Digital Logic: From Transistors to Gates
The Transistor

• Transistor: building block of computers
• Microprocessors contain tons of transistors
  – Intel Core i7-5960X (2015): 2.6 Billion
  – Qualcomm Snapdragon 810 (2015): multi billions
  – AMD 6-core Opteron (2009): 904 million
  – Intel Core i7 Quad (2008): 731 million
  – Intel 4004 (1971): 2300
  – Intel 8008
The Transistor: Past and Present

- 1947 first point-contact transistor

Discrete transistor
Moore’s Law

Microprocessor Transistor Counts 1971-2011 & Moore’s Law

The number of transistors on a microprocessor chip doubles approximately every 18 months.

Curriculum Map

Moore’s Law (1965): The number of transistors on a microprocessor chip doubles approximately every 18 months.

In 1971, Jack Kilby of Texas Instruments and Robert Noyce of Fairchild Semiconductor independently invented the first integrated circuit, which was a combination of a chip and a microprocessor.

In 1971, David Packard of Hewlett-Packard (HP) made the first computer using integrated circuits.

In 1971, Intel introduced the 4004 microprocessor, which was the first successful microprocessor.

In 1971, AMD was founded by Wally淇er and Jim Clark.

In 1971, MOS Technology was founded by Andrew Grove.

In 1971, Busicom introduced the BC-1000, the first business computer.

In 1971, Digital Equipment Corporation (DEC) introduced the PDP-8, the first minicomputer.

In 1971, Xerox introduced the Xerox Alto, the first graphical user interface computer.

In 1971, Apple Inc. was founded by Steve Jobs and Steve Wozniak.

In 1971, IBM introduced the System/360, the first commercial mainframe computer.

In 1971, Xerox introduced the Xerox Star, the first commercial GUI computer.

In 1971, Honeywell introduced the Honeywell 9800, the first commercial minicomputer.

In 1971, International Business Machines (IBM) introduced the IBM 360, the first commercial mainframe computer.

In 1971, Digital Equipment Corporation (DEC) introduced the PDP-10, the first commercial minicomputer.

In 1971, Xerox introduced the Xerox 7000, the first commercial GUI computer.

In 1971, Apple Inc. introduced the Apple II, the first personal computer.

In 1971, IBM introduced the IBM 3081, the first commercial mainframe computer.

In 1971, Digital Equipment Corporation (DEC) introduced the PDP-11, the first commercial minicomputer.

In 1971, Xerox introduced the Xerox 8200, the first commercial GUI computer.

In 1971, Apple Inc. introduced the Apple III, the first successful personal computer.

In 1971, IBM introduced the IBM 3085, the first commercial mainframe computer.

In 1971, Digital Equipment Corporation (DEC) introduced the PDP-20, the first commercial minicomputer.

In 1971, Xerox introduced the Xerox 9200, the first commercial GUI computer.
What Is a Transistor?

• A switch, which can close between the **source** and the **drain**

• Changing the **voltage** of the gate lets you change the current flow between the **source** and drain (closing or opening the switch)

• Think of a light switch, the gate is the switch that allows electricity to flow from the **source** to the **drain**
MOSFET

Metal-Oxide-Semiconductor transistor

NMOS Transistor (n-channel MOSFET)

Silicon Dioxide (insulator)

source

gate

drain

gate electrode

p-type silicon

n-type silicon

n-channel

Silicon Substrate
How big is a transistor?

• If a CPU die were as big as this whole classroom...

• A transistor would be...
  • Core i7-5960X (8 cores and 20MB cache)
    • 2.6 billion transistors in ~355 mm²
    • 1 mm² has over 7.3 million transistors

• If you take the CPU and stretch it to the size of a 10 m² room, a transistor would still be tiny.
  • About 7 transistors would fit on the end of a ball point pen (1 sq mm).
What is a transistor?

- Logically, each transistor is used as a switch
- Combined to implement logic functions
  - AND, OR, NOT
- Combined to build higher-level structures
  - Adder, multiplexer, decoder, register, ...
- Combined to build a processor
  - LC-3, Core i7, A9, etc
Simple switch circuit

Switch open:
- No current through circuit
- 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

n-type MOS (nMOS)

- 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 in this example.
Digital Values for Analog Signals

- Use the switch behavior of MOS transistors to implement logical functions: AND, OR, NOT
- Digital symbols:
  - We assign a range of analog voltages to each digital (logic) symbol
  - Assignment of voltage ranges depends on electrical properties of transistors being used
CMOS circuit

- CMOS is Complementary Metal Oxide Semiconductor
- Uses both n-type and p-type MOS transistors
  - p-type (pMOS)
    - Attached to + voltage
    - Pulls output voltage UP when input is zero
  - n-type (nMOS)
    - Attached to GND
    - Pulls output voltage DOWN when input is one
- Faster than using just one type
Truth Table: Inverter

- Inverted signals are denoted with an overbar
- Or with a prime symbol
  - \( A' \)
- Or with a bubble in a circuit diagram

<table>
<thead>
<tr>
<th>Input</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
Inverter (NOT gate)

<table>
<thead>
<tr>
<th>In</th>
<th>Out</th>
<th>In</th>
<th>Out</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 V</td>
<td>2.9 V</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>2.9 V</td>
<td>0 V</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

P-type

N-type

In=0 → Out=1

In=1 → Out=0
Truth Table: AND Gate

- The result of an AND operation is 1 if and only if all inputs are 1
- Depict AND by the multiplication symbol
  - $A \cdot B$
- Or by lumping the signals together
  - $AB$
- We don’t really build these gates...

<table>
<thead>
<tr>
<th>Inputs</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A$</td>
<td>$B$</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
NAND gate (NOT-AND)

Note: Parallel structure on top, serial on bottom.
AND gate

Add an inverter to a NAND.
Truth Table: OR Gate

- The result of an OR operation is 1 if and only if any inputs are 1
- Depict OR by the addition symbol
  - A+B

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>Y = A + B</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
NOR Gate: NOT-OR

Note: Serial structure on top, parallel on bottom.
OR gate

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Add an inverter to a NOR gate.
Truth table to transistors

- So giving some arbitrary truth table, how do you go about creating a transistor-based circuit for it?
- Typically this is only done for a handful of gate types.
- Recall:
  - PMOS (with the bubbles) on top
  - NMOS (no bubbles) on bottom
  - Series structure makes AND
  - Parallel structure makes OR
Transistor based designing

- How do you get from a truth table a transistor based circuit?
- Procedure:
  1. Find the rows with the ‘1’ output
  2. Use these to form the “pull-up” part of the circuit, remember p-type are active low
  3. Find the rows with the ‘0’ output
  4. Use these to form the “pull-down” part of the circuit, remember n-type are active high

Note: This is not optimal
Simple example

- XOR Gate – one or the other, but not both

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
Synthesis of an AOI Gate

- AOI means AND-OR-Invert
Synthesis of AOI Gate

2-1 AOI

<table>
<thead>
<tr>
<th>INPUT</th>
<th>OUTPUT</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>B</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

3 inputs 5 outputs

3 \times 3 = 9 P MOS

5 \times 3 = 15 N MOS
Synthesis of AOI Gate

3 inputs 5 0's 3.3 = 9 PMOS
5.3 = 15 NMOS

<table>
<thead>
<tr>
<th>INPUT</th>
<th>OUTPUT</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>B</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

\[ A \rightarrow \overline{A} \rightarrow A \rightarrow \overline{C} \rightarrow C \rightarrow F \]
Why are our circuits so big?

• The circuits for the NAND and NOR were a lot smaller than the one we just did. Why is that?
• We just used the brute force method.
• To do correctly you need the Function and the dual of the function. E.g. for NAND:
  • $F = (AB)' = A' + B'$
  • $F' = ((AB)')' = AB$
Nvidia
980

960

$888$

$55$