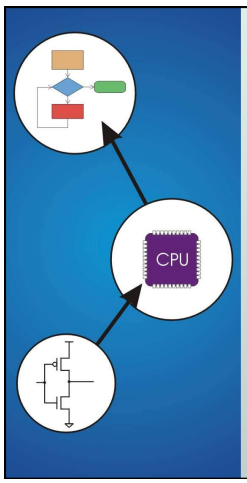
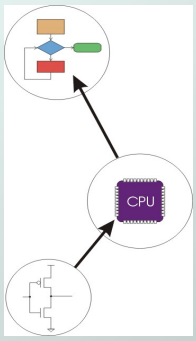


Chapter 3
Digital Logic Structures

Original slides from Gregory Byrd, North Carolina State University

Modified slides by Chris Wilcox, Colorado State University

Computing Layers



Problems

Algorithms

Language

Instruction Set Architecture

Microarchitecture

Circuits

Devices ←

Transistor: Building Block of Computers

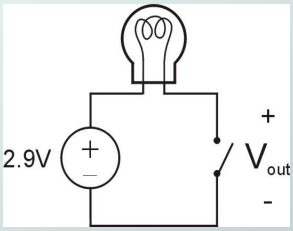
- Microprocessors contain lots of transistors
  - **Intel 8086 (1978):** 29 thousand
  - **Intel 80186 (1982):** 55 thousand
  - **Intel 80386 (1985):** 275 thousand
  - **Intel 80486 (1989):** 1.1 million
  - **Intel Pentium (1993):** 3.1 million
  - **Intel Pentium II (1998):** 7.5 million
  - **Intel Pentium III (2001):** 45 million
  - **Intel Pentium 4 (2006):** 184 million
  - **Intel Core 2 Duo (2006):** 291 million
  - **Intel Quad Core i7 (2011):** 1.1 billion
  - **Intel 8-core Xeon (2012):** 2.3 billion

Transistor: Building Block of Computers

Logically, each transistor acts as a switch

- Combined to implement logic functions (gates)
  - AND, OR, NOT
- Combined to build higher-level structures
  - Adder, multiplexer, decoder, register, memory …
  - Adder, multiplier …
- Combined to build simple processor
  - LC-3

Simple Switch Circuit

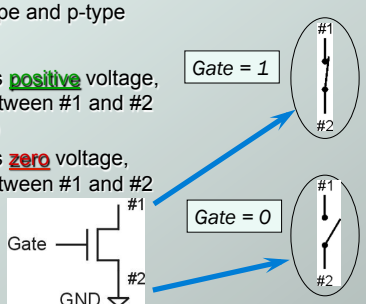


- Switch **open**:
  - Open circuit, no current
  - Light is **off**
  - V_{out} is **+2.9V**
- Switch **closed**:
  - Short circuit across switch, current flows
  - Light is **on**
  - V_{out} is **0V**

Switch-based circuits can easily represent two states: on/off, open/closed, voltage/no voltage.

n-type MOS Transistor

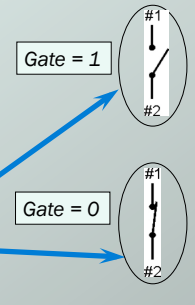
- MOS = Metal Oxide Semiconductor
  - two types: n-type and p-type
- n-type
  - when Gate has **positive** voltage, short circuit between #1 and #2 (switch **closed**)
  - when Gate has **zero** voltage, open circuit between #1 and #2 (switch **open**)



Terminal #2 must be connected to GND (0V).

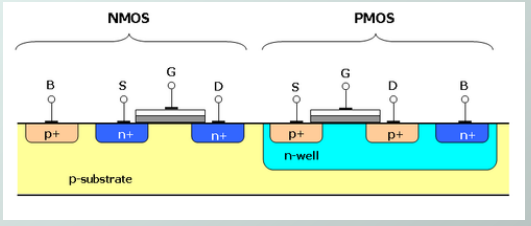
p-type MOS Transistor

- p-type is *complementary* to n-type
  - when Gate has **positive** voltage, open circuit between #1 and #2 (switch **open**)
  - when Gate has **zero** voltage, short circuit between #1 and #2 (switch **closed**)



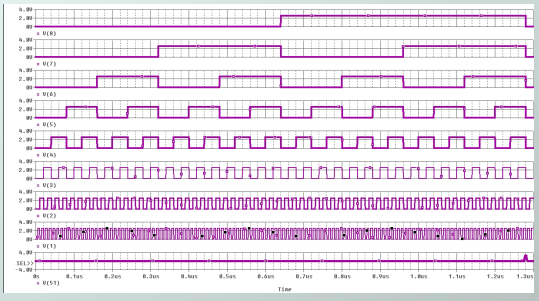
Terminal #1 must be connected to +2.9V.

Physical Transistor



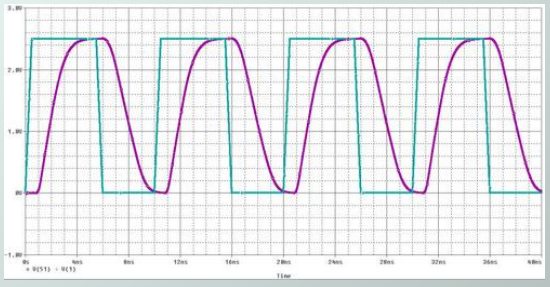
http://en.wikipedia.org/wiki/CMOS

Transistor Output (Ideal)



Logic analyzer view of waveforms

Transistor Output (Actual)



Actual waveform is not ideal!

Logic Gates

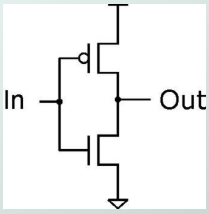
- Use switch behavior of MOS transistors to implement logical functions: AND, OR, NOT.
- Digital symbols:
  - recall that we assign a range of analog voltages to each digital (logic) symbol

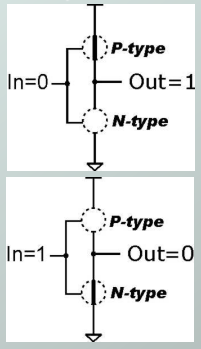
  - assignment of voltage ranges depends on electrical properties of transistors being used
    - typical values for "1": +5V, +3.3V, +2.9V
    - from now on we'll use +2.9V

CMOS Circuit

- **Complementary** MOS
- Uses both **n-type** and **p-type** MOS transistors
  - p-type
    - Attached to + voltage
    - Pulls output voltage UP when input is zero
  - n-type
    - Attached to GND
    - Pulls output voltage DOWN when input is one
- **For all inputs, make sure that output is either connected to GND or to +, but not both!**

Inverter (NOT Gate)



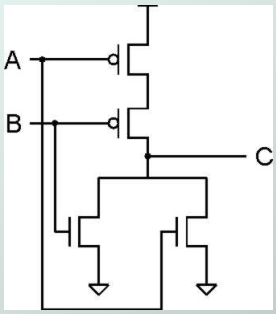


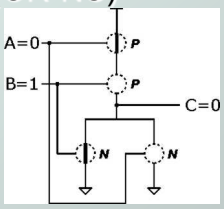
In	Out
0 V	2.9 V
2.9 V	0 V

In	Out
0	1
1	0

Truth table

NOR Gate (OR-NO)



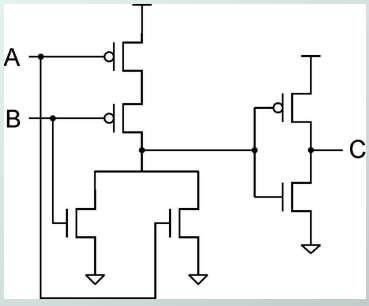


A	B	C
0	0	1
0	1	0
1	0	0
1	1	0

Truth table

Note: Serial structure on top, parallel on bottom.

OR Gate

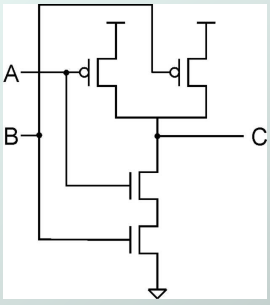


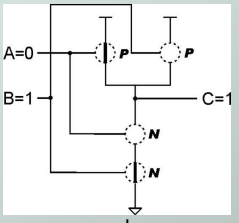
A	B	C
0	0	0
0	1	1
1	0	1
1	1	1

Truth table

Add inverter to NOR.

NAND Gate (AND-NOT)



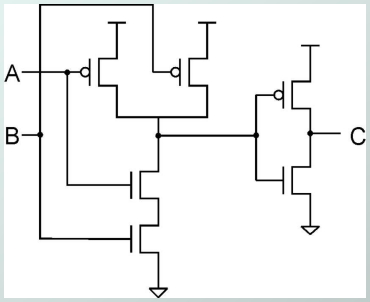


A	B	C
0	0	1
0	1	1
1	0	1
1	1	0

Truth table

Note: Parallel structure on top, serial on bottom.

AND Gate

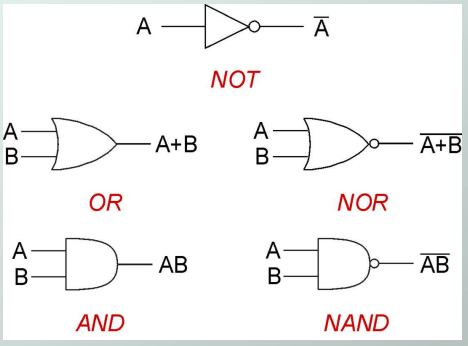


A	B	C
0	0	0
0	1	0
1	0	0
1	1	1

Truth table

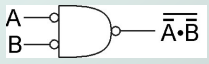
Add inverter to NAND.

Basic Logic Gates



DeMorgan's Law

- Converting AND to OR (with some help from NOT)
- Consider the following gate:



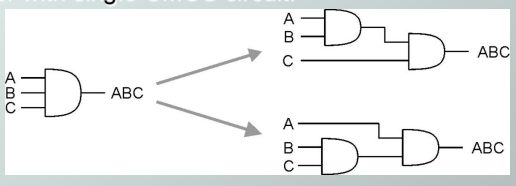
A	B	$\overline{A}$	$\overline{B}$	$\overline{A}\cdot\overline{B}$	$\overline{\overline{A}\cdot\overline{B}}$
0	0	1	1	1	0
0	1	1	0	0	1
1	0	0	1	0	1
1	1	0	0	0	1

Same as A OR B!

To convert AND to OR (or vice versa), invert inputs and output.

More than 2 Inputs?

- AND/OR can take any number of inputs.
  - AND = 1 if all inputs are 1.
  - OR = 1 if any input is 1.
  - Similar for NAND/NOR.
- Can implement with multiple two-input gates, or with single CMOS circuit.



Summary

- MOS transistors are used as switches to implement logic functions.
  - n-type: connect to GND, turn on (1) to pull down to 0
  - p-type: connect to +2.9V, turn on (0) to pull up to 1
- Basic gates: NOT, NOR, NAND
  - Logic functions are usually expressed with AND, OR, and NOT
- DeMorgan's Law
  - Convert AND to OR (and vice versa) by inverting inputs and output

Building Functions from Logic Gates

- ***Combinational Logic Circuit***
  - output depends only on the current inputs
  - stateless
- ***Sequential Logic Circuit***
  - output depends on the sequence of inputs (past and present)
  - stores information (state) from past inputs
- We'll first look at some useful combinational circuits, then show how to use sequential circuits to store information.