# Digital Logic & Computer Design
# CS 4341

Professor Dan Moldovan

Spring 2010

Copyright © 2007 Elsevier

# Chapter 1 :: From Zero to One

*Digital Design and Computer Architecture*

David Money Harris and Sarah L. Harris

Copyright © 2007 Elsevier

# Chapter 1 :: Topics

- Background
- The Game Plan
- The Art of Managing Complexity
- The Digital Abstraction
- Number Systems
- Logic Gates
- Logic Levels
- CMOS Transistors
- Power Consumption

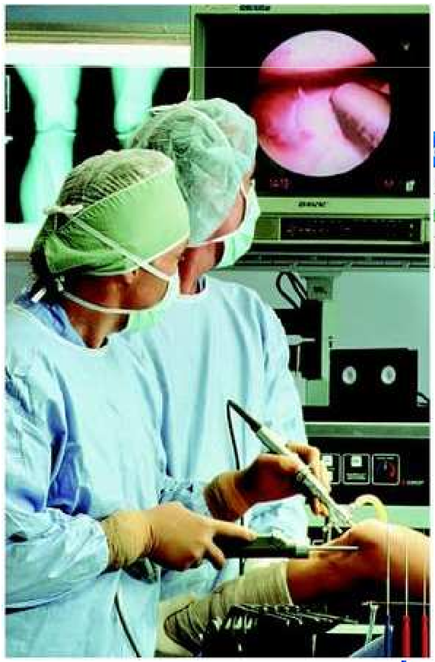
Copyright © 2007 Elsevier

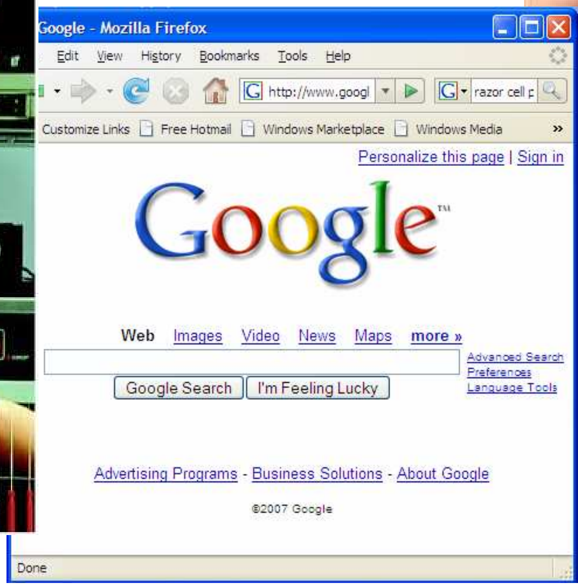


ELSEVIER

# Background

- Microprocessors have revolutionized our world
  - Cell phones, Internet, rapid advances in medicine, etc.
- The semiconductor industry has grown from $21 billion in 1985 to $213 billion in 2004






Copyright © 2007 Elsevier

# The Game Plan

- The purpose of this course is that you:
  - Learn what's under the hood of a computer
  - Learn the principles of digital design
  - Learn to systematically debug increasingly complex designs
  - Design and build a microprocessor

Copyright © 2007 Elsevier



ELSEVIER

# The Art of Managing Complexity

- Abstraction
- Discipline
- The Three –Y's
  - Hierarchy
  - Modularity
  - Regularity

Copyright © 2007 Elsevier

# Abstraction

- Hiding details when they aren't important

| | |
|---|---|
| Application Software | programs |
| Operating Systems | device drivers |
| Architecture | instructions registers |
| Micro-architecture | datapaths controllers |
| Logic | adders memories |
| Digital Circuits | AND gates NOT gates |
| Analog Circuits | amplifiers filters |
| Devices | transistors diodes |
| Physics | electrons |

focus of this course

Copyright © 2007 Elsevier



ELSEVIER

# Discipline

- Intentionally restricting your design choices
  - to work more productively at a higher level of abstraction
- Example: Digital discipline
  - Considering discrete voltages instead of continuous voltages used by analog circuits
  - Digital circuits are simpler to design than analog circuits – can build more sophisticated systems
  - Digital systems replacing analog predecessors:
    - I.e., digital cameras, digital television, cell phones, CDs

Copyright © 2007 Elsevier



ELSEVIER

# 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

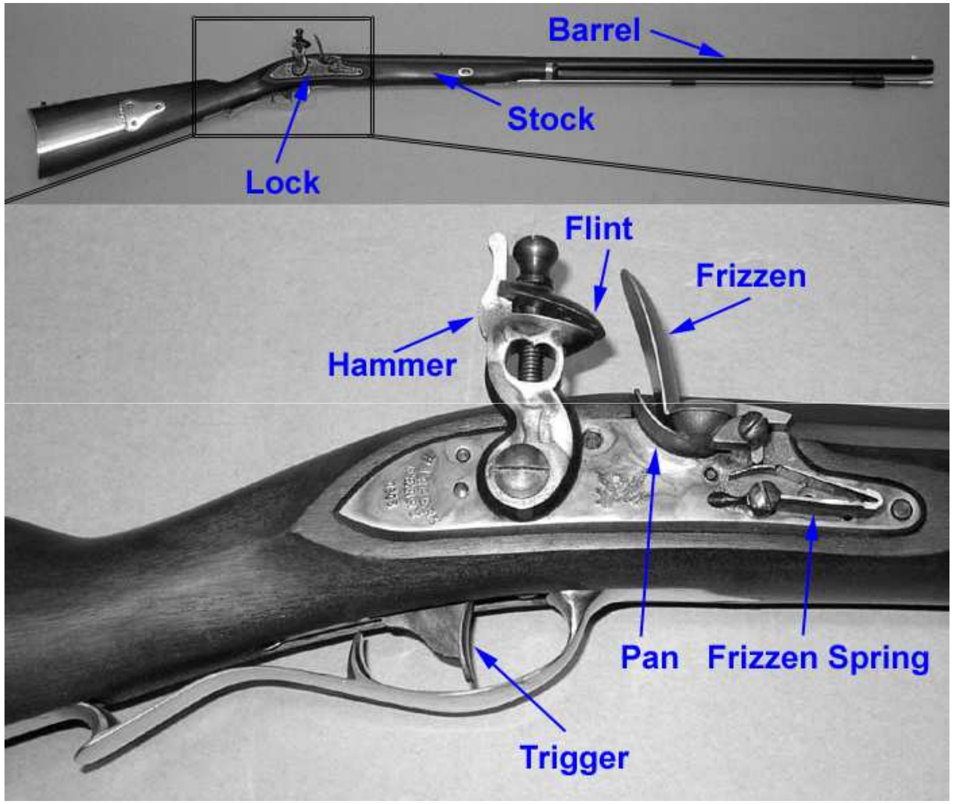
Copyright © 2007 Elsevier



ELSEVIER

# Example: Flintlock Rifle

- Hierarchy
  - Three main modules: lock, stock, and barrel
  - Submodules of lock: hammer, flint, frizzen, etc.

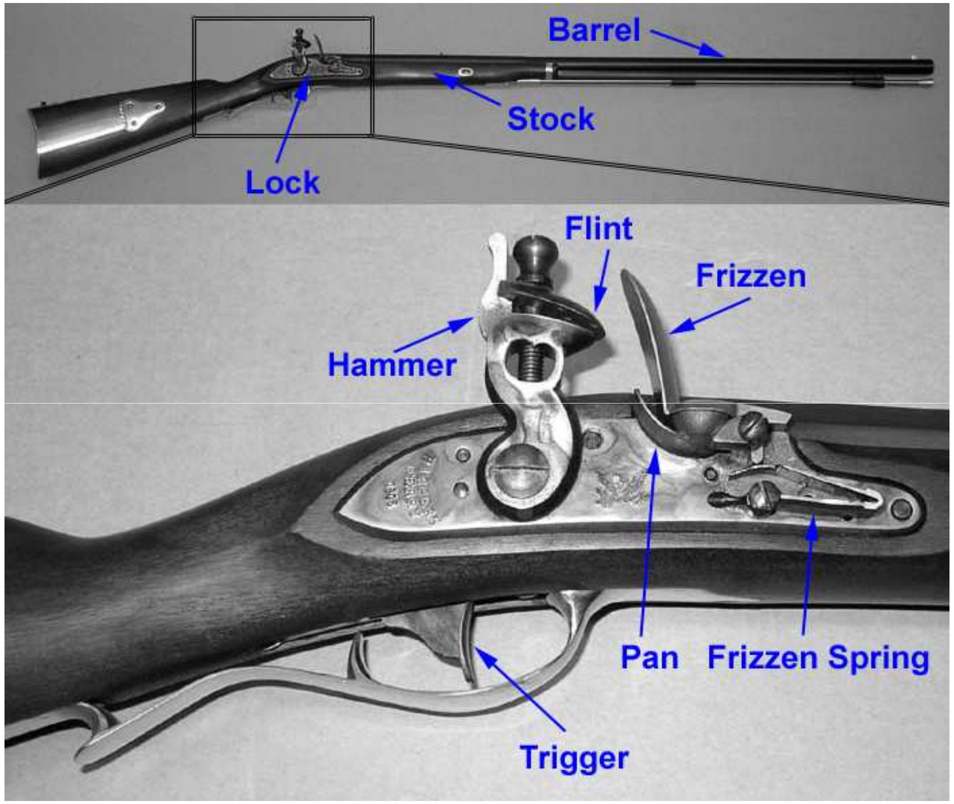



Copyright © 2007 Elsevier

ELSEVIER

# Example: Flintlock Rifle

- Modularity
  - Function of stock: mount barrel and lock
  - Interface of stock: length and location of mounting pins
- Regularity
  - Interchangeable parts




Copyright © 2007 Elsevier



ELSEVIER

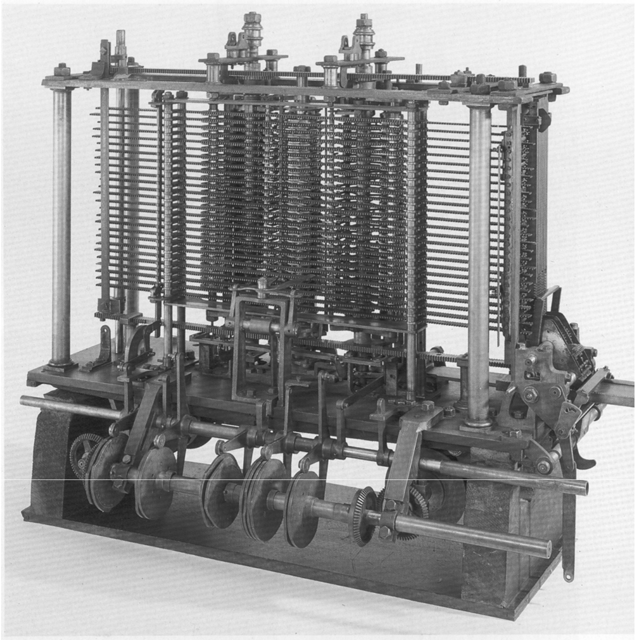
# The Digital Abstraction

- Most physical variables are continuous, for example
  - Voltage on a wire
  - Frequency of an oscillation
  - Position of a mass
- Instead of considering all values, the digital abstraction considers only a discrete subset of values

Copyright © 2007 Elsevier

# The Analytical Engine

- Designed by Charles Babbage from 1834 – 1871
- Considered to be the first digital computer
- Built from mechanical gears, where each gear represented a discrete value (0-9)
- Babbage died before it was finished



Copyright © 2007 Elsevier



ELSEVIER

# Digital Discipline: Binary Values

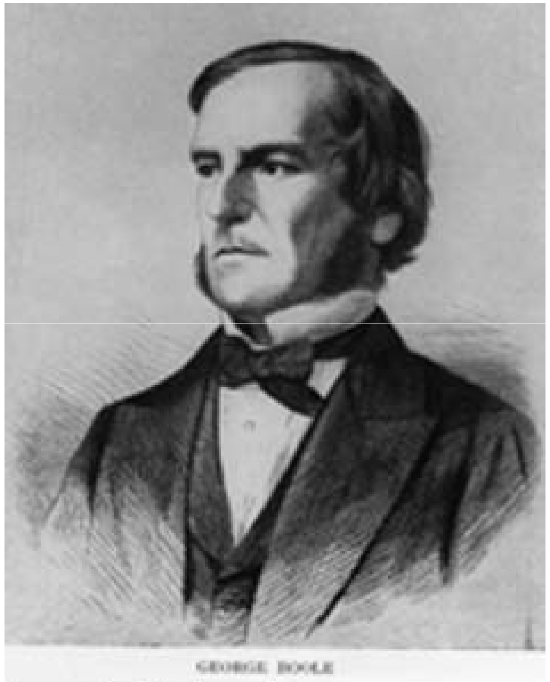
- Typically consider only two discrete values:
  - 1's and 0's
  - 1, TRUE, HIGH
  - 0, FALSE, LOW
- 1 and 0 can be represented by specific voltage levels, rotating gears, fluid levels, etc.
- Digital circuits usually depend on specific voltage levels to represent 1 and 0
- *Bit*: *B*inary dig*it*

Copyright © 2007 Elsevier

ELSEVIER

# George Boole, 1815 - 1864

- Born to working class parents
- Taught himself mathematics and joined the faculty of Queen's College in Ireland.
- Wrote *An Investigation of the Laws of Thought* (1854)
- Introduced binary variables
- Introduced the three fundamental logic operations: AND, OR, and NOT.



Scanned at the American Institute of Physics

Copyright © 2007 Elsevier

ELSEVIER

# Number Systems

- Decimal numbers

1000's column
100's column
10's column
1's column

$5374_{10} =$

- Binary numbers

8's column
4's column
2's column
1's column

$1101_2 =$

Copyright © 2007 Elsevier



ELSEVIER

# Number Systems

- Decimal numbers

1000's column
100's column
10's column
1's column

$$5374_{10} = \underbrace{5 \times 10^3}_{\text{five thousands}} + \underbrace{3 \times 10^2}_{\text{three hundreds}} + \underbrace{7 \times 10^1}_{\text{seven tens}} + \underbrace{4 \times 10^0}_{\text{four ones}}$$

- Binary numbers

8's column
4's column
2's column
1's column

$$1101_2 = \underbrace{1 \times 2^3}_{\text{one eight}} + \underbrace{1 \times 2^2}_{\text{one four}} + \underbrace{0 \times 2^1}_{\text{no two}} + \underbrace{1 \times 2^0}_{\text{one one}} = 13_{10}$$

Copyright © 2007 Elsevier

# Powers of Two

- $2^0 =$
- $2^1 =$
- $2^2 =$
- $2^3 =$
- $2^4 =$
- $2^5 =$
- $2^6 =$
- $2^7 =$
- $2^8 =$
- $2^9 =$
- $2^{10} =$
- $2^{11} =$
- $2^{12} =$
- $2^{13} =$
- $2^{14} =$
- $2^{15} =$

Copyright © 2007 Elsevier



ELSEVIER

# Powers of Two

- $2^0 = 1$
- $2^1 = 2$
- $2^2 = 4$
- $2^3 = 8$
- $2^4 = 16$
- $2^5 = 32$
- $2^6 = 64$
- $2^7 = 128$
- $2^8 = 256$
- $2^9 = 512$
- $2^{10} = 1024$
- $2^{11} = 2048$
- $2^{12} = 4096$
- $2^{13} = 8192$
- $2^{14} = 16384$
- $2^{15} = 32768$
- Handy to memorize up to $2^9$

Copyright © 2007 Elsevier

# Number Conversion

- Decimal to binary conversion:
  - Convert $10101_2$ to decimal

- Decimal to binary conversion:
  - Convert $47_{10}$ to binary

Copyright © 2007 Elsevier

# Number Conversion

- Decimal to binary conversion:
  - Convert $10011_2$ to decimal
  - $16\times1 + 8\times0 + 4\times0 + 2\times1 + 1\times1 = 19_{10}$

- Decimal to binary conversion:
  - Convert $47_{10}$ to binary
  - $32\times1 + 16\times0 + 8\times1 + 4\times1 + 2\times1 + 1\times1 = 101111_2$

Copyright © 2007 Elsevier

# 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 $111_2]$

Copyright © 2007 Elsevier



ELSEVIER

# Hexadecimal Numbers

| Hex Digit | Decimal Equivalent | Binary Equivalent |
|---|---|---|
| 0 | 0 | |
| 1 | 1 | |
| 2 | 2 | |
| 3 | 3 | |
| 4 | 4 | |
| 5 | 5 | |
| 6 | 6 | |
| 7 | 7 | |
| 8 | 8 | |
| 9 | 9 | |
| A | 10 | |
| B | 11 | |
| C | 12 | |
| D | 13 | |
| E | 14 | |
| F | 15 | |

Copyright © 2007 Elsevier



ELSEVIER

# Hexadecimal Numbers

| Hex Digit | Decimal Equivalent | Binary Equivalent |
|---|---|---|
| 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 |

Copyright © 2007 Elsevier



ELSEVIER

# Hexadecimal Numbers

- Base 16
- Shorthand to write long binary numbers

Copyright © 2007 Elsevier

# Hexadecimal to Binary Conversion

- Hexadecimal to binary conversion:
  - Convert $4AF_{16}$ (also written 0x4AF) to binary

- Hexadecimal to decimal conversion:
  - Convert 0x4AF to decimal

Copyright © 2007 Elsevier

ELSEVIER

# 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
  - $16^2 \times 4 + 16^1 \times 10 + 16^0 \times 15 = 1199_{10}$

Copyright © 2007 Elsevier

# Bits, Bytes, Nibbles…

- Bits

$$\underbrace{1}_{\text{most significant bit}}001011\underbrace{0}_{\text{least significant bit}}$$

- Bytes & Nibbles

$$\overbrace{1001\underbrace{0110}_{\text{nibble}}}^{\text{byte}}$$

- Bytes

$$\underbrace{\mathrm{CE}}_{\text{most significant byte}}\mathrm{BF9A}\underbrace{\mathrm{D7}}_{\text{least significant byte}}$$

Copyright © 2007 Elsevier

# 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)

Copyright © 2007 Elsevier

# Estimating Powers of Two

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

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

Copyright © 2007 Elsevier

ELSEVIER

# Estimating Powers of Two

- 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**

Copyright © 2007 Elsevier

ELSEVIER

# Addition

- Decimal

$$\begin{array}{r} \color{blue}{11\ \ } \\ 3734 \\ +\ \ 5168 \\ \hline 8902 \end{array}$$

← carries

- Binary

$$\begin{array}{r} \color{blue}{11\ \ } \\ 1011 \\ +\ \ 0011 \\ \hline 1110 \end{array}$$

← carries

Copyright © 2007 Elsevier

# Binary Addition Examples

- Add the following 4-bit binary numbers

$$
\begin{array}{r}
1001 \\
+\ \ 0101 \\
\hline
\end{array}
$$

- Add the following 4-bit binary numbers

$$
\begin{array}{r}
1011 \\
+\ \ 0110 \\
\hline
\end{array}
$$

Copyright © 2007 Elsevier

# Binary Addition Examples

- Add the following 4-bit binary numbers

```
     1
  1001
+ 0101
------
  1110
```

- Add the following 4-bit binary numbers

```
  111
   1011
+  0110
-------
  10001
```

Overflow!

Copyright © 2007 Elsevier



ELSEVIER

# Overflow

- Digital systems operate on a fixed number of bits
- Addition overflows when the result is too big to fit in the available number of bits
- See previous example of 11 + 6

Copyright © 2007 Elsevier

# Signed Binary Numbers

- Sign/Magnitude Numbers
- Two's Complement Numbers

Copyright © 2007 Elsevier

ELSEVIER

# Sign/Magnitude Numbers

- 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 =

  - 6 =

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

Copyright © 2007 Elsevier

# Sign/Magnitude Numbers

- 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:

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

Copyright © 2007 Elsevier

# Sign/Magnitude Numbers

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

$$
\begin{array}{r}
1110 \\
+\ 0110 \\
\hline
10100
\end{array}
\text{ (wrong!)}
$$

  - Two representations of 0 (± 0):

    1000

    0000

Copyright © 2007 Elsevier

# Two's Complement Numbers

- Don't have same problems as sign/magnitude numbers:
  - Addition works
  - Single representation for 0

Copyright © 2007 Elsevier



ELSEVIER

# Two's Complement Numbers

- Same as unsigned binary, but the most significant bit (msb) has value of $-2^{N-1}$

$$A = a_{n-1}\left(-2^{n-1}\right) + \sum_{i=0}^{n-2} a_i 2^i$$

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

# Two's Complement Numbers

- Same as unsigned binary, but the most significant bit (msb) has value of $-2^{N-1}$

$$A = a_{n-1}\left(-2^{n-1}\right) + \sum_{i=0}^{n-2} a_i 2^i$$

- 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 comp number:

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

Copyright © 2007 Elsevier

# “Taking the Two’s Complement”

- Flip the sign of a two’s complement number
- Method:
  1. Invert the bits
  2. Add 1
- Example: Flip the sign of $3_{10} = 0011_2$

Copyright © 2007 Elsevier

# "Taking the Two's Complement"

- Flip 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**

  **$1101 = -3_{10}$**

Copyright © 2007 Elsevier



ELSEVIER

# Two's Complement Examples

- Take the two's complement of $6_{10} = 0110_2$

- What is the decimal value of $1001_2$?

Copyright © 2007 Elsevier

# Two's Complement Examples

- Take the two's complement of $6_{10} = 0110_2$

  1. $1001$
  2. $+ \quad 1$

  $1010_2 = -6_{10}$

- What is the decimal value of the two's complement number $1001_2$?

  1. $0110$
  2. $+ \quad 1$

  $0111_2 = 7_{10}$, so $1001_2 = -7_{10}$

Copyright © 2007 Elsevier

# Two's Complement Addition

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

$$\begin{array}{r} 0110 \\ +\ 1010 \\ \hline \end{array}$$

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

$$\begin{array}{r} 1110 \\ +\ 0011 \\ \hline \end{array}$$

Copyright © 2007 Elsevier



ELSEVIER

# 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
```

Copyright © 2007 Elsevier

# Increasing Bit Width

- A value can be extended from $N$ bits to $M$ bits (where $M > N$) by using:
  - Sign-extension
  - Zero-extension

# Sign-Extension

- Sign bit is copied into most significant bits.
- Number value remains the same.

- **Example 1:**
  - 4-bit representation of 3 = 0011
  - 8-bit sign-extended value: 00000011
- **Example 2:**
  - 4-bit representation of -5 = 1011
  - 8-bit sign-extended value: 11111011

Copyright © 2007 Elsevier

# Zero-Extension

- Zeros are copied into most significant bits.
- Value will change for negative numbers.

- **Example 1:**
  - 4-bit value = $0011_2 = 3_{10}$
  - 8-bit zero-extended value: $00000011 = 3_{10}$
- **Example 2:**
  - 4-bit value = $1011 = -5_{10}$
  - 8-bit zero-extended value: $00001011 = 11_{10}$

Copyright © 2007 Elsevier



ELSEVIER

# Number System Comparison

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

For example, 4-bit representation:

| | -8 | -7 | -6 | -5 | -4 | -3 | -2 | -1 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Unsigned | | | | | | | | | 0000 | 0001 | 0010 | 0011 | 0100 | 0101 | 0110 | 0111 | 1000 | 1001 | 1010 | 1011 | 1100 | 1101 | 1110 | 1111 |
| Two's Complement | 1000 | 1001 | 1010 | 1011 | 1100 | 1101 | 1110 | 1111 | 0000 | 0001 | 0010 | 0011 | 0100 | 0101 | 0110 | 0111 | | | | | | | | |
| Sign/Magnitude | | 1111 | 1110 | 1101 | 1100 | 1011 | 1010 | 1001 | 0000<br>1000 | 0001 | 0010 | 0011 | 0100 | 0101 | 0110 | 0111 | | | | | | | | |

Copyright © 2007 Elsevier

# Logic Gates

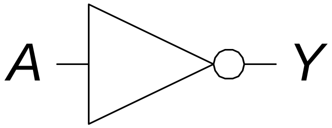
- Perform logic functions:
  - inversion (NOT), AND, OR, NAND, NOR, etc.
- Single-input:
  - NOT gate, buffer
- Two-input:
  - AND, OR, XOR, NAND, NOR, XNOR
- Multiple-input

Copyright © 2007 Elsevier

ELSEVIER

# Single-Input Logic Gates

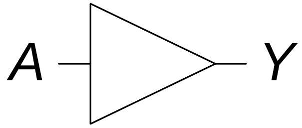
## NOT




$$Y = \overline{A}$$

| $A$ | $Y$ |
|---|---|
| 0 | |
| 1 | |

## BUF



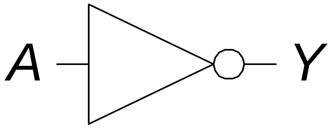

$$Y = A$$

| $A$ | $Y$ |
|---|---|
| 0 | |
| 1 | |

Copyright © 2007 Elsevier

# Single-Input Logic Gates

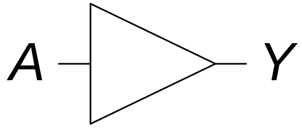
**NOT**




$$Y = \overline{A}$$

| $A$ | $Y$ |
|---|---|
| 0 | 1 |
| 1 | 0 |

**BUF**



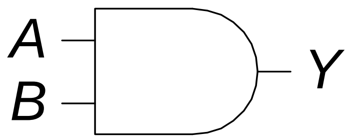

$$Y = A$$

| $A$ | $Y$ |
|---|---|
| 0 | 0 |
| 1 | 1 |

Copyright © 2007 Elsevier

# Two-Input Logic Gates

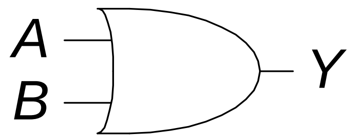
## AND




$$Y = AB$$

| A | B | Y |
|---|---|---|
| 0 | 0 | |
| 0 | 1 | |
| 1 | 0 | |
| 1 | 1 | |

## OR



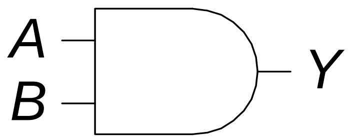

$$Y = A + B$$

| A | B | Y |
|---|---|---|
| 0 | 0 | |
| 0 | 1 | |
| 1 | 0 | |
| 1 | 1 | |

Copyright © 2007 Elsevier

# Two-Input Logic Gates

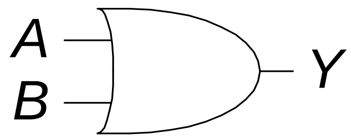
## AND




$$Y = AB$$

| $A$ | $B$ | $Y$ |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

## OR



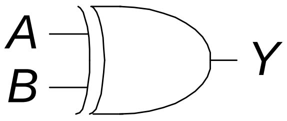

$$Y = A + B$$

| $A$ | $B$ | $Y$ |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

Copyright © 2007 Elsevier

# More Two-Input Logic Gates

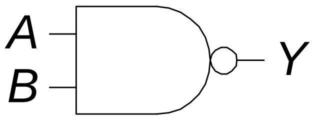
## XOR




$$Y = A \oplus B$$

| $A$ | $B$ | $Y$ |
|---|---|---|
| 0 | 0 | |
| 0 | 1 | |
| 1 | 0 | |
| 1 | 1 | |

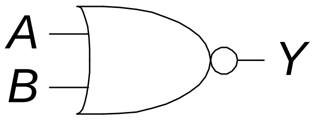
## NAND




$$Y = \overline{AB}$$

| $A$ | $B$ | $Y$ |
|---|---|---|
| 0 | 0 | |
| 0 | 1 | |
| 1 | 0 | |
| 1 | 1 | |

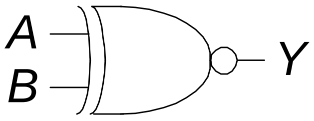
## NOR




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

| $A$ | $B$ | $Y$ |
|---|---|---|
| 0 | 0 | |
| 0 | 1 | |
| 1 | 0 | |
| 1 | 1 | |

## XNOR



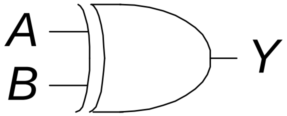

$$Y = \overline{A \oplus B}$$

| $A$ | $B$ | $Y$ |
|---|---|---|
| 0 | 0 | |
| 0 | 1 | |
| 1 | 0 | |
| 1 | 1 | |

Copyright © 2007 Elsevier

# More Two-Input Logic Gates

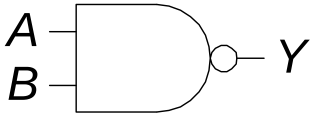
## XOR




$$Y = A \oplus B$$

| A | B | Y |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

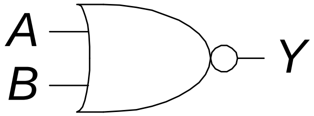
## NAND




$$Y = \overline{AB}$$

| A | B | Y |
|---|---|---|
| 0 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

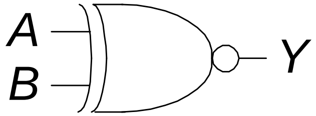
## NOR




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

| A | B | Y |
|---|---|---|
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 0 |

## XNOR



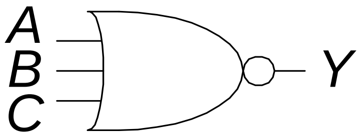

$$Y = \overline{A \oplus B}$$

| A | B | Y |
|---|---|---|
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

Copyright © 2007 Elsevier

# Multiple-Input Logic Gates

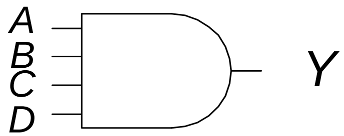
## NOR3




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

| $A$ | $B$ | $C$ | $Y$ |
|---|---|---|---|
| 0 | 0 | 0 | |
| 0 | 0 | 1 | |
| 0 | 1 | 0 | |
| 0 | 1 | 1 | |
| 1 | 0 | 0 | |
| 1 | 0 | 1 | |
| 1 | 1 | 0 | |
| 1 | 1 | 1 | |

## AND4




$$Y = ABCD$$

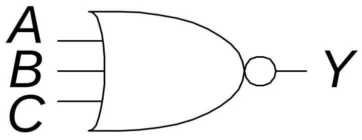
Copyright © 2007 Elsevier



ELSEVIER

# Multiple-Input Logic Gates

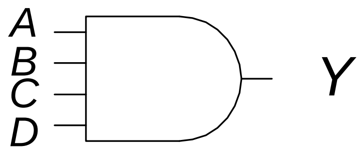
## NOR3




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

| $A$ | $B$ | $C$ | $Y$ |
|---|---|---|---|
| 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 |

## AND4




$$Y = ABCD$$

| $A$ | $B$ | $C$ | $Y$ |
|---|---|---|---|
| 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 |

- Multi-input XOR: Odd parity

Copyright © 2007 Elsevier

# Logic Levels

- Define discrete voltages to represent 1 and 0
- For example, we could define:
  - 0 to be *ground* or 0 volts
  - 1 to be $V_{DD}$ or 5 volts
- What about 4.99 volts? Is that a 0 or a 1?
- What about 3.2 volts?

Copyright © 2007 Elsevier



ELSEVIER

# Logic Levels

- Define a *range* of voltages to represent 1 and 0
- Define different ranges for outputs and inputs to allow for *noise* in the system
- What is noise?

Copyright © 2007 Elsevier



ELSEVIER

# Logic Levels

- Define a *range* of voltages to represent 1 and 0
- Define different ranges for outputs and inputs to allow for *noise* in the system

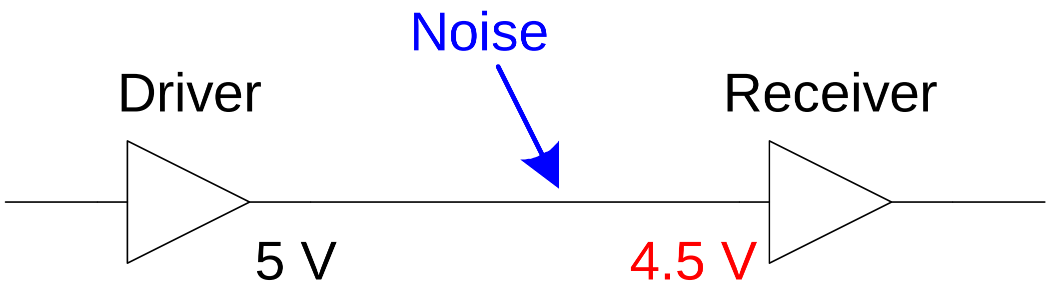
Copyright © 2007 Elsevier

# What is Noise?

Copyright © 2007 Elsevier

# What is Noise?

- **Anything that degrades the signal**
  - E.g., resistance, power supply noise, coupling to neighboring wires, etc.
- **Example:** a gate (driver) could output a 5 volt signal but, because of resistance in a long wire, the signal could arrive at the receiver with a degraded value, for example, 4.5 volts




Copyright © 2007 Elsevier

# The Static Discipline

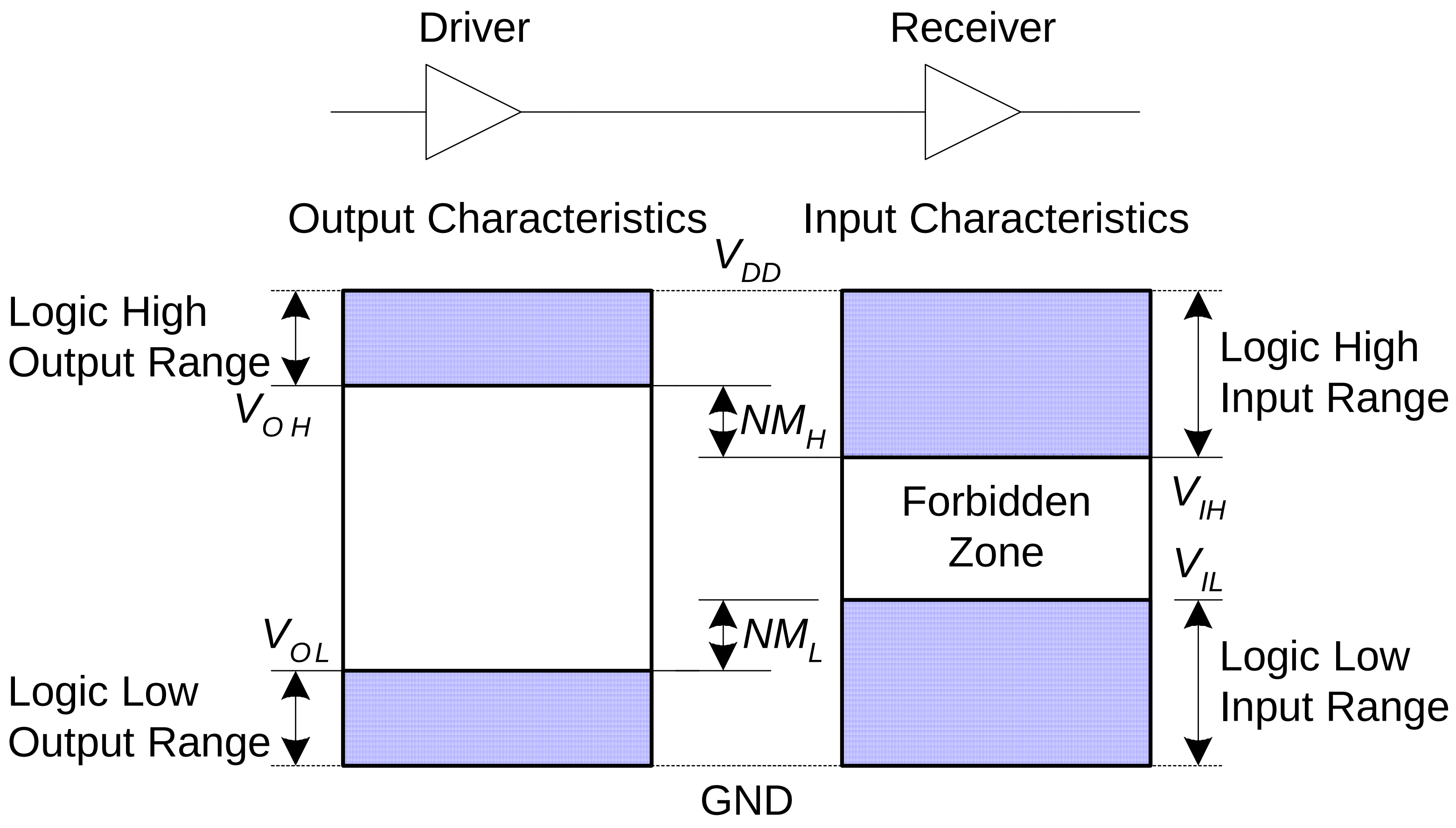
- Given logically valid inputs, every circuit element must produce logically valid outputs

- Discipline ourselves to use limited ranges of voltages to represent discrete values

Copyright © 2007 Elsevier

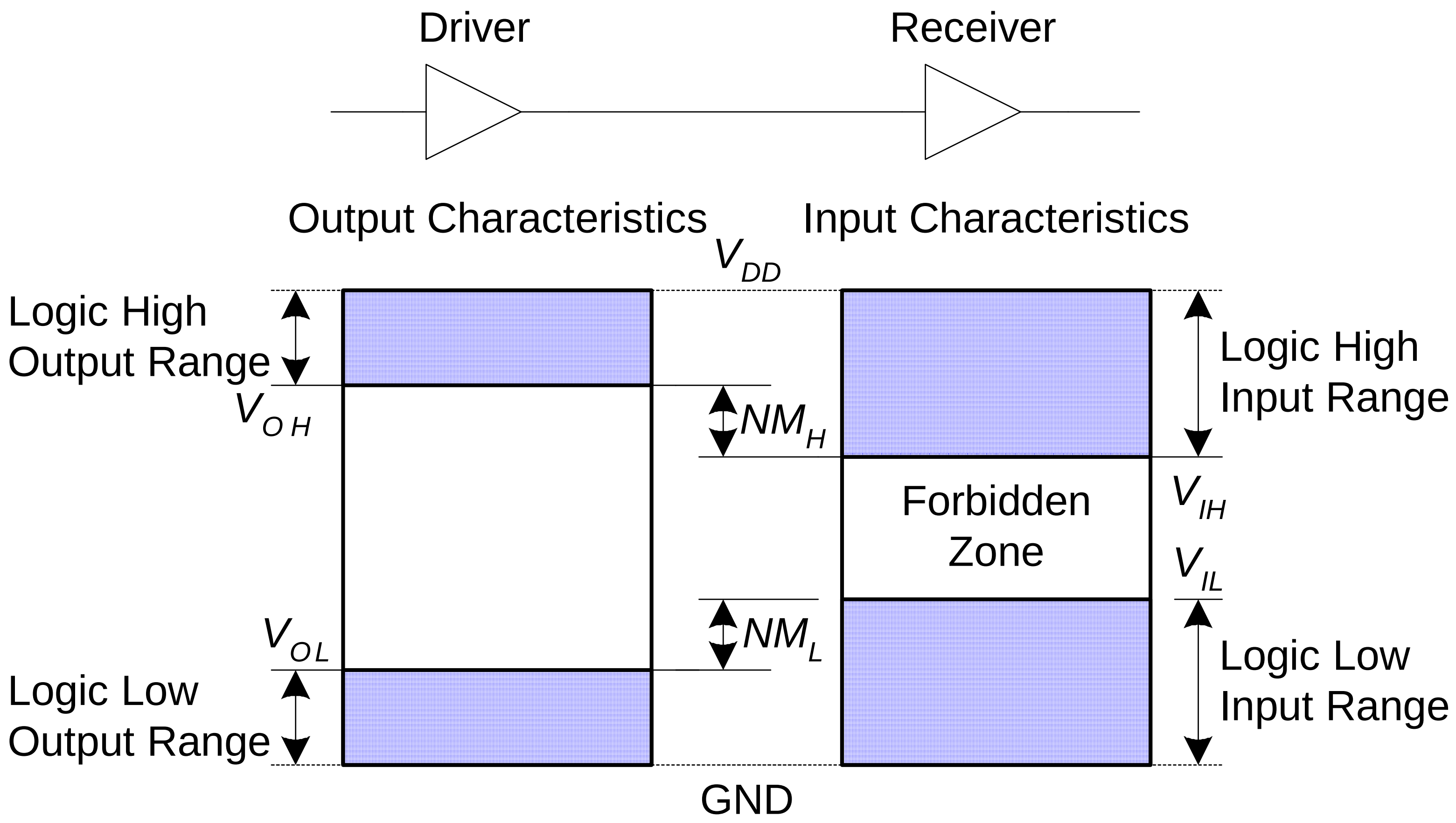
ELSEVIER

# Logic Levels




Copyright © 2007 Elsevier

# Noise Margins



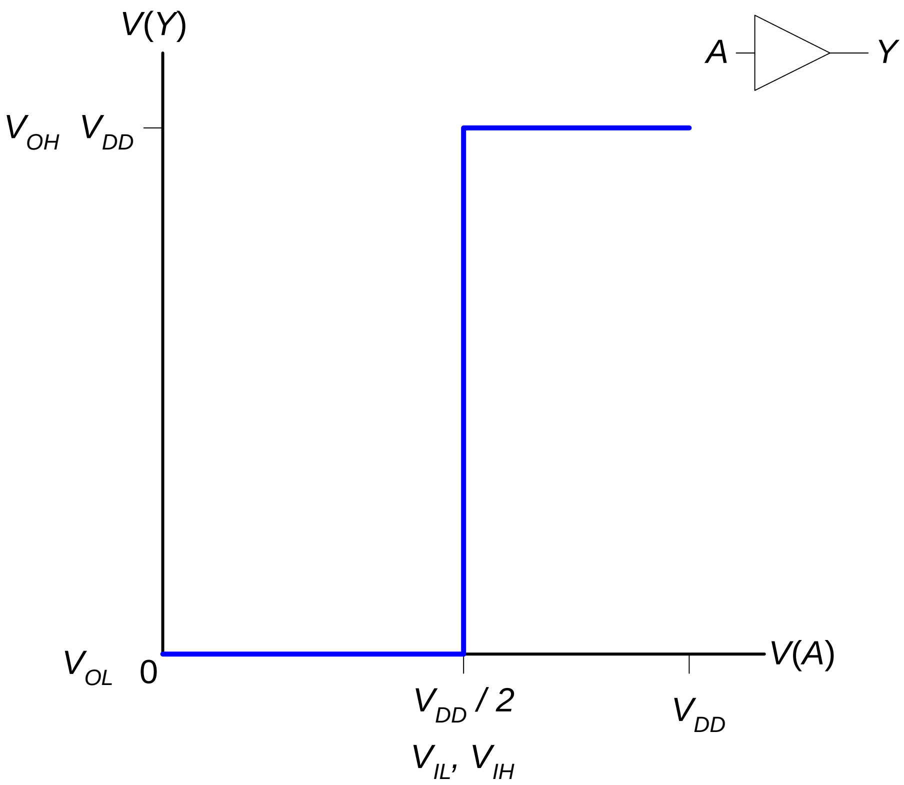

$$NM_H = V_{OH} - V_{IH}$$

$$NM_L = V_{IL} - V_{OL}$$

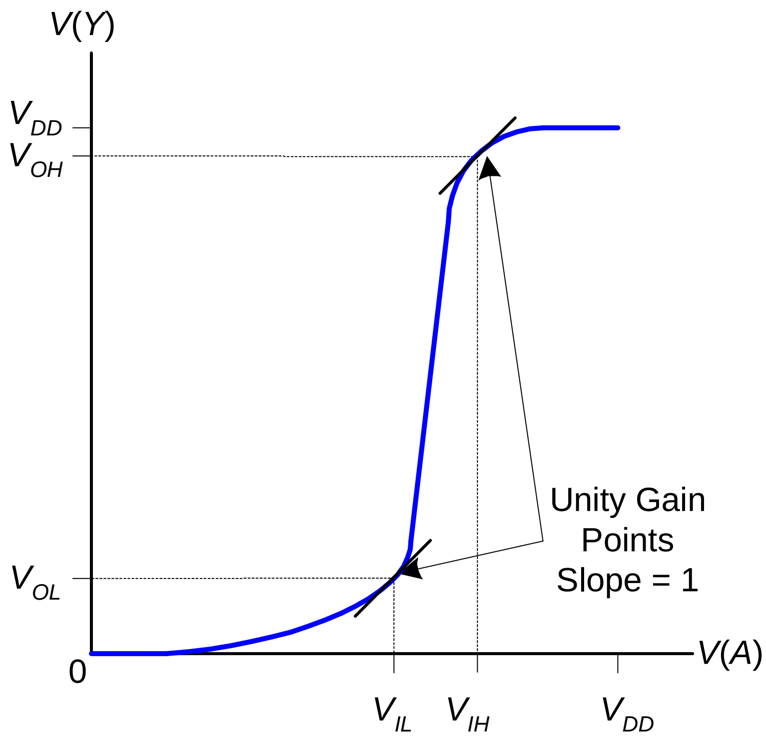
Copyright © 2007 Elsevier

# DC Transfer Characteristics

Ideal Buffer:




Real Buffer:

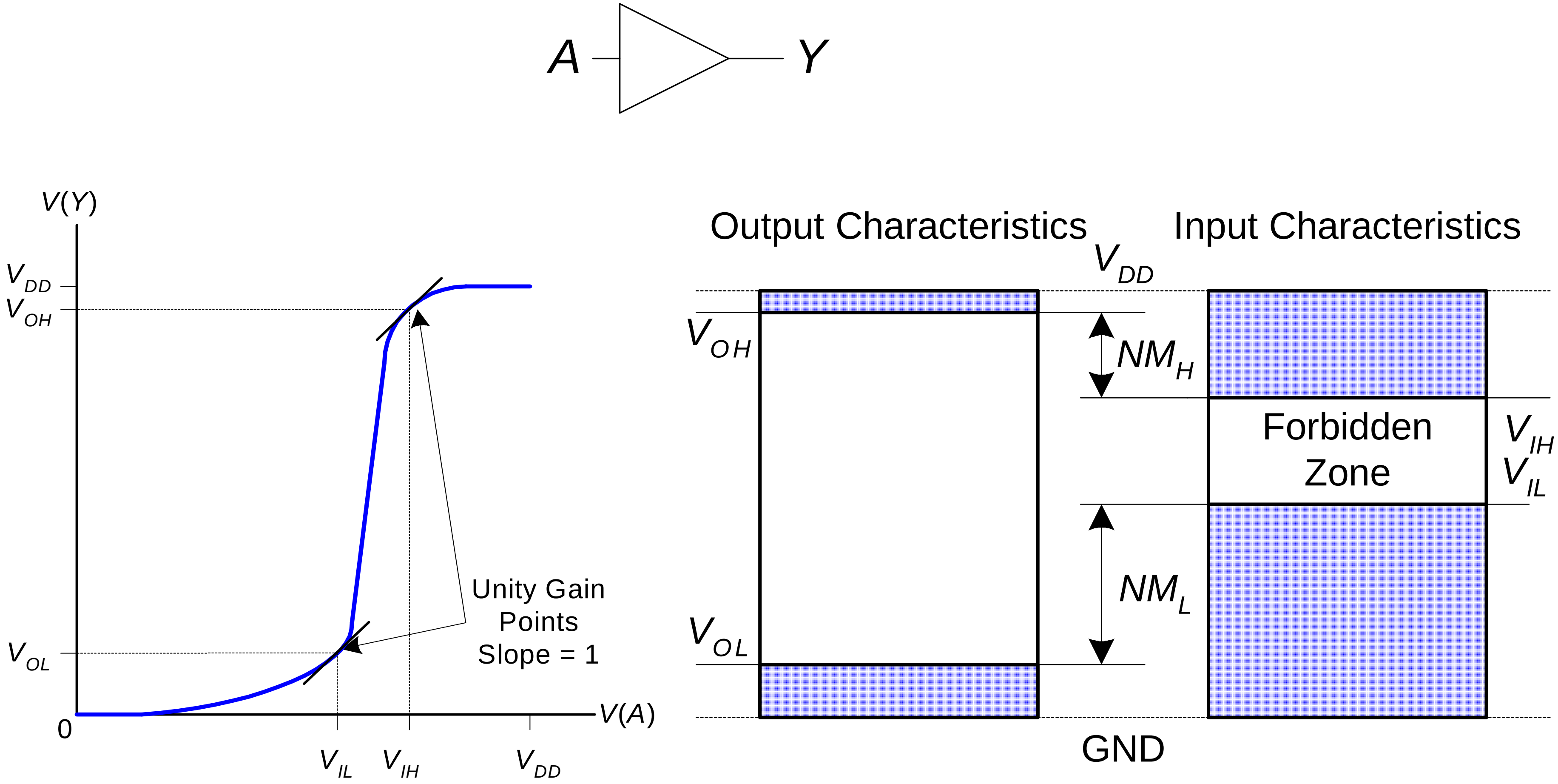



$$NM_H = NM_L = V_{DD}/2$$

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

Copyright © 2007 Elsevier

# DC Transfer Characteristics




Copyright © 2007 Elsevier

# $V_{DD}$ Scaling

- Chips in the 1970's and 1980's were designed using $V_{DD}$ = 5 V
- As technology improved, $V_{DD}$ 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

Chips operate because they contain magic smoke

Proof:

- if the magic smoke is let out, the chip stops working

Copyright © 2007 Elsevier



ELSEVIER

# Logic Family Examples

| 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 |

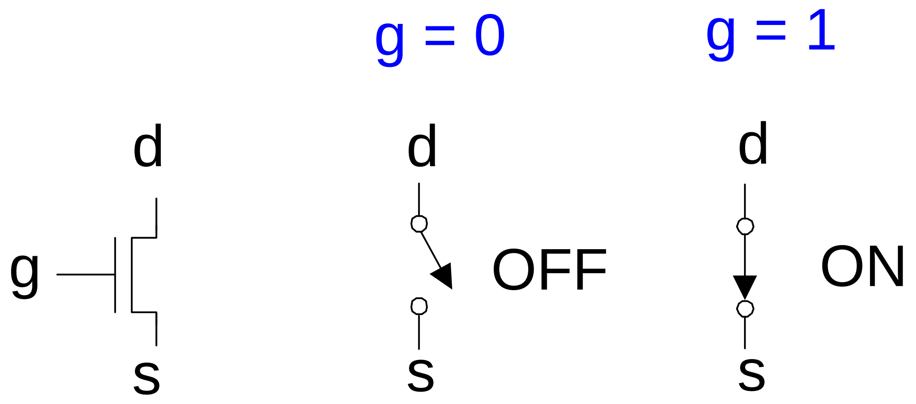
Copyright © 2007 Elsevier



ELSEVIER

# Transistors

- Logic gates are usually built out of transistors
- Transistor is a three-ported voltage-controlled switch
  - Two of the ports are connected depending on the voltage on the third port
  - For example, in the switch below the two terminals (d and s) are connected (ON) only when the third terminal (g) is 1




Copyright © 2007 Elsevier

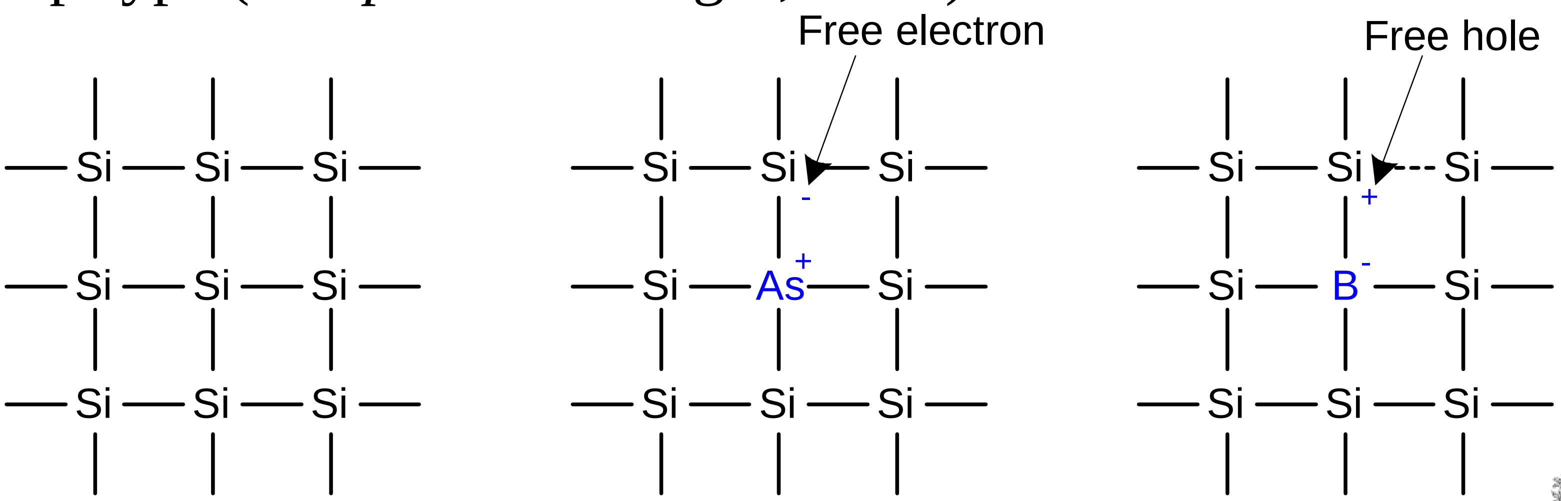
# Robert Noyce, 1927 - 1990

- Nicknamed “Mayor of Silicon Valley”
- Cofounded Fairchild Semiconductor in 1957
- Cofounded Intel in 1968
- Co-invented the integrated circuit

Copyright © 2007 Elsevier

# Silicon

- Transistors are built out of 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)



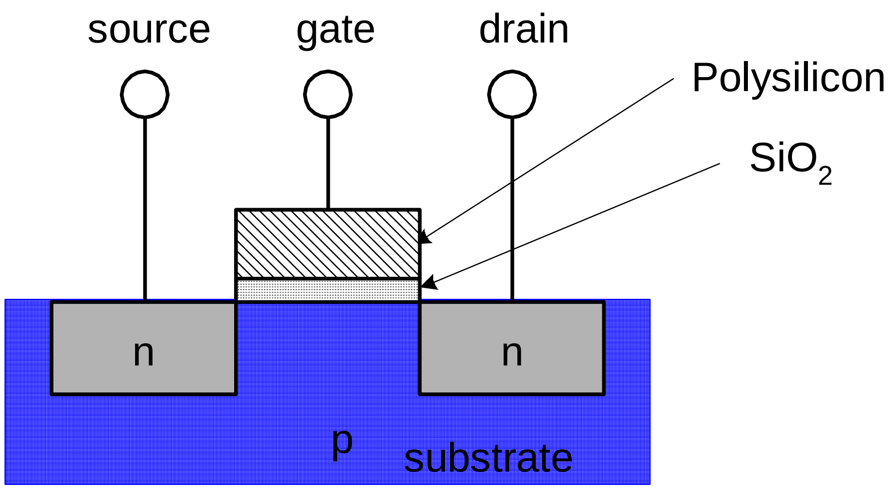

**Silicon Lattice** **n-Type** **p-Type**

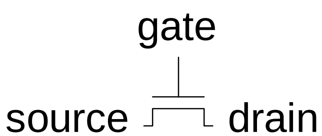
Copyright © 2007 Elsevier

ELSEVIER

# MOS Transistors

- Metal oxide silicon (MOS) transistors:
  - Polysilicon (used to be **metal**) gate
  - **Oxide** (silicon dioxide) insulator
  - Doped **silicon**



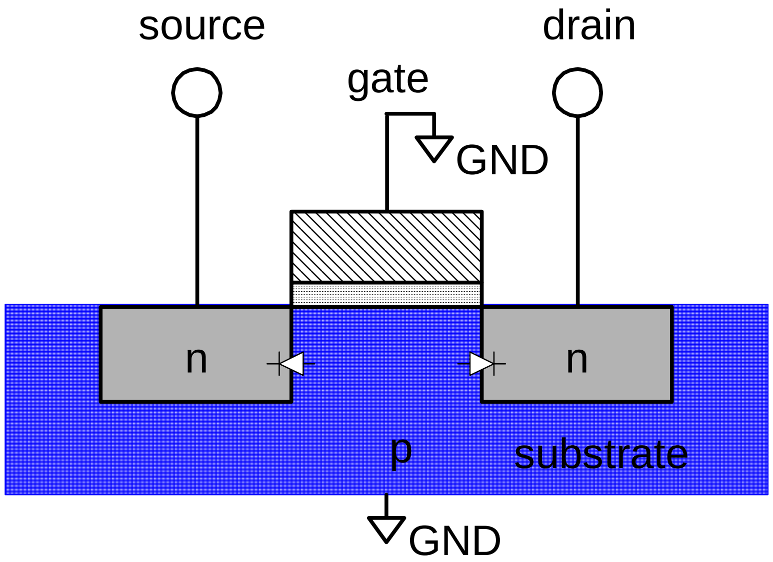




nMOS

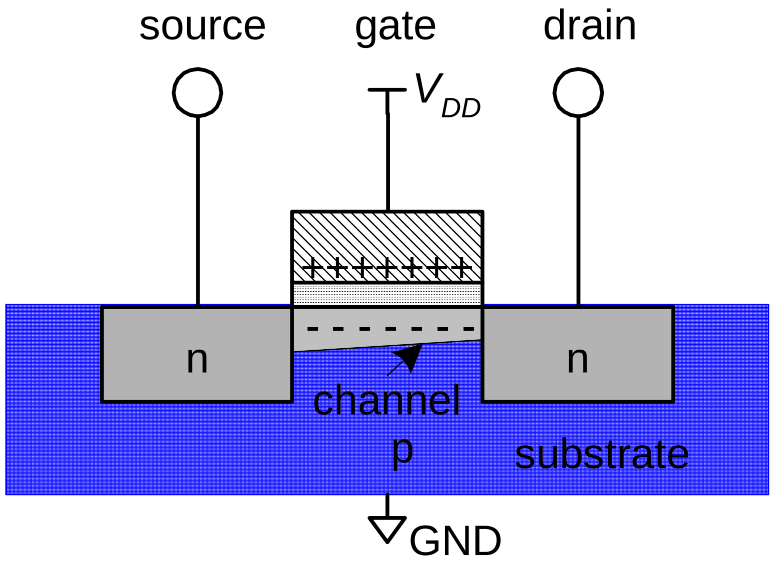
Copyright © 2007 Elsevier

# Transistors: nMOS

Gate = 0, so it is OFF (no connection between source and drain)



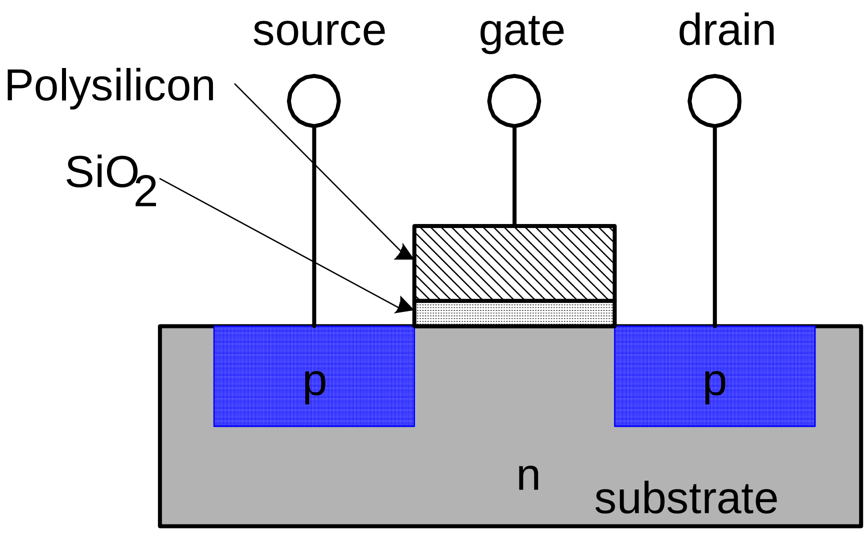

Gate = 1, so it is ON (channel between source and drain)

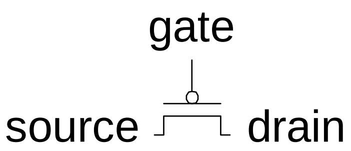



Copyright © 2007 Elsevier

# Transistors: pMOS

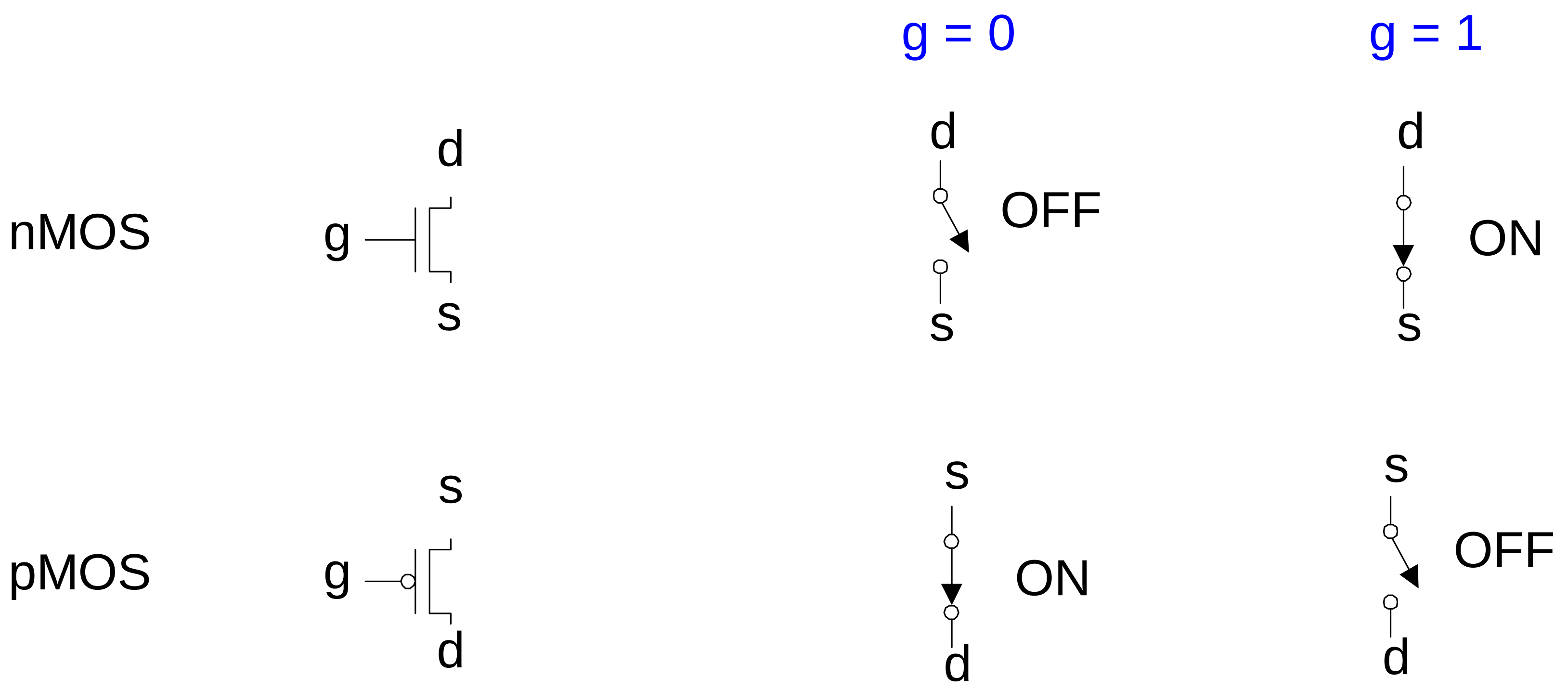
- pMOS transistor is just the opposite
  - ON when Gate = 0
  - OFF when Gate = 1




Copyright © 2007 Elsevier

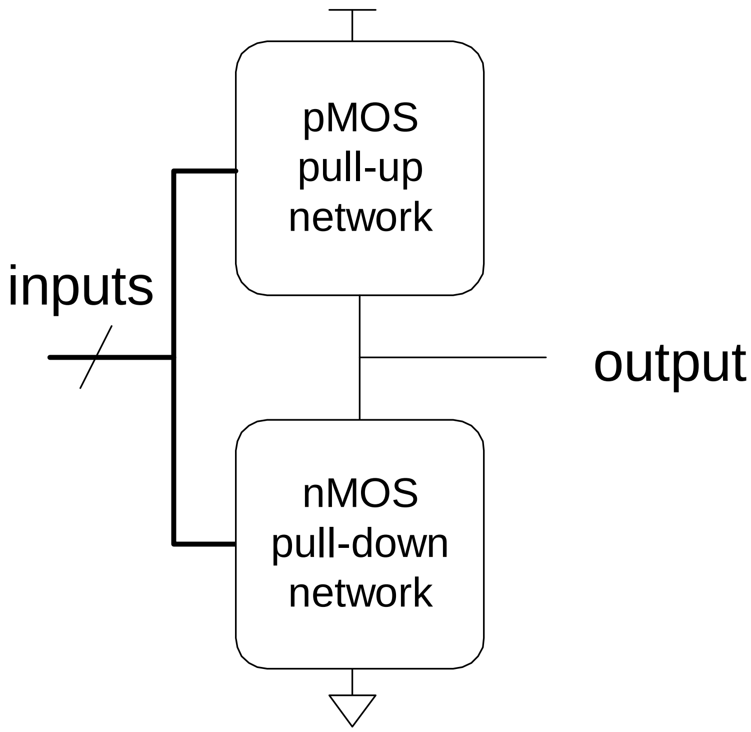
# Transistor Function




Copyright © 2007 Elsevier

# Transistor Function

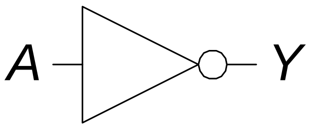
- nMOS transistors pass good 0's, so connect source to GND
- pMOS transistors pass good 1's, so connect source to $V_{DD}$




Copyright © 2007 Elsevier

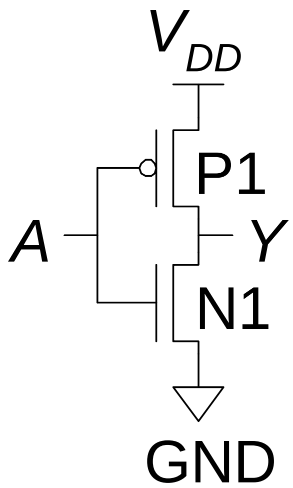
# CMOS Gates: NOT Gate

## NOT




$$Y = \overline{A}$$

| A | Y |
|---|---|
| 0 | 1 |
| 1 | 0 |




| *A* | P1 | N1 | *Y* |
|---|---|---|---|
| 0 | | | |
| 1 | | | |

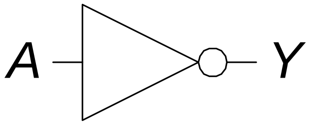
Copyright © 2007 Elsevier



ELSEVIER

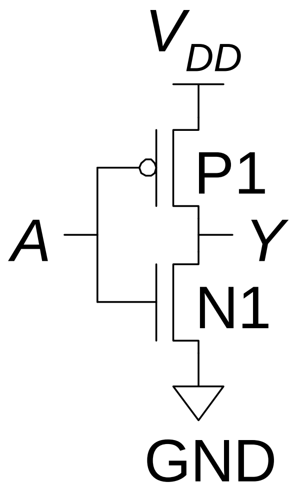
# CMOS Gates: NOT Gate

**NOT**




$Y = \overline{A}$

| $A$ | $Y$ |
|---|---|
| 0 | 1 |
| 1 | 0 |




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

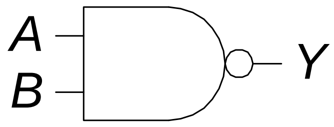
Copyright © 2007 Elsevier



ELSEVIER

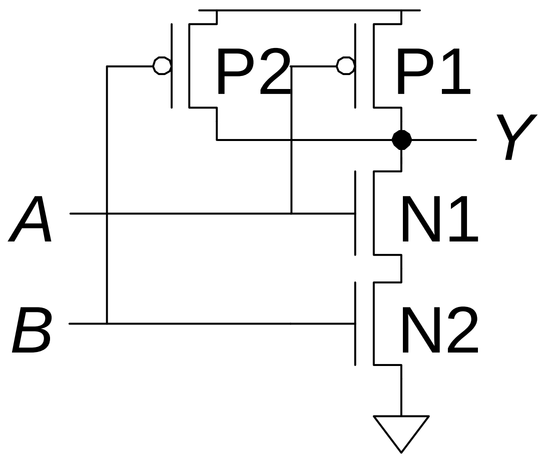
# CMOS Gates: NAND Gate

**NAND**




$Y = \overline{AB}$

| $A$ | $B$ | $Y$ |
|---|---|---|
| 0 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

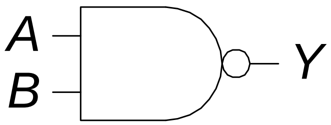



| $A$ | $B$ | P1 | P2 | N1 | N2 | $Y$ |
|---|---|---|---|---|---|---|
| 0 | 0 | | | | | |
| 0 | 1 | | | | | |
| 1 | 0 | | | | | |
| 1 | 1 | | | | | |

Copyright © 2007 Elsevier

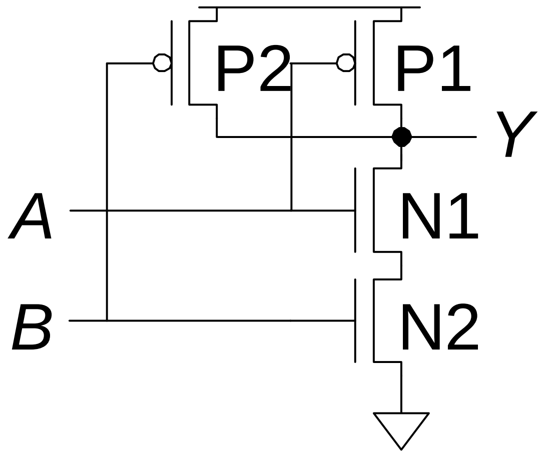
# CMOS Gates: NAND Gate

## NAND




$Y = \overline{AB}$

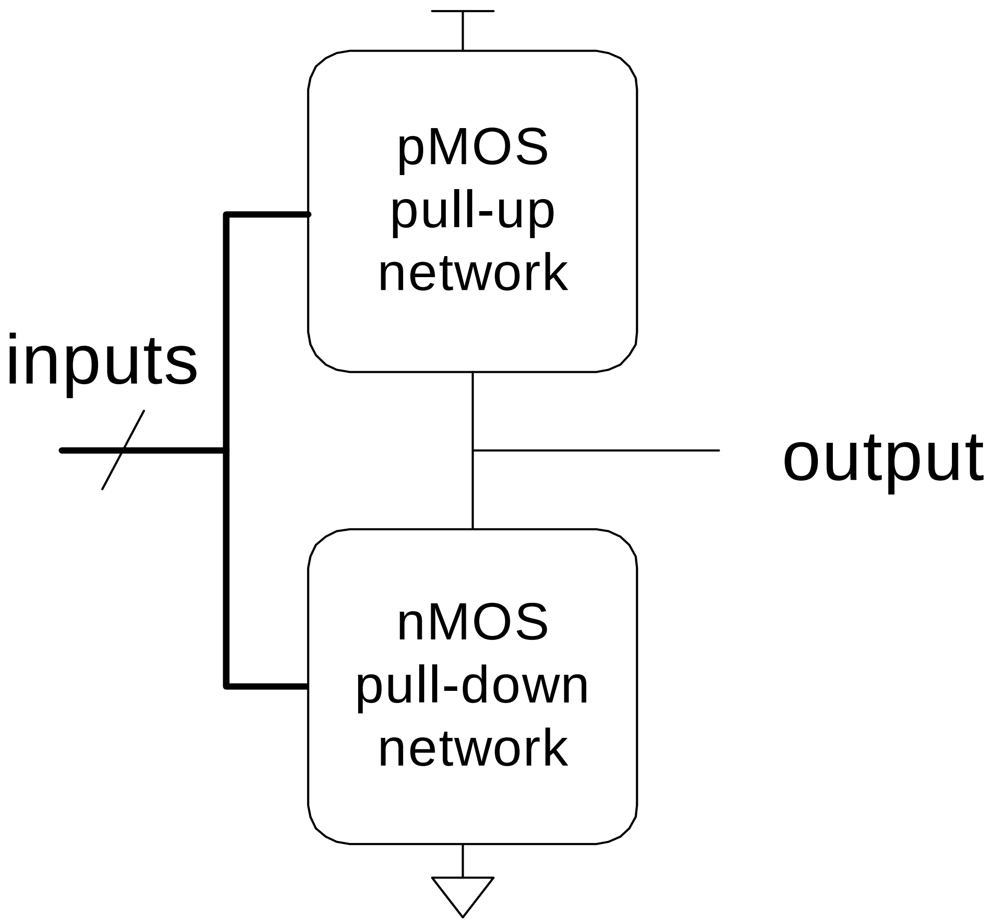
| $A$ | $B$ | $Y$ |
|---|---|---|
| 0 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |




| *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 |

Copyright © 2007 Elsevier

# CMOS Gate Structure




Copyright © 2007 Elsevier

# NOR Gate

How do you build a three-input NOR gate?

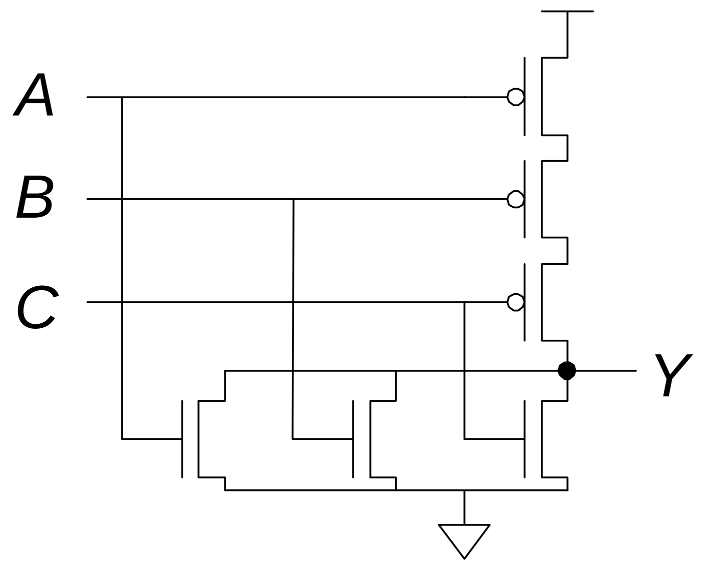
Copyright © 2007 Elsevier



ELSEVIER

# NOR3 Gate

Three-input NOR gate




Copyright © 2007 Elsevier



ELSEVIER

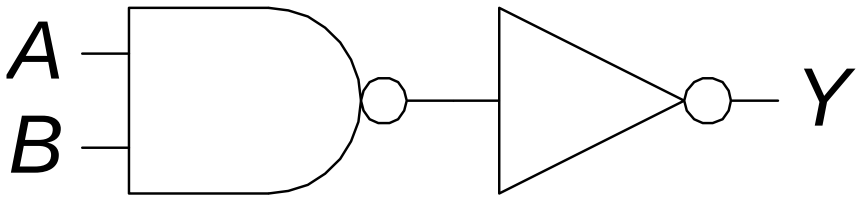
# Other CMOS Gates

How do you build a two-input AND gate?

Copyright © 2007 Elsevier

# Other CMOS Gates

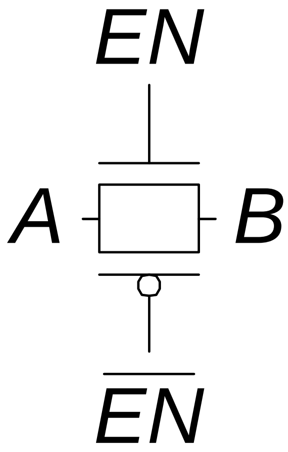
## Two-input AND gate




Copyright © 2007 Elsevier

# Transmission Gates

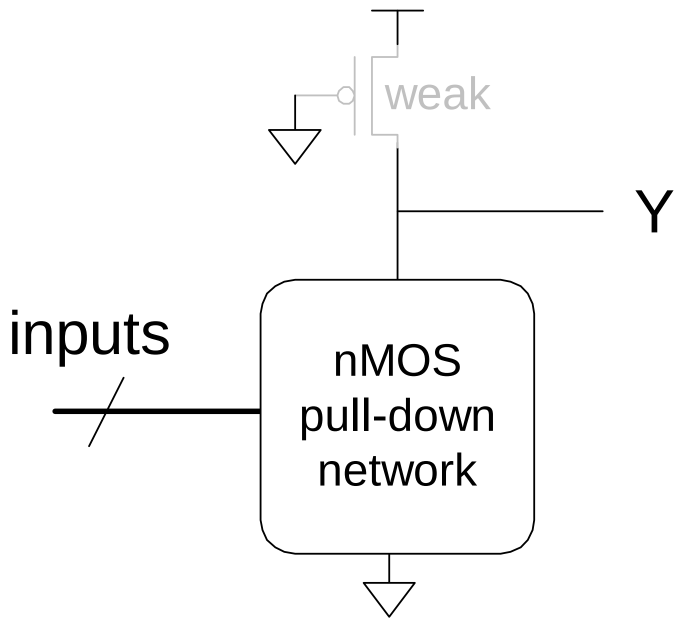
- nMOS pass 1's poorly
- pMOS pass 0's poorly
- Transmission gate is a better switch
  - passes both 0 and 1 well
- When *EN* = 1, the switch is ON:
  - *EN* = 0 and *A* is connected to *B*
- When *EN* = 0, the switch is OFF:
  - *A* is not connected to *B*




Copyright © 2007 Elsevier

# Pseudo-nMOS Gates

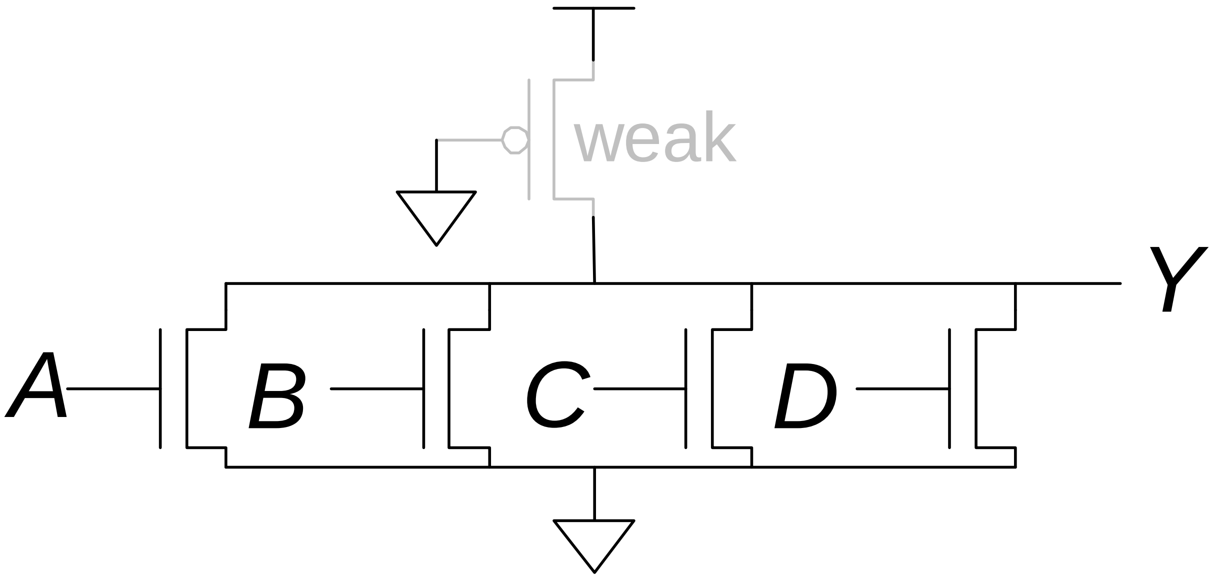
- nMOS gates replace the pull-up network with a *weak* pMOS transistor that is always on
- The pMOS transistor is called weak because it pulls the output HIGH only when the nMOS network is not pulling it LOW




Copyright © 2007 Elsevier

# Pseudo-nMOS Example

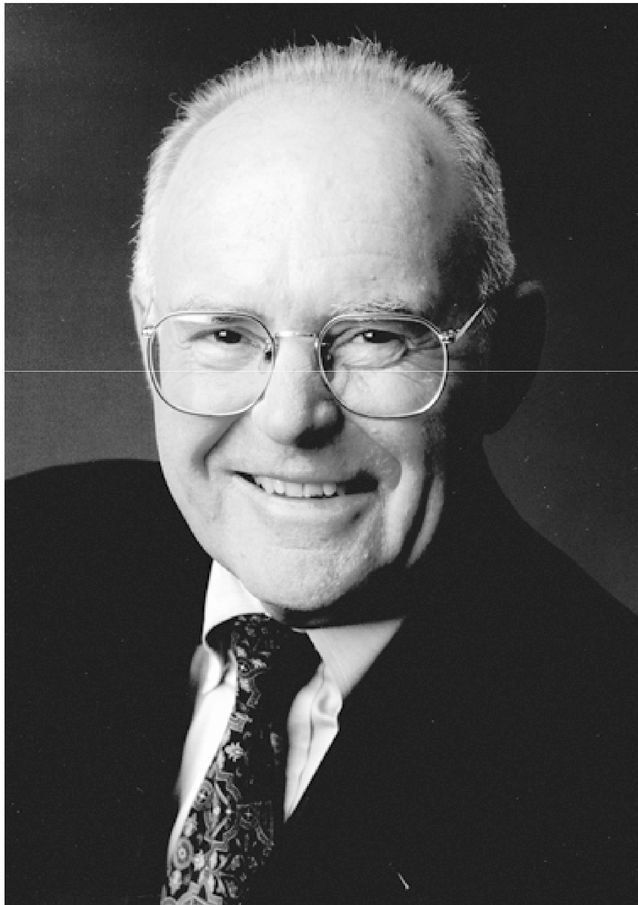
Pseudo-nMOS NOR4




Copyright © 2007 Elsevier

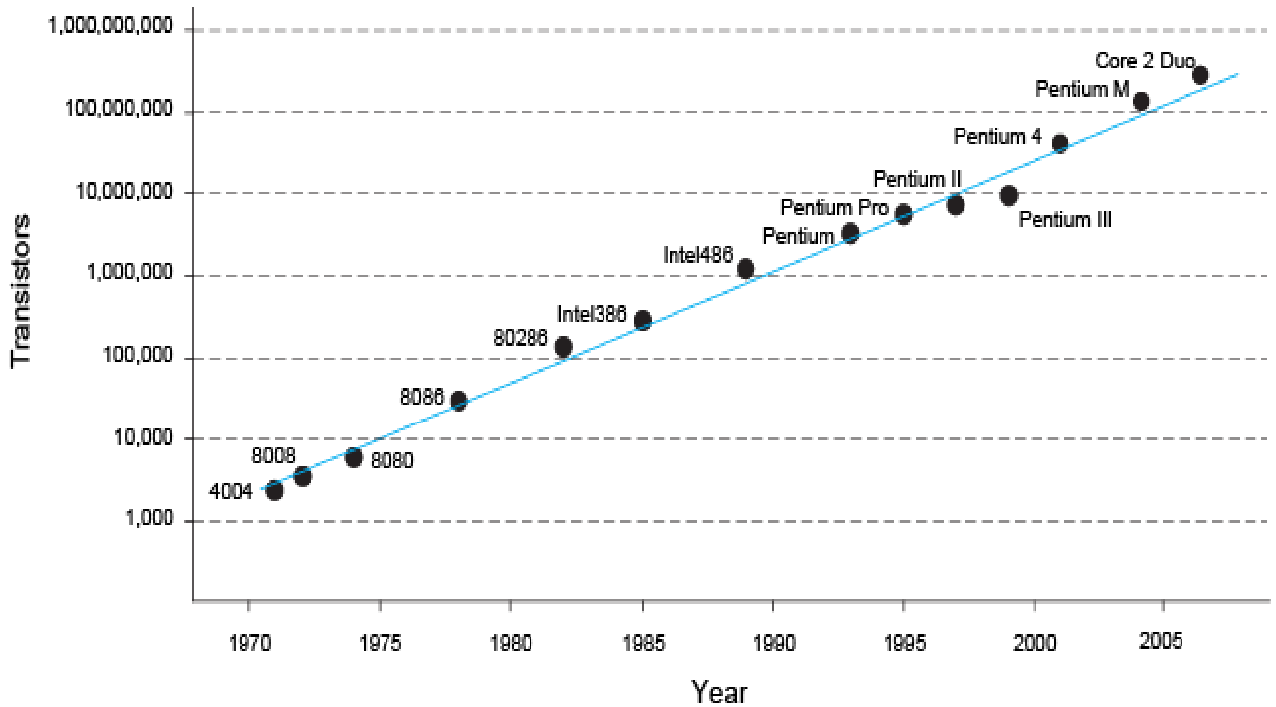
# Gordon Moore, 1929 -

- Cofounded Intel in 1968 with Robert Noyce.
- **Moore's Law:** the number of transistors on a computer chip doubles every year (observed in 1965)
- Since 1975, transistor counts have doubled every two years.



Copyright © 2007 Elsevier

# Moore's Law




- *"If the automobile had followed the same development cycle as the computer, a Rolls-Royce would today cost $100, get one million miles to the gallon, and explode once a year . . ."*

  *– Robert Cringley*

Copyright © 2007 Elsevier



ELSEVIER

# Power Consumption

- Power = Energy consumed per unit time
- Two types of power consumption:
  - Dynamic power consumption
  - Static power consumption

Copyright © 2007 Elsevier



ELSEVIER

# Dynamic Power Consumption

- Power to charge transistor gate capacitances
- The energy required to charge a capacitance, $C$, to $V_{DD}$ is $CV_{DD}^2$
- If the circuit is running at frequency $f$, and all transistors switch (from 1 to 0 or vice versa) at that frequency, the capacitor is charged $f/2$ times per second (discharging from 1 to 0 is free).
- Thus, the total dynamic power consumption is:

$$\boxed{P_{dynamic} = \tfrac{1}{2} C V_{DD}^2 f}$$

Copyright © 2007 Elsevier



ELSEVIER

# Static Power Consumption

- Power consumed when no gates are switching
- It is caused by the *quiescent supply current*, $I_{DD}$, also called the *leakage current*
- Thus, the total static power consumption is:

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

Copyright © 2007 Elsevier

# Power Consumption Example

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

Copyright © 2007 Elsevier



ELSEVIER

# Power Consumption Example

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

$$P = \tfrac{1}{2} C V_{DD}^2 f + \mathrm{I}_{DD} V_{DD}$$

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

**= 14.4 W**

Copyright © 2007 Elsevier



ELSEVIER