Computer Architecture
(CSC-3501)
Lecture 4
(24 Jan 2008)

Seung-Jong Park (Jay)
http://www.csc.lsu.edu/~sjpark

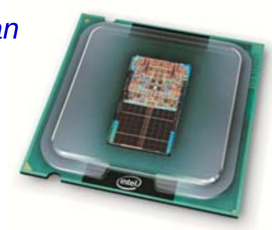
Announcement

- 1^{st} Homework's due date is tomorrow
  - Due date will be 72 hours later

Objectives of Ch. 4

- Understand the relationship between Boolean logic and digital computer circuits.
- Learn how to design simple logic circuits.
- Understand how digital circuits work together to form complex computer systems

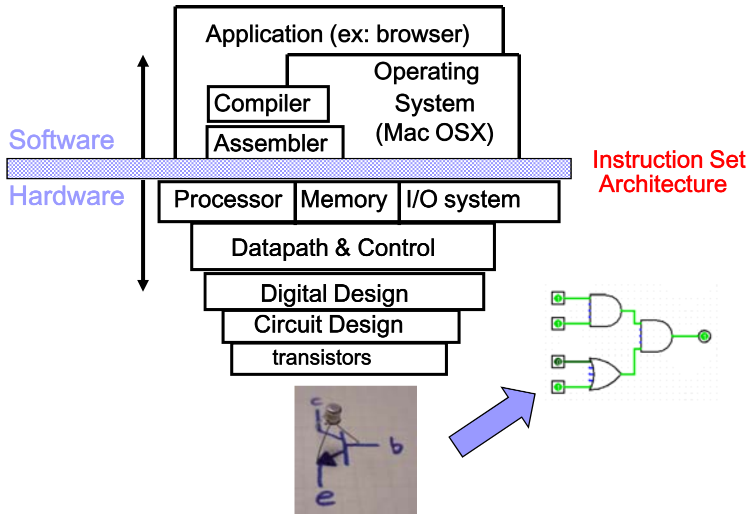
Why do we need to learn logic ?



- "*The Intel Core 2 Duo desktop processor is an energy-efficient marvel, packing 291 million transistors yet consuming lower power*"
  from Intel

- We need to know relationships between transistors and computers
  - Transistors comprise basic logic gates, e.g., AND, OR, NOT, NAND, NOR, XOR, etc.
  - Basic logic gates comprise complicated functional units, e.g., adder, counter, memory, CPU
  - Finally, computer is built with those complicated functional units

Computer Architecture

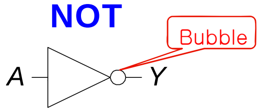


Basic Logic Gates

- 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

Single-Input Logic Gates

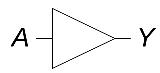
NOT



$Y = \overline{A}$

A	Y
0	1
1	0

BUF

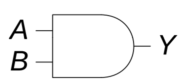


$Y = A$

A	Y
0	0
1	1

Two-Input Logic Gates

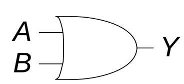
AND



$Y = AB$

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

OR

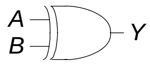


$Y = A + B$

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

More Two-Input Logic Gates

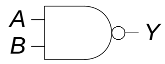
XOR



$Y = A \oplus B$

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

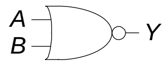
NAND



$Y = \overline{AB}$

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

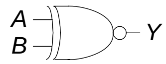
NOR



$Y = \overline{A + B}$

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

XNOR

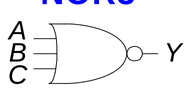


$Y = \overline{A \oplus B}$

A	B	Y
0	0	
0	1	
1	0	
1	1	

Multiple-Input Logic Gates

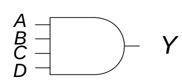
NOR3



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

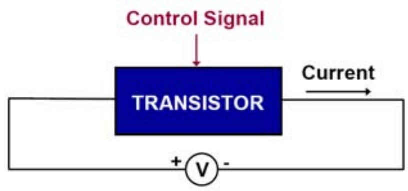
A	B	C	Y
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

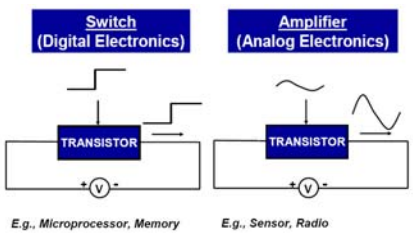
AND4

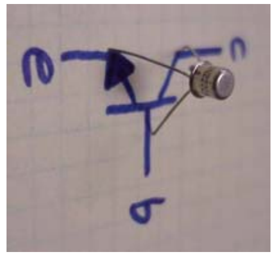


$Y = ABCD$

Transistor

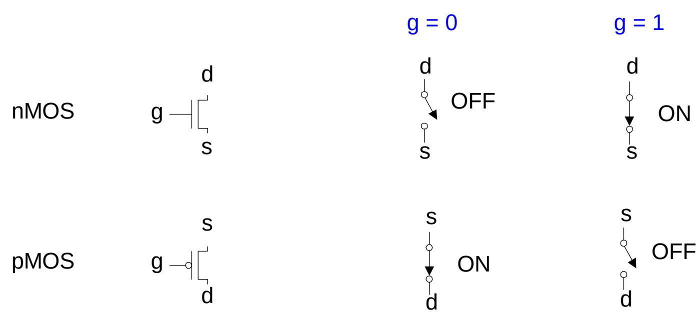






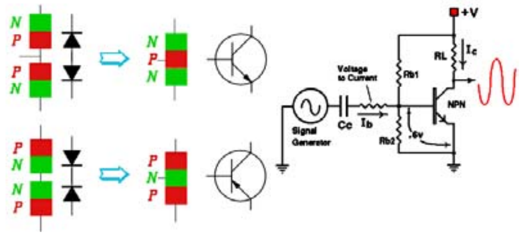
Transistor

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

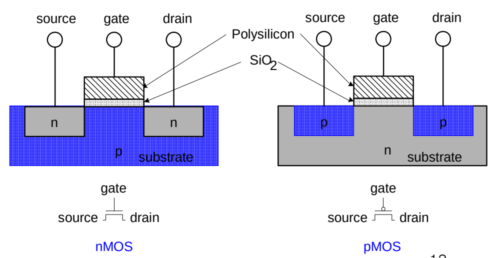


Different Kinds of Transistors

- Junction Transistor
  - A Bipolar Transistor essentially consists of a pair of PN Junction Diodes that are joined back-to-back.
  - It acts as an amplifier or a switch



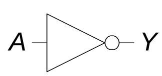
- Metal oxide silicon (MOS) transistors
  - Polysilicon (used to be **metal**) gate
  - **Oxide** (silicon dioxide) insulator
  - Doped **silicon** substrate and wells



How to Build Basic Logic Gates

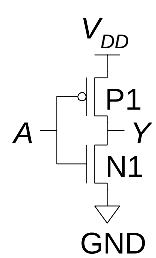
- Build NOT gate with transistors

NOT



$Y = \overline{A}$

A	Y
0	1
1	0

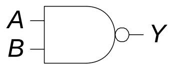


A	P1	N1	Y
0	ON	OFF	1
1	OFF	ON	0

How to build Basic Logic Gates

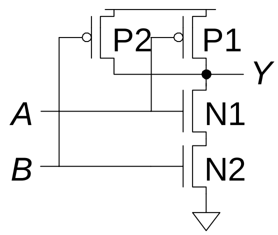
- Build a NAND gate with transistors

NAND



$Y = \overline{AB}$

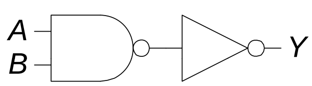
A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0



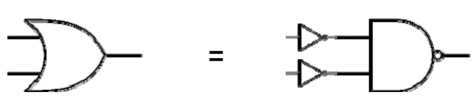
A	B	P1	P2	N1	N2	Y
0	0	ON	ON	OFF	OFF	1
0	1	ON	OFF	OFF	ON	1
1	0	OFF	ON	ON	OFF	1
1	1	OFF	OFF	ON	ON	0

How to Build Other Logic Gates

- Build a AND gate with NAND gate



- Build a OR gate with NAND gate



- Therefore, NAND gate is a basic unit to build complicated functional logic circuits
- NOW!, we know that transistors are fundamental components to build a computer !!!

Boolean Algebra

- Boolean algebra is a mathematical system for the manipulation of variables that can have one of two values.
  - In formal logic, these values are "true" and "false."
  - In digital systems, these values are "on" and "off," 1 and 0, or "high" and "low."
- Boolean expressions are created by performing operations on Boolean variables.
  - Common Boolean operators include AND, OR, and NOT.

Boolean Algebra

- A Boolean operator can be completely described using a truth table.
- The truth table for the Boolean operators AND and OR are shown at the right.
- The AND operator is also known as a Boolean product. The OR operator is the Boolean sum.
- The truth table for the Boolean NOT operator i s shown at the right.
- The NOT operation is most often designated by an overbar. It is sometimes indicated by a prime mark (') or an "elbow" (¬).

X AND Y

X	Y	XY
0	0	0
0	1	0
1	0	0
1	1	1

X OR Y

X	Y	X+Y
0	0	0
0	1	1
1	0	1
1	1	1

NOT X

X	$\overline{X}$
0	1
1	0

Boolean Function

- A Boolean function has:
  - At least one Boolean variable,
  - At least one Boolean operator, and
  - At least one input from the set {0,1}.
- It produces an output that is also a member of the set {0, 1}.

Now you know why the binary numbering system is so handy in digital systems.

Boolean Function

- The truth table for the Bool ean function:

  $F(x,y,z) = x\bar{z}+y$

  is shown at the right.
- To make evaluation of the Boolean function easier, th e truth table contains extra (shaded) columns to hold e valuations of subparts of th e function

$F(x,y,z) = x\bar{z}+y$

x	y	z	$\bar{z}$	$x\bar{z}$	$x\bar{z}+y$
0	0	0	1	0	0
0	0	1	0	0	0
0	1	0	1	0	1
0	1	1	0	0	1
1	0	0	1	1	1
1	0	1	0	0	0
1	1	0	1	1	1
1	1	1	0	0	1

Boolean Function

- As with common arithmetic, Boolean operations have rules of precedence.
- The NOT operator has highest priority, followed by AND and then OR.
- This is how we chose the (shaded) function subparts in our table.

$F(x,y,z) = x\bar{z}+y$

x	y	z	$\bar{z}$	$x\bar{z}$	$x\bar{z}+y$
0	0	0	1	0	0
0	0	1	0	0	0
0	1	0	1	0	1
0	1	1	0	0	1
1	0	0	1	1	1
1	0	1	0	0	0
1	1	0	1	1	1
1	1	1	0	0	1

To be continued …more complicated Boolean Algebra.