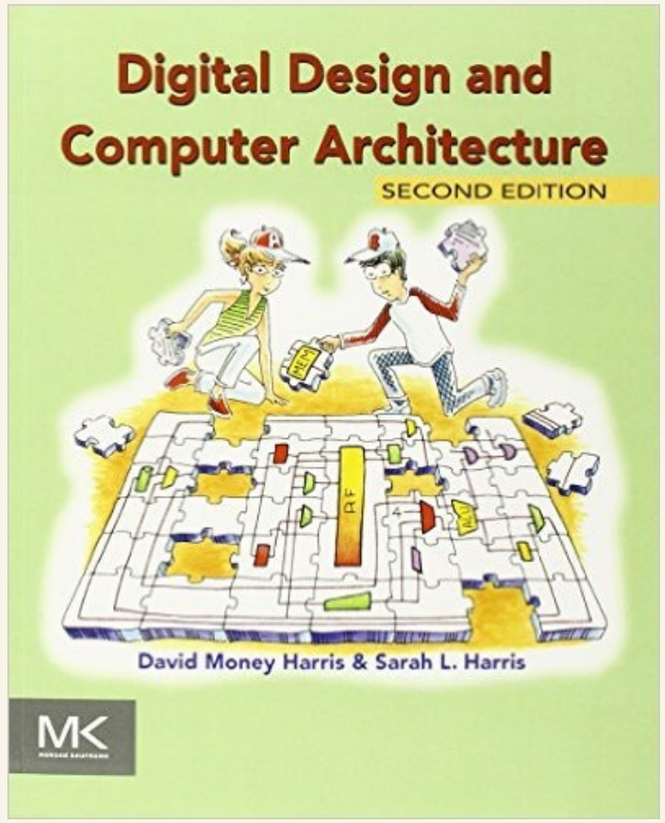
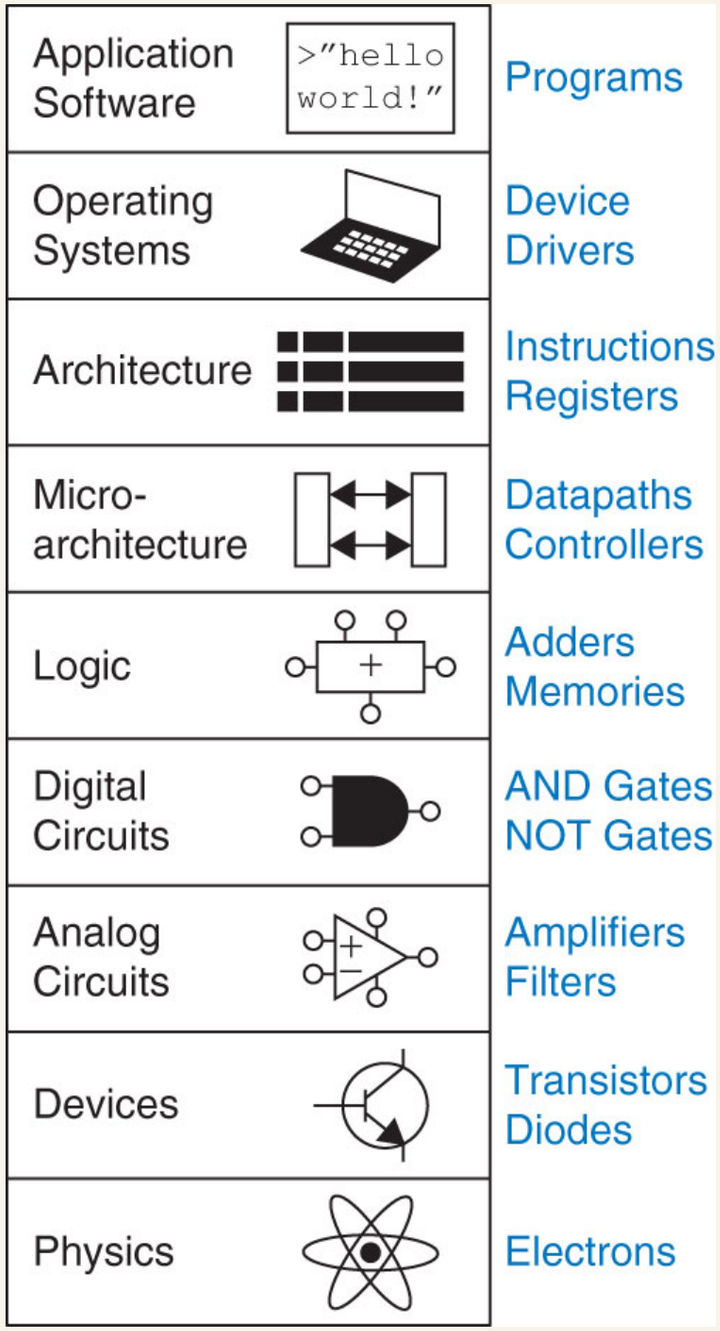
Introduction to Digital Design

- The Digital Abstraction
- Number Systems
- Logic Gates
- Beneath the Digital Abstraction



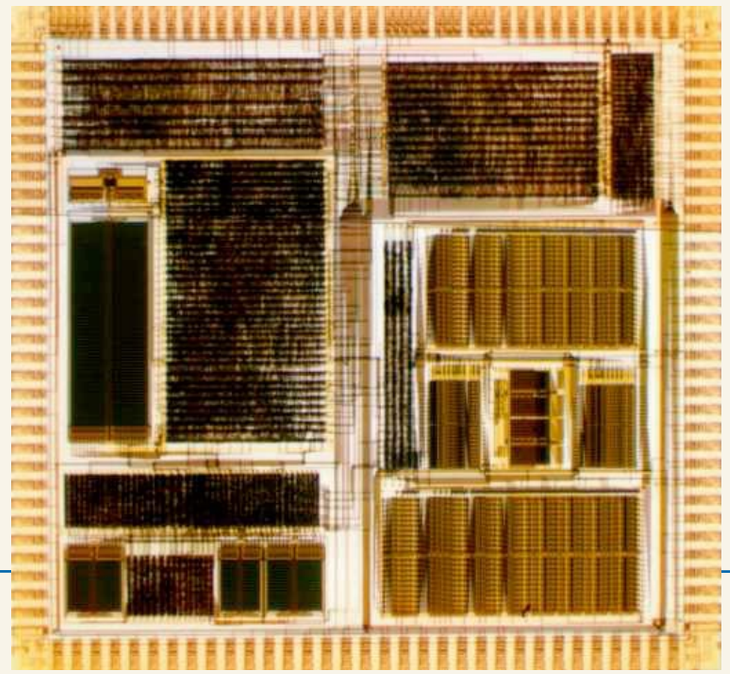
Chap 1

The Digital Abstraction



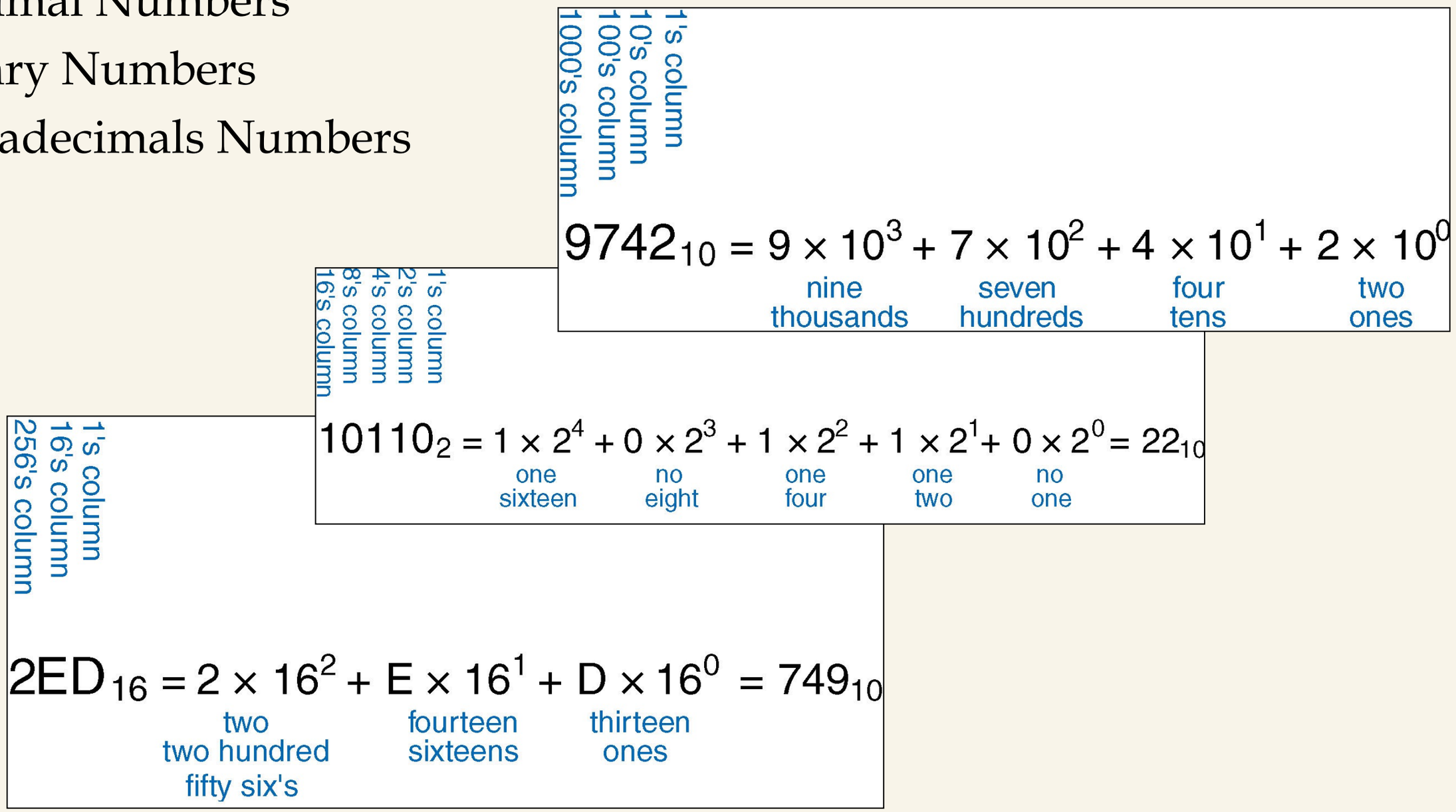
- The Three –Y's
  - Hierarchy
    - » A system divided into modules and submodules
  - Modularity
    - » Having well-defined functions and interfaces
  - Regularity
    - » Encouraging uniformity, so modules can be easily reused

- Positive voltage is commonly used to represent '1' and zero volt to represent '0'



Number Systems

- Decimal Numbers
- Binary Numbers
- Hexadecimals Numbers



- Decimal to Binary Conversion

Method 1: Find the largest power of 2 that fits, subtract and repeat

53_{10}	32×1
53-32 = 21	16×1
21-16 = 5	4×1
5-4 = 1	1×1

= 110101_2

Method 2: Repeatedly divide by 2, remainder goes in next most significant bit

53_{10} =	53/2 = 26	R1
	26/2 = 13	R0
	13/2 = 6	R1
	6/2 = 3	R0
	3/2 = 1	R1
	1/2 = 0	R1

= 110101_2

- Binary Values and Range

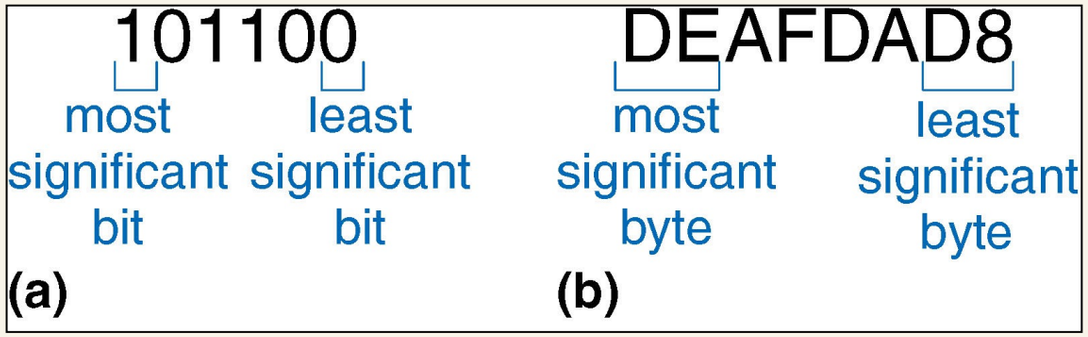
  - ***N*-digit decimal number**
    - How many values? **10^N**
    - Range? **$[0, 10^N - 1]$**
    - Example: 3-digit decimal number:
      - **10^3 = 1000 possible values**
      - **Range: [0, 999]**

  - ***N*-bit binary number**
    - How many values? **2^N**
    - Range: **$[0, 2^N - 1]$**
    - Example: 3-digit binary number:
      - **2^3 = 8 possible values**
      - **Range: [0, 7] = [000_2 to 111_2]**

Hex Digit	Decimal	Binary
0	0	0000
1	1	0001
2	2	0010
3	3	0011
4	4	0100
5	5	0101
6	6	0110
7	7	0111
8	8	1000
9	9	1001
A	10	1010
B	11	1011
C	12	1100
D	13	1101
E	14	1110
F	15	1111

- Hexadecimal to Binary Conversion
  - Hexadecimal to binary conversion:
    - Convert $4AF_{16}$ (also written 0x4AF) to binary
    - $0100\ 1010\ 1111_2$
  - Hexadecimal to decimal conversion:
    - Convert $4AF_{16}$ to decimal
    - $16^2 \times 4 + 16^1 \times 10 + 16^0 \times 15 = 1199_{10}$

- Byte, Word, MSB, LSB



- Estimating Powers of Two
  - 2^{10} = 1 kilo ≈ 1000 (1024)
  - 2^{20} = 1 mega ≈ 1 million (1,048,576)
  - 2^{30} = 1 giga ≈ 1 billion (1,073,741,824)

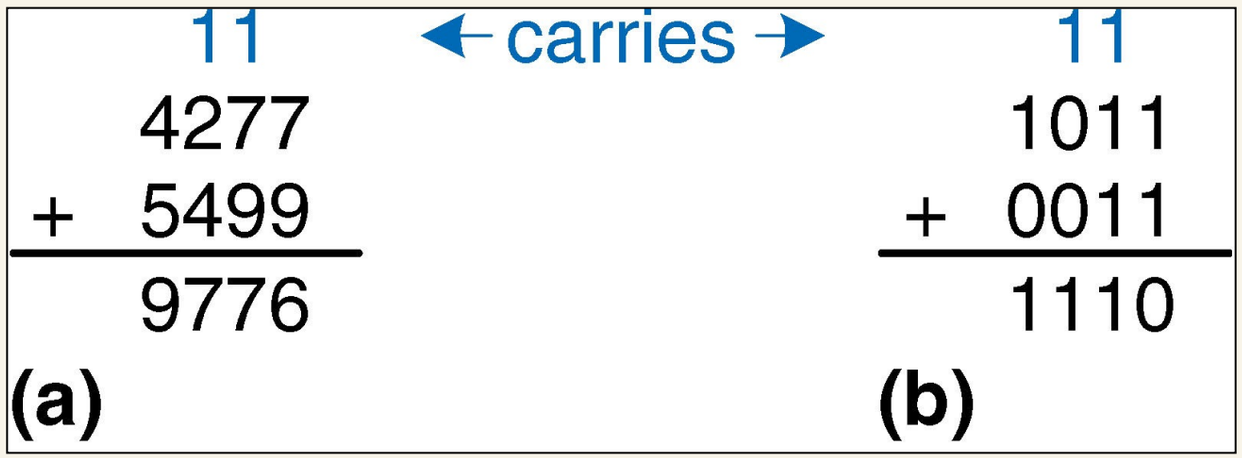
- What is the value of 2^{24}?

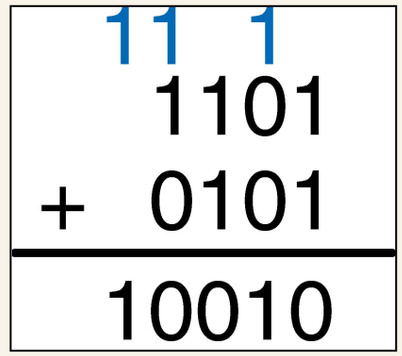
  $2^4 \times 2^{20} \approx$ 16 million

- How many values can a 32-bit variable represent?

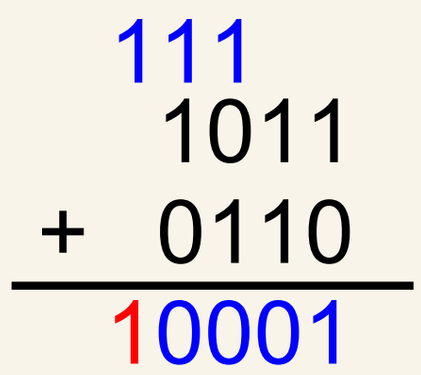
  $2^2 \times 2^{30} \approx$ 4 billion

- Binary Addition





- Overflow
  - » when result is too big to fit in the available number of bits



- Signed Binary Numbers

  - Sign and Magnitude

    - 1 sign bit, *N*-1 magnitude bits
    - Sign bit is the most significant (left-most) bit
      - Positive number: sign bit = 0
      - Negative number: sign bit = 1

$$A: \{a_{N-1}, a_{N-2}, \ldots a_2, a_1, a_0\}$$

$$A = (-1)^{a_{N-1}} \sum_{i=0}^{N-2} a_i \, 2^i$$

    - Example, 4-bit sign/mag representations of ± 6:

      +6 = **0110**

      - 6 = **1110**

    - Range of an *N*-bit sign/magnitude number:

      $[-(2^{N-1}-1), 2^{N-1}-1]$

Problems:

- Addition doesn't work, for example -6 + 6:

  1110

  + 0110

  10100 (wrong!)

- Two representations of 0 (± 0):

  1000

  0000

– Two's Complement

- msb has value of -2^{N-1}

$$A = a_{N-1}(-2^{N-1}) + \sum_{i=0}^{N-2} a_i \, 2^i$$

- Most positive 4-bit number: **0111**
- Most negative 4-bit number: **1000**
- The most significant bit still indicates the sign (1 = negative, 0 = positive)
- Range of an *N*-bit two's complement number:

  $[-(2^{N-1}), 2^{N-1}-1]$

» Taking the Two's Complement

- "Taking the Two's complement" **flips the sign** of a two's complement number
- **Method:**
  1. Invert the bits
  2. Add 1
- **Example:** Flip the sign of $3_{10} = 0011_2$
  1. **1100**
  2. **+ 1**

  $1101 = -3_{10}$

» Two's Complement Addition

- Add 6 + (-6) using two's complement numbers

```
  111
   0110
+  1010
------
 10000
```

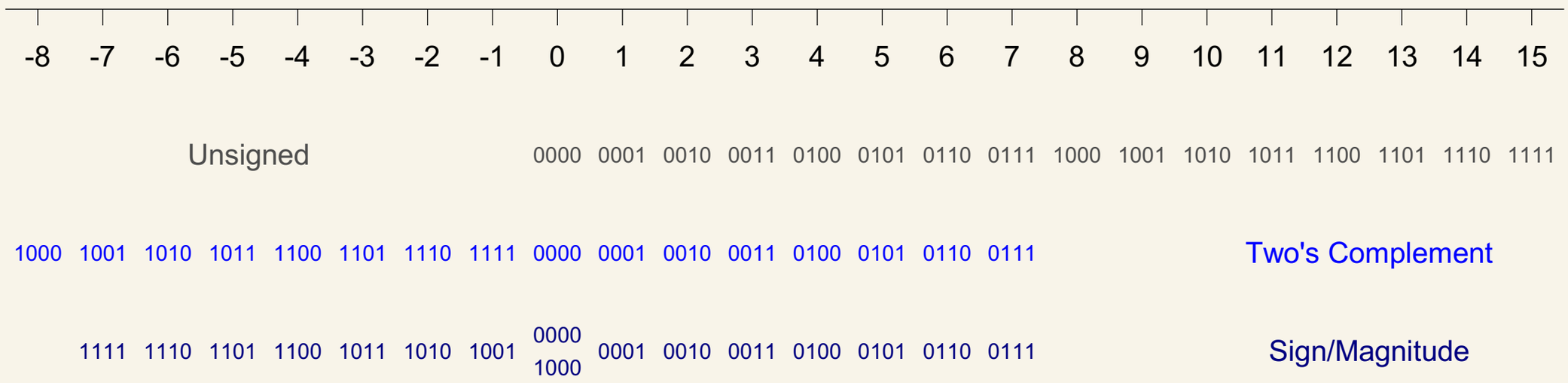
- Add -2 + 3 using two's complement numbers

```
  111
   1110
+  0011
------
 10001
```

- Sign-Extension
  - Sign bit copied to msb's
  - Number value is same
  - **Example 1:**
    - 4-bit representation of 3 = **0**011
    - 8-bit sign-extended value: **0000**0011
  - **Example 2:**
    - 4-bit representation of -5 = **1**011
    - 8-bit sign-extended value: **1111**1011

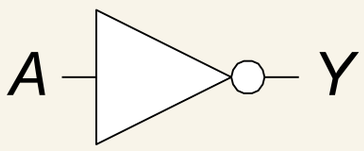
- Number System Comparison

Number System	Range
Unsigned	$[0, 2^N\text{-}1]$
Sign/Magnitude	$[-(2^{N-1}\text{-}1), 2^{N-1}\text{-}1]$
Two's Complement	$[-2^{N-1}, 2^{N-1}\text{-}1]$



Logic Gates

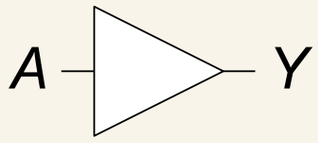
NOT



$Y = \overline{A}$

A	Y
0	1
1	0

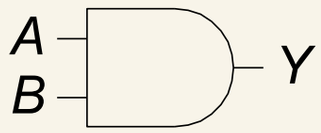
BUF



$Y = A$

A	Y
0	0
1	1

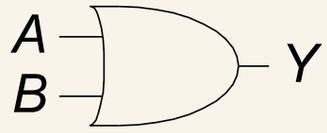
AND



$Y = AB$

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

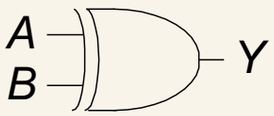
OR



$Y = A + B$

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

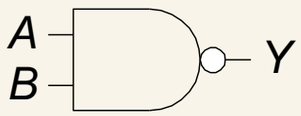
XOR



$Y = A \oplus B$

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

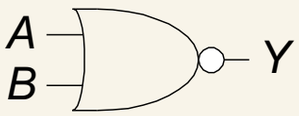
NAND



$Y = \overline{AB}$

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

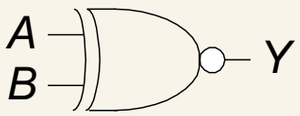
NOR



$Y = \overline{A + B}$

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

XNOR

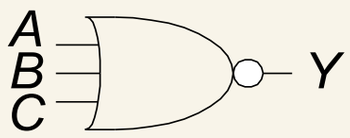


$Y = \overline{A \oplus B}$

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

- Multiple-Input Logic Gates

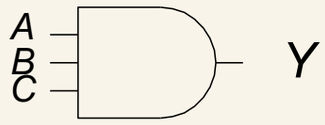
NOR3



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

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

AND3



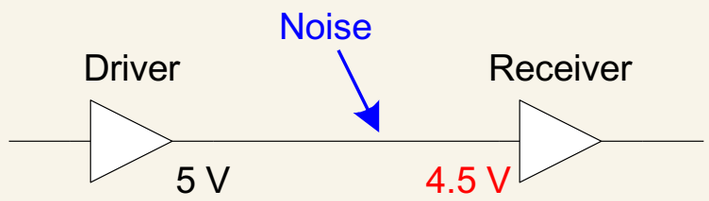
$Y = ABC$

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

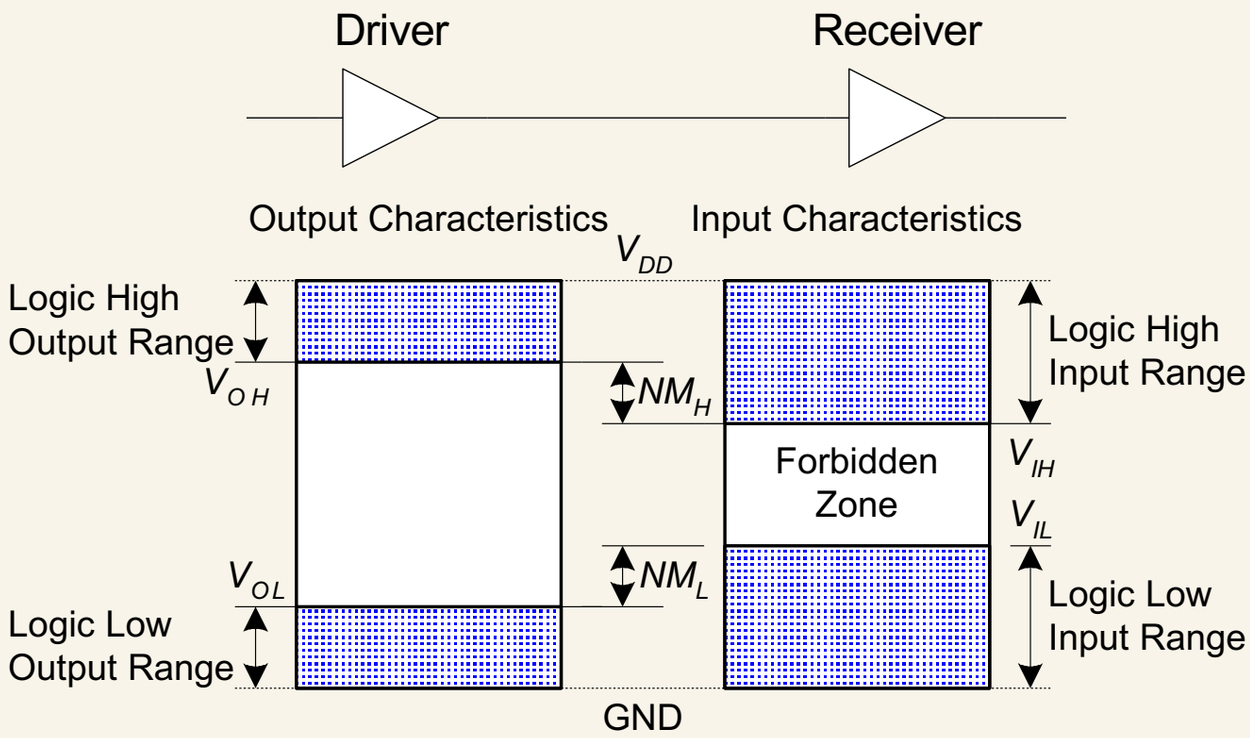
Beneath the Digital Abstraction

- Logic levels
  - Discrete voltages represent 1 and 0
  - For example:
    - 0 = *ground* (GND) or 0 volts
    - 1 = V_{DD} or 5 volts
  - What about 4.99 volts? Is that a 0 or a 1?
  - What about 3.2 volts?

- What is Noise ?
  - **Anything that degrades the signal**
    - E.g., resistance, power supply noise, coupling to neighboring wires, etc.
  - **Example:** a gate (driver) outputs 5 V but, because of resistance in a long wire, receiver gets 4.5 V



- Noise Margins

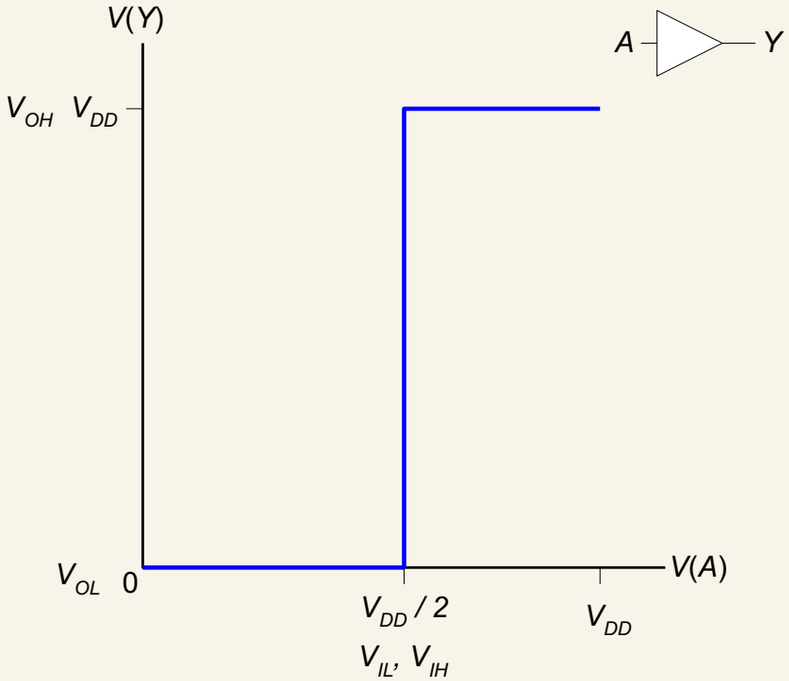


High Noise Margin: $NM_H = V_{OH} - V_{IH}$

Low Noise Margin: $NM_L = V_{IL} - V_{OL}$

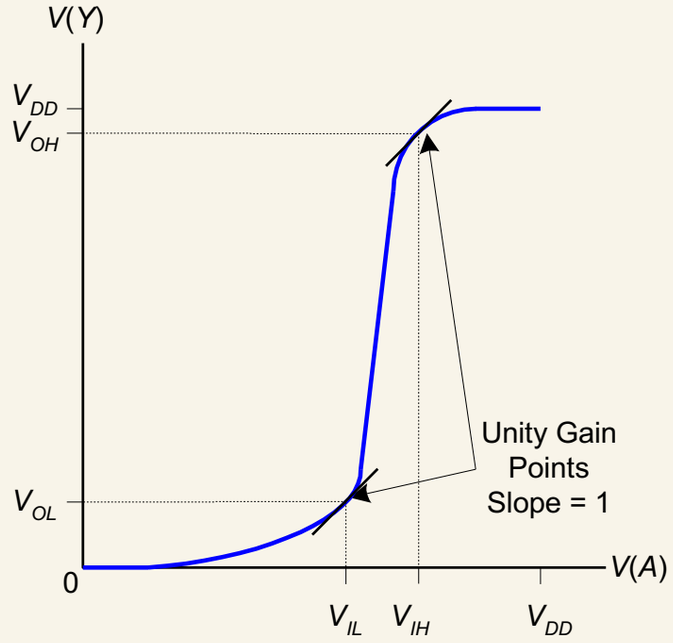
- DC Transfer Characteristics

Ideal Buffer:

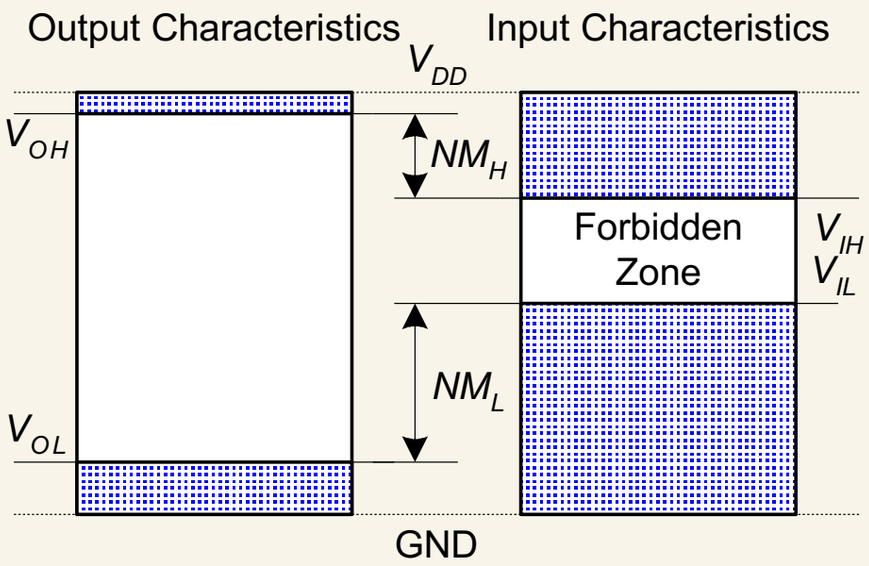


$NM_H = NM_L = V_{DD}/2$

Real Buffer:



$NM_H, NM_L < V_{DD}/2$

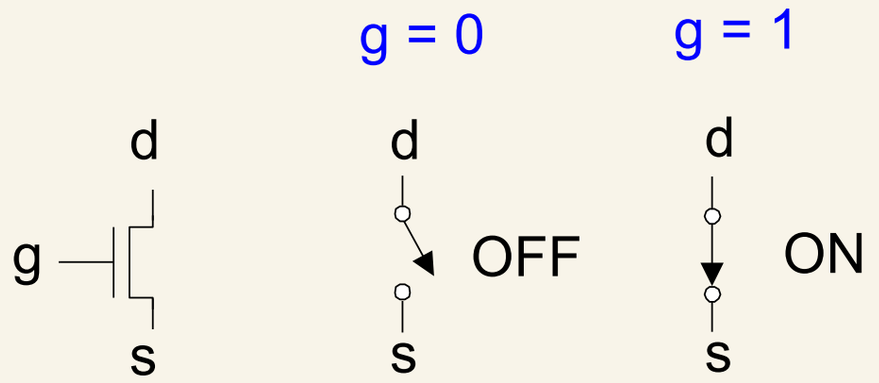


- V_{DD} Scaling

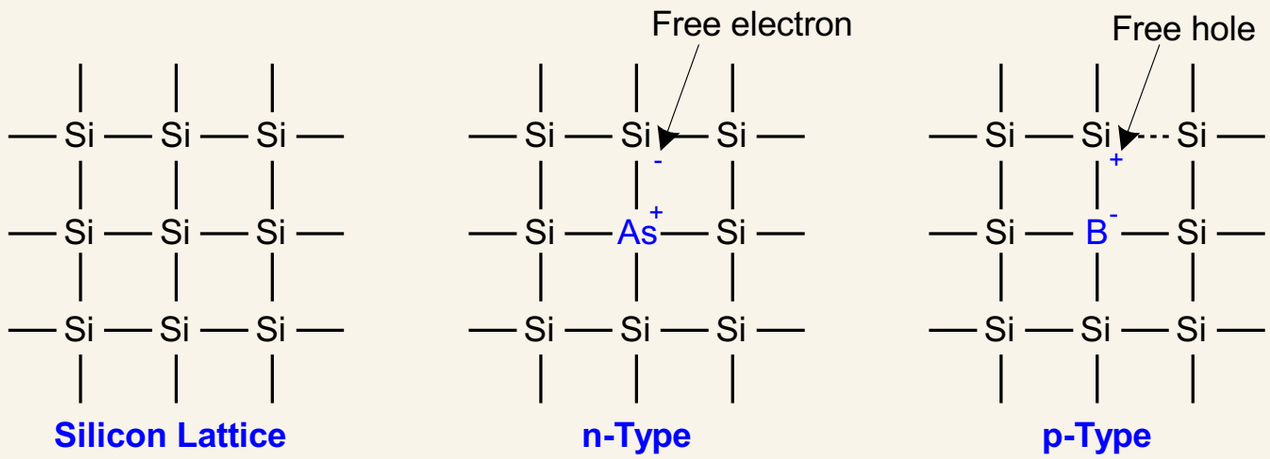
  - In 1970's and 1980's, V_{DD} = 5 V
  - V_{DD} has dropped
    - Avoid frying tiny transistors
    - Save power
  - 3.3 V, 2.5 V, 1.8 V, 1.5 V, 1.2 V, 1.0 V, …
    - Be careful connecting chips with different supply voltages

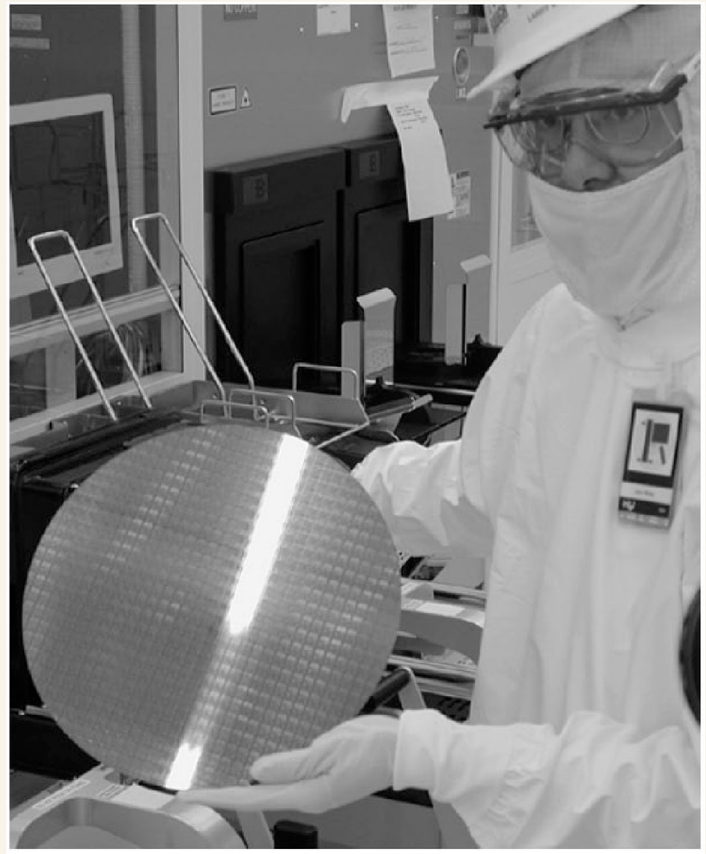
Logic Family	V_{DD}	V_{IL}	V_{IH}	V_{OL}	V_{OH}
TTL	5 (4.75 - 5.25)	0.8	2.0	0.4	2.4
CMOS	5 (4.5 - 6)	1.35	3.15	0.33	3.84
LVTTL	3.3 (3 - 3.6)	0.8	2.0	0.4	2.4
LVCMOS	3.3 (3 - 3.6)	0.9	1.8	0.36	2.7

- Transistors
  - Logic gates built from transistors
  - 3-ported voltage-controlled switch
    - 2 ports connected depending on voltage of 3rd
    - d and s are connected (ON) when g is 1



  - Transistors built from silicon, a semiconductor
  - Pure silicon is a poor conductor (no free charges)
  - Doped silicon is a good conductor (free charges)
    - n-type (free ***n***egative charges, electrons)
    - p-type (free ***p***ositive charges, holes)

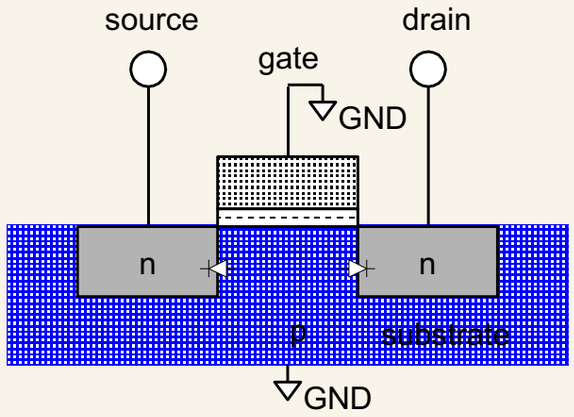




- nMOS Transistors

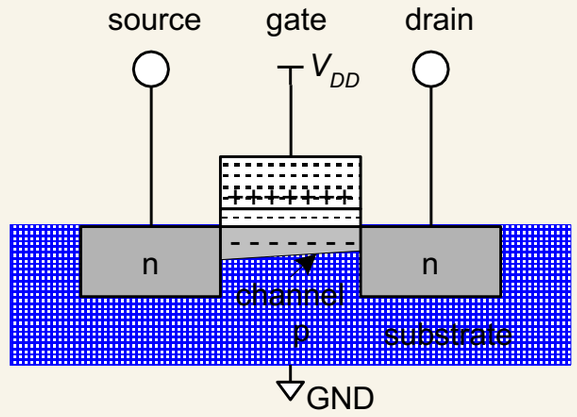
Gate = 0

OFF (no connection between source and drain)



Gate = 1

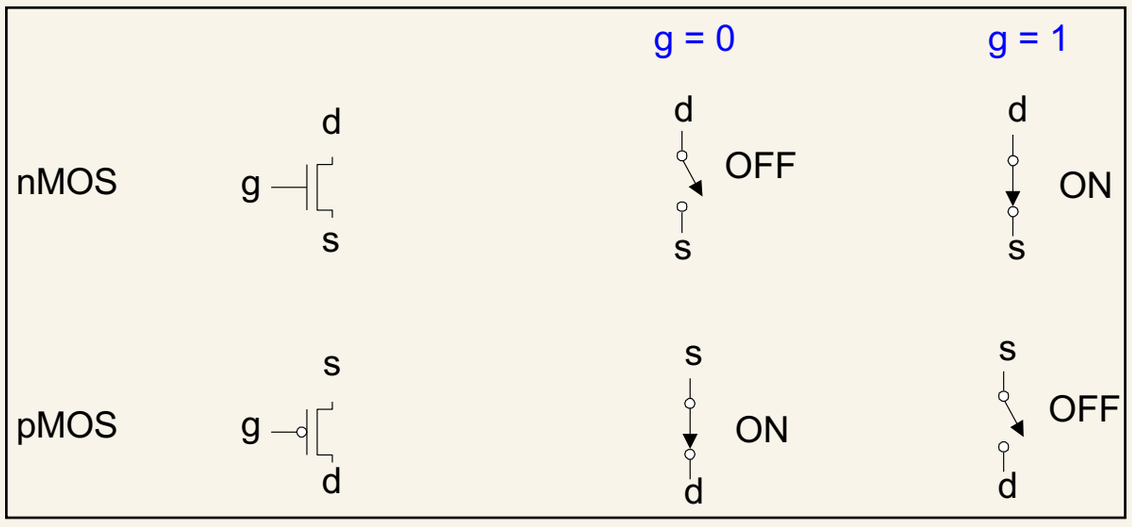
ON (channel between source and drain)

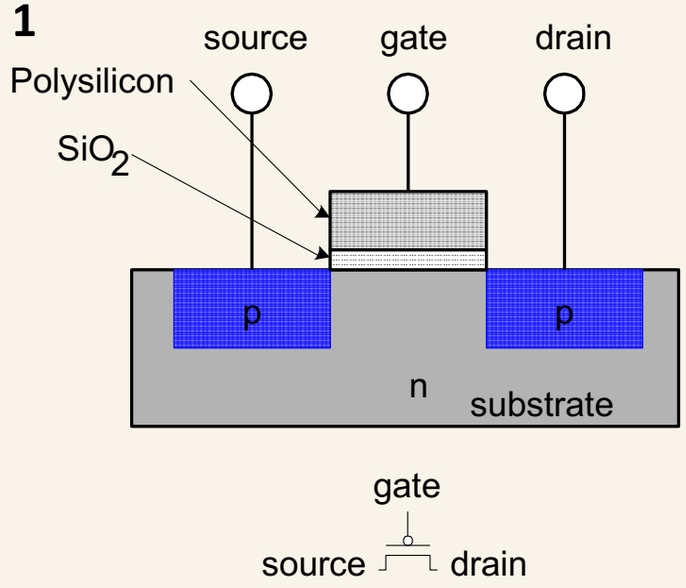


- pMOS Transistors

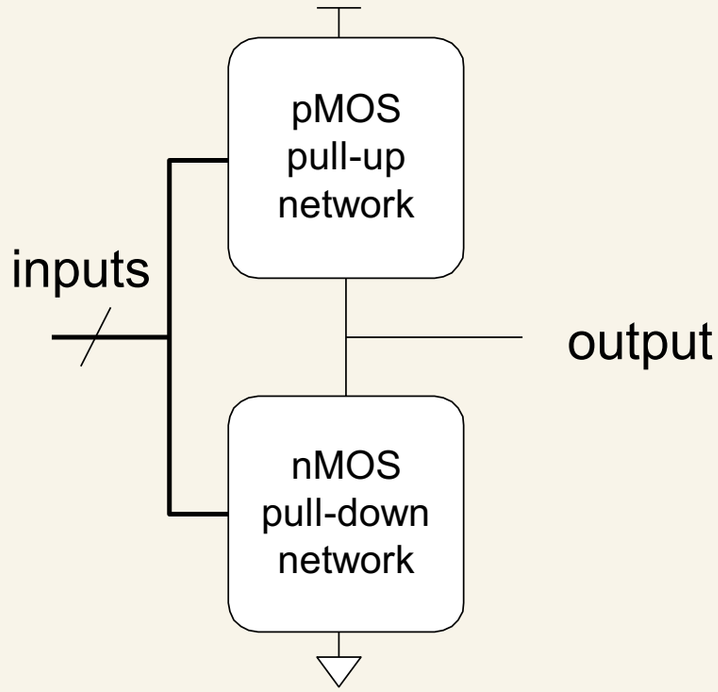
pMOS transistor is opposite

- **ON** when **Gate = 0**
- **OFF** when **Gate = 1**

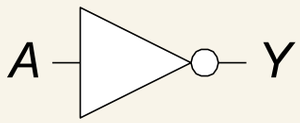




- Gate Implementation
  - **nMOS**: pass good **0**'s, so connect source to GND
  - **pMOS**: pass good **1**'s, so connect source to V_{DD}

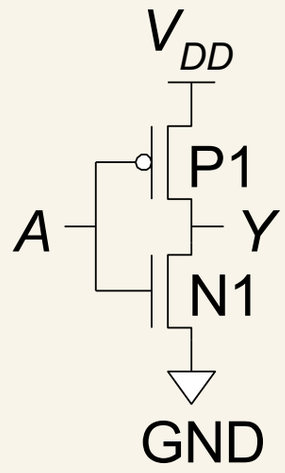


NOT



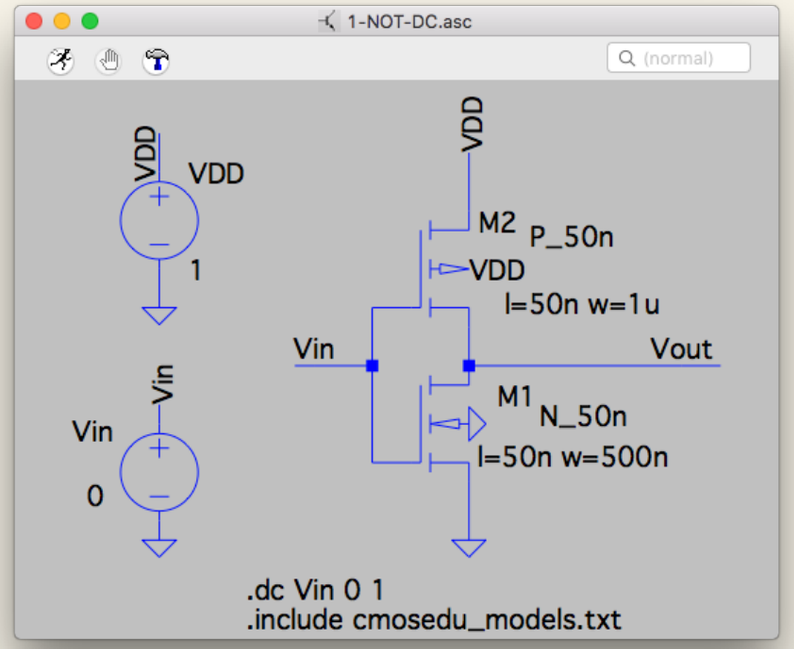
$Y = \overline{A}$

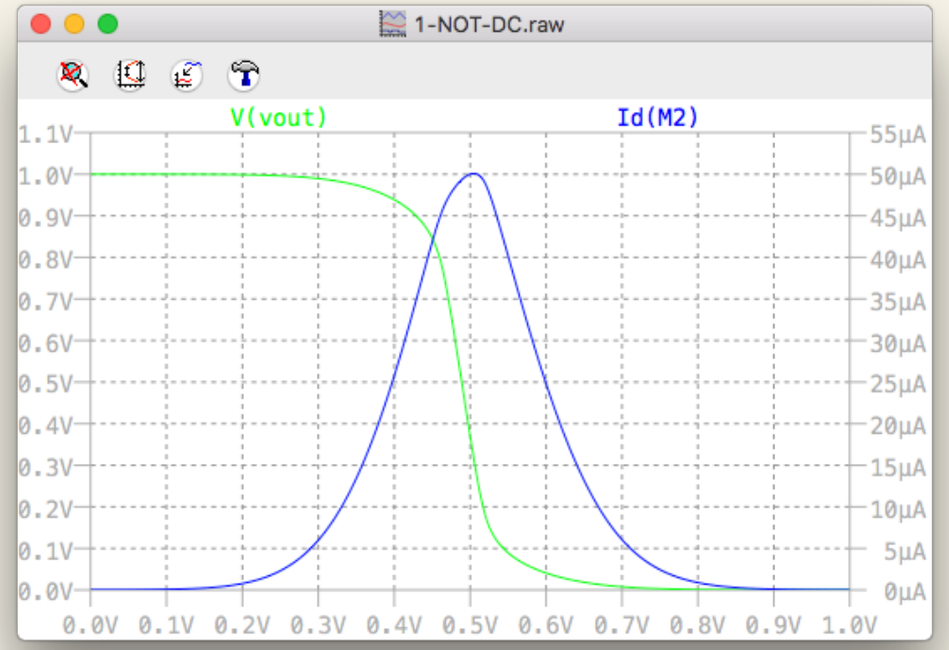
A	Y
0	1
1	0



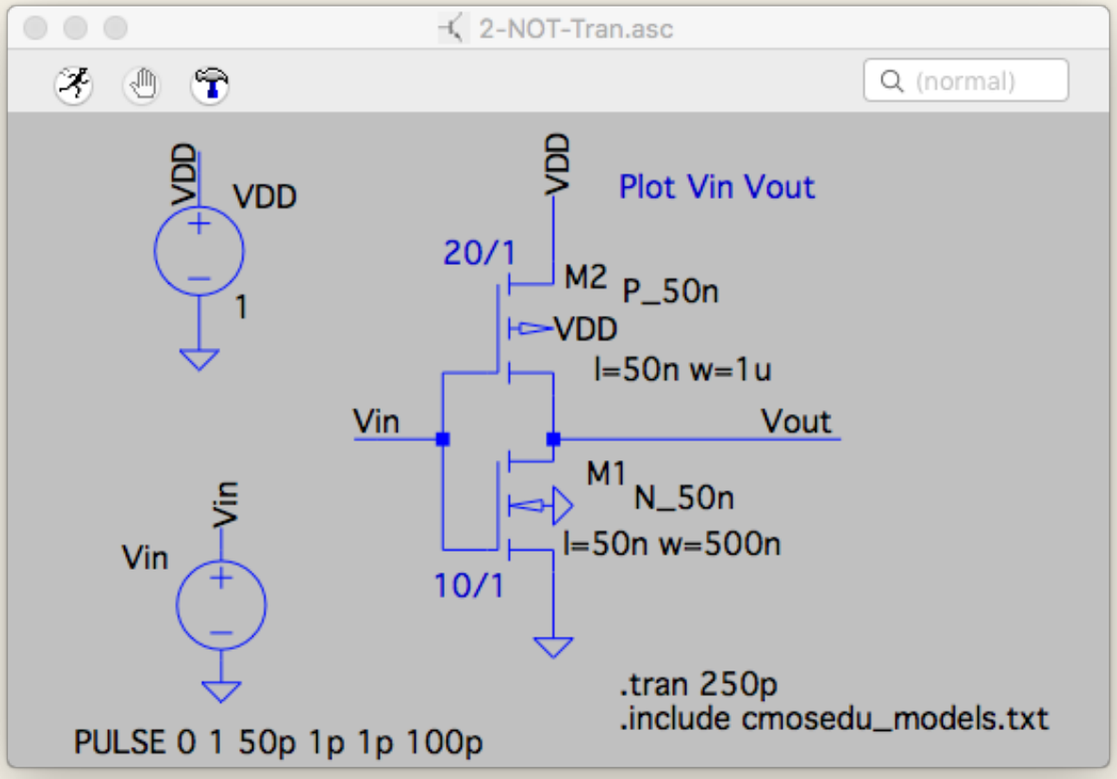
A	P1	N1	*Y*
0	ON	OFF	1
1	OFF	ON	0

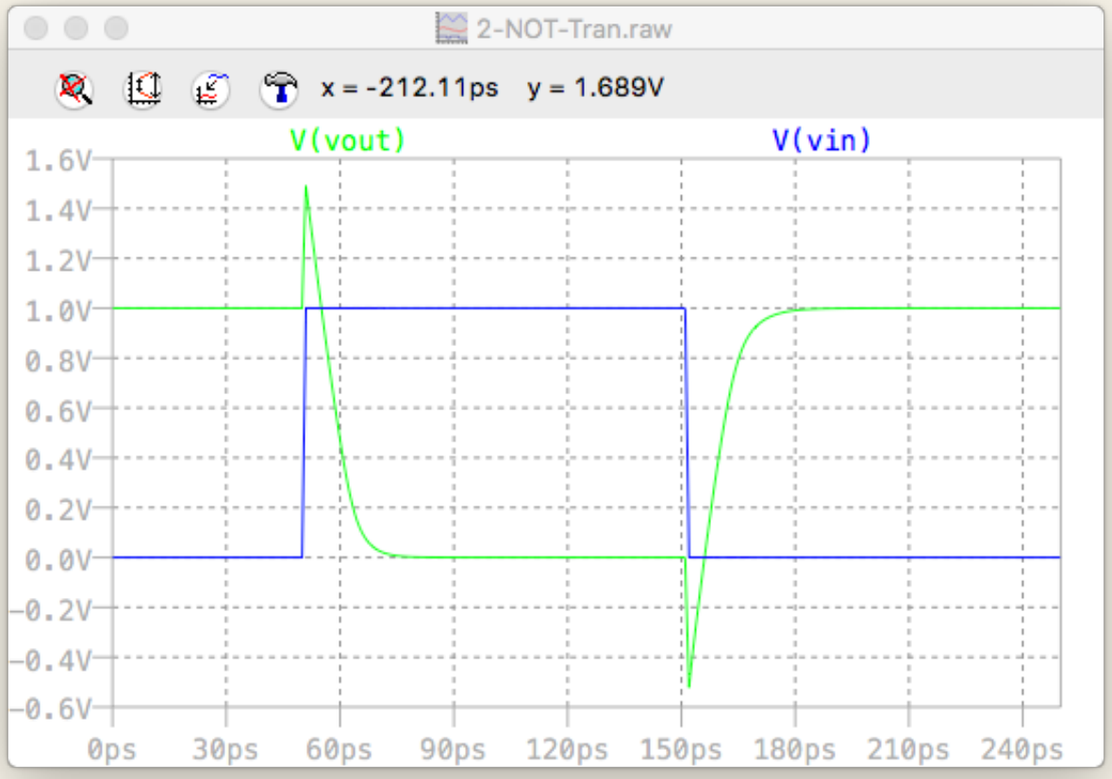
- Gate Simulation
  - 50 nm CMOS
  - V_{DD} = 1 V
  - 2 parameters : L (long) W (Width)
    - » L_n = 50 nm, W_n = 500 nm
    - » L_p = 50 nm, W_p = 2 W_n
  - Transfer Characteristics :

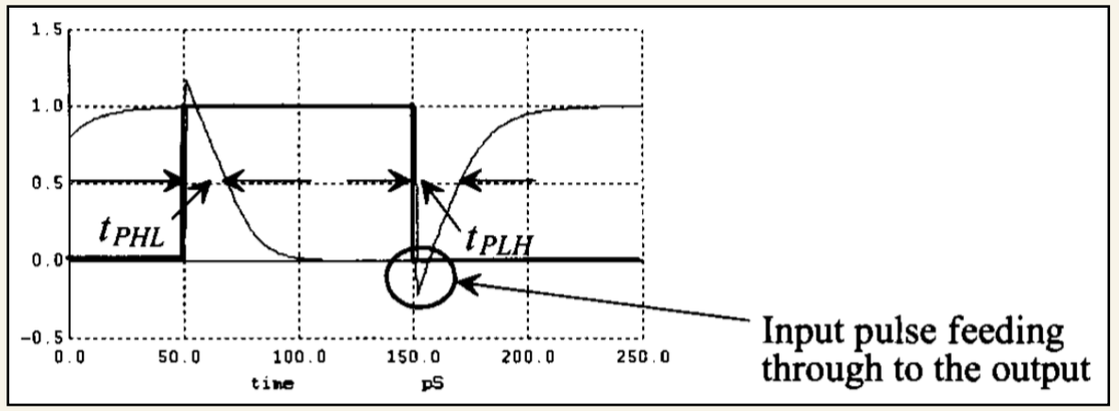


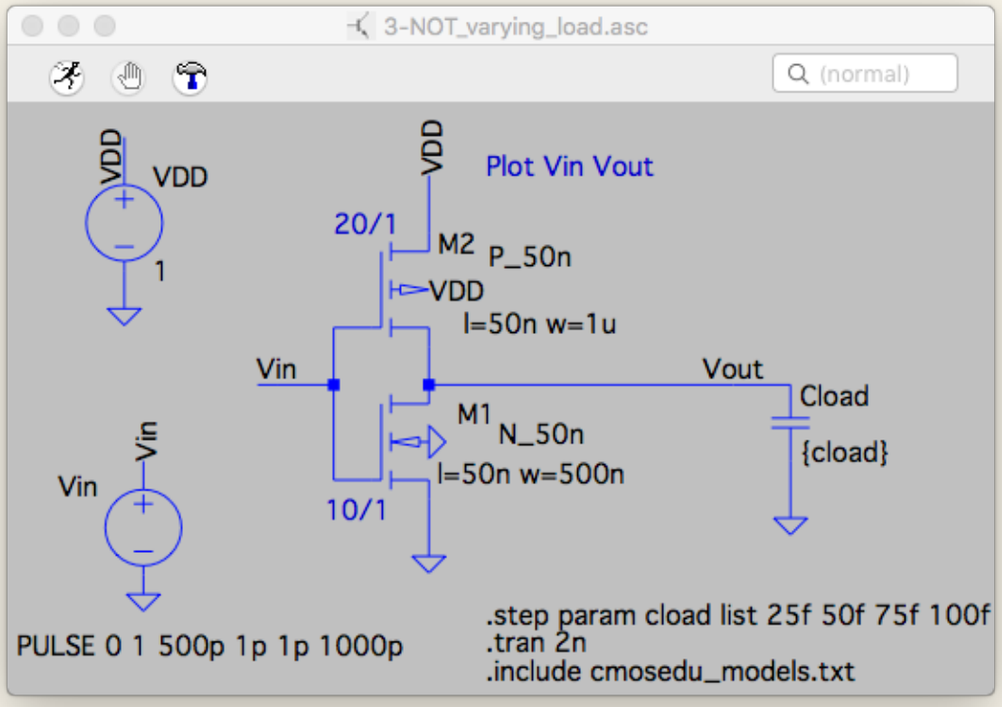


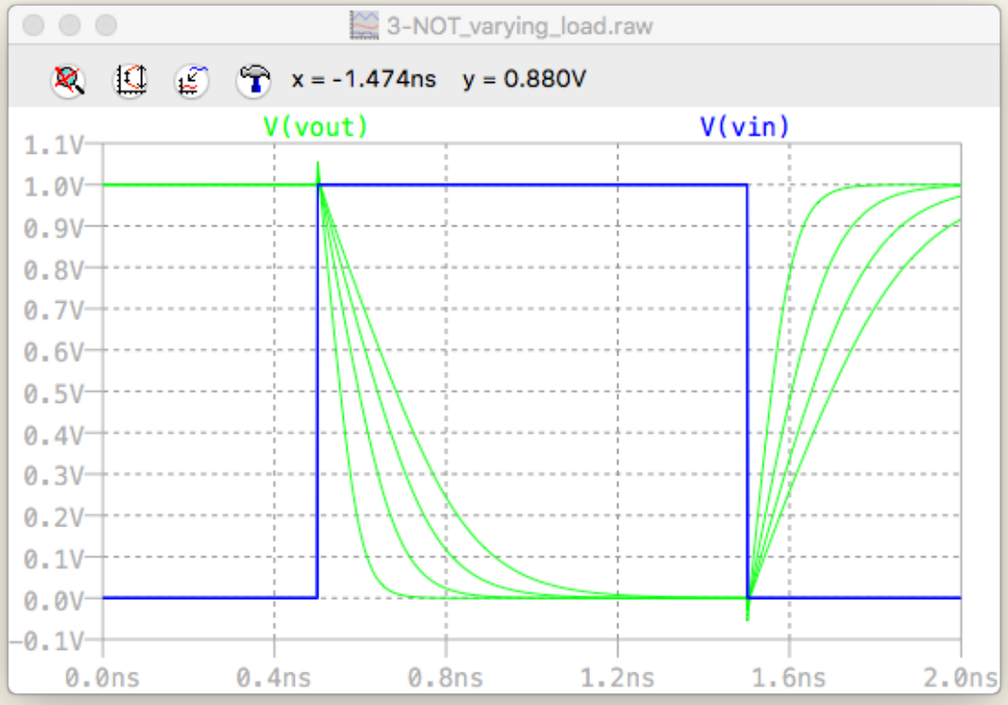
– Switching Characteristics

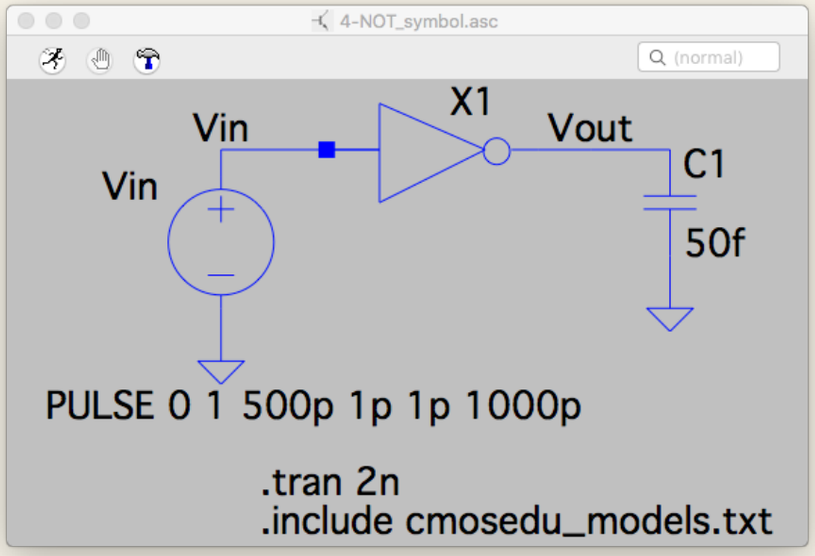


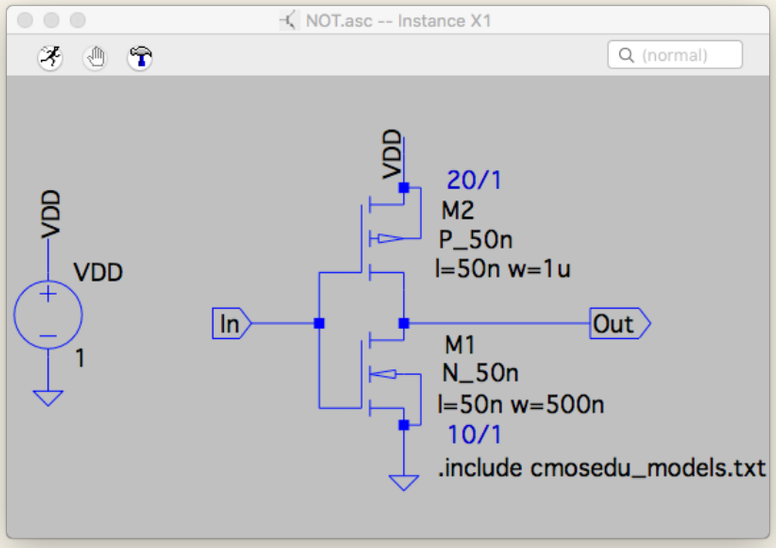


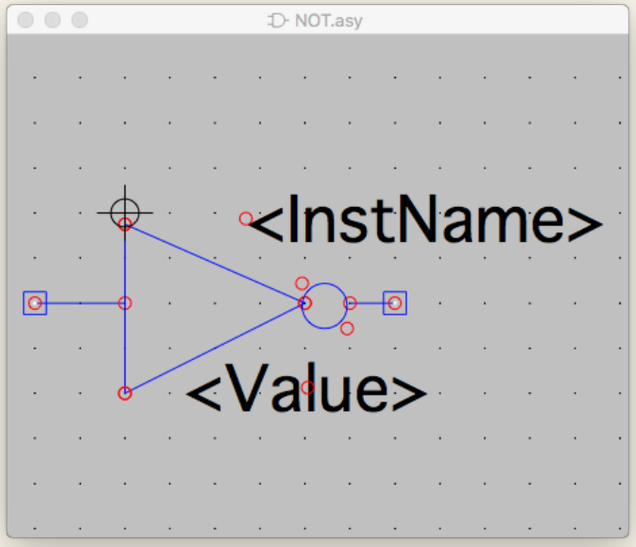



3-NOT_varying_load.asc
(normal)
VDD
VDD
1
Plot Vin Vout
VDD
20/1
M2
P_50n
VDD
l=50n w=1u
Vin
Vout
Cload
{cload}
M1
N_50n
l=50n w=500n
10/1
Vin
Vin
PULSE 0 1 500p 1p 1p 1000p
.step param cload list 25f 50f 75f 100f
.tran 2n
.include cmosedu_models.txt


3-NOT_varying_load.raw
x = -1.474ns y = 0.880V
V(vout)
V(vin)
1.1V
1.0V
0.9V
0.8V
0.7V
0.6V
0.5V
0.4V
0.3V
0.2V
0.1V
0.0V
-0.1V
0.0ns
0.4ns
0.8ns
1.2ns
1.6ns
2.0ns

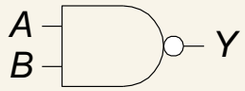

4-NOT_symbol.asc
(normal)
X1
Vin
Vout
Vin
C1
50f
PULSE 0 1 500p 1p 1p 1000p
.tran 2n
.include cmosedu_models.txt


NOT.asc -- Instance X1
(normal)
VDD
VDD
1
VDD
20/1
M2
P_50n
l=50n w=1u
In
Out
M1
N_50n
l=50n w=500n
10/1
.include cmosedu_models.txt


NOT.asy
<InstName>
<Value>

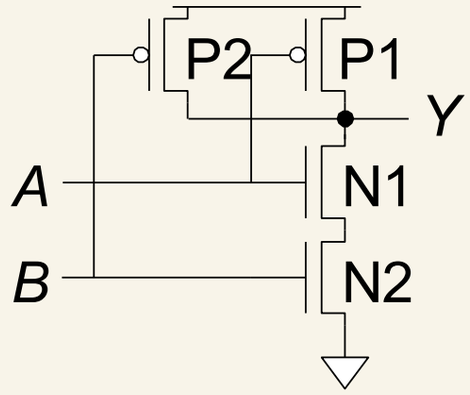
– NAND

NAND

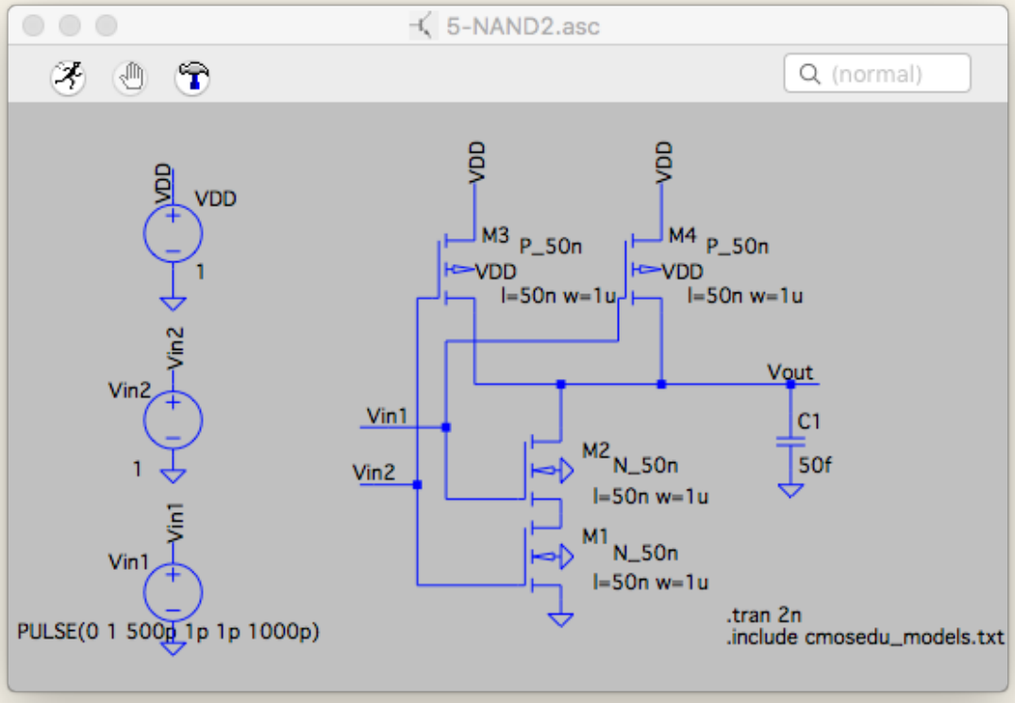


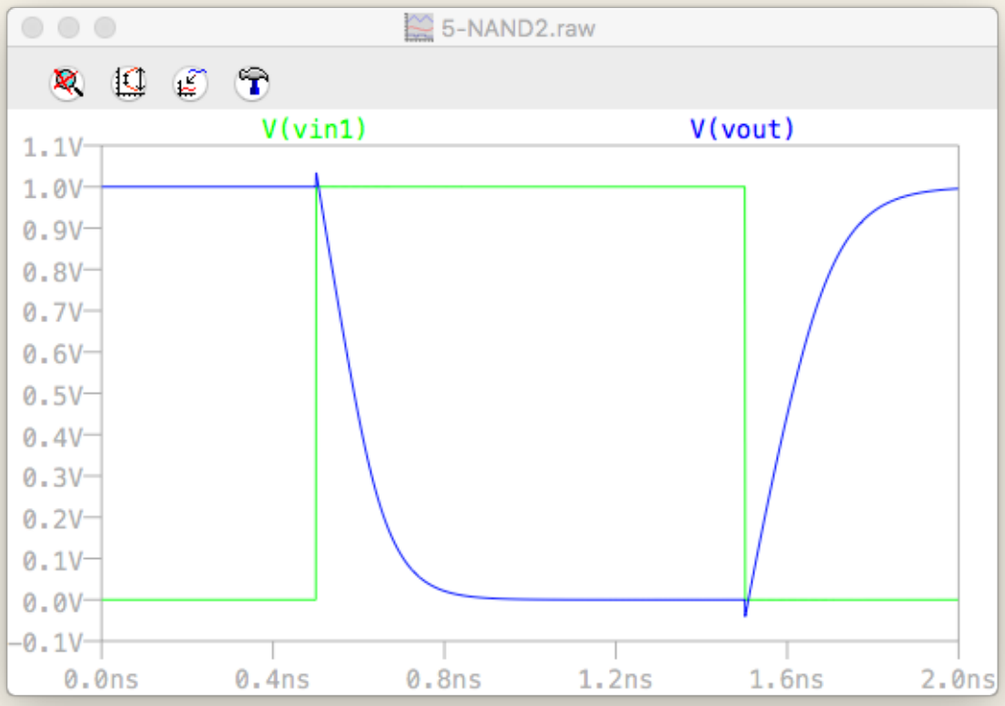
$Y = \overline{AB}$

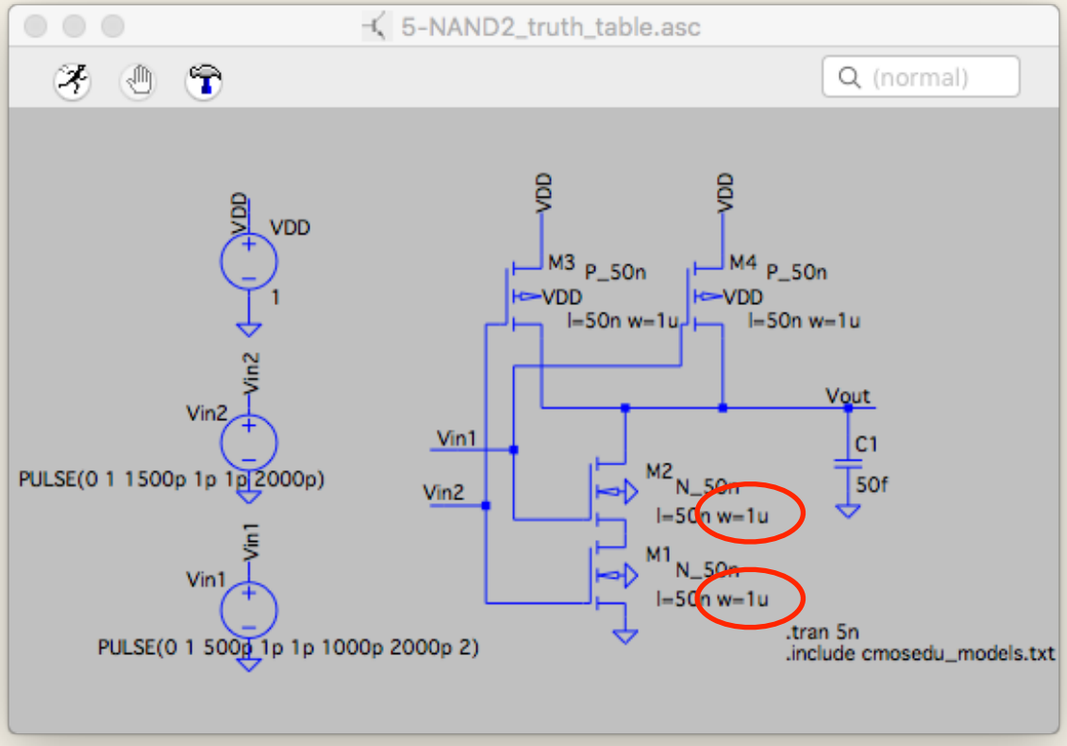
A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0

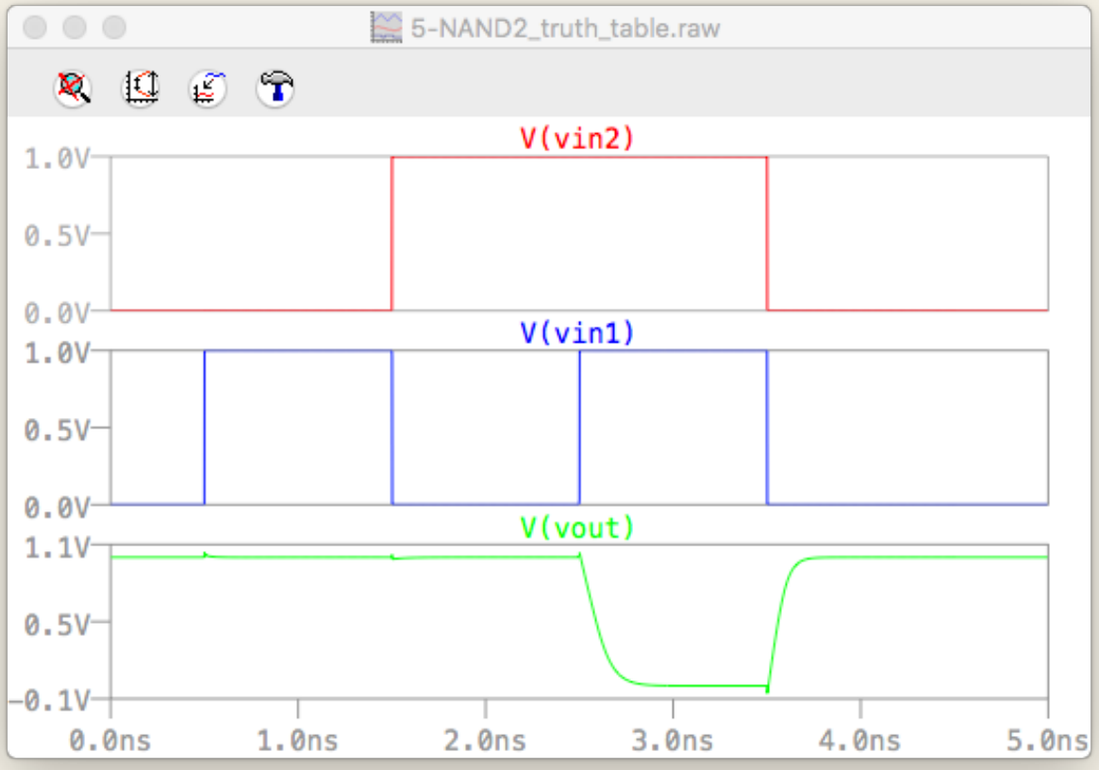


A	*B*	P1	P2	N1	N2	*Y*
0	0	ON	ON	OFF	OFF	1
0	1	ON	OFF	OFF	ON	1
1	0	OFF	ON	ON	OFF	1
1	1	OFF	OFF	ON	ON	0

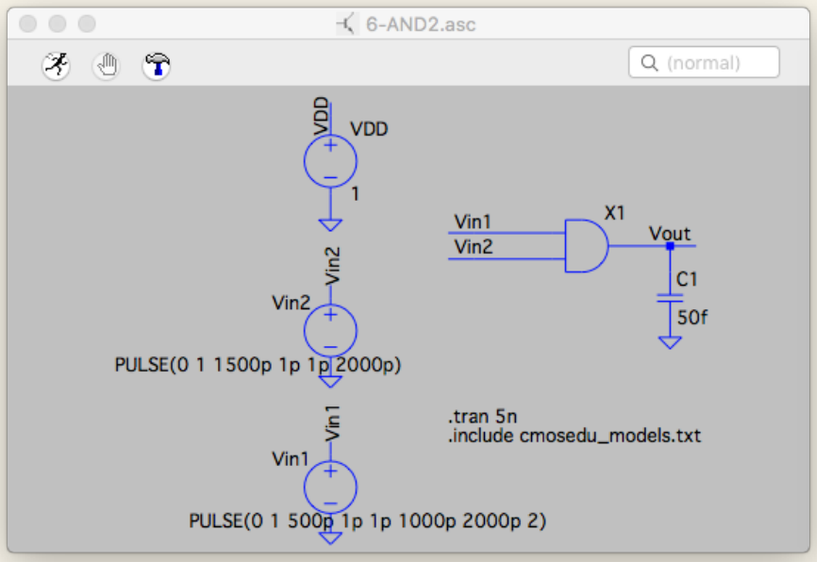


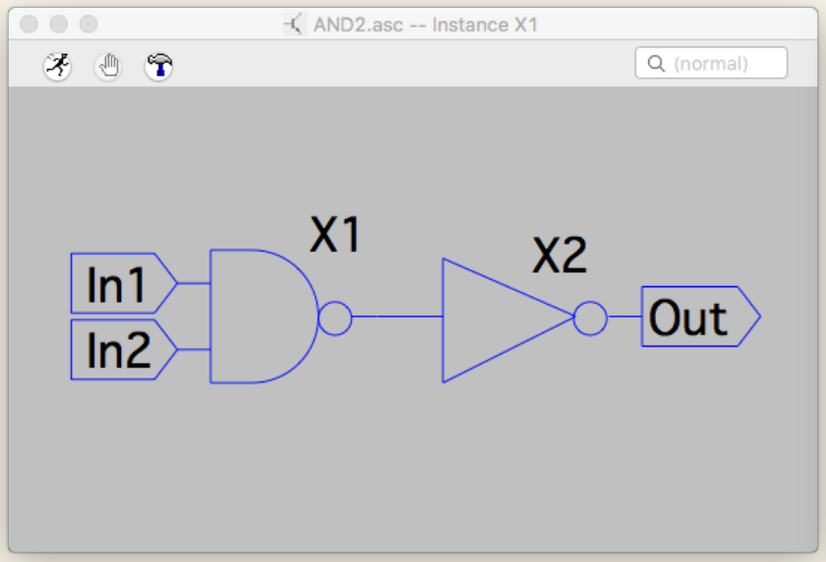


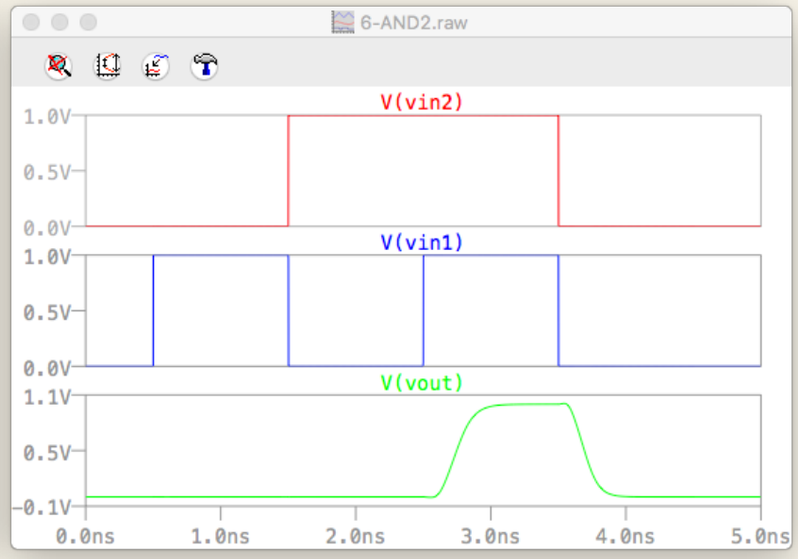

5-NAND2_truth_table.asc
(normal)
VDD
VDD
1
Vin2
Vin2
PULSE(0 1 1500p 1p 1p 2000p)
Vin1
Vin1
PULSE(0 1 500p 1p 1p 1000p 2000p 2)
M3 P_50n
VDD
l=50n w=1u
M4 P_50n
VDD
l=50n w=1u
Vout
C1
50f
M2 N_50n
l=50n w=1u
M1 N_50n
l=50n w=1u
.tran 5n
.include cmosedu_models.txt


5-NAND2_truth_table.raw
V(vin2)
1.0V
0.5V
0.0V
V(vin1)
1.0V
0.5V
0.0V
V(vout)
1.1V
0.5V
-0.1V
0.0ns
1.0ns
2.0ns
3.0ns
4.0ns
5.0ns

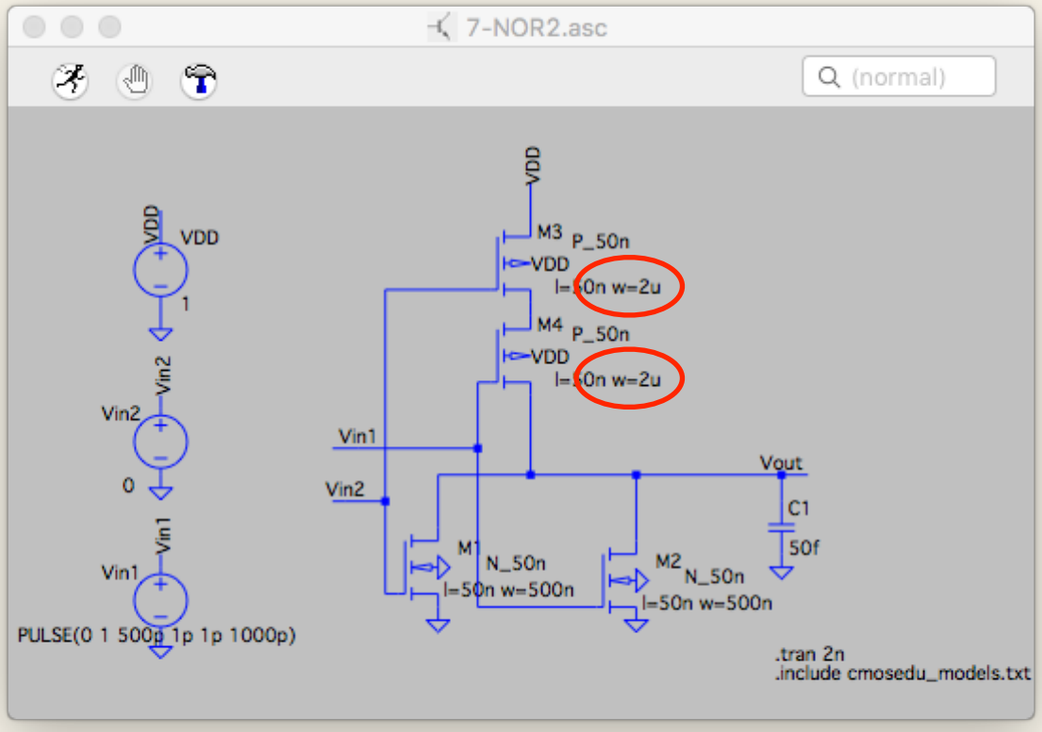
- AND

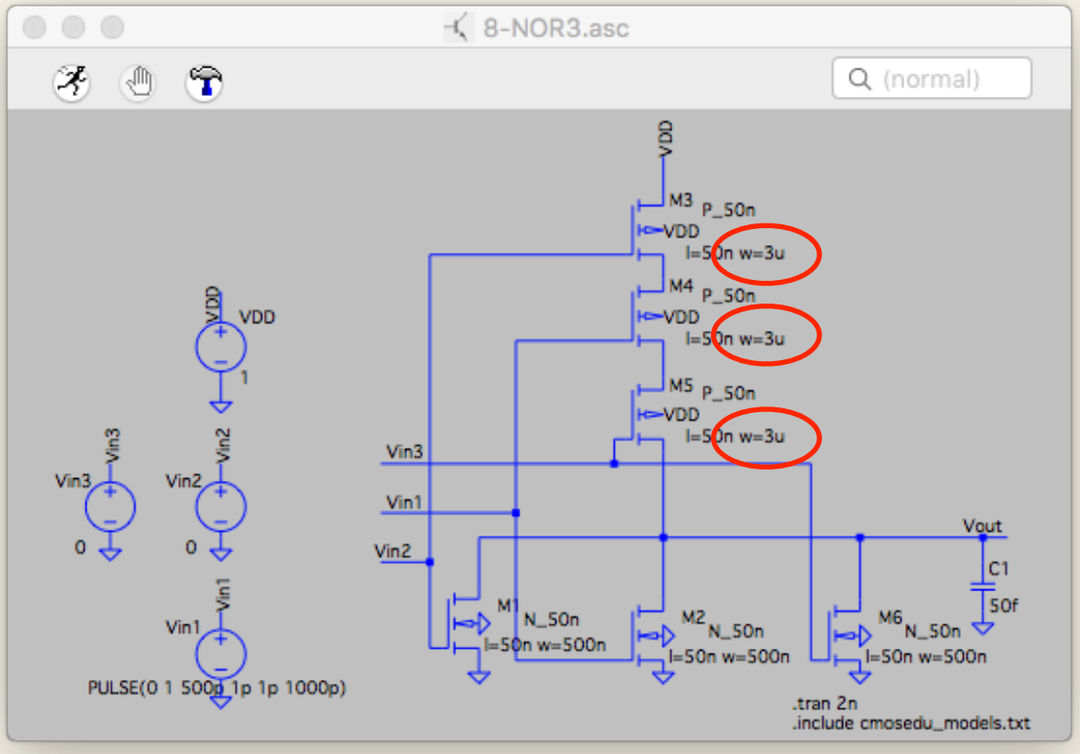

6-AND2.asc
(normal)
VDD
VDD
1
Vin2
Vin2
PULSE(0 1 1500p 1p 1p 2000p)
Vin1
Vin1
PULSE(0 1 500p 1p 1p 1000p 2000p 2)
X1
Vin1
Vin2
Vout
C1
50f
.tran 5n
.include cmosedu_models.txt

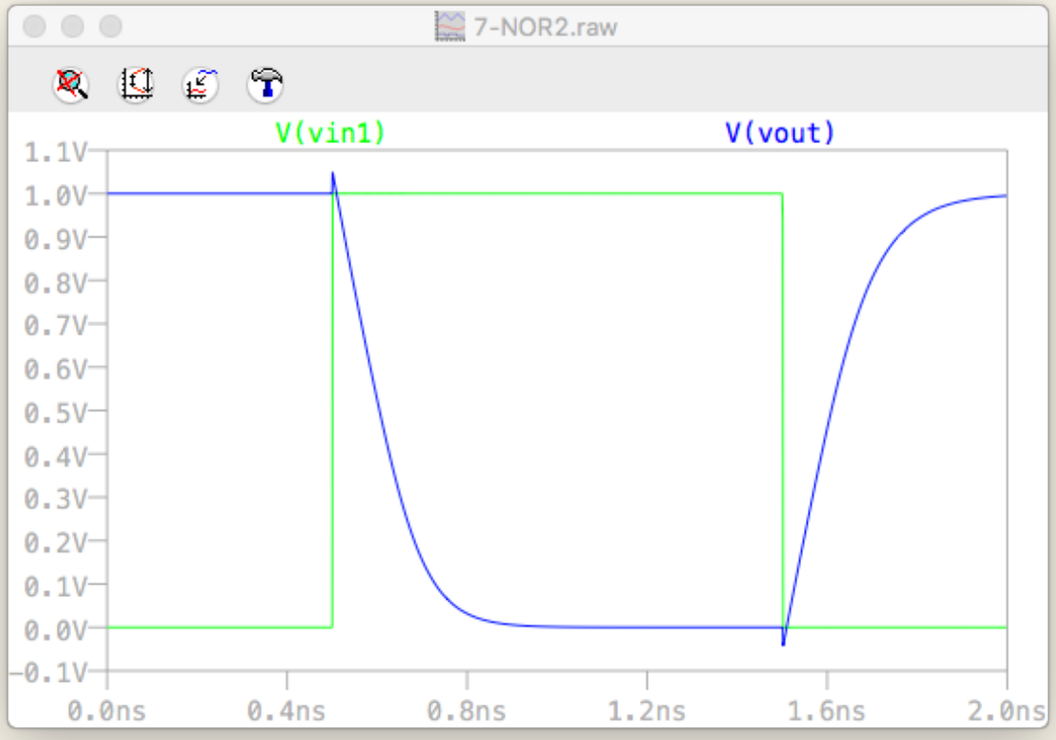

AND2.asc -- Instance X1
(normal)
In1
In2
X1
X2
Out

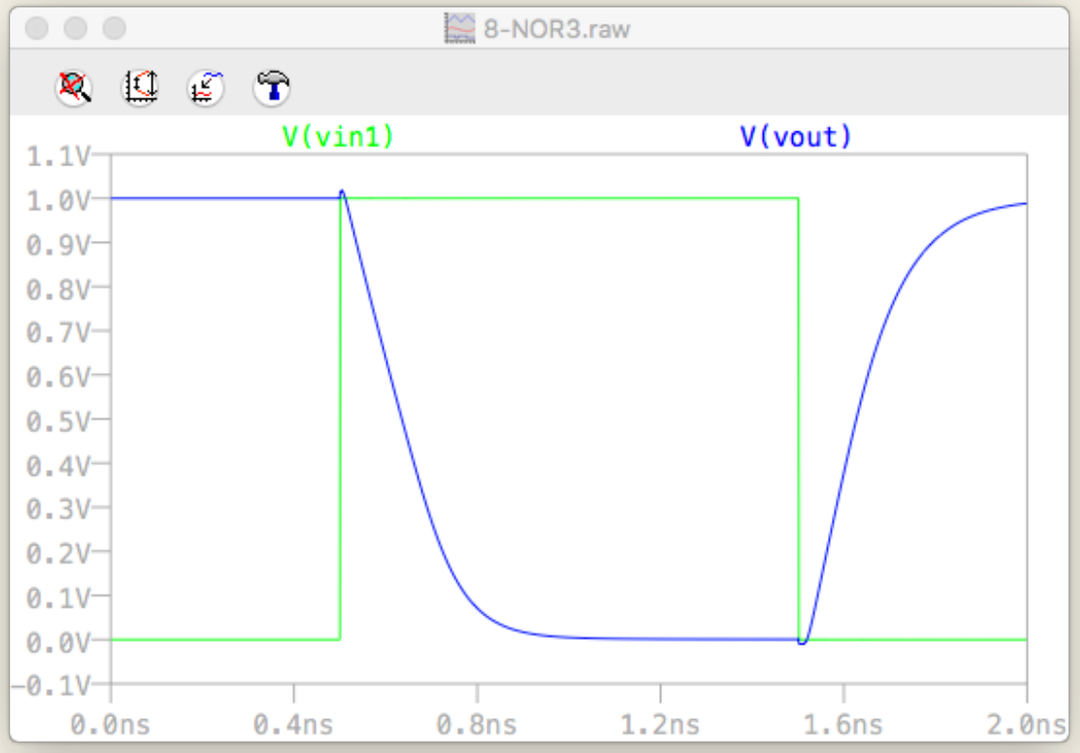

6-AND2.raw
V(vin2)
1.0V
0.5V
0.0V
V(vin1)
1.0V
0.5V
0.0V
V(vout)
1.1V
0.5V
-0.1V
0.0ns
1.0ns
2.0ns
3.0ns
4.0ns
5.0ns

- NOR

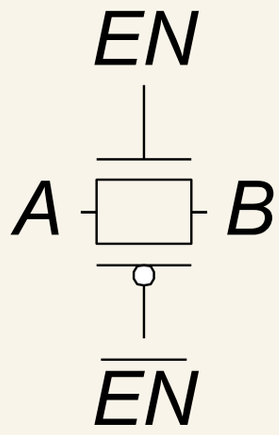

7-NOR2.asc
(normal)
VDD
VDD
1
Vin2
Vin2
0
Vin1
Vin1
PULSE(0 1 500p 1p 1p 1000p)
M3 P_50n
VDD
l=50n w=2u
M4 P_50n
VDD
l=50n w=2u
Vin1
Vin2
Vout
C1
50f
M1 N_50n
l=50n w=500n
M2 N_50n
l=50n w=500n
.tran 2n
.include cmosedu_models.txt

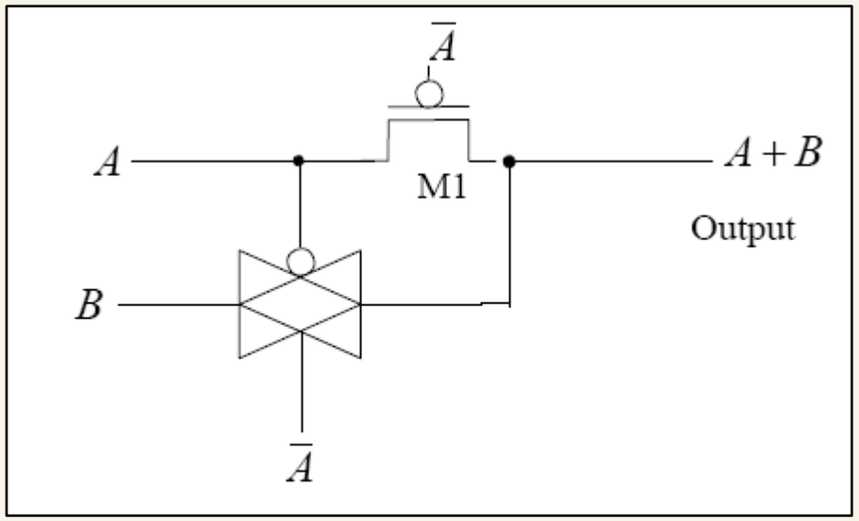

8-NOR3.asc
(normal)
VDD
VDD
1
Vin3
Vin3
0
Vin2
Vin2
0
Vin1
Vin1
PULSE(0 1 500p 1p 1p 1000p)
M3 P_50n
VDD
l=50n w=3u
M4 P_50n
VDD
l=50n w=3u
M5 P_50n
VDD
l=50n w=3u
Vin3
Vin1
Vin2
Vout
C1
50f
M1 N_50n
l=50n w=500n
M2 N_50n
l=50n w=500n
M6 N_50n
l=50n w=500n
.tran 2n
.include cmosedu_models.txt

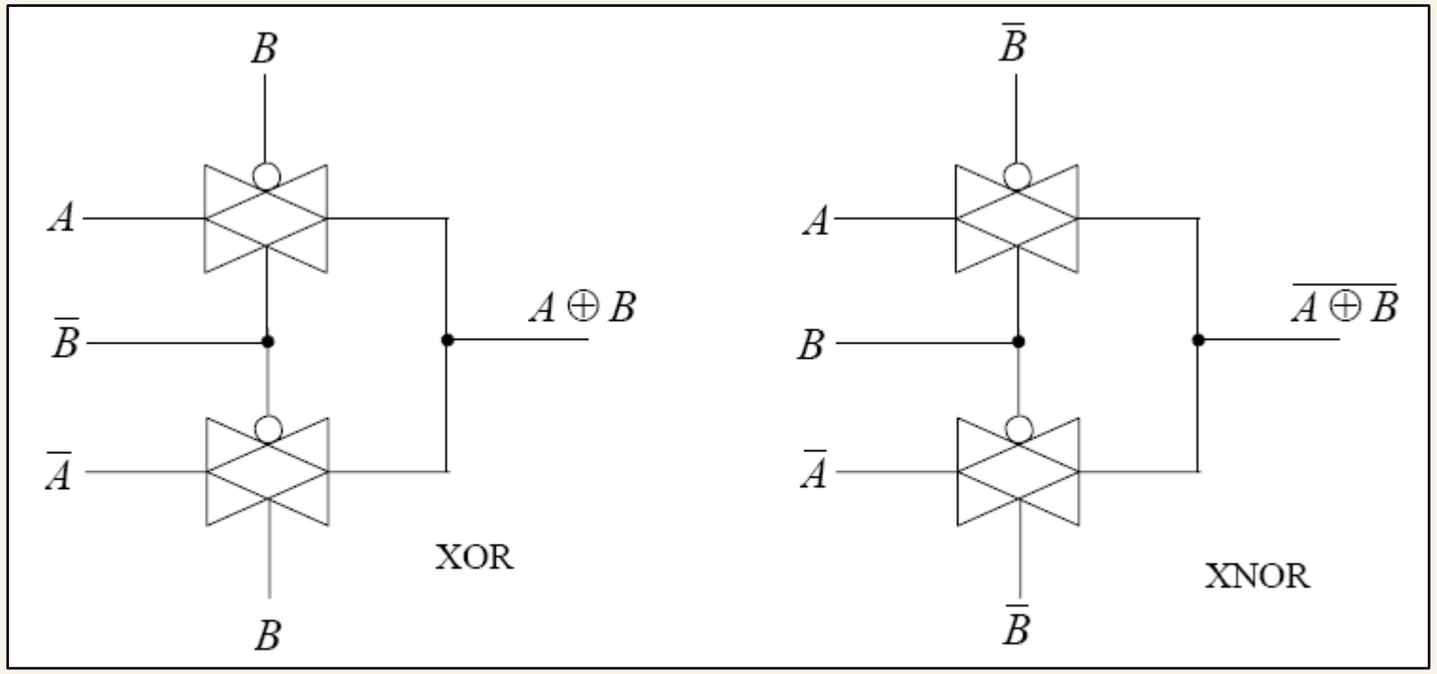

7-NOR2.raw
V(vin1)
V(vout)
1.1V
1.0V
0.9V
0.8V
0.7V
0.6V
0.5V
0.4V
0.3V
0.2V
0.1V
0.0V
-0.1V
0.0ns
0.4ns
0.8ns
1.2ns
1.6ns
2.0ns

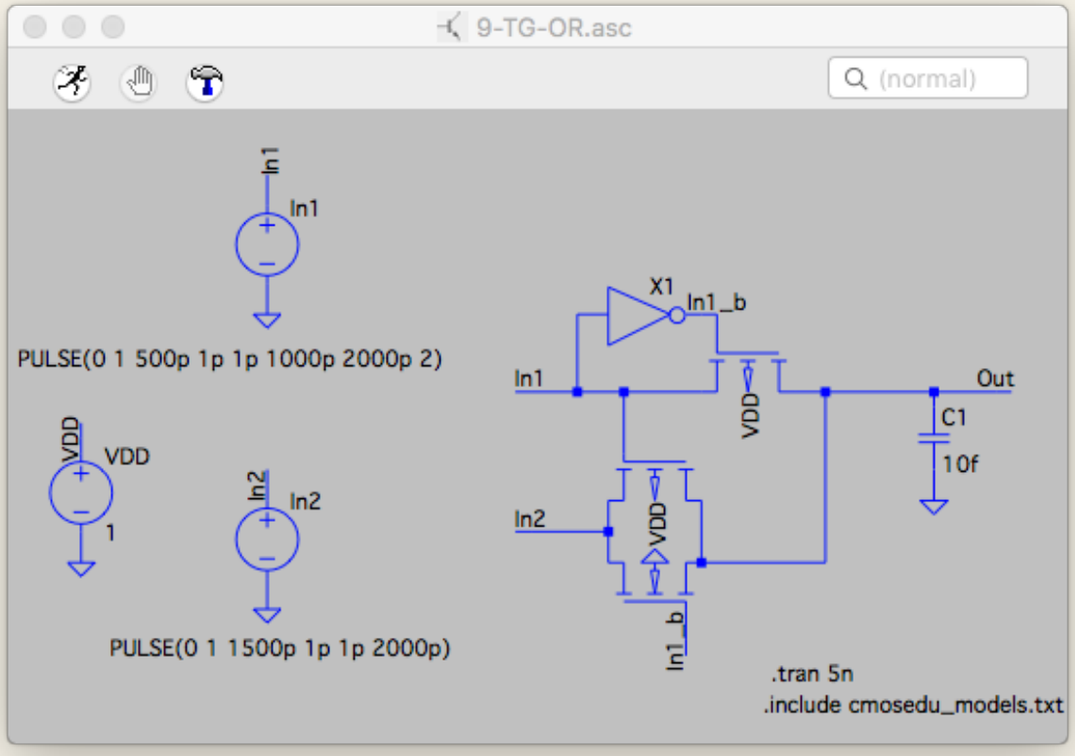

8-NOR3.raw
V(vin1)
V(vout)
1.1V
1.0V
0.9V
0.8V
0.7V
0.6V
0.5V
0.4V
0.3V
0.2V
0.1V
0.0V
-0.1V
0.0ns
0.4ns
0.8ns
1.2ns
1.6ns
2.0ns

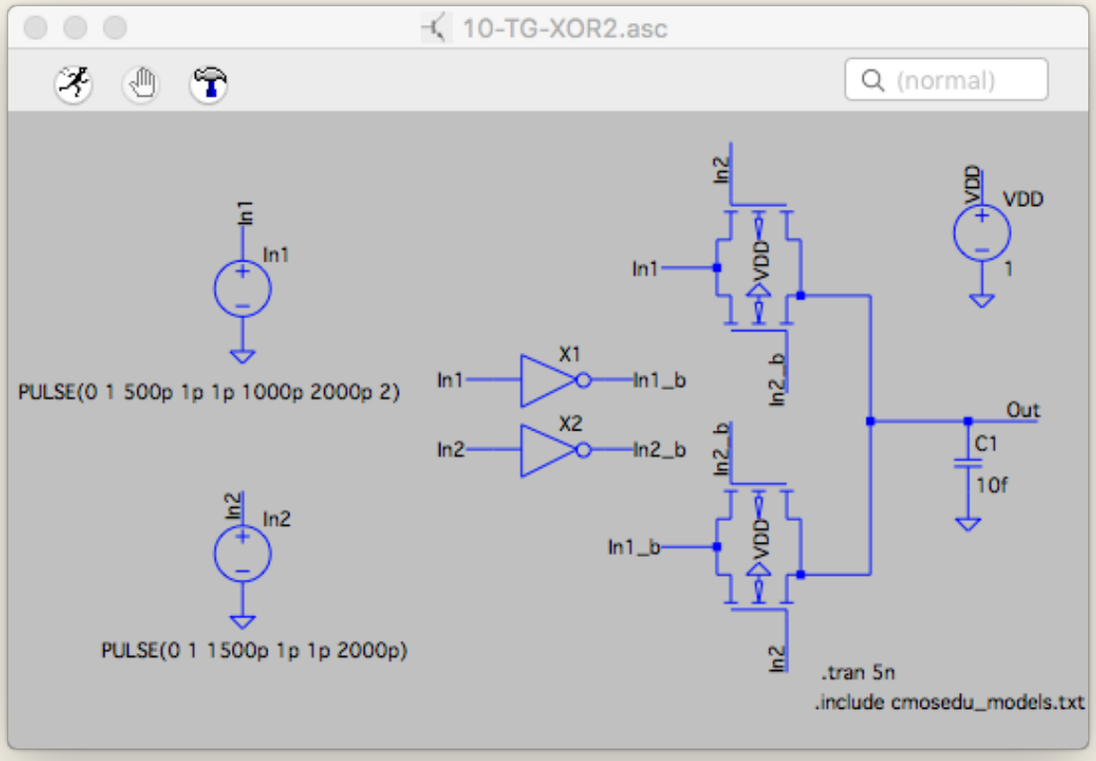
- Transmission Gates
  - nMOS pass 1's poorly
  - pMOS pass 0's poorly
  - Transmission gate is a better switch
    - passes both 0 and 1 well
  - When *EN* = 1, the switch is ON:
    - *EN* = 0 and *A* is connected to *B*
  - When *EN* = 0, the switch is OFF:
    - *A* is not connected to *B*

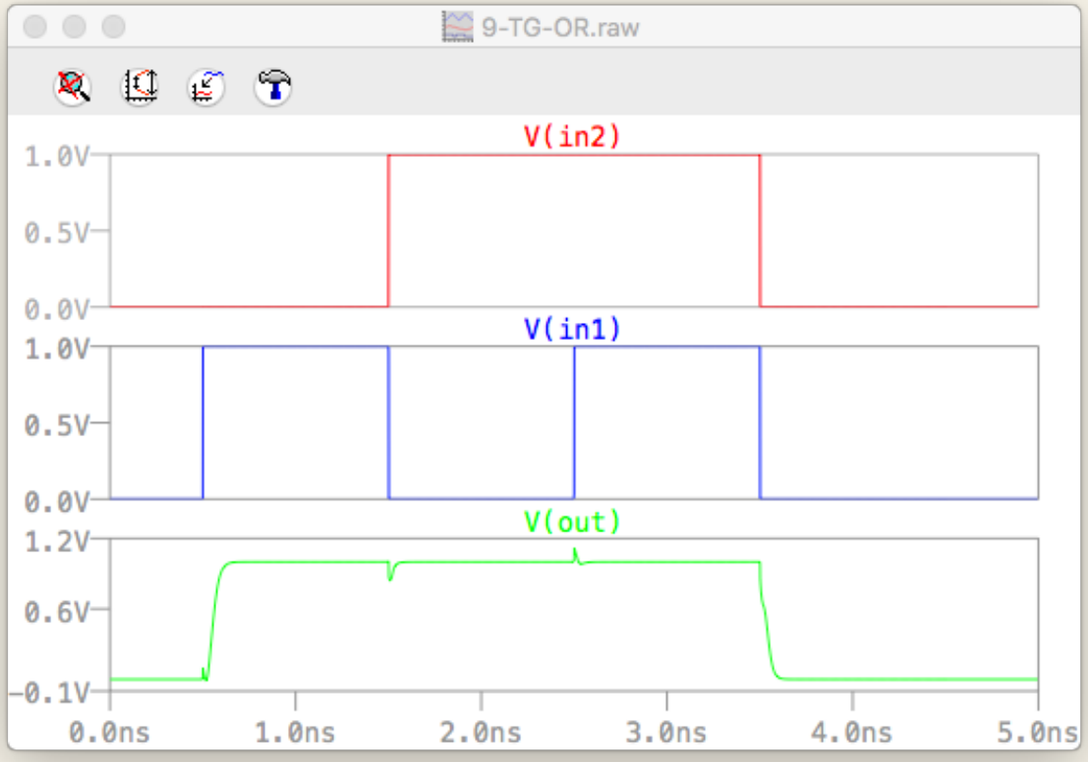


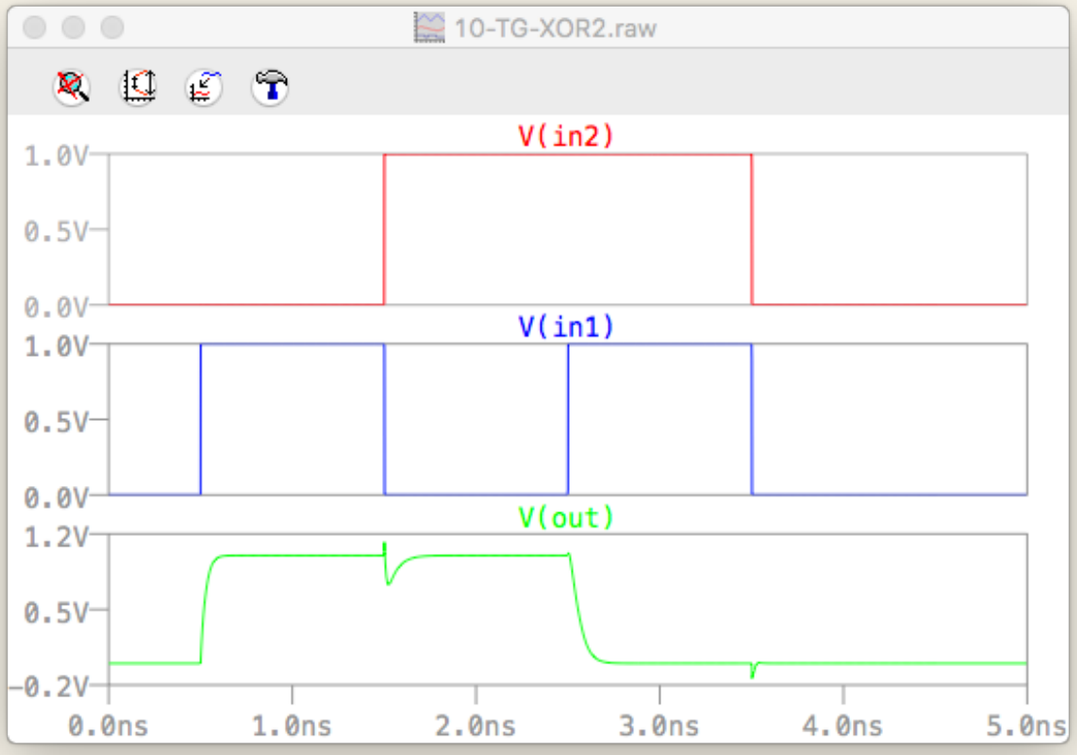





9-TG-OR.asc
(normal)
In1
In1
PULSE(0 1 500p 1p 1p 1000p 2000p 2)
VDD
VDD
1
In2
In2
PULSE(0 1 1500p 1p 1p 2000p)
X1
In1_b
In1
VDD
Out
C1
10f
In2
VDD
In1_b
.tran 5n
.include cmosedu_models.txt

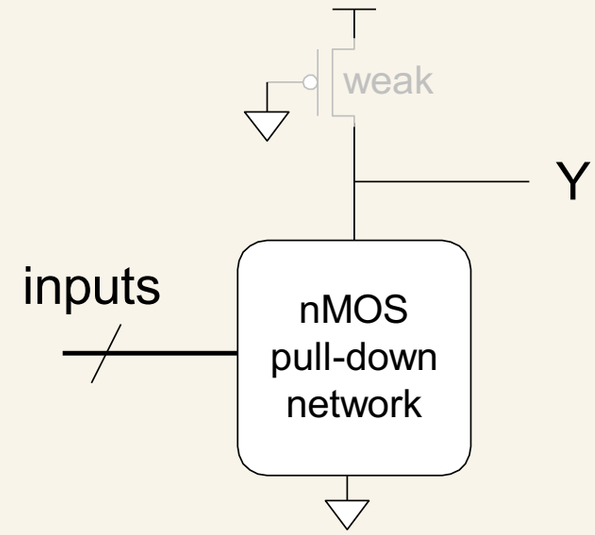

10-TG-XOR2.asc
(normal)
In1
In1
PULSE(0 1 500p 1p 1p 1000p 2000p 2)
In2
In2
PULSE(0 1 1500p 1p 1p 2000p)
In2
VDD
VDD
1
In1
VDD
In2_b
X1
In1
In1_b
X2
In2
In2_b
In2_b
Out
C1
10f
In1_b
VDD
In2
.tran 5n
.include cmosedu_models.txt

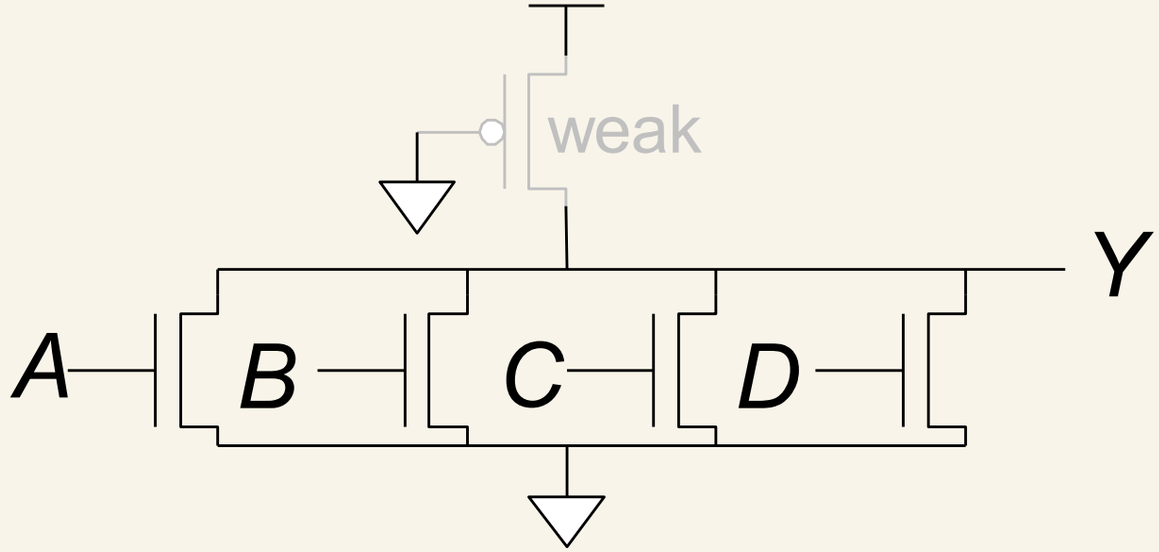

9-TG-OR.raw
V(in2)
1.0V
0.5V
0.0V
V(in1)
1.0V
0.5V
0.0V
V(out)
1.2V
0.6V
-0.1V
0.0ns
1.0ns
2.0ns
3.0ns
4.0ns
5.0ns


10-TG-XOR2.raw
V(in2)
1.0V
0.5V
0.0V
V(in1)
1.0V
0.5V
0.0V
V(out)
1.2V
0.5V
-0.2V
0.0ns
1.0ns
2.0ns
3.0ns
4.0ns
5.0ns

- Pseudo-nMOS Gates

  - Replace pull-up network with *weak* pMOS transistor that is always on
  - pMOS transistor: pulls output HIGH *only* when nMOS network not pulling it LOW





- Power Consumption

Power = Energy consumed per unit time

- Dynamic power consumption
- Static power consumption

- **Power to charge transistor gate capacitances**
  - Energy required to charge a capacitance, C, to V_{DD} is CV_{DD}^2
  - Circuit running at frequency f: transistors switch (from 1 to 0 or vice versa) at that frequency
  - Capacitor is charged $f/2$ times per second (discharging from 1 to 0 is free)
- **Dynamic power consumption:**

$$P_{dynamic} = \tfrac{1}{2}CV_{DD}^2 f$$

- Power consumed when no gates are switching
- Caused by the *quiescent supply current*, I_{DD} (also called the *leakage current*)
- Static power consumption:

$$P_{static} = I_{DD}V_{DD}$$

- Estimate the power consumption of a wireless handheld computer
  - V_{DD} = 1.2 V
  - C = 20 nF
  - f = 1 GHz
  - I_{DD} = 20 mA

$$P = \tfrac{1}{2}CV_{DD}^2 f + I_{DD}V_{DD}$$

$$= \tfrac{1}{2}(20\text{ nF})(1.2\text{ V})^2(1\text{ GHz}) + (20\text{ mA})(1.2\text{ V})$$

$$= (14.4 + 0.024)\text{ W} \approx 14.4\text{ W}$$