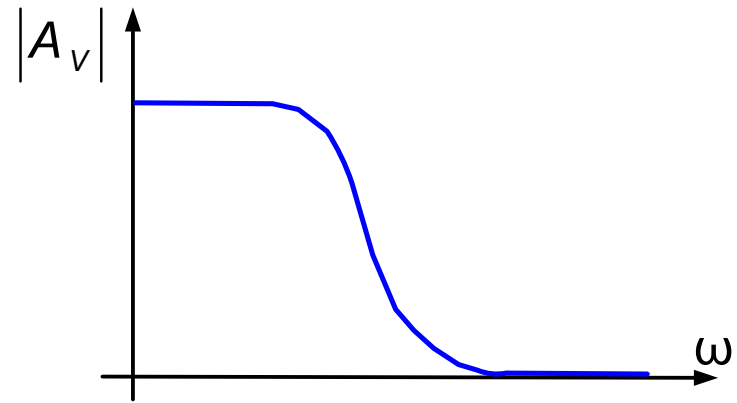
Causes gain to decrease at high frequencies

Solving:

$$\frac{v_{OUT}}{v_{IN}} = -\frac{g_{m1}}{sC_{DB} + G_D}$$

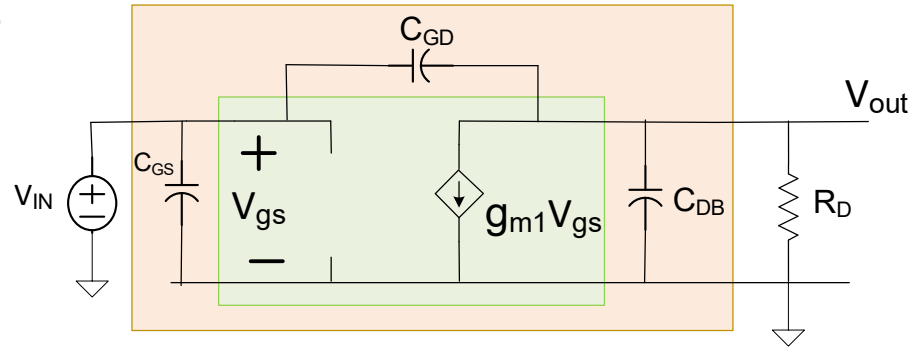
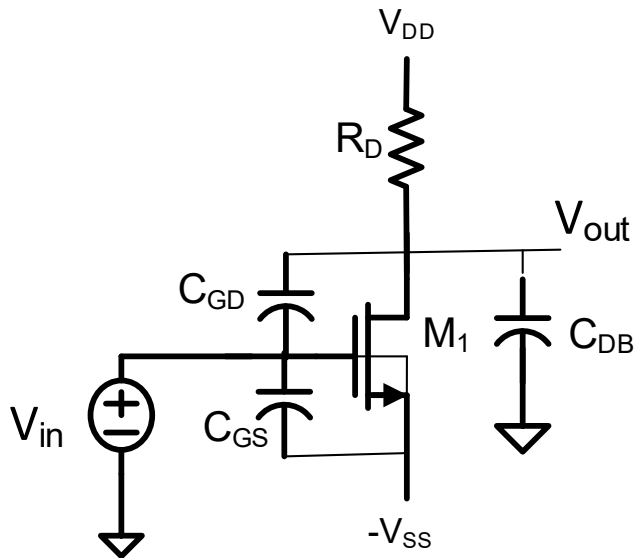
Equivalently:

$$\frac{v_{OUT}}{v_{IN}} = -\frac{g_{m1} R_D}{sC_{DB} R_D + 1}$$



# Amplifiers with Small Capacitors

Consider parasitic  $C_{GS}$ ,  $C_{GD}$ , and  $C_{DB}$



By KCL:

$$v_{OUT} (s[C_{DB} + C_{GD}] + G_D) = -g_{m1} v_{IN} + sC_{GD} v_{IN}$$

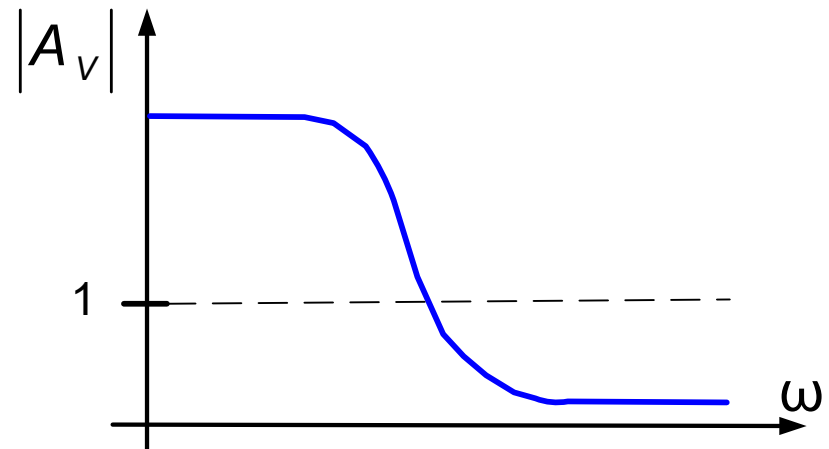
Causes gain to decrease at high frequencies  
Has one LHP pole and one RHP zero

Solving:

$$\frac{v_{OUT}}{v_{IN}} = -\frac{-g_{m1} + sC_{GD}}{s[C_{DB} + C_{GD}] + G_D}$$

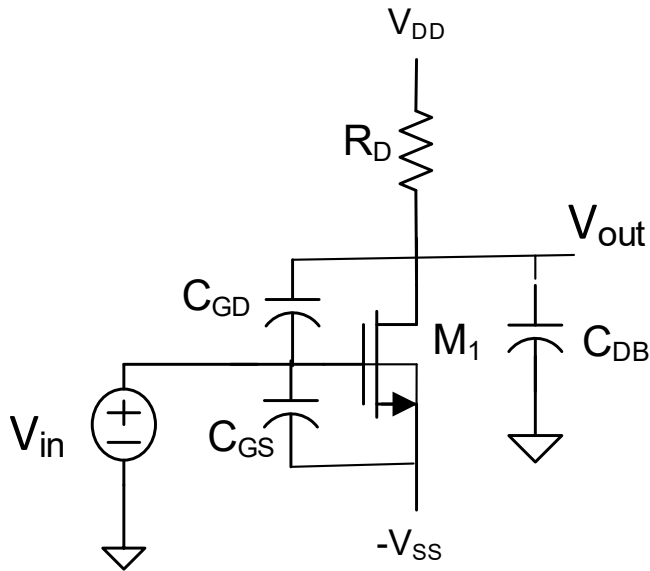
Equivalently:

$$\frac{v_{OUT}}{v_{IN}} = -\frac{-R_D (g_{m1} - sC_{GD})}{s[C_{DB} + C_{GD}]R_D + 1}$$



# Amplifiers with Small Capacitors

Consider parasitic  $C_{GS}$ ,  $C_{GD}$ , and  $C_{DB}$



Device parasitics problematic at high frequencies

$C_{DB}$ ,  $C_{GD}$  and  $C_{GS}$  effects can be significant

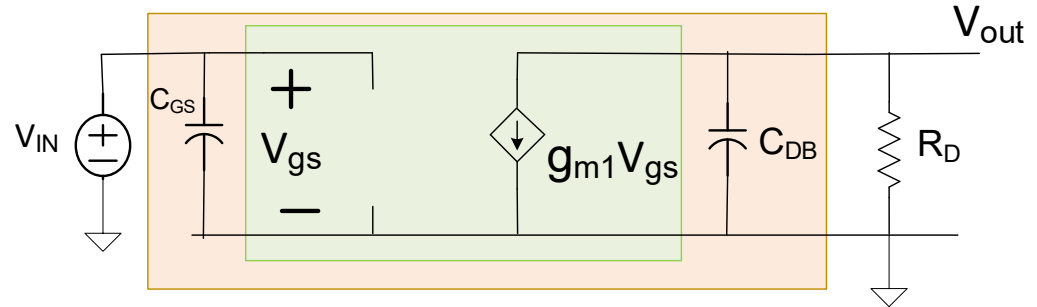
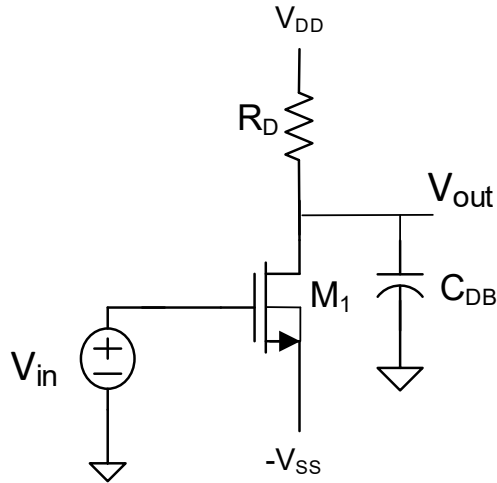
Value of parasitic capacitances strongly dependent upon layout

Device parasitics usually not a problem at audio frequencies

Causes gain to decrease at high frequencies:  
has one high frequency LHP pole and one high frequency RHP zero.

# Amplifiers with Small Capacitors

Consider parasitic  $C_{GS}$  and  $C_{DB}$

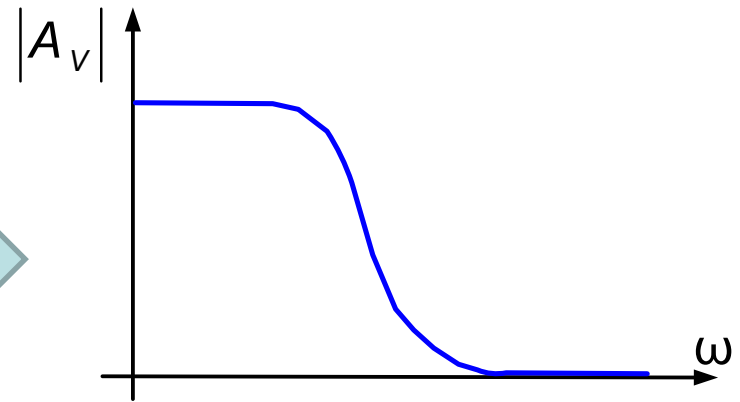


$$\frac{v_{OUT}}{v_{IN}} = A_V(s) = -\frac{-g_{m1}R_D}{sC_{DB}R_D + 1}$$

$$A_V(j\omega) = \frac{-g_{m1}R_D}{j\omega C_{DB}R_D + 1}$$

$$|A_V(j\omega)| = \frac{g_{m1}R_D}{\sqrt{(\omega C_{DB}R_D)^2 + 1}}$$

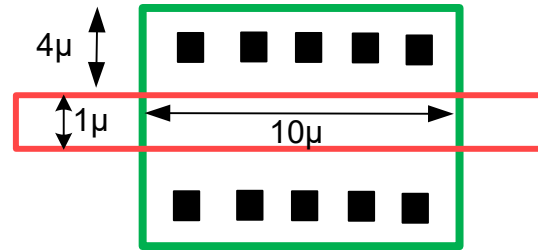
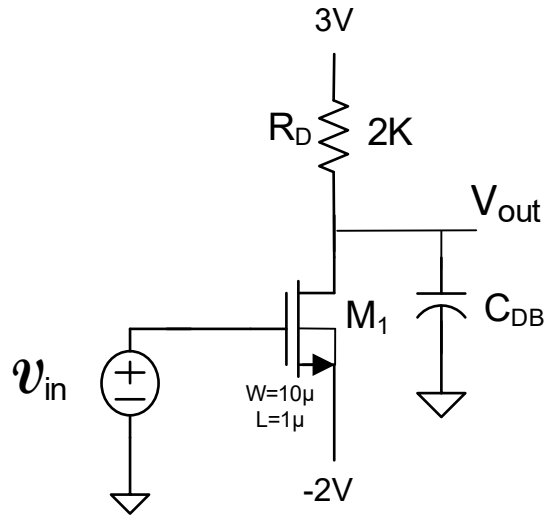
$$\angle A_V(j\omega) = -180^\circ - \tan^{-1}\left(\frac{\omega C_{DB}R_D}{1}\right)$$



Since first-order low-pass, half-power frequency given by

$$\omega_{3dB} = \frac{1}{R_D C_{DB}}$$

Example: Determine the small-signal voltage gain and the 3dB bandwidth. Consider only the effects of  $C_{DB}$  on the BW. Assume a 0.5u process with  $V_{TH}=0.75V$  and the layout of the transistor shown.



From PDK

$$C_{DB} = C_{BOT} * 40u^2 + C_{SW} * 28u$$

$$C_{BOT} = 942aF/u^2 \quad C_{SW} = 212aF/u$$

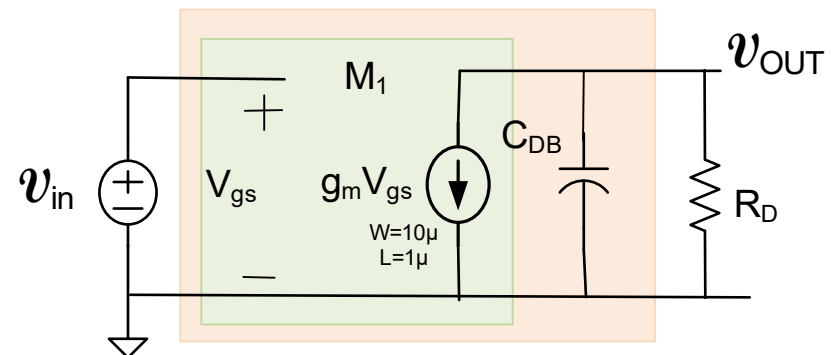
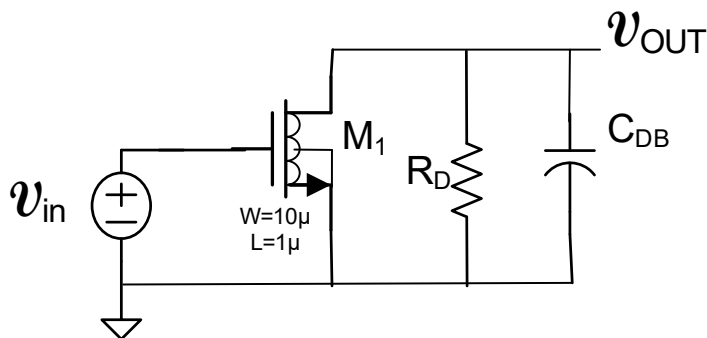
$$C_{DB} = 942aF/u^2 * 40u^2 + 212aF/u * 28u$$

$$C_{DB} = 37.7fF + 5.9fF = 43.6fF$$

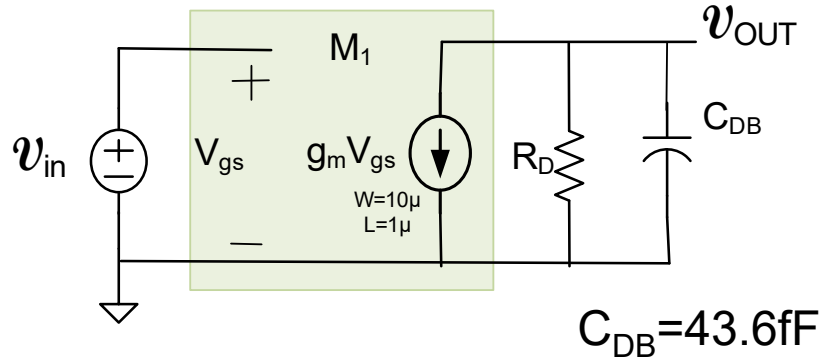
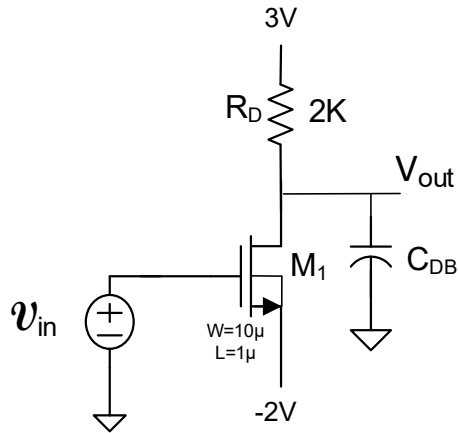
Solution:

$$I_{DQ} = 100\mu A / V^2 \frac{10}{2 \cdot 1} (2 - 0.75)^2 = 0.78mA$$

$$I_{DQ} R_D = 0.78mA \cdot 2K = 1.56$$



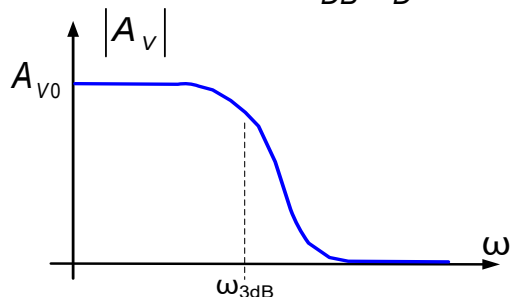
Example: Determine the small-signal dc voltage gain and the 3dB bandwidth. Consider only the effects of  $C_{DB}$  on the BW. Assume a 0.5u process with  $V_{TH}=0.75V$  and the layout of the transistor shown.



Solution continued:

$$v_{OUT} (G_D + sC_{DB}) + g_m v_{IN} = 0$$

$$v_{OUT} = -v_{IN} \frac{g_m R_D}{1 + sC_{DB} R_D}$$

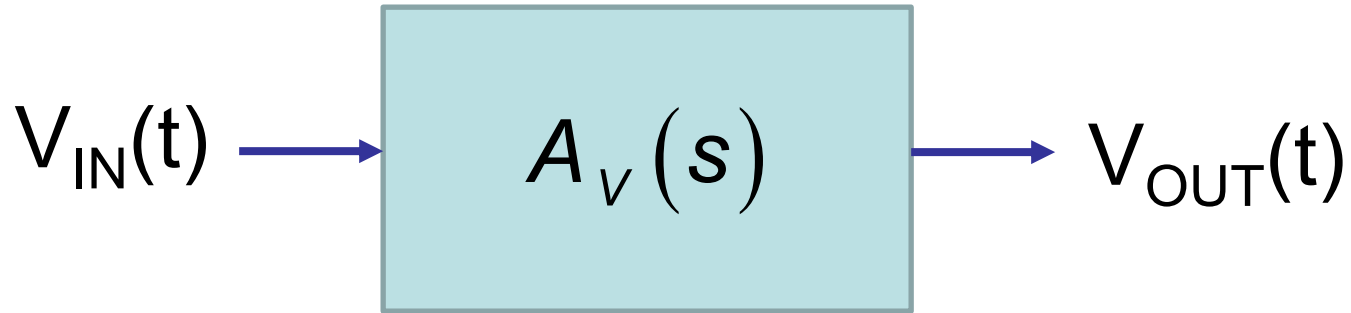


$$A_{V0} = -g_m R_D = -\frac{2I_{DQ} R_D}{V_{EB}} = -\frac{3.12}{1.25} = -2.5$$

$$f_{3dB} = \frac{1}{2\pi} \cdot \frac{1}{R_D C_{DB}} = 1.8GHz$$



# Sinusoidal Steady State Response for Linear Systems



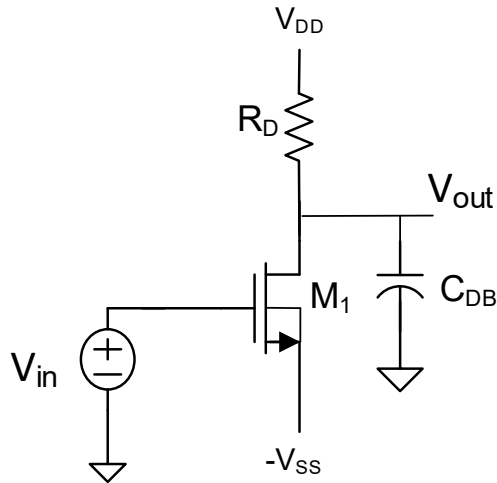
## Key Result from EE 201

If  $V_{IN} = V_m \sin(\omega t + \theta)$  where  $V_m$  is small (so linear operation maintained)

Steady state output is also a sinusoid given by

$$V_{OUT}(t) = V_m |A_V(j\omega)| \sin(\omega t + \theta + \angle A_V(j\omega))$$

# Sinusoidal Steady State Response for Linear Systems



$$|A_V(j\omega)| = \frac{g_{m1}R_D}{\sqrt{(\omega C_{DB}R_D)^2 + 1}}$$

$$\angle A_V(j\omega) = -180^\circ - \tan^{-1}\left(\frac{\omega C_{DB}R_D}{1}\right)$$

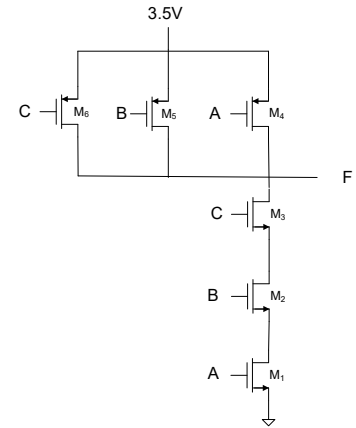
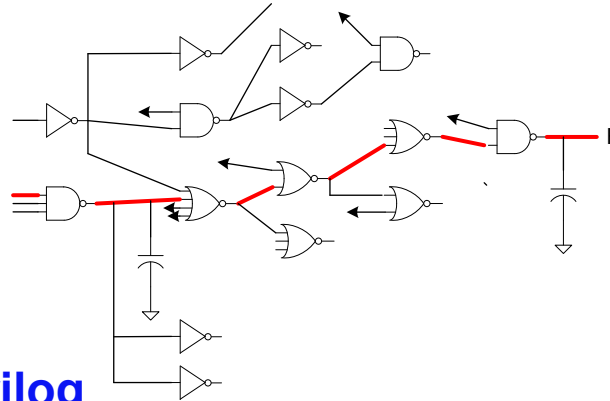
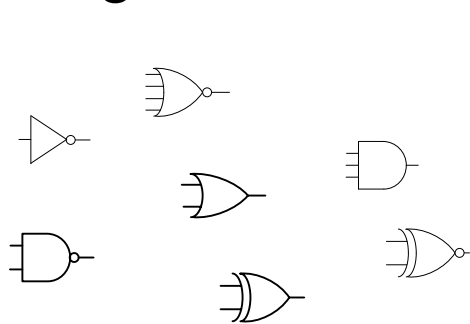
If  $V_{IN} = V_m \sin(\omega t + \theta)$

For  $V_m$  small, small-signal steady state output given by

$$V_{OUT}(t) = V_m \frac{g_{m1}R_D}{\sqrt{(\omega C_{DB}R_D)^2 + 1}} \sin\left(\omega t + \theta - 180^\circ - \tan^{-1}\left(\frac{\omega C_{DB}R_D}{1}\right)\right)$$

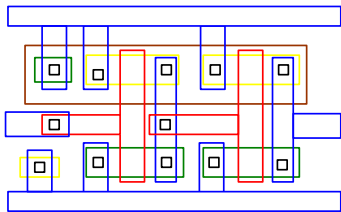
# Digital Circuit Design

Most of the remainder of the course will be devoted to digital circuit design



**Verilog**

**VHDL**



```

module gates (input logic [3:0] a,b,
              output logic [3:0] y1,y2,y3,y4,y5);
  assign y1 = a&b; //AND
  assign y2 = a | b; //OR
  assign y3 = a ^ b; //XOR
  assign y4 = ~(a & b); //NAND
  assign y5 = ~( a | b); //NOR
endmodule
    
```

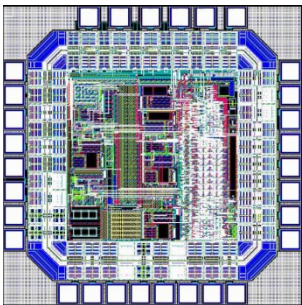
```

library IEEE; use IEEE.STD_LOGIC_1164.all;

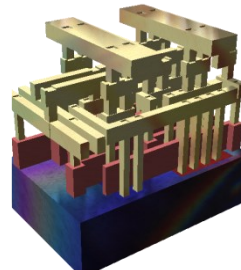
entity gates is
  port(a,b: in STD_LOGIC_VECTOR(3 downto 0);
        y1,y2,y3,y4,y5:out STD_LOGIC_VECTOR(3 downto 0));
end;

architecture synth of gates is
begin

y1 <= a and b;
y2 <= a or b;
y3 <= a xor b;
y4 <= a nand b;
y5 <= a nor b;
end;
    
```

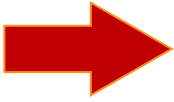


A rendering of a small standard cell with three metal layers (dielectric has been removed). The sand-colored structures are metal interconnect, with the vertical pillars being contacts, typically plugs of tungsten. The reddish structures are polysilicon gates, and the solid at the bottom is the crystalline silicon bulk



Standard Cell Library

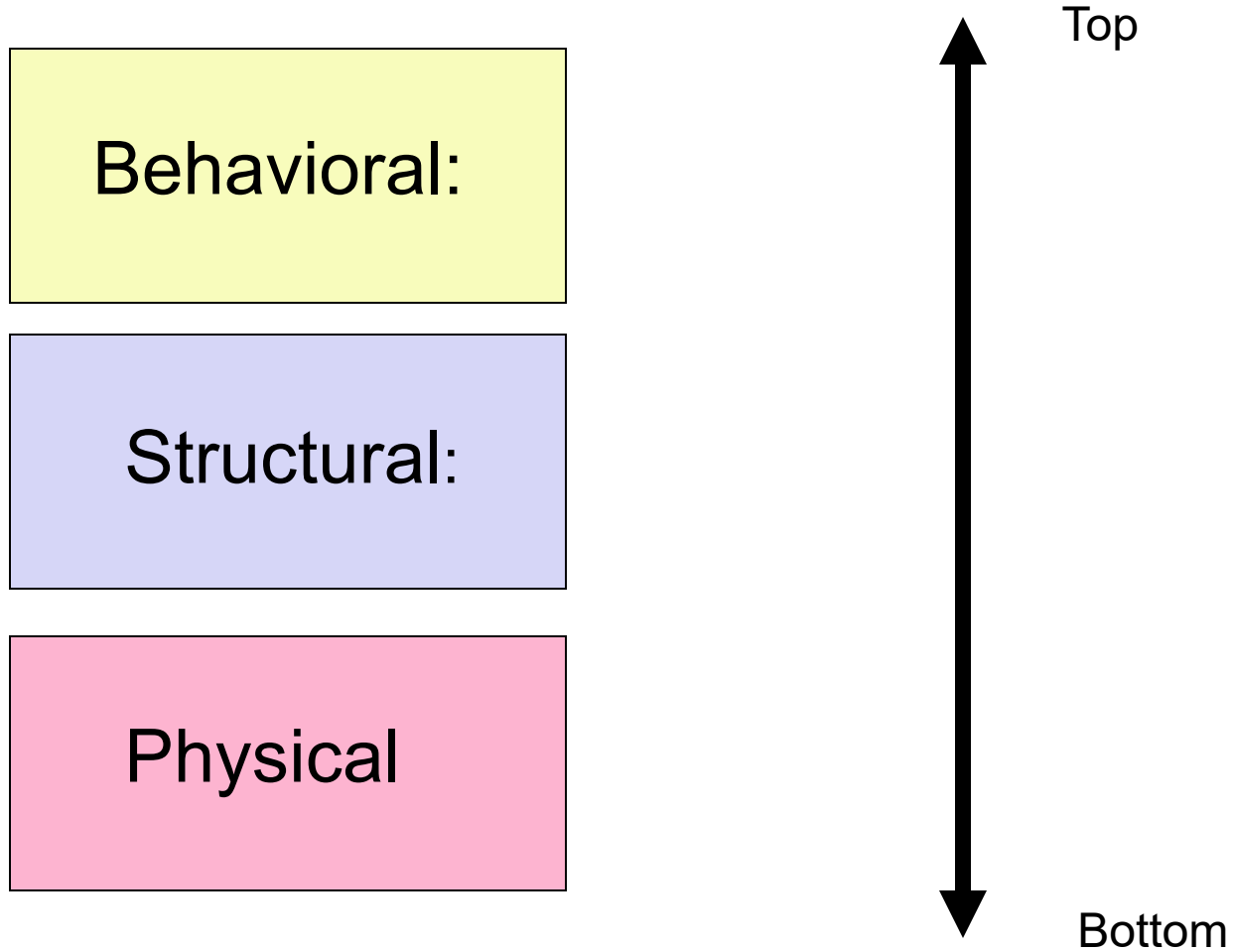
# Digital Circuit Design



## Hierarchical Design

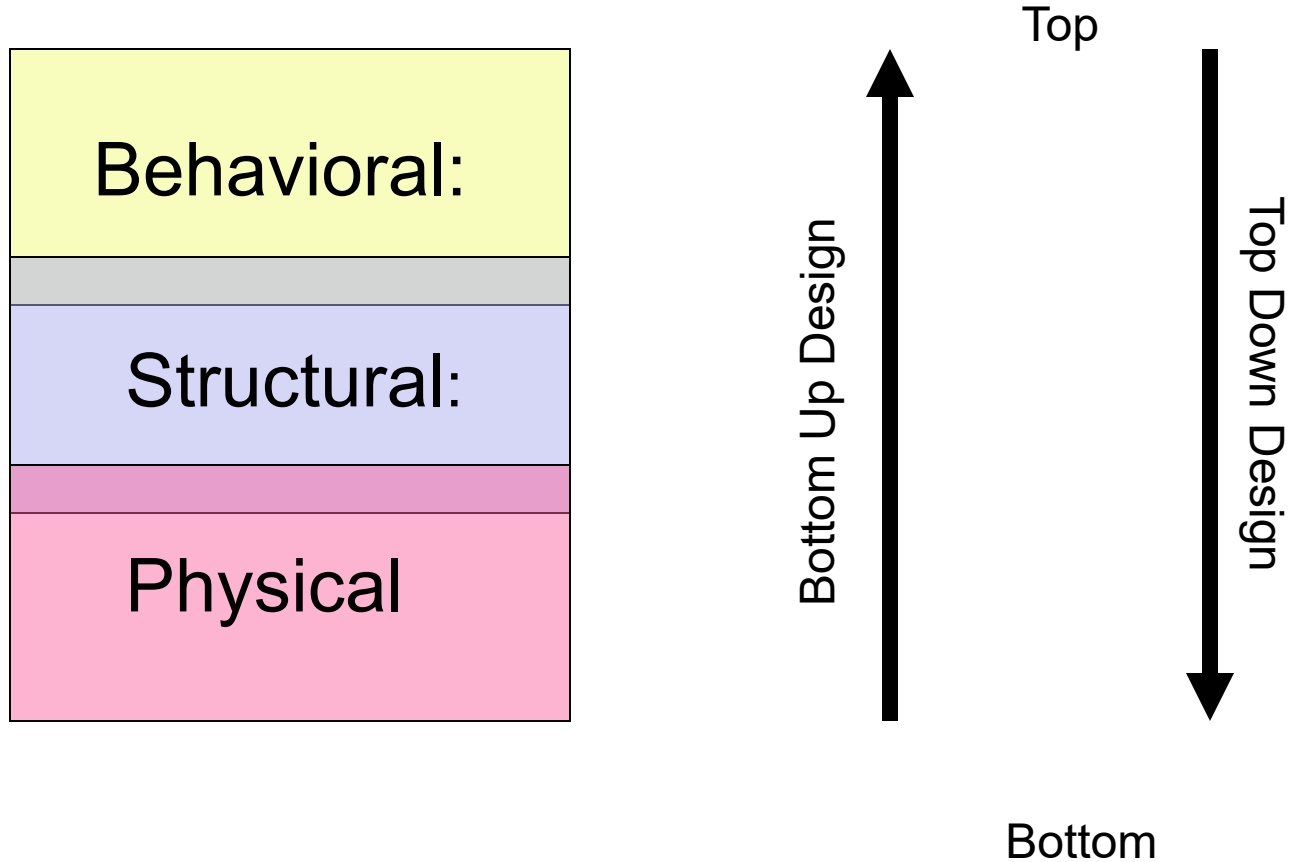
- Basic Logic Gates
  - Properties of Logic Families
  - Characterization of CMOS Inverter
  - Static CMOS Logic Gates
    - Ratio Logic
  - Propagation Delay
    - Simple analytical models
      - FI/OD
      - Logical Effort
    - Elmore Delay
  - Sizing of Gates
    - The Reference Inverter
- 
- Propagation Delay with Multiple Levels of Logic
  - Optimal driving of Large Capacitive Loads
  - Power Dissipation in Logic Circuits
  - Other Logic Styles
  - Array Logic
  - Ring Oscillators

# Hierarchical Digital Design Domains:

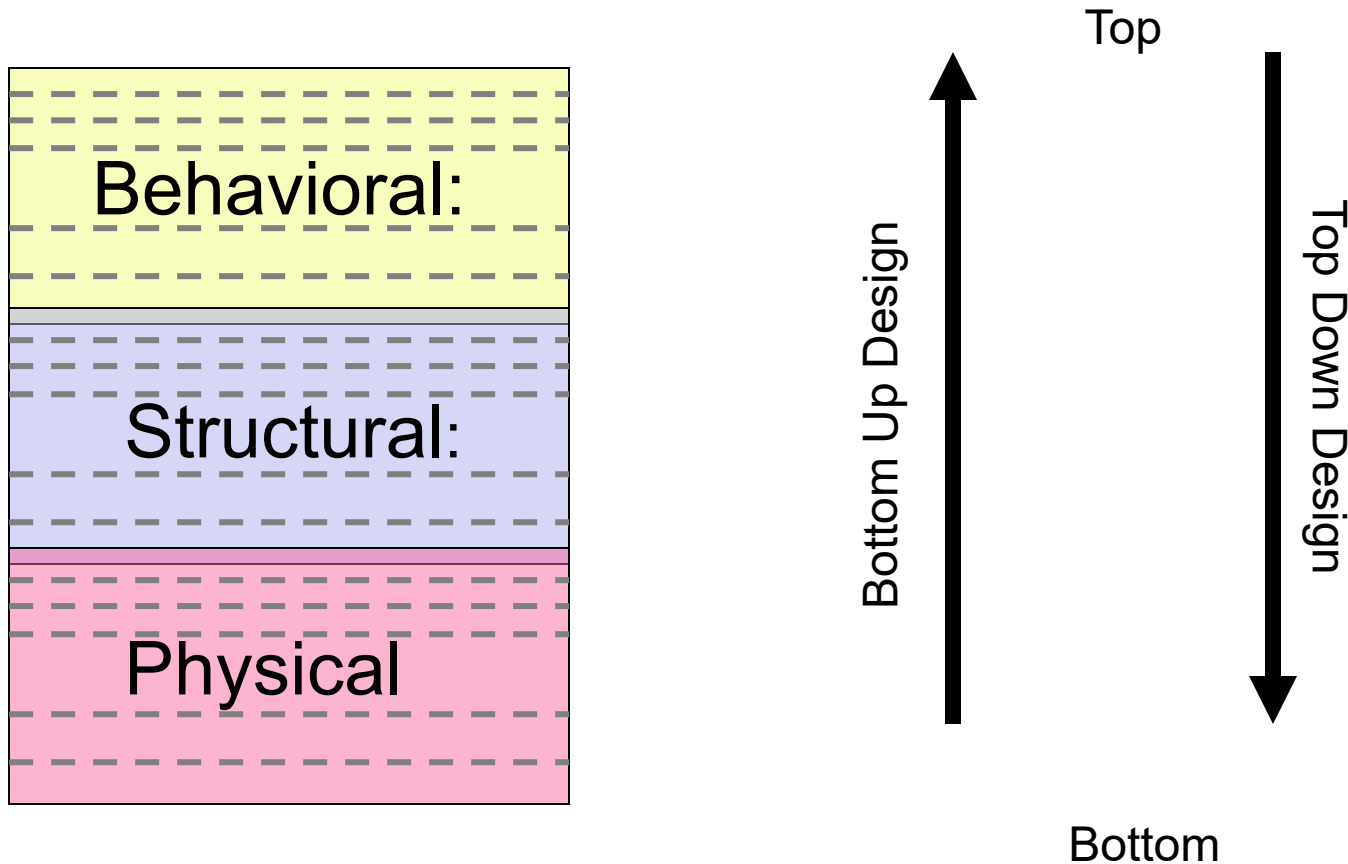


Multiple Levels of Abstraction

# Hierarchical Digital Design Domains:



# Hierarchical Digital Design Domains:



Multiple Sublevels in Each Major Level

All Design Steps may not Fit Naturally in this Description

# Hierarchical Digital Design Domains:

**Behavioral** : Describes what a system does or what it should do

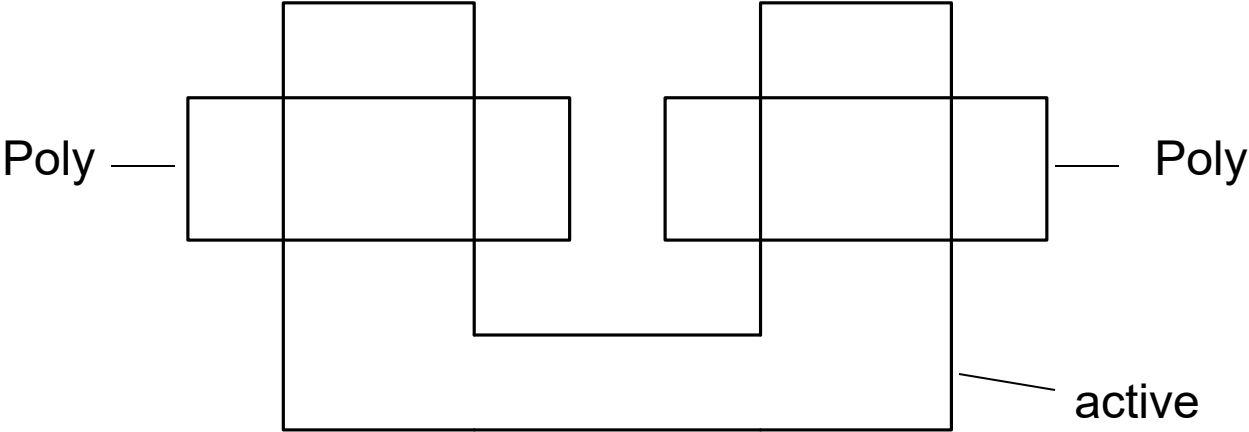
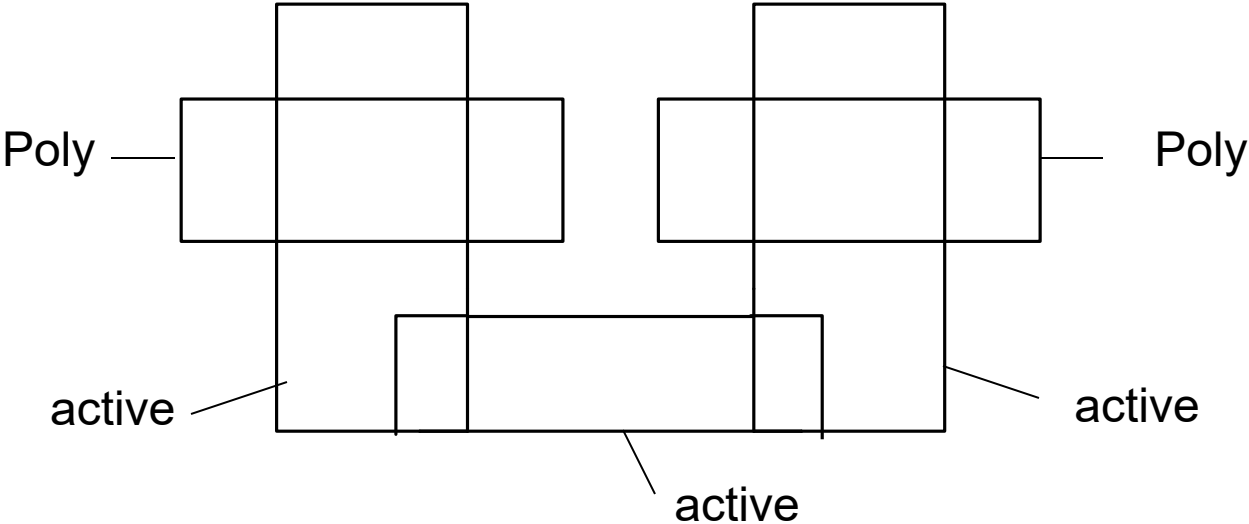
**Structural** : Identifies constituent blocks and describes how these blocks are interconnected and how they interact

**Physical** : Describes the constituent blocks to both the transistor and polygon level and their physical placement and interconnection

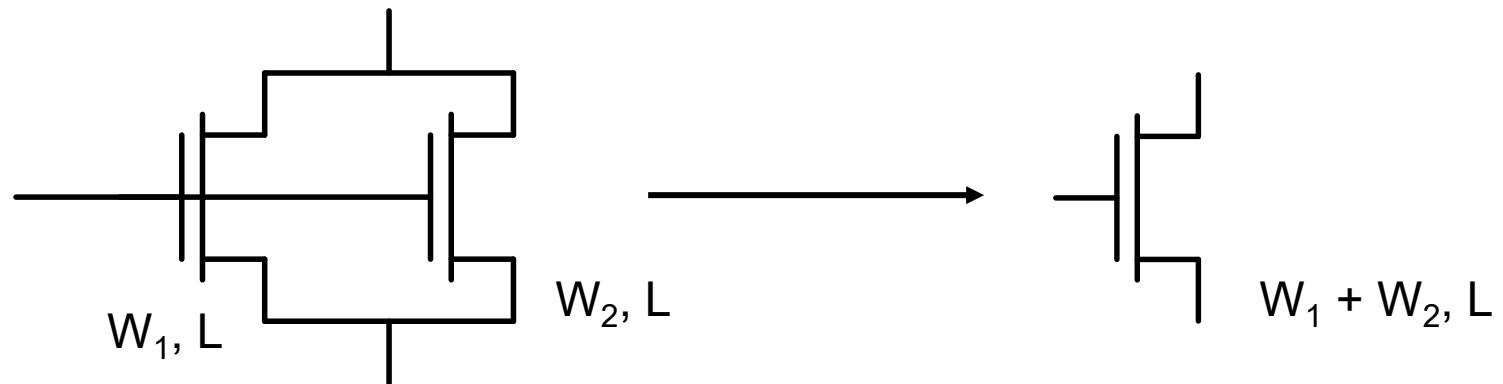
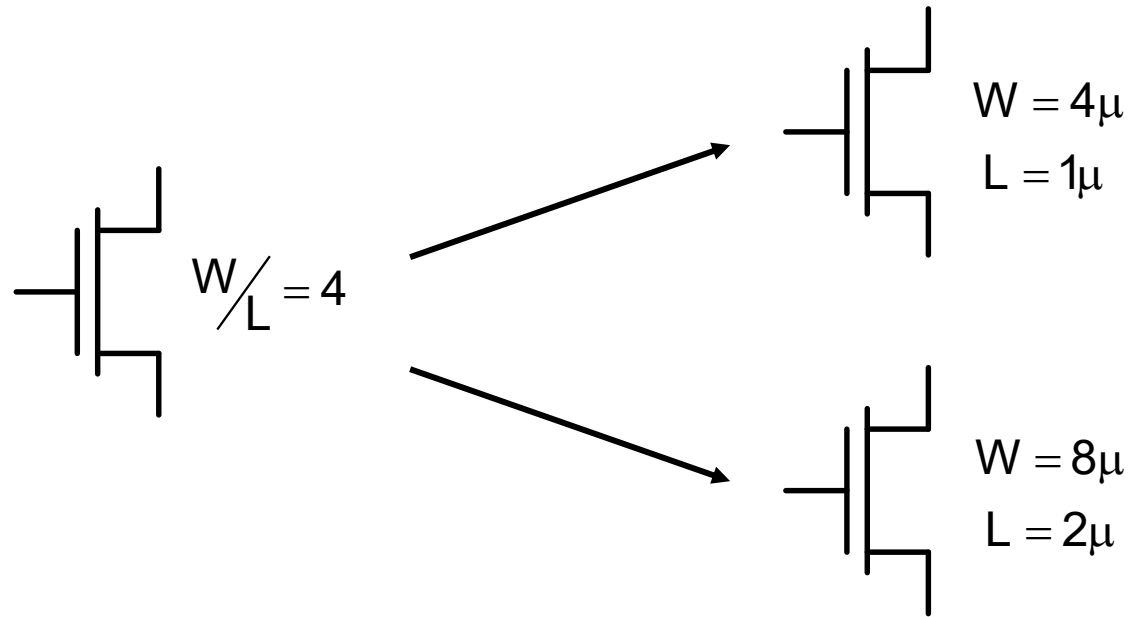
Multiple representations often exist at any level or sublevel



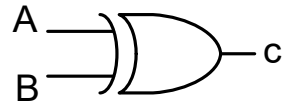
Example: Two distinct representations at the physical level (polygon sublevel)



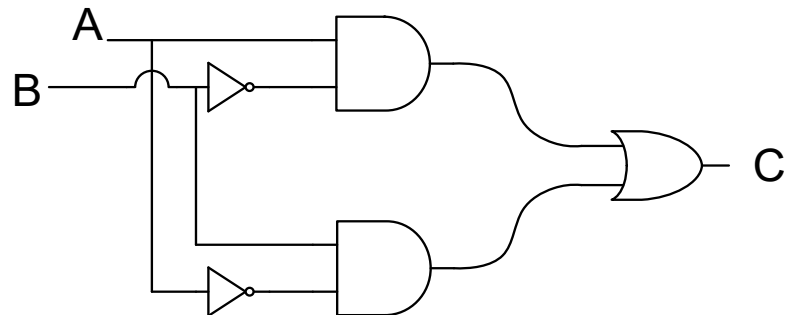
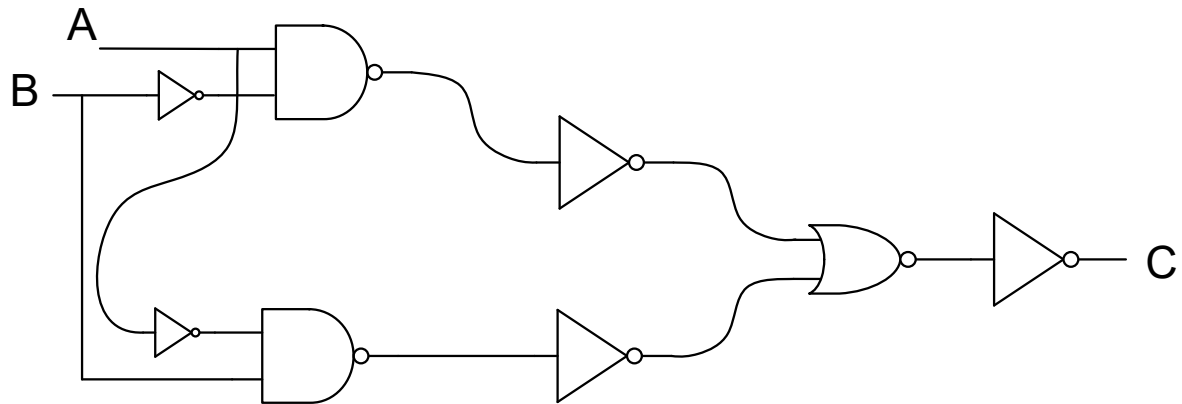
# Example: Two distinct representations at physical level (schematic sublevel)



Example: Three distinct representations at the structural/behavioral level (gate sublevel)



$$C = A \oplus B$$



In each domain, multiple levels of abstraction are generally used.

### Consider Physical Domain

- Consider lowest level to highest
- 0 - placement of diffusions, thin oxide regions, field oxide, ect. on a substrate.
- 1 - polygons identify all mask information  
(not unique)
- 2 - transistors  
(not unique)
- 3 - gate level  
(not unique)
- 4 - cell level  
Adders, Flip Flop, MUTs,...

### Information Type

**PG data**

**G.D.F**

**Netlist**

**HDL Description**

## Structural Domain:

- DSP
- Blocks (Adders, Memory, Registers, etc.)
- Gates
- Transistor

## Information Type

HDL

Netlists

## Behavior Domain (top down):

- Application
- Programs
- Subroutines
- Boolean Expressions

## Information Type

High-Level Language  
HDL

# Representation of Digital Systems

## Standard Approach to Digital Circuit Design

### 8 – level representation

1. Behavioral Description
  - Technology independent
2. RTL Description (Register Transfer Level)  
(must verify (1)  $\Leftrightarrow$  (2))
3. RTL Compiler
  - Registers and Combinational Logic Functions
4. Logic Optimizer
5. Logic Synthesis
  - Generally use a standard cell library for synthesis

(sublevels 6-8 not shown on this slide)

# Frontend design

## Representation of Digital Systems

### Standard Approach to Digital Circuit Design

1. Behavioral Description
  - Technology independent
2. RTL Description
  - (must verify (1)  $\Leftrightarrow$  (2))
3. RTL Compiler
  - Registers and Combinational Logic Functions
4. Logic Optimizer

## 5. Logic Synthesis

Generally use a standard cell library for synthesis





# Backend design

## 6. Place and Route

(physically locates all gates and registers and interconnects them)

### 7. Layout Extraction

- DRC
- Back Annotation

### 8. Post Layout simulation

May necessitate a return to a higher level in the design flow

Logic synthesis, though extensively used, often is not as efficient nor as optimal for implementing some important blocks or some important functions

These applications generally involve transistor level logic circuit design that may combine one or more different logic design styles

# Logic Optimization

What is optimized (or minimized) ?





- Number of Gates
- Number or Levels of Logic
- Speed
- Delay
- Power Dissipation
- Area
- Cost
- Peak Current
- • •

Depends Upon What User Is Interested In

# Standard Cell Library

- Set of primitive building blocks that have been pre-characterized for dc and high frequency performance
- Generally includes basic multiple-input gates and flip flops
- P-cells often included
- Can include higher-level blocks
  - Adders, multipliers, shift registers, counters, ...
- Cell library often augmented by specific needs of a group or customer

# Digital Circuit Design

-  • Hierarchical Design
  -  • Basic Logic Gates
  -  • Properties of Logic Families
  -  • Characterization of CMOS Inverter
    - Static CMOS Logic Gates
      - Ratio Logic
    - Propagation Delay
      - Simple analytical models
        - FI/OD
        - Logical Effort
      - Elmore Delay
    - Sizing of Gates
      - The Reference Inverter
  - Propagation Delay with Multiple Levels of Logic
  - Optimal driving of Large Capacitive Loads
  - Power Dissipation in Logic Circuits
  - Other Logic Styles
  - Array Logic
  - Ring Oscillators
- 

 done

 partial

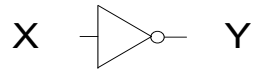
# Logic Circuit Block Design

Many different logic design styles

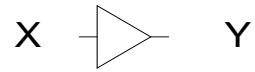
- Static Logic Gates
- Complex Logic Gates
- Pseudo NMOS
- Pass Transistor Logic
- Dynamic Logic Gates
  - Domino Logic
  - Zipper Logic
  - Output Prediction Logic

Various logic design styles often combined in the implementation of one logic block

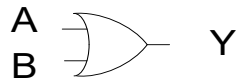
# The basic logic gates



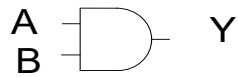
$$Y = \bar{X}$$



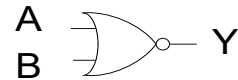
$$Y = X$$



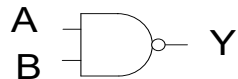
$$Y = A + B$$



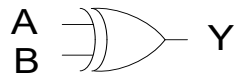
$$Y = A \cdot B$$



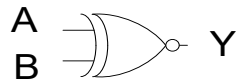
$$Y = \overline{A + B}$$



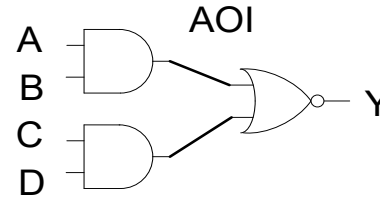
$$Y = \overline{A \cdot B}$$



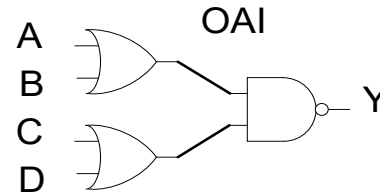
$$Y = A \oplus B$$



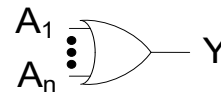
$$Y = \overline{A \oplus B}$$



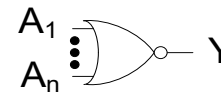
$$Y = \overline{A \cdot B + C \cdot D}$$



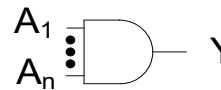
$$Y = \overline{(A + B) \cdot (C + D)}$$



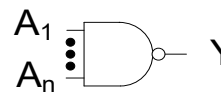
$$Y = A_1 + A_2 + \dots + A_n$$



$$Y = \overline{A_1 + A_2 + \dots + A_n}$$

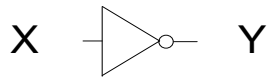


$$Y = A_1 \cdot A_2 \cdot \dots \cdot A_n$$

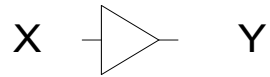


$$Y = \overline{A_1 \cdot A_2 \cdot \dots \cdot A_n}$$

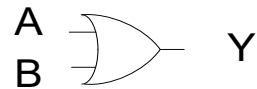
# The basic logic gates



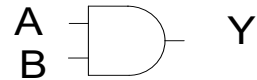
$$Y = \bar{X}$$



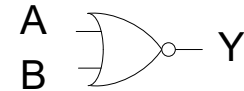
$$Y = X$$



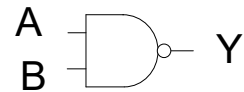
$$Y = A + B$$



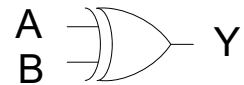
$$Y = A \cdot B$$



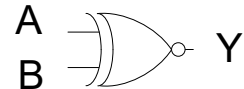
$$Y = \overline{A + B}$$



$$Y = \overline{A \cdot B}$$



$$Y = A \oplus B$$



$$Y = \overline{A \oplus B}$$

Question: How many basic one and two input gates exist and how many of these are useful?

# The basic logic gates

The set of NOR gates is complete

Any combinational logic function can be realized with only multiple-input NOR gates

The set of NAND gates is complete

Any combinational logic function can be realized with only multiple-input NAND gates

Performance of the BASIC gates is critical!



# The basic logic gates

A gate logic family can be formed based upon a specific design style for implementing logic functions

Many different gate logic family types exist

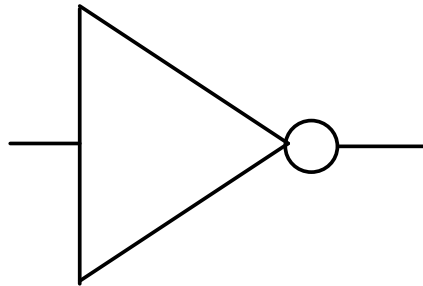
NMOS, PMOS, CMOS, TTL, ECL, RTL, DCTL,...

Substantial differences in performance from one family type to another

Power, Area, Noise Margins, ....

# The basic logic gates

It suffices to characterize the inverter of a logic family and then express the performance of other gates in that family in terms of the performance of the inverter.



What characteristics are required and desirable for an inverter to form the basis for a useful logic family?



Stay Safe and Stay Healthy !

End of Lecture 37